

HC32A4A8 系列

32 位 ARM[®] Cortex[®]-M4 微控制器

参考手册

Rev1.00
2025 年 6 月

说明事项

版权所有©2025 小华半导体有限公司。保留所有权利

本文件及附件包含的信息有关知识产权权益全部属于小华半导体有限公司（以下简称“XHSC”）；客户对本文件及附件包含的信息只享有内部使用权，未经 XHSC 书面允许，任何单位和个人不得擅自摘抄、复制、改动或以其他任何形式使用本文件的部分或全部内容，并不得以任何形式进行传播。

商标声明

 小华半导体有限公司  小华半导体和其他商标均为 XHSC 的商标。所有其他在 XHSC 产品上显示的产品或服务名称均为其各自所有者的财产。

注意事项

- XHSC 保留随时更改、更正、增强、修改产品和/或本文档的权利，恕不另行通知。用户可在下单前获取最新相关信息。XHSC 产品依据购销基本合同中载明的销售条款和条件进行销售。
- 客户应针对您的应用选择合适的 XHSC 产品，并设计、验证和测试您的应用，以确保您的应用满足相应标准以及任何安全、安保或其它要求。客户应对此独自承担全部责任。
- XHSC 在此确认未以明示或暗示方式授予任何知识产权许可。
- XHSC 产品的转售，若其条款与此处规定不同，XHSC 对此类产品的任何保修承诺无效。
- 本通知中的信息取代并替换先前版本中的信息。

小华半导体有限公司

地址：	上海市浦东新区中科路 1867 号 A 座 4 楼
网址：	https://www.xhsc.com.cn
邮箱：	XHSC_MCU@xhsc.com.cn
电话：	021-38880888-887

前言

位/位域

寄存器包含位（一个位宽）或者位域（多个位宽）的字段，位域通常描述成[y:x]，从 x 位开始到 y 位结束。没有特殊说明的情况下，位域字段使用的是十进制数。

不建议使用没有指定的字段，否则可能导致不可预知后果。

寄存器地址

每个寄存器的地址可以通过系统地址划分的基地址以及寄存器的偏移地址（详见各模块的寄存器总表）获得。

寄存器属性

寄存器说明中使用以下缩写词：

缩写词	说明
RW	读写，可以读写该位。
RO	只读，只能读取该位。
WO	只写，只能写入该位。
RWA	读写，可以读写该位。 仅解锁或者安全模式下可写。
RWC	读写，可以读写该位。 写入任意值将该位清零。
RC	只读，只能读取该位。 读取后该位自动清零。
RS	只读，只能读取该位。 读取后该位自动置位。

数据格式

- 0x 前缀表示十六进制数据
- 0b 前缀表示二进制数据
- 数字没有前缀表示十进制数据

Reserved (Res.)

标记为 Reserved 的寄存器和位/位域是为了将来使用预留，必须保持复位值（寄存器另有说明的场景除外）。Reserved 位的读取值在不同设备或者特性中可能不同。

复位值 (Reset)

复位值表示寄存器重置之后的值。寄存器在使用前需要进行初始化重置，否则可能导致不可预知后果。

外设可用性说明

HC32A4A8 系列 32 位 ARM® Cortex®-M4 微控制器所支持的外设以及其数目，请参见相应的《数据手册》。

安全声明

由于使用某个功能或者协议，可能会存在潜在的安全问题，需要进行声明，提醒用户慎用，规避安全风险。

修订记录

文档版本	发布日期	修改说明
Rev1.00	2025/06/16	第一次正式发布。

目录

前言.....	iii
修订记录.....	iv
1 产品简介.....	1
2 存储器映射 (Memory Mapping)	2
2.1 存储器映射.....	2
2.2 外部空间映射.....	6
2.3 位段空间.....	7
2.4 地址重映射.....	7
2.5 寄存器.....	8
2.5.1 寄存器总表.....	8
2.5.2 访问保护寄存器 (MMF_REMPRT)	9
2.5.3 重映射寄存器 (MMF_REMCrx) (x=0~1)	10
3 总线架构 (BUS)	11
3.1 简介.....	11
3.2 总线架构.....	12
3.3 总线功能.....	13
4 复位控制 (RMU)	14
4.1 简介.....	14
4.2 功能说明.....	14
4.2.1 复位方式和复位标志位.....	14
4.2.2 复位时序.....	16
4.2.3 复位方式判断.....	21
4.2.4 模块功能复位.....	22
4.2.5 复位条件.....	22
4.3 寄存器.....	25
4.3.1 寄存器总表.....	25
4.3.2 功能复位控制寄存器 0 (RMU_FRST0)	26
4.3.3 功能复位控制寄存器 1 (RMU_FRST1)	28
4.3.4 功能复位控制寄存器 2 (RMU_FRST2)	30
4.3.5 功能复位控制寄存器 3 (RMU_FRST3)	32
4.3.6 复位控制寄存器 0 (RMU_PRSTCR0)	34
4.3.7 复位标志寄存器 0 (RMU_RSTF0)	35
5 时钟控制器 (CMU)	38
5.1 简介.....	38
5.2 主要特性.....	38
5.3 功能说明.....	40
5.3.1 系统框图.....	40
5.3.2 工作时钟规格.....	41
5.3.3 晶振电路.....	43
5.3.4 内部 RC 时钟.....	46
5.3.5 PLL 时钟.....	48
5.3.6 时钟切换步骤.....	48
5.3.7 时钟输出功能.....	50
5.3.8 时钟频率测量.....	51
5.4 寄存器.....	53
5.4.1 寄存器总表.....	53
5.4.2 CMU XTAL 配置寄存器 (CMU_XTALCFGR)	55
5.4.3 CMU XTAL 稳定配置寄存器 (CMU_XTALSTBCR)	56

5.4.4	CMU XTAL 小数分频配置寄存器 (CMU_XTALDIVR)	57
5.4.5	CMU XTAL 小数分频控制寄存器 (CMU_XTALDIVCR)	57
5.4.6	CMU XTAL 控制寄存器 (CMU_XTALCR)	58
5.4.7	CMU XTAL 振荡故障控制寄存器 (CMU_XTALSTDCCR)	59
5.4.8	CMU XTAL 振荡故障状态寄存器 (CMU_XTALSTDSCR)	60
5.4.9	CMU XTAL32 配置寄存器 (CMU_XTAL32CFGR)	60
5.4.10	CMU XTAL32 滤波寄存器 (CMU_XTAL32NFR)	61
5.4.11	CMU XTAL32 控制寄存器 (CMU_XTAL32CR)	61
5.4.12	CMU HRC 校准寄存器 (CMU_HRCTRM)	62
5.4.13	CMU HRC 控制寄存器 (CMU_HRCCR)	63
5.4.14	CMU MRC 校准寄存器 (CMU_MRCTRM)	64
5.4.15	CMU MRC 控制寄存器 (CMU_MRCCR)	65
5.4.16	CMU LRC 校准寄存器 (CMU_LRCTRM)	66
5.4.17	CMU LRC 控制寄存器 (CMU_LRCCR)	66
5.4.18	CMU RTCLRC 校准寄存器 (CMU_RTCLRCTRM)	67
5.4.19	CMU PLLH 配置寄存器 (CMU_PLLHCFGR)	68
5.4.20	CMU PLLH 控制寄存器 (CMU_PLLHCR)	70
5.4.21	CMU PLLA 配置寄存器 (CMU_PLLACFGR)	71
5.4.22	CMU PLLA 控制寄存器 (CMU_PLLACR)	73
5.4.23	CMU 时钟源稳定状态寄存器 (CMU_OSCSTBSR)	74
5.4.24	CMU 系统时钟源切换寄存器 (CMU_CKSWR)	75
5.4.25	CMU 时钟分频配置寄存器 (CMU_SCFGR)	76
5.4.26	CMU USB 时钟配置寄存器 (CMU_USBCKCFGR)	78
5.4.27	CMU CAN 时钟配置寄存器 (CMU_CANCKCFGR)	79
5.4.28	CMU ADC/DAC 时钟配置寄存器 (CMU_PERICKSEL)	81
5.4.29	CMU 调试时钟配置寄存器 (CMU_TPIUCKCFGR)	82
5.4.30	CMU I2S 时钟配置寄存器 (CMU_I2SCKCFGR)	83
5.4.31	CMU MCO1 时钟输出配置寄存器 (CMU_MCO1CFGR)	84
5.4.32	CMU MCO2 时钟输出配置寄存器 (CMU_MCO2CFGR)	85
5.4.33	FCM 下限比较值寄存器 (FCMn_LVR) (n=1~4)	86
5.4.34	FCM 上限比较值寄存器 (FCMn_UVR) (n=1~4)	86
5.4.35	FCM 计数器值寄存器 (FCMn_CNTR) (n=1~4)	87
5.4.36	FCM 开始停止寄存器 (FCM_STR)	87
5.4.37	FCM 测量对象控制寄存器 (FCMn_MCCR) (n=1~4)	88
5.4.38	FCM 测量基准控制寄存器 (FCMn_RCCR) (n=1~4)	89
5.4.39	FCM 中断复位控制寄存器 (FCMn_RIER) (n=1~4)	90
5.4.40	FCM 标志寄存器 (FCM_SR)	91
5.4.41	FCM 标志位清除寄存器 (FCMn_CLR) (n=1~4)	92
6	电源控制 (PWC)	93
6.1	简介	93
6.2	功能说明	94
6.2.1	电源分布	94
6.2.2	电源电压检测单元 (PVD) 说明	95
6.2.3	唤醒定时器	101
6.2.4	动作模式与低功耗模式	101
6.3	寄存器	111
6.3.1	寄存器总表	111
6.3.2	电源模式控制寄存器 0 (PWC_PWRC0)	113
6.3.3	电源模式控制寄存器 1 (PWC_PWRC1)	114
6.3.4	电源模式控制寄存器 2 (PWC_PWRC2)	115
6.3.5	电源模式控制寄存器 3 (PWC_PWRC3)	115
6.3.6	电源模式控制寄存器 4 (PWC_PWRC4)	116

6.3.7	掉电唤醒使能寄存器 0 (PWC_PDWKE0)	117
6.3.8	掉电唤醒使能寄存器 1 (PWC_PDWKE1)	118
6.3.9	掉电唤醒使能寄存器 2 (PWC_PDWKE2)	119
6.3.10	掉电唤醒事件边沿选择寄存器 (PWC_PDWKES)	120
6.3.11	掉电唤醒标志寄存器 0 (PWC_PDWKF0)	121
6.3.12	掉电唤醒标志寄存器 1 (PWC_PDWKF1)	122
6.3.13	RAM 功耗控制寄存器 0 (PWC_RAMPC0)	123
6.3.14	RAM 运行条件寄存器 (PWC_RAMOPM)	124
6.3.15	外设 RAM 低功耗控制寄存器 (PWC_PRAMLPC)	125
6.3.16	低功耗状态寄存器 (PWC_LPMCSR)	126
6.3.17	低功耗状态清除寄存器 (PWC_LPMCSCR)	126
6.3.18	PVD 控制寄存器 0 (PWC_PVDCR0)	127
6.3.19	PVD 控制寄存器 1 (PWC_PVDCR1)	128
6.3.20	PVD 滤波控制寄存器 (PWC_PVDFCR)	129
6.3.21	PVD 电平控制寄存器 (PWC_PVDLCR)	130
6.3.22	PVD 中断控制寄存器 (PWC_PVDICR)	131
6.3.23	PVD 检测状态寄存器 (PWC_PVDDSR)	132
6.3.24	功能时钟控制 0 (PWC_FCG0)	133
6.3.25	功能时钟控制 1 (PWC_FCG1)	135
6.3.26	功能时钟控制 2 (PWC_FCG2)	137
6.3.27	功能时钟控制 3 (PWC_FCG3)	139
6.3.28	PWC_FCG0 保护控制 (PWC_FCG0PC)	141
6.3.29	功能保护控制寄存器 (PWC_FPRC)	142
6.3.30	STOP 模式控制寄存器 (PWC_STPMCRR)	143
6.3.31	备份域复位寄存器 (PWC_VBATRSTR)	143
6.3.32	备份域控制寄存器 (PWC_VBATCR)	144
6.3.33	备份域上电标志寄存器 (PWC_VBATPORF)	145
6.3.34	备份域上电标志清除寄存器 (PWC_VBATPORFCLR)	145
6.3.35	唤醒计时器控制寄存器 0 (PWC_WKTC0)	146
6.3.36	唤醒计时器控制寄存器 1 (PWC_WKTC1)	146
6.3.37	唤醒计时器控制寄存器 2 (PWC_WKTC2)	147
6.3.38	VBAT 备份寄存器 n (PWC_BKRn) (n=0~127)	147
7	初始化配置 (ICG)	148
7.1	简介	148
7.2	寄存器	148
7.2.1	寄存器总表	148
7.2.2	初始化配置寄存器 0 (ICG0)	149
7.2.3	初始化配置寄存器 1 (ICG1)	151
7.2.4	初始化配置寄存器 3 (ICG3)	151
7.2.5	初始化配置寄存器 8x (ICG8x) (x=A/B/C)	152
8	嵌入式 Flash (EFM)	153
8.1	简介	153
8.2	主要特性	153
8.3	功能说明	154
8.3.1	嵌入式 Flash	154
8.3.2	读接口	154
8.3.3	Flash 读加速缓存	155
8.3.4	Flash 编程和擦除操作	156
8.3.5	一次性可编程字节 (OTP)	160
8.3.6	引导交换	161
8.3.7	数据可靠性和完整性	163

8.3.8 中断.....	163
8.4 寄存器.....	165
8.4.1 寄存器总表.....	165
8.4.2 Flash 访问写保护寄存器 (EFM_FAPRT)	167
8.4.3 Flash 密钥 1 寄存器 (EFM_KEY1)	167
8.4.4 Flash 密钥 2 寄存器 (EFM_KEY2)	168
8.4.5 Flash 停止寄存器 (EFM_FSTP)	168
8.4.6 Flash 读模式寄存器 (EFM_FRMC)	169
8.4.7 Flash 擦写模式寄存器 (EFM_FWMC)	170
8.4.8 Flash 状态寄存器 (EFM_FSR)	171
8.4.9 Flash 状态清除寄存器 (EFM_FSCLR)	174
8.4.10 Flash 中断许可寄存器 (EFM_FITE)	176
8.4.11 Flash 引导交换状态寄存器 (EFM_FSWP)	177
8.4.12 Flash 读模式寄存器 2 (EFM_FRMC2)	177
8.4.13 芯片专属标志寄存器 (EFM_CHIPID)	178
8.4.14 UNIQUE ID 寄存器 (EFM_UQID0)	178
8.4.15 UNIQUE ID 寄存器 (EFM_UQID1)	178
8.4.16 UNIQUE ID 寄存器 (EFM_UQID2)	179
8.4.17 Flash 写保护锁定寄存器 (EFM_WLOCK)	180
8.4.18 Flash0 写保护寄存器 0 (EFM_F0NWPRT0)	182
8.4.19 Flash0 写保护寄存器 1 (EFM_F0NWPRT1)	182
8.4.20 Flash0 写保护寄存器 2 (EFM_F0NWPRT2)	183
8.4.21 Flash0 写保护寄存器 3 (EFM_F0NWPRT3)	183
8.4.22 Flash1 写保护寄存器 0 (EFM_F1NWPRT0)	184
8.4.23 Flash1 写保护寄存器 1 (EFM_F1NWPRT1)	184
8.4.24 Flash1 写保护寄存器 2 (EFM_F1NWPRT2)	185
8.4.25 Flash1 写保护寄存器 3 (EFM_F1NWPRT3)	185
8.4.26 Flash 校验控制寄存器 (EFM_CKCR)	186
8.4.27 Flash 校验保护寄存器 (EFM_CKPR)	187
8.4.28 Flash 校验状态寄存器 (EFM_CKSR)	188
8.4.29 Flash 错误注入使能寄存器 (EFM_EIEN)	189
8.4.30 Flash0 ECC 控制寄存器 (EFM_F0ECCCR)	189
8.4.31 Flash0 ECC 数据寄存器 (EFM_F0ECCDR)	190
8.4.32 Flash0 错误记录寄存器 n (EFM_F0ERRn) (n=0~1)	191
8.4.33 Flash0 错误注入位使能寄存器 n (EFM_F0EIRn) (n=0~4)	192
8.4.34 Flash1 ECC 控制寄存器 (EFM_F1ECCCR)	193
8.4.35 Flash1 ECC 数据寄存器 (EFM_F1ECCDR)	193
8.4.36 Flash1 错误记录寄存器 n (EFM_F1ERRn) (n=0~1)	194
8.4.37 Flash1 错误注入位使能寄存器 n (EFM_F1EIRn) (n=0~4)	195
8.5 注意事项.....	196
9 内置 SRAM (SRAM)	197
9.1 简介.....	197
9.2 功能说明.....	197
9.2.1 等待周期.....	197
9.2.2 SRAM 空间分配.....	197
9.2.3 ECC 校验错误注入.....	198
9.3 寄存器.....	200
9.3.1 寄存器总表.....	200
9.3.2 SRAM 等待控制寄存器 (SRAMC_WTCR)	202
9.3.3 SRAM 等待控制保护寄存器 (SRAMC_WTPR)	203
9.3.4 SRAM 校验控制寄存器 (SRAMC_CKCR)	204
9.3.5 SRAM 校验控制保护寄存器 (SRAMC_CKPR)	207

9.3.6	SRAM 校验状态寄存器 (SRAMC_CKSR)	208
9.3.7	SRAM1 错误注入使能寄存器 (SRAM1_EIEN)	210
9.3.8	SRAM1 错误注入位使能寄存器 0 (SRAM1_EIBIT0)	210
9.3.9	SRAM1 错误注入位使能寄存器 1 (SRAM1_EIBIT1)	211
9.3.10	SRAM1 错误地址寄存器 (SRAM1_ECCERRADDR)	211
9.3.11	SRAM2 错误注入使能寄存器 (SRAM2_EIEN)	212
9.3.12	SRAM2 错误注入位使能寄存器 0 (SRAM2_EIBIT0)	212
9.3.13	SRAM2 错误注入位使能寄存器 1 (SRAM2_EIBIT1)	213
9.3.14	SRAM2 错误地址寄存器 (SRAM2_ECCERRADDR)	213
9.3.15	SRAM3 错误注入使能寄存器 (SRAM3_EIEN)	214
9.3.16	SRAM3 错误注入位使能寄存器 0 (SRAM3_EIBIT0)	214
9.3.17	SRAM3 错误注入位使能寄存器 1 (SRAM3_EIBIT1)	215
9.3.18	SRAM3 错误地址寄存器 (SRAM3_ECCERRADDR)	215
9.3.19	SRAM4 错误注入使能寄存器 (SRAM4_EIEN)	216
9.3.20	SRAM4 错误注入位使能寄存器 0 (SRAM4_EIBIT0)	216
9.3.21	SRAM4 错误注入位使能寄存器 1 (SRAM4_EIBIT1)	217
9.3.22	SRAM4 错误地址寄存器 (SRAM4_ECCERRADDR)	217
9.3.23	SRAMH 错误注入使能寄存器 (SRAMH_EIEN)	218
9.3.24	SRAMH 错误注入位使能寄存器 0 (SRAMH_EIBIT0)	218
9.3.25	SRAMH 错误注入位使能寄存器 1 (SRAMH_EIBIT1)	219
9.3.26	SRAMH 错误地址寄存器 (SRAMH_ECCERRADDR)	219
9.3.27	Ret SRAM 错误注入使能寄存器 (SRAMB_EIEN)	220
9.3.28	Ret SRAM 错误注入位使能寄存器 0 (SRAMB_EIBIT0)	220
9.3.29	Ret SRAM 错误注入位使能寄存器 1 (SRAMB_EIBIT1)	221
9.3.30	Ret SRAM 错误地址寄存器 (SRAMB_ECCERRADDR)	221
9.3.31	CACHERAM 错误注入使能寄存器 (CACHE_EIEN)	222
9.3.32	CACHERAM 错误注入位使能寄存器 0 (CACHE_EIBIT0)	222
9.3.33	CACHERAM 错误注入位使能寄存器 1 (CACHE_EIBIT1)	223
9.3.34	CACHERAM 错误注入位使能寄存器 2 (CACHE_EIBIT2)	223
9.3.35	CACHERAM 错误注入位使能寄存器 3 (CACHE_EIBIT3)	224
9.3.36	CACHERAM 错误注入位使能寄存器 4 (CACHE_EIBIT4)	224
9.3.37	CACHERAM 错误地址寄存器 (CACHE_ECCERRADDR)	225
9.4	注意事项	225
10	通用 IO (GPIO)	226
10.1	简介	226
10.2	主要特性	226
10.3	功能说明	227
10.3.1	功能框图	227
10.3.2	通用输入输出 GPIO 功能	227
10.3.3	周边功能	228
10.3.4	双周边功能	228
10.3.5	Event Port 输入输出功能	228
10.3.6	外部中断 EIRQ 输入功能	228
10.3.7	模拟功能	229
10.3.8	通用控制	229
10.4	寄存器	230
10.4.1	寄存器总表	230
10.4.2	通用输入寄存器 (PIDRx)	232
10.4.3	通用输出数据寄存器 (PODRx)	232
10.4.4	通用输出许可寄存器 (POERx)	233
10.4.5	通用输出置位寄存器 (POSRx)	233
10.4.6	通用输出复位寄存器 (PORRx)	234

10.4.7	通用输出翻转寄存器 (POTRx)	234
10.4.8	特殊控制寄存器 (PSPCR)	235
10.4.9	公共控制寄存器 (PCCR)	236
10.4.10	输入控制寄存器 (PINAER)	237
10.4.11	写保护寄存器 (PWPR)	238
10.4.12	通用控制寄存器 (PCRxy)	239
10.4.13	功能选择寄存器 (PFSRxy)	240
10.4.14	Event Port 方向选择寄存器 (PEVNTDIRRm)	241
10.4.15	Event Port 输入数据寄存器 (PEVNTIDRm)	241
10.4.16	Event Port 输出数据寄存器 (PEVNTODRm)	242
10.4.17	Event Port 输出数据复位寄存器 (PEVNTORRm)	242
10.4.18	Event Port 输出数据置位寄存器 (PEVNTOSRm)	243
10.4.19	Event Port 上升沿输入许可寄存器 (PEVNTRISRm)	243
10.4.20	Event Port 下降沿输入许可寄存器 (PEVNTFALRm)	244
10.4.21	Event Port 输入滤波控制寄存器 (PEVNTNFCR)	245
10.5	注意事项	246
11	中断控制器 (INTC)	247
11.1	简介	247
11.2	向量表	248
11.2.1	中断向量表	248
11.2.2	中断事件表	251
11.3	功能说明	264
11.3.1	功能框图	264
11.3.2	不可屏蔽中断	264
11.3.3	外部中断	265
11.3.4	中断事件多路器	265
11.3.5	软件中断	265
11.3.6	事件输入	265
11.3.7	WFE 事件管理	266
11.3.8	噪声滤波器	266
11.3.9	低功耗模式返回	267
11.4	寄存器	268
11.4.1	寄存器总表	268
11.4.2	外部管脚中断滤波控制寄存器 (INTC_NOCCR)	269
11.4.3	不可屏蔽中断使能寄存器 (INTC_NMIER)	270
11.4.4	不可屏蔽中断标志寄存器 (INTC_NMIFR)	271
11.4.5	不可屏蔽中断标志清除寄存器 (INTC_NMIFCR)	272
11.4.6	外部中断控制寄存器 (INTC_EIRQCRx) (x=0~15)	273
11.4.7	外部中断标志寄存器 (INTC_EIFR)	274
11.4.8	外部中断标志清除寄存器 (INTC_EIFCR)	274
11.4.9	中断事件选择寄存器 (INTC_INTSELx) (x=0~31)	275
11.4.10	中断选择寄存器 (INTC_INTSELy) (y=32~127)	276
11.4.11	向量共享中断选择寄存器 (INTC_VSSELz) (z=128~143)	277
11.4.12	停止模式唤醒事件使能寄存器 (INTC_WKEN)	278
11.4.13	软件中断事件寄存器 (INTC_SWIER)	279
11.4.14	事件使能寄存器 (INTC_EVTER)	279
11.4.15	中断使能寄存器 (INTC_IER)	280
11.4.16	FPU 中断使能寄存器 (INTC_FPUIER)	280
11.5	注意事项	280
12	自动运行系统 (AOS)	281
12.1	简介	281

12.2 主要特性.....	281
12.3 功能说明.....	282
12.3.1 功能框图.....	282
12.3.2 AOS 源事件和目标.....	282
12.3.3 专用触发源.....	284
12.3.4 公共触发源.....	284
12.3.5 可编程逻辑运算单元 (PLU)	284
12.4 寄存器.....	286
12.4.1 寄存器总表.....	286
12.4.2 软件触发事件寄存器 (INTSFTRG)	287
12.4.3 DCU 触发源选择寄存器 (DCU_TRGSELx) (x=1~4)	288
12.4.4 DMA1 传输启动触发源选择寄存器 (DMA1_TRGSELx) (x=0~7)	289
12.4.5 DMA2 传输启动触发源选择寄存器 (DMA2_TRGSELx) (x=0~7)	290
12.4.6 DMA 通道重置触发源选择寄存器 (DMA_RC_TRGSEL)	291
12.4.7 Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3)	292
12.4.8 Event Port 触发源选择寄存器 (PEVNT_TRGSEL12/PEVNT_TRGSEL34)	293
12.4.9 Timer0 硬件触发事件选择寄存器 (TMR0_TRGSEL)	294
12.4.10 Timer2 硬件触发事件选择寄存器 (TMR2_TRGSEL)	295
12.4.11 HASH 硬件触发事件选择寄存器 (HASH_TRGSELx) (x=A/B)	296
12.4.12 TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3)	297
12.4.13 OTS 硬件触发事件选择寄存器 (OTS_TRGSEL)	298
12.4.14 ADC1 转换开始片内触发源选择寄存器 (ADC1_TRGSELx) (x=0~1)	299
12.4.15 ADC2 转换开始片内触发源选择寄存器 (ADC2_TRGSELx) (x=0~1)	300
12.4.16 ADC3 转换开始片内触发源选择寄存器 (ADC3_TRGSELx) (x=0~1)	301
12.4.17 公共触发源选择寄存器 (AOS_COMTRGSELx) (x=1~2)	302
12.4.18 PLU 控制寄存器 (AOS_PLUX_CR) (x=0~3)	303
12.4.19 PLU 触发源选择寄存器 A (AOS_PLUX_TRGSELA) (x=0~3)	304
12.4.20 PLU 触发源选择寄存器 B (AOS_PLUX_TRGSELB) (x=0~3)	304
12.4.21 PLU 触发源选择寄存器 C (AOS_PLUX_TRGSELC) (x=0~3)	305
12.4.22 PLU 触发源选择寄存器 D (AOS_PLUX_TRGSELD) (x=0~3)	305
13 存储保护单元 (MPU)	306
13.1 简介.....	306
13.2 功能说明.....	307
13.2.1 区域范围设置.....	307
13.2.2 权限设置.....	307
13.2.3 MPU 动作选择.....	307
13.2.4 启动 MPU.....	307
13.3 操作示例.....	308
13.3.1 只允许部分空间访问.....	308
13.3.2 只禁止部分空间访问.....	308
13.4 寄存器.....	309
13.4.1 寄存器总表.....	309
13.4.2 区域范围描述寄存器 (MPU_RGDn) (n=0~15)	311
13.4.3 状态标志寄存器 (MPU_SR)	312
13.4.4 标志清除寄存器 (MPU_ECLR)	313
13.4.5 写保护寄存器 (MPU_WP)	313
13.4.6 IP 访问保护寄存器 (MPU_IPPR)	314
13.4.7 SMPU1 区域使能寄存器 (MPU_S1RGE)	317
13.4.8 SMPU1 区域写权限寄存器 (MPU_S1RGWP)	318
13.4.9 SMPU1 区域读权限寄存器 (MPU_S1RGRP)	319
13.4.10 SMPU1 控制寄存器 (MPU_S1CR)	320
13.4.11 SMPU2 区域使能寄存器 (MPU_S2RGE)	321

13.4.12	SMPU2 区域写权限寄存器 (MPU_S2RGWP)	322
13.4.13	SMPU2 区域读权限寄存器 (MPU_S2RGRP)	323
13.4.14	SMPU2 控制寄存器 (MPU_S2CR)	324
13.4.15	FMPU 区域使能寄存器 (MPU_FRGE)	325
13.4.16	FMPU 区域写权限寄存器 (MPU_FRGWP)	326
13.4.17	FMPU 区域读权限寄存器 (MPU_FRGRP)	327
13.4.18	FMPU 控制寄存器 (MPU_FCR)	328
13.4.19	HMPU 区域使能寄存器 (MPU_HRGE)	329
13.4.20	HMPU 区域写权限寄存器 (MPU_HRGWP)	330
13.4.21	HMPU 区域读权限寄存器 (MPU_HRGRP)	331
13.4.22	HMPU 控制寄存器 (MPU_HCR)	332
13.4.23	EMPU 区域使能寄存器 (MPU_ERGE)	333
13.4.24	EMPU 区域写权限寄存器 (MPU_ERGWP)	334
13.4.25	EMPU 区域读权限寄存器 (MPU_ERGRP)	335
13.4.26	EMPU 控制寄存器 (MPU_ECR)	336
14	键盘扫描控制模块 (KEYSCAN)	337
14.1	简介	337
14.2	主要特性	337
14.3	功能说明	338
14.3.1	功能框图	338
14.3.2	按键识别功能	338
14.3.3	键盘扫描功能	339
14.4	寄存器	340
14.4.1	寄存器总表	340
14.4.2	扫描控制寄存器 (KEYSCAN_SCR)	341
14.4.3	扫描使能寄存器 (KEYSCAN_SER)	342
14.4.4	扫描状态寄存器 (KEYSCAN_SSR)	342
14.5	注意事项	343
15	内部时钟校准器 (CTC)	344
15.1	简介	344
15.2	主要特性	344
15.3	功能说明	344
15.3.1	功能框图	344
15.3.2	参考时钟	344
15.3.3	频率校准	346
15.3.4	中断	346
15.4	操作示例	347
15.4.1	使用 XTAL32 作为参考时钟将 HRC 校准到 20MHz 且校准目标偏差小于 0.5%	347
15.5	寄存器	348
15.5.1	寄存器总表	348
15.5.2	时钟校准控制寄存器 1 (CTC_CR1)	349
15.5.3	时钟校准控制寄存器 2 (CTC_CR2)	351
15.5.4	时钟校准状态寄存器 (CTC_STR)	352
16	DMA 控制器 (DMA)	353
16.1	简介	353
16.2	主要特性	353
16.3	功能说明	354
16.3.1	DMA 框图	354
16.3.2	使能 DMA 控制器	354
16.3.3	通道选择和通道优先级	354
16.3.4	启动 DMA	354

16.3.5	数据块.....	355
16.3.6	传输地址控制.....	355
16.3.7	传输次数.....	355
16.3.8	连锁传输.....	355
16.3.9	不连续地址传输.....	357
16.3.10	通道重置.....	358
16.3.11	传输提前终止.....	360
16.3.12	中断.....	360
16.4	操作示例.....	361
16.4.1	存储器到存储器的传输.....	361
16.4.2	存储器到外围电路的传输.....	362
16.4.3	存储器到存储器的连锁传输.....	363
16.5	寄存器.....	365
16.5.1	寄存器总表.....	365
16.5.2	DMA 使能寄存器 (DMA_EN)	367
16.5.3	中断状态寄存器 0 (DMA_INTSTAT0)	367
16.5.4	中断状态寄存器 1 (DMA_INTSTAT1)	368
16.5.5	中断屏蔽寄存器 0 (DMA_INTMASK0)	369
16.5.6	中断屏蔽寄存器 1 (DMA_INTMASK1)	370
16.5.7	中断复位寄存器 0 (DMA_INTCLR0)	371
16.5.8	中断复位寄存器 1 (DMA_INTCLR1)	372
16.5.9	通道使能寄存器 (DMA_CHEN)	373
16.5.10	传输请求状态寄存器 (DMA_REQSTAT)	374
16.5.11	通道状态观测寄存器 (DMA_CHSTAT)	375
16.5.12	通道重置控制寄存器 (DMA_RCFGCTL)	376
16.5.13	软件触发寄存器 (DMA_SWREQ)	378
16.5.14	通道使能复位寄存器 (DMA_CHENCLR)	379
16.5.15	传输源地址寄存器 (DMA_SARx) (x=0~7)	379
16.5.16	传输目标地址寄存器 (DMA_DARx) (x=0~7)	380
16.5.17	数据控制寄存器 (DMA_DTCTLx) (x=0~7)	380
16.5.18	重复区域大小寄存器 (DMA_RPTx) (x=0~7)	381
16.5.19	重复区域大小寄存器 B (DMA_RPTBx) (x=0~7)	382
16.5.20	源地址不连续传输控制寄存器 (DMA_SNSEQCTLx) (x=0~7)	383
16.5.21	源地址不连续传输控制寄存器 B (DMA_SNSEQCTLBx) (x=0~7)	384
16.5.22	目标地址不连续传输控制寄存器 (DMA_DNSEQCTLx) (x=0~7)	385
16.5.23	目标地址不连续传输控制寄存器 B (DMA_DNSEQCTLBx) (x=0~7)	386
16.5.24	链指针寄存器 (DMA_LL Px) (x=0~7)	387
16.5.25	通道控制寄存器 (DMA_CHCTLx) (x=0~7)	387
16.5.26	传输源地址监视寄存器 (DMA_MONSARx) (x=0~7)	388
16.5.27	传输目标地址监视寄存器 (DMA_MONDARx) (x=0~7)	389
16.5.28	数据控制监视寄存器 (DMA_MONDTCTLx) (x=0~7)	390
16.5.29	重复区域计数器监视寄存器 (DMA_MONRPTx) (x=0~7)	391
16.5.30	源地址不连续传输计数器监视寄存器 (DMA_MONSSEQCTLx) (x=0~7)	392
16.5.31	目标地址不连续传输计数器监视寄存器 (DMA_MONDSEQCTLx) (x=0~7)	393
16.6	注意事项.....	393
17	电压比较器 (CMP)	394
17.1	简介.....	394
17.2	主要特性.....	394
17.3	功能说明.....	394
17.3.1	功能框图.....	394
17.3.2	输入选择.....	395
17.3.3	普通比较.....	396

17.3.4	窗口比较.....	397
17.3.5	空白窗口.....	397
17.3.6	迟滞功能.....	398
17.3.7	噪声滤波器.....	398
17.3.8	外部管脚输出.....	398
17.3.9	中断和事件.....	398
17.4	寄存器.....	399
17.4.1	寄存器总表.....	399
17.4.2	比较器工作模式寄存器 (CMP_MDR)	400
17.4.3	比较器滤波和中断寄存器 (CMP_FIR)	401
17.4.4	比较器输出控制寄存器 (CMP_OCR)	402
17.4.5	比较器正负端输入选择寄存器 (CMP_PMSR)	403
17.4.6	比较器电压输入源选择寄存器 (CMP_VISR) (仅 CMP1 和 CMP3 存在)	404
17.4.7	比较器输出空白窗口选择寄存器 (CMP_BWSR)	405
17.4.8	比较器输出空白窗口极性寄存器 (CMP_BWPR)	406
17.5	注意事项.....	407
17.5.1	模块停止功能.....	407
17.5.2	模块停止时的功耗.....	407
17.5.3	停止低功耗模式时的功耗.....	407
18	模数转换模块 (ADC)	408
18.1	简介.....	408
18.2	主要特性.....	408
18.3	功能说明.....	409
18.3.1	功能框图.....	409
18.3.2	ADC 时钟.....	410
18.3.3	通道选择.....	411
18.3.4	触发源选择.....	412
18.3.5	序列 A 单次扫描模式.....	412
18.3.6	序列 A 连续扫描模式.....	413
18.3.7	序列 A 数据缓冲模式.....	413
18.3.8	双序列扫描模式.....	414
18.3.9	模拟看门狗功能.....	416
18.3.10	模拟输入的采样时间和转换时间.....	417
18.3.11	ADC 数据寄存器自动清除功能.....	417
18.3.12	转换数据平均计算功能.....	418
18.3.13	过采样.....	418
18.3.14	可编程增益放大器 PGA.....	418
18.3.15	通道专用采样保持电路 SH.....	419
18.3.16	多 ADC 协同工作模式.....	420
18.3.17	中断和事件.....	424
18.4	寄存器.....	426
18.4.1	寄存器总表.....	426
18.4.2	启动寄存器 (ADC_STR)	428
18.4.3	控制寄存器 0 (ADC_CR0)	429
18.4.4	控制寄存器 1 (ADC_CR1)	431
18.4.5	控制寄存器 2 (ADC_CR2)	432
18.4.6	转换开始触发寄存器 (ADC_TRGSR)	433
18.4.7	通道选择寄存器 A (ADC_CHSELRA)	435
18.4.8	通道选择寄存器 B (ADC_CHSELRB)	436
18.4.9	平均通道选择寄存器 (ADC_AVCHSELR)	437
18.4.10	扩展通道选择寄存器 (ADC_EXCHSELR)	438
18.4.11	采样周期寄存器 (ADC_SSTRx) (x=0~15、L)	439

18.4.12	通道映射控制寄存器 (ADC_CHMUXRx) (x=0~3)	440
18.4.13	中断状态寄存器 (ADC_ISR)	442
18.4.14	中断许可寄存器 (ADC_ICR)	443
18.4.15	中断状态复位寄存器 (ADC_ISCLRR)	443
18.4.16	协同模式控制寄存器 (ADC_SYNCCR)	444
18.4.17	ADC 数据寄存器 (ADC_DRx)	446
18.4.18	模拟看门狗控制寄存器 (ADC_AWDCR)	448
18.4.19	模拟看门狗状态寄存器 (ADC_AWDSR)	449
18.4.20	模拟看门狗状态复位寄存器 (ADC_AWDSCCLR)	450
18.4.21	模拟看门狗窗口 x 阈值寄存器 (ADC_AWDxDRy) (x=0~1, y=0~1)	451
18.4.22	模拟看门狗窗口 x 比较通道选择寄存器 (ADC_AWDxCHSR) (x=0~1)	452
18.4.23	专用采样保持电路控制寄存器 (ADC_SHCR)	453
18.4.24	可编程增益放大器控制寄存器 (ADC_PGACRx) (x=1~3)	454
18.4.25	可编程增益放大器参考地选择寄存器 (ADC_PGAVSSENR)	455
18.5	注意事项	456
18.5.1	数据寄存器读取时注意事项	456
18.5.2	扫描完成中断处理注意事项	456
18.5.3	模块停止与低功耗设定的注意事项	456
18.5.4	ADC 转换模拟通道输入的引脚设定	456
18.5.5	噪声控制	456
19	数模转换器 (DAC)	457
19.1	简介	457
19.2	主要特性	457
19.3	功能说明	457
19.3.1	功能框图	457
19.3.2	DAC 转换	458
19.3.3	同步转换	458
19.3.4	外部数据转换	458
19.3.5	ADC 转换优先模式	458
19.4	寄存器	460
19.4.1	寄存器总表	460
19.4.2	数据寄存器 (DAC_DADRx) (y=1~2)	461
19.4.3	控制寄存器 (DAC_DACR)	462
19.4.4	模拟输出控制寄存器 (DAC_DAOCR)	463
19.4.5	ADC 转换优先控制寄存器 (DAC_DAADPCR)	464
19.5	注意事项	465
19.5.1	模块停止功能的设置	465
19.5.2	模块停止状态下的 DAC 动作	465
19.5.3	停止低功耗模式下的 DAC 动作	465
19.5.4	掉电低功耗模式下的 DAC 动作	465
19.5.5	使用输出放大器	465
20	温度传感器 (OTS)	466
20.1	简介	466
20.2	功能说明	466
20.2.1	功能框图	466
20.2.2	使用方法	466
20.3	寄存器	468
20.3.1	寄存器总表	468
20.3.2	OTS 控制寄存器 (OTS_CTL)	469
20.3.3	OTS 数据寄存器 1 (OTS_DR1)	470
20.3.4	OTS 数据寄存器 2 (OTS_DR2)	470

20.3.5	OTS 误差补偿寄存器 (OTS_ECR)	470
20.3.6	OTS 预置温度数据寄存器 1 (OTS_PDR1)	471
20.3.7	OTS 预置温度数据寄存器 2 (OTS_PDR2)	471
20.3.8	OTS 预置温度数据寄存器 3 (OTS_PDR3)	472
21	高级控制定时器 (Timer6)	473
21.1	简介	473
21.2	主要特性	473
21.3	功能说明	474
21.3.1	Timer6 框图	474
21.3.2	基本动作	475
21.3.3	时钟源选择	477
21.3.4	计数方向	477
21.3.5	计数器刷新	477
21.3.6	数字滤波	478
21.3.7	软件同步	478
21.3.8	硬件同步	479
21.3.9	脉宽测量	481
21.3.10	周期测量	482
21.3.11	缓存功能	482
21.3.12	通用 PWM 输出	487
21.3.13	正交编码计数	491
21.3.14	周期间隔响应	495
21.3.15	EMB 控制	496
21.3.16	PTO 功能	496
21.3.17	功能汇总表	503
21.3.18	中断和事件	504
21.4	操作示例	506
21.4.1	基本计数及中断动作	506
21.4.2	比较输出及中断动作	506
21.4.3	捕获输入及中断动作	506
21.4.4	缓存传送动作 (周期基准值)	506
21.4.5	缓存传送动作 (通用比较基准值)	507
21.4.6	缓存传送动作 (专用比较基准值)	507
21.4.7	缓存传送动作 (死区基准值)	507
21.4.8	同步启动动作 (软件方式)	508
21.4.9	同步启动动作 (硬件方式)	508
21.4.10	正交编码计数动作 (2 相)	508
21.4.11	正交编码计数动作 (3 相)	508
21.4.12	单路 PWM 输出	509
21.4.13	互补 PWM 输出 (软件死区)	509
21.4.14	互补 PWM 输出 (硬件死区)	509
21.4.15	EMB 监控及中断动作	510
21.5	寄存器	511
21.5.1	寄存器总表	511
21.5.2	通用计数值寄存器 (TMR6_CNTER)	514
21.5.3	通用刷新值寄存器 (TMR6_UPDAR)	514
21.5.4	通用周期基准值寄存器 (TMR6_PERmR) (m=A~C)	515
21.5.5	通用比较基准值寄存器 (TMR6_GCMmR) (m=A~F)	515
21.5.6	专用比较基准值寄存器 (TMR6_SCMmR) (m=A~F)	516
21.5.7	死区时间基准值寄存器 (TMR6_DTmR) (m=U/D, n=A/B)	516
21.5.8	通用控制寄存器 (TMR6_GCONR)	517
21.5.9	中断控制寄存器 (TMR6_ICONR)	519

21.5.10	缓存控制寄存器 (TMR6_BCONR)	521
21.5.11	死区控制寄存器 (TMR6_DCONR)	524
21.5.12	端口控制寄存器 A (TMR6_PCNAR)	525
21.5.13	端口控制寄存器 B (TMR6_PCNBR)	528
21.5.14	滤波控制寄存器 (TMR6_FCNGR)	531
21.5.15	有效周期寄存器 (TMR6_VPERR)	532
21.5.16	状态标志寄存器 (TMR6_STFLR)	533
21.5.17	硬件启动事件选择寄存器 (TMR6_HSTAR)	535
21.5.18	硬件停止事件选择寄存器 (TMR6_HSTPR)	537
21.5.19	硬件清零事件选择寄存器 (TMR6_HCLRR)	539
21.5.20	硬件刷新事件选择寄存器 (TMR6_HUPDR)	541
21.5.21	硬件捕获事件选择寄存器 A (TMR6_HCPAR)	543
21.5.22	硬件捕获事件选择寄存器 B (TMR6_HCPBR)	545
21.5.23	硬件递加事件选择寄存器 (TMR6_HCUPR)	547
21.5.24	硬件递减事件选择寄存器 (TMR6_HCDOR)	549
21.5.25	PTO 脉冲个数寄存器 (TMR6_PTNUMR)	551
21.5.26	PTO 脉冲周期寄存器 (TMR6_PTPERR)	551
21.5.27	PTO 算法周期余数寄存器 (TMR6_PTRMDR)	551
21.5.28	PTO 模式控制寄存器 (TMR6_PTCTLR)	552
21.5.29	PTO A 相输出位置寄存器 (TMR6_PTAOPR)	552
21.5.30	PTO B 相输出位置寄存器 (TMR6_PTBOPR)	553
21.5.31	PTO Z 相输出开始位置寄存器 (TMR6_PTZOSPR)	553
21.5.32	PTO Z 相输出结束位置寄存器 (TMR6_PTZOEPR)	553
21.5.33	PTO Z 相输出脉宽压缩 B 寄存器 (TMR6_PTZCMPBR)	554
21.5.34	PTO 整圈脉冲个数寄存器 (TMR6_PTZPERR)	554
21.5.35	PTO 脉冲初始位置寄存器 (TMR6_PTPSPR)	554
21.5.36	PTO 脉冲初始端口寄存器 (TMR6_PTPSTR)	555
21.5.37	PTO 脉冲当前位置寄存器 (TMR6_PTPCPR)	555
21.5.38	PTO 脉冲当前端口寄存器 (TMR6_PTPCTR)	556
21.5.39	公共滤波控制寄存器 (TMR6_FCNTNTR)	557
21.5.40	公共软件同步启动控制寄存器 (TMR6_SSTAR)	558
21.5.41	公共软件同步停止控制寄存器 (TMR6_SSTPR)	559
21.5.42	公共软件同步清零控制寄存器 (TMR6_SCLRR)	559
21.5.43	公共软件同步刷新控制寄存器 (TMR6_SUPDR)	560
21.6	注意事项.....	561
22	通用控制定时器 (Timer4)	563
22.1	简介.....	563
22.2	主要特性.....	563
22.3	功能说明.....	564
22.3.1	Timer4 框图.....	564
22.3.2	基本动作.....	565
22.3.3	缓存功能.....	568
22.3.4	通用 PWM 输出.....	573
22.3.5	周期间隔响应.....	577
22.3.6	EMB 控制.....	579
22.3.7	监测输出.....	579
22.3.8	同步启动.....	580
22.3.9	中断和事件.....	580
22.4	寄存器.....	582
22.4.1	寄存器总表.....	582
22.4.2	计数值寄存器 (TMR4_CNTR)	585
22.4.3	周期基准寄存器 (TMR4_CPSR)	585

22.4.4	控制状态寄存器 (TMR4_CCSR)	586
22.4.5	有效周期寄存器 (TMR4_CVPR)	588
22.4.6	通用比较基准寄存器 (TMR4_OCCRm)	589
22.4.7	通用控制状态寄存器 (TMR4_OCSRn)	590
22.4.8	通用扩展控制寄存器 (TMR4_OCERn)	591
22.4.9	通用模式控制寄存器 (TMR4_OCMRm)	593
22.4.10	专用比较基准寄存器 (TMR4_SCCRm)	598
22.4.11	专用控制状态寄存器 (TMR4_SCSRm)	599
22.4.12	专用扩展控制寄存器 (TMR4_SCER)	601
22.4.13	专用模式控制寄存器 (TMR4_SCMRm)	602
22.4.14	PWM 基本控制寄存器 (TMR4_POCRn)	603
22.4.15	PWM 状态控制寄存器 (TMR4_PSCR)	604
22.4.16	PWM 滤波控制寄存器 (TMR4_PFSRn)	607
22.4.17	PWM 死区控制寄存器 A (TMR4_PDARn)	607
22.4.18	PWM 死区控制寄存器 B (TMR4_PDBRn)	607
22.4.19	重载控制状态寄存器 (TMR4_RCSR)	608
23	紧急刹车模块 (EMB)	610
23.1	简介	610
23.2	功能说明	610
23.2.1	EMB 通道功能框图	610
23.2.2	EMB 结构框图	612
23.2.3	外部端口输入电平变化时控制 PWM 信号输出	612
23.2.4	PWM 输出端口电平发生同相 (同高或同低) 时停止 PWM 信号输出	613
23.2.5	根据电压比较器比较结果停止 PWM 信号输出	614
23.2.6	系统错误发生时停止 PWM 信号输出	615
23.2.7	写寄存器软件控制 PWM 信号输出	615
23.3	寄存器	616
23.3.1	寄存器总表	616
23.3.2	EMB 控制寄存器 1 (EMB_CTL1) (Group0~3)	617
23.3.3	EMB 控制寄存器 1 (EMB_CTL1) (Group4~6)	619
23.3.4	EMB 控制寄存器 2 (EMB_CTL2) (Group0~3)	621
23.3.5	EMB 控制寄存器 2 (EMB_CTL2) (Group4~6)	623
23.3.6	EMB 软件输出使能控制寄存器 (EMB_SOE)	624
23.3.7	EMB 状态寄存器 (EMB_STAT)	625
23.3.8	EMB 状态复位寄存器 (EMB_STATCLR)	626
23.3.9	EMB 中断许可寄存器 (EMB_INTEN)	627
23.3.10	EMB 控制 PWM 输出释放方式选择寄存器 (EMB_RLSSEL)	628
24	通用定时器 (TimerA)	629
24.1	简介	629
24.2	主要特性	629
24.3	功能说明	630
24.3.1	TimerA 框图	630
24.3.2	对称单元	630
24.3.3	基本动作	631
24.3.4	时钟源选择	633
24.3.5	计数方向	633
24.3.6	数字滤波	634
24.3.7	同步启动	634
24.3.8	缓存功能	635
24.3.9	级联计数	636
24.3.10	通用 PWM 输出	636

24.3.11	正交编码计数.....	638
24.3.12	中断和事件.....	640
24.4	寄存器.....	642
24.4.1	寄存器总表.....	642
24.4.2	通用计数值寄存器 (TMRA_CNTER)	643
24.4.3	周期基准值寄存器 (TMRA_PERAR)	643
24.4.4	比较基准值寄存器 (TMRA_CMPARm) (m=1~4)	643
24.4.5	控制状态寄存器 L (TMRA_BCSTRL)	644
24.4.6	控制状态寄存器 H (TMRA_BCSTRH)	645
24.4.7	中断控制寄存器 (TMRA_ICONR)	646
24.4.8	事件控制寄存器 (TMRA_ECONR)	647
24.4.9	滤波控制寄存器 (TMRA_FCONR)	648
24.4.10	状态标志寄存器 (TMRA_STFLR)	649
24.4.11	缓存控制寄存器 (TMRA_BCONRm) (m=1~2)	651
24.4.12	捕获控制寄存器 (TMRA_CCONRm) (m=1~4)	652
24.4.13	端口控制寄存器 (TMRA_PCONRm) (m=1~4)	654
24.4.14	硬件触发事件选择寄存器 (TMRA_HCONR)	656
24.4.15	硬件递加事件选择寄存器 (TMRA_HCUPR)	658
24.4.16	硬件递减事件选择寄存器 (TMRA_HCDOR)	660
25	通用定时器 (Timer2)	662
25.1	简介.....	662
25.2	主要特性.....	662
25.3	功能说明.....	662
25.3.1	基本框图.....	662
25.3.2	时钟源选择.....	663
25.3.3	比较输出.....	664
25.3.4	硬件触发.....	665
25.3.5	数字滤波.....	667
25.3.6	软件同步启动.....	667
25.3.7	中断和事件.....	668
25.4	寄存器.....	669
25.4.1	寄存器总表.....	669
25.4.2	计数值寄存器 m (TMR2_CNTmR) (m=A/B)	670
25.4.3	基准值寄存器 m (TMR2_CMPmR) (m=A/B)	670
25.4.4	基本控制寄存器 (TMR2_BCONR)	671
25.4.5	中断控制寄存器 (TMR2_ICONR)	674
25.4.6	端口控制寄存器 (TMR2_PCONR)	675
25.4.7	硬件控制寄存器 (TMR2_HCONR)	677
25.4.8	状态标志寄存器 (TMR2_STFLR)	680
25.5	注意事项.....	681
26	通用定时器 (Timer0)	682
26.1	简介.....	682
26.2	功能说明.....	682
26.2.1	Timer0 框图.....	682
26.2.2	时钟源选择.....	682
26.2.3	基本计数.....	683
26.2.4	硬件触发.....	683
26.2.5	软件同步启动.....	683
26.2.6	中断和事件.....	684
26.3	寄存器.....	685
26.3.1	寄存器总表.....	685

26.3.2	计数值寄存器 m (TMR0_CNTmR) (m=A/B)	686
26.3.3	基准值寄存器 m (TMR0_CMPmR) (m=A/B)	686
26.3.4	基本控制寄存器 (TMR0_BCONR)	687
26.3.5	状态标志寄存器 (TMR0_STFLR)	690
26.3.6	计数同步使能寄存器 1 (TMR_SYNNR1)	691
26.3.7	计数同步使能寄存器 2 (TMR_SYNNR2)	694
26.4	注意事项	697
27	实时时钟 (RTC)	698
27.1	简介	698
27.2	主要特性	698
27.3	功能说明	699
27.3.1	RTC 框图	699
27.3.2	上电设定	699
27.3.3	系统低功耗模式切换	699
27.3.4	时钟误差补偿	699
27.3.5	入侵检测	700
27.3.6	时间戳功能	700
27.3.7	中断	700
27.4	操作示例	700
27.4.1	RTC 计数开始设定	700
27.4.2	读出计数寄存器	701
27.4.3	写入计数寄存器	701
27.4.4	闹钟设定	701
27.4.5	1Hz 输出	701
27.5	寄存器	703
27.5.1	寄存器总表	703
27.5.2	控制寄存器 0 (RTC_CR0)	704
27.5.3	控制寄存器 1 (RTC_CR1)	705
27.5.4	控制寄存器 2 (RTC_CR2)	706
27.5.5	控制寄存器 3 (RTC_CR3)	707
27.5.6	秒计数寄存器 (RTC_SEC)	708
27.5.7	分计数寄存器 (RTC_MIN)	708
27.5.8	时计数寄存器 (RTC_HOUR)	709
27.5.9	日计数寄存器 (RTC_DAY)	710
27.5.10	周计数寄存器 (RTC_WEEK)	711
27.5.11	月计数寄存器 (RTC_MON)	711
27.5.12	年计数寄存器 (RTC_YEAR)	712
27.5.13	分闹钟寄存器 (RTC_ALMMIN)	712
27.5.14	时闹钟寄存器 (RTC_ALMHOUR)	713
27.5.15	周闹钟寄存器 (RTC_ALMWEEK)	713
27.5.16	时钟误差补偿寄存器 (RTC_ERRCRH、RTC_ERRCRL)	714
27.5.17	入侵控制寄存器 0 (RTC_TPCR0)	716
27.5.18	入侵控制寄存器 1 (RTC_TPCR1)	717
27.5.19	入侵状态寄存器 (RTC_TPSR)	718
27.5.20	秒时间戳寄存器 (RTC_SECTP)	719
27.5.21	分时间戳寄存器 (RTC_MINTP)	719
27.5.22	时时间戳寄存器 (RTC_HOURTP)	720
27.5.23	日时间戳寄存器 (RTC_DAYTP)	720
27.5.24	月时间戳寄存器 (RTC_MONTP)	721
28	看门狗计数器 (WDT/SWDT)	722
28.1	简介	722

28.2 主要特性.....	722
28.3 功能说明.....	722
28.3.1 启动看门狗.....	722
28.3.2 硬件启动方式.....	722
28.3.3 软件启动方式.....	723
28.3.4 刷新动作.....	724
28.3.5 标志位.....	724
28.3.6 中断复位.....	724
28.3.7 计数下溢.....	725
28.3.8 刷新错误.....	725
28.4 寄存器.....	727
28.4.1 寄存器总表.....	727
28.4.2 控制寄存器 (SWDT_CR、WDT_CR)	728
28.4.3 状态寄存器 (SWDT_SR、WDT_SR)	729
28.4.4 刷新寄存器 (SWDT_RR、WDT_RR)	730
28.5 注意事项.....	730
29 通用同步异步收发器 (USART)	731
29.1 简介.....	731
29.2 主要特性.....	731
29.3 功能说明.....	733
29.3.1 系统框图.....	733
29.3.2 UART 功能.....	734
29.3.3 多处理器通信.....	743
29.3.4 UART-LIN.....	746
29.3.5 智能卡.....	748
29.3.6 时钟同步模式.....	751
29.3.7 数字滤波功能.....	754
29.3.8 中断.....	756
29.4 寄存器.....	757
29.4.1 寄存器总表.....	757
29.4.2 状态寄存器 (USART_SR)	758
29.4.3 发送数据寄存器 (USART_TDR)	762
29.4.4 接收数据寄存器 (USART_RDR)	762
29.4.5 波特率寄存器 (USART_BRR)	763
29.4.6 控制寄存器 1 (USART_CR1)	764
29.4.7 控制寄存器 2 (USART_CR2)	767
29.4.8 控制寄存器 3 (USART_CR3)	770
29.4.9 预分频寄存器 (USART_PR)	772
29.4.10 LIN 波特率测量计数寄存器 (USART_LBMC)	773
29.4.11 USART1 滤波控制寄存器 (USART1_NFC)	774
29.5 注意事项.....	775
29.5.1 UART 注意事项.....	775
29.5.2 时钟同步模式注意事项.....	775
29.5.3 其他注意事项.....	775
30 集成电路总线 (I2C)	776
30.1 简介.....	776
30.2 主要特性.....	776
30.3 功能说明.....	776
30.3.1 I2C 系统框图.....	776
30.3.2 I2C 协议.....	777
30.3.3 地址匹配.....	784

30.3.4	SMBus 动作.....	787
30.3.5	复位.....	787
30.3.6	可编程数字滤波.....	788
30.3.7	中断和事件.....	788
30.4	操作示例.....	789
30.4.1	I2C 初始化流程.....	789
30.5	寄存器.....	791
30.5.1	寄存器总表.....	791
30.5.2	I2C 控制寄存器 1 (I2C_CR1)	792
30.5.3	I2C 控制寄存器 2 (I2C_CR2)	794
30.5.4	I2C 控制寄存器 3 (I2C_CR3)	795
30.5.5	I2C 控制寄存器 4 (I2C_CR4)	796
30.5.6	I2C 从机地址寄存器 0 (I2C_SLR0)	797
30.5.7	I2C 从机地址寄存器 1 (I2C_SLR1)	799
30.5.8	I2C SCL 电平超时控制寄存器 (I2C_SLTR)	800
30.5.9	I2C 状态寄存器 (I2C_SR)	801
30.5.10	I2C 状态清零寄存器 (I2C_CLR)	806
30.5.11	I2C 数据发送寄存器 (I2C_DTR)	807
30.5.12	I2C 数据接收寄存器 (I2C_DRR)	808
30.5.13	I2C 时钟控制寄存器 (I2C_CCR)	809
30.5.14	I2C 滤波控制寄存器 (I2C_FLTR)	810
30.5.15	I2C FIFO 状态寄存器 (I2C_FSTR)	811
30.5.16	I2C 从机地址接收器寄存器 (I2C_SLVADRR)	812
31	串行外设接口 (SPI)	813
31.1	简介.....	813
31.2	主要特性.....	813
31.3	功能说明.....	814
31.3.1	系统框图.....	814
31.3.2	SPI 动作系统说明.....	815
31.3.3	数据通信说明.....	817
31.3.4	运行说明.....	823
31.3.5	奇偶校验位自诊断.....	828
31.3.6	错误检测.....	828
31.3.7	SPI 的初始化.....	831
31.3.8	中断和事件.....	832
31.4	寄存器.....	833
31.4.1	寄存器总表.....	833
31.4.2	SPI 数据寄存器 (SPI_DR)	834
31.4.3	SPI 控制寄存器 (SPI_CR)	835
31.4.4	SPI 通信配置寄存器 1 (SPI_CFG1)	837
31.4.5	SPI 状态寄存器 (SPI_SR)	839
31.4.6	SPI 通信配置寄存器 2 (SPI_CFG2)	840
32	四线式串行外设接口 (QSPI)	842
32.1	简介.....	842
32.2	主要特性.....	842
32.3	内存映射.....	843
32.3.1	内部总线空间.....	843
32.3.2	ROM 空间及总线的地址宽度.....	843
32.4	功能说明.....	845
32.4.1	基本框图.....	845
32.4.2	QSPI 总线.....	845

32.4.3	QSPI 总线的时序调整.....	847
32.4.4	用于 ROM 访问的 SPI 指令介绍.....	849
32.4.5	QSPI 总线周期的安排.....	857
32.4.6	XIP 控制.....	859
32.4.7	QSIO2 和 QSIO3 的管脚状态.....	860
32.4.8	直接通信模式.....	860
32.4.9	中断.....	861
32.5	寄存器.....	862
32.5.1	寄存器总表.....	862
32.5.2	QSPI 控制寄存器 (QSPI_CR)	863
32.5.3	QSPI 片选控制寄存器 (QSPI_CSCR)	864
32.5.4	QSPI 格式控制寄存器 (QSPI_FCR)	865
32.5.5	QSPI 状态寄存器 (QSPI_SR)	866
32.5.6	QSPI 直接通信指令寄存器 (QSPI_DCOM)	867
32.5.7	QSPI 指令代码寄存器 (QSPI_CCMD)	867
32.5.8	QSPI XIP 模式代码寄存器 (QSPI_XCMD)	868
32.5.9	QSPI 标志清除寄存器 (QSPI_CLR)	868
32.5.10	QSPI 外部扩展地址寄存器 (QSPI_EXAR)	868
32.6	注意事项.....	869
32.6.1	QSPI 寄存器的设置.....	869
32.6.2	模块停止位的设置.....	869
33	集成电路内置音频总线模块 (I2S)	870
33.1	简介.....	870
33.2	主要特性.....	870
33.3	功能说明.....	871
33.3.1	系统框图.....	871
33.3.2	管脚说明.....	871
33.3.3	通信方式.....	872
33.3.4	支持的音频协议.....	872
33.3.5	时钟发生器.....	876
33.3.6	主从模式.....	877
33.3.7	中断.....	879
33.4	寄存器.....	880
33.4.1	寄存器总表.....	880
33.4.2	I2S 控制寄存器 (I2S_CTRL)	881
33.4.3	I2S 状态寄存器 (I2S_SR)	883
33.4.4	I2S 错误状态寄存器 (I2S_ER)	884
33.4.5	I2S 配置寄存器 (I2S_CFGR)	885
33.4.6	I2S 发送缓冲 FIFO 数据寄存器 (I2S_TXBUF)	885
33.4.7	I2S 接收缓冲 FIFO 数据寄存器 (I2S_RXBUF)	886
33.4.8	I2S 分频寄存器 (I2S_PR)	886
33.5	注意事项.....	887
33.5.1	作为主机使用时的注意事项.....	887
33.5.2	作为从机使用时的注意事项.....	888
33.5.3	抗干扰支持功能.....	889
33.5.4	数据通道标识功能.....	889
33.5.5	四路同步控制功能.....	889
33.5.6	其他注意事项.....	889
34	USB2.0 高速模块 (USBHS)	890
34.1	简介.....	890
34.2	主要特性.....	890

34.2.1	通用特性.....	890
34.2.2	主机模式特性.....	890
34.2.3	设备模式特性.....	891
34.3	功能说明.....	892
34.3.1	系统框图.....	892
34.3.2	管脚说明.....	892
34.3.3	时钟以及工作模式.....	893
34.3.4	模式决定.....	893
34.3.5	主机功能.....	893
34.3.6	设备功能.....	896
34.3.7	SOF 脉冲管脚输出功能.....	898
34.3.8	功耗控制.....	899
34.3.9	动态更新 USBHS_HFIR 寄存器.....	899
34.3.10	数据 FIFO.....	899
34.3.11	主机 FIFO 架构.....	900
34.3.12	设备 FIFO 架构.....	901
34.3.13	FIFO RAM 分配.....	902
34.3.14	系统性能.....	902
34.3.15	中断和事件.....	903
34.4	编程模型.....	904
34.4.1	模块初始化.....	904
34.4.2	主机初始化.....	904
34.4.3	设备初始化.....	905
34.4.4	DMA 模式.....	905
34.4.5	主机编程模型.....	906
34.4.6	设备编程模型.....	907
34.5	操作模型.....	909
34.5.1	SETUP 和 OUT 数据传输.....	909
34.5.2	IN 数据传输.....	914
34.6	寄存器.....	920
34.6.1	寄存器总表.....	921
34.6.2	USB 系统控制寄存器.....	924
34.6.3	USBHS 全局寄存器.....	926
34.6.4	USBHS 主机模式寄存器.....	953
34.6.5	USBHS 设备模式寄存器.....	965
34.6.6	USBHS 时钟门控控制寄存器.....	994
35	USB2.0 全速模块 (USBFS)	995
35.1	简介.....	995
35.2	主要特性.....	995
35.2.1	通用特性.....	995
35.2.2	主机模式特性.....	995
35.2.3	设备模式特性.....	996
35.3	功能说明.....	997
35.3.1	系统框图.....	997
35.3.2	管脚说明.....	997
35.3.3	时钟以及工作模式.....	997
35.3.4	模式决定.....	998
35.3.5	主机功能.....	998
35.3.6	设备功能.....	1000
35.3.7	SOF 脉冲管脚输出功能.....	1003
35.3.8	功耗控制.....	1003
35.3.9	动态更新 USBFS_HFIR 寄存器.....	1004

35.3.10	数据 FIFO.....	1004
35.3.11	主机 FIFO 架构.....	1004
35.3.12	设备 FIFO 架构.....	1006
35.3.13	FIFO RAM 分配.....	1007
35.3.14	系统性能.....	1007
35.3.15	中断和事件.....	1008
35.4	编程模型.....	1009
35.4.1	模块初始化.....	1009
35.4.2	主机初始化.....	1009
35.4.3	设备初始化.....	1010
35.4.4	DMA 模式.....	1010
35.4.5	主机编程模型.....	1010
35.4.6	设备编程模型.....	1011
35.5	操作模型.....	1014
35.5.1	SETUP 和 OUT 数据传输.....	1014
35.5.2	IN 数据传输.....	1019
35.6	寄存器.....	1025
35.6.1	寄存器总表.....	1025
35.6.2	USB 系统控制寄存器.....	1028
35.6.3	USBFS 全局寄存器.....	1030
35.6.4	USBFS 主机模式寄存器.....	1056
35.6.5	USBFS 设备模式寄存器.....	1067
35.6.6	USBFS 时钟门控控制寄存器.....	1091
36	CAN FD 控制器 (CAN FD)	1092
36.1	简介.....	1092
36.2	主要特性.....	1092
36.3	功能说明.....	1093
36.3.1	系统框图.....	1093
36.3.2	管脚说明.....	1093
36.3.3	工作模式.....	1093
36.3.4	波特率设定.....	1094
36.3.5	发送处理.....	1096
36.3.6	接收处理.....	1098
36.3.7	错误处理.....	1099
36.3.8	节点关闭.....	1099
36.3.9	仲裁失败位置捕捉.....	1100
36.3.10	时间触发 TTCAN.....	1100
36.3.11	TDC 和 RDC.....	1101
36.3.12	向上兼容 CAN-FD 功能.....	1102
36.3.13	RAM ECC 功能.....	1102
36.3.14	软件复位功能.....	1102
36.3.15	中断.....	1105
36.4	寄存器.....	1106
36.4.1	寄存器总表.....	1106
36.4.2	CAN 接收 BUF 寄存器 (CAN_RBUF)	1108
36.4.3	CAN 发送 BUF 寄存器 (CAN_TBUF)	1110
36.4.4	CAN 配置和状态寄存器 (CAN_CFG_STAT)	1111
36.4.5	CAN 命令寄存器 (CAN_TCMD)	1112
36.4.6	CAN 发送控制寄存器 (CAN_TCTRL)	1114
36.4.7	CAN 接收控制寄存器 (CAN_RCTRL)	1115
36.4.8	CAN 接收和发送中断使能寄存器 (CAN_RTIE)	1116
36.4.9	CAN 接收和发送中断标志寄存器 (CAN_RTIF)	1117

36.4.10	CAN 错误中断使能和标志寄存器 (CAN_ERRINT)	1119
36.4.11	CAN Slow 位时序寄存器 (CAN_SBT)	1120
36.4.12	CAN Fast 位时序寄存器 (CAN_FBT)	1120
36.4.13	CAN 发送器延迟补偿寄存器 (CAN_TDC)	1121
36.4.14	CAN 错误和仲裁失败捕捉寄存器 (CAN_EALCAP)	1121
36.4.15	CAN 警告限定寄存器 (CAN_LIMIT)	1122
36.4.16	CAN 接收错误计数器寄存器 (CAN_RECNT)	1122
36.4.17	CAN 发送错误计数器寄存器 (CAN_TECNT)	1122
36.4.18	CAN 筛选器组控制寄存器 (CAN_ACFCTRL)	1123
36.4.19	CAN 筛选器组使能寄存器 (CAN_ACFEN)	1124
36.4.20	CAN 筛选器组 code 和 mask 寄存器 (CAN_ACF)	1125
36.4.21	TTCAN TB slot 指针寄存器 (CAN_TBSLOT)	1126
36.4.22	TTCAN 时间触发配置寄存器 (CAN_TTCFG)	1127
36.4.23	TTCAN 参考消息寄存器 (CAN_REF_MSG)	1128
36.4.24	TTCAN 触发配置寄存器 (CAN_TRG_CFG)	1129
36.4.25	TTCAN 触发时间寄存器 (CAN_TT_TRIG)	1130
36.4.26	TTCAN 触发看门时间寄存器 (CAN_TT_WTRIG)	1130
36.4.27	CAN RAM 校验控制寄存器 (CAN_RAM_CKCR)	1131
36.4.28	CAN RAM 校验状态寄存器 (CAN_RAM_CKSR)	1132
36.4.29	CAN RAM 错误注入使能寄存器 (CAN_RAM_EIEN)	1132
36.4.30	CAN RAM 错误注入位使能寄存器 0 (CAN_RAM_EIBIT0)	1133
36.4.31	CAN RAM 错误注入位使能寄存器 1 (CAN_RAM_EIBIT1)	1133
36.4.32	CAN RAM 校验错误地址寄存器 (CAN_RAM_ECCERRADDR)	1134
36.4.33	CAN 全局控制寄存器 (CAN_SYCTLREG)	1134
36.5	注意事项	1135
36.5.1	CAN 总线抗干扰措施	1135
36.5.2	CAN 控制器噪声制约	1135
37	控制器局域网 (MCAN)	1136
37.1	简介	1136
37.2	主要特性	1136
37.3	功能说明	1137
37.3.1	MCAN 系统框图	1137
37.3.2	引脚说明	1138
37.3.3	工作模式	1138
37.3.4	时间戳生成	1145
37.3.5	超时计数器	1145
37.3.6	接收处理	1146
37.3.7	发送处理	1151
37.3.8	FIFO 确认处理	1155
37.3.9	消息 RAM	1156
37.4	寄存器	1166
37.4.1	寄存器总表	1166
37.4.2	MCAN 字节序寄存器 (MCAN_ENDN)	1168
37.4.3	MCAN 数据位时间和预分频寄存器 (MCAN_DBTP)	1169
37.4.4	MCAN 测试寄存器 (MCAN_TEST)	1170
37.4.5	MCAN RAM 看门狗寄存器 (MCAN_RWD)	1171
37.4.6	MCAN CC 控制寄存器 (MCAN_CCCR)	1172
37.4.7	MCAN 标称位时间和预分频寄存器 (MCAN_NBTP)	1174
37.4.8	MCAN 时间戳计数器配置寄存器 (MCAN_TSCC)	1175
37.4.9	MCAN 时间戳计数器值寄存器 (MCAN_TSCV)	1176
37.4.10	MCAN 超时计数器配置寄存器 (MCAN_TOCC)	1177
37.4.11	MCAN 超时计数器值寄存器 (MCAN_TOCV)	1178

37.4.12	MCAN 错误计数器寄存器 (MCAN_ECR)	1179
37.4.13	MCAN 协议状态寄存器 (MCAN_PSR)	1180
37.4.14	MCAN 发送延迟补偿寄存器 (MCAN_TDCR)	1183
37.4.15	MCAN 中断寄存器 (MCAN_IR)	1184
37.4.16	MCAN 中断使能寄存器 (MCAN_IE)	1187
37.4.17	MCAN 中断线选择寄存器 (MCAN_ILS)	1189
37.4.18	MCAN 中断线使能寄存器 (MCAN_ILE)	1191
37.4.19	MCAN 全局过滤器配置寄存器 (MCAN_GFC)	1192
37.4.20	MCAN 标准 ID 过滤器配置寄存器 (MCAN_SIDFC)	1193
37.4.21	MCAN 扩展 ID 过滤器配置寄存器 (MCAN_XIDFC)	1193
37.4.22	MCAN 扩展 ID 与掩码寄存器 (MCAN_XIDAM)	1194
37.4.23	MCAN 高优先级消息状态寄存器 (MCAN_HPMS)	1194
37.4.24	MCAN 新数据 1 寄存器 (MCAN_NDAT1)	1195
37.4.25	MCAN 新数据 2 寄存器 (MCAN_NDAT2)	1195
37.4.26	MCAN 接收 FIFO0 配置寄存器 (MCAN_RXF0C)	1196
37.4.27	MCAN 接收 FIFO0 状态寄存器 (MCAN_RXF0S)	1197
37.4.28	MCAN 接收 FIFO0 确认寄存器 (MCAN_RXF0A)	1198
37.4.29	MCAN 接收缓冲区配置寄存器 (MCAN_RXBC)	1198
37.4.30	MCAN 接收 FIFO1 配置寄存器 (MCAN_RXF1C)	1199
37.4.31	MCAN 接收 FIFO1 状态寄存器 (MCAN_RXF1S)	1200
37.4.32	MCAN 接收 FIFO1 确认寄存器 (MCAN_RXF1A)	1201
37.4.33	MCAN 接收缓冲区和 FIFO 元素大小配置寄存器 (MCAN_RXESC)	1202
37.4.34	MCAN 发送缓冲区配置寄存器 (MCAN_TXBC)	1203
37.4.35	MCAN 发送 FIFO/队列状态寄存器 (MCAN_TXFQS)	1204
37.4.36	MCAN 发送缓冲区元素大小配置寄存器 (MCAN_TXESC)	1205
37.4.37	MCAN 发送缓冲区请求挂起寄存器 (MCAN_TXBRP)	1206
37.4.38	MCAN 发送缓冲区添加请求寄存器 (MCAN_TXBAR)	1207
37.4.39	MCAN 发送缓冲区取消请求寄存器 (MCAN_TXBCR)	1207
37.4.40	MCAN 发送缓冲区发送已发生寄存器 (MCAN_TXBTO)	1208
37.4.41	MCAN 发送缓冲区取消已完成寄存器 (MCAN_TXBCF)	1208
37.4.42	MCAN 发送缓冲区发送中断使能寄存器 (MCAN_TXBTIE)	1209
37.4.43	MCAN 发送缓冲区取消已完成中断使能寄存器 (MCAN_TXBCIE)	1209
37.4.44	MCAN 发送事件 FIFO 配置寄存器 (MCAN_TXEFC)	1210
37.4.45	MCAN 发送事件 FIFO 状态寄存器 (MCAN_TXEFS)	1211
37.4.46	MCAN 发送事件 FIFO 确认寄存器 (MCAN_TXEFA)	1212
37.4.47	MCAN 消息 RAM 校验控制寄存器 (MCANRAM_CKCR)	1213
37.4.48	MCAN 消息 RAM 校验状态寄存器 (MCANRAM_CKSR)	1214
37.4.49	MCAN 消息 RAM 错误注入使能寄存器 (MCANRAM_EIEN)	1214
37.4.50	MCAN 消息 RAM 错误注入位使能寄存器 0 (MCANRAM_EIBIT0)	1215
37.4.51	MCAN 消息 RAM 错误注入位使能寄存器 1 (MCANRAM_EIBIT1)	1215
37.4.52	MCAN 消息 RAM 校验错误地址寄存器 (MCANRAM_ECCERRADDR)	1216
37.5	注意事项	1216
37.5.1	CAN 总线抗干扰措施	1216
37.5.2	CAN 控制器噪声制约	1216
38	以太网 MAC 控制器 (ETHMAC)	1217
38.1	简介	1217
38.2	主要特性	1217
38.2.1	ETH_MAC 特性	1217
38.2.2	ETH_PTP 特性	1218
38.2.3	ETH_DMA 特性	1219
38.3	功能说明	1220
38.3.1	基本框图	1220

38.3.2	接口说明.....	1220
38.3.3	ETH_MAC 功能.....	1224
38.3.4	ETH_PTP 功能.....	1238
38.3.5	ETH_DMA 功能.....	1243
38.3.6	中断.....	1261
38.4	寄存器.....	1263
38.4.1	寄存器总表.....	1263
38.4.2	ETH_MAC 寄存器.....	1266
38.4.3	ETH_PTP 寄存器.....	1295
38.4.4	ETH_DMA 寄存器.....	1305
38.4.5	ETH_MMC 寄存器.....	1322
39	SDIO 控制器 (SDIOC)	1335
39.1	简介.....	1335
39.2	主要特性.....	1335
39.3	功能说明.....	1336
39.3.1	端口分配.....	1336
39.3.2	基本访问方式.....	1336
39.3.3	数据传输.....	1336
39.3.4	SD 时钟.....	1337
39.3.5	中断和 DMA 启动请求.....	1337
39.4	操作示例.....	1338
39.4.1	主机和设备初始化.....	1338
39.4.2	SD/MMC 单数据块 (single block) 读写.....	1339
39.4.3	SD/MMC 多数据块 (multi block) 读写.....	1339
39.4.4	传输终止 (abort), 挂起 (suspend) 和恢复 (resume)	1340
39.4.5	读等待 (read wait)	1341
39.4.6	唤醒 (wakeup)	1341
39.5	寄存器.....	1343
39.5.1	寄存器总表.....	1343
39.5.2	数据块长度寄存器 (SDIOC_BLKSIZE)	1345
39.5.3	数据块计数寄存器 (SDIOC_BLKCNT)	1345
39.5.4	参数寄存器 0 (SDIOC_ARG0)	1345
39.5.5	参数寄存器 1 (SDIOC_ARG1)	1346
39.5.6	传输模式寄存器 (SDIOC_TRANSMODE)	1346
39.5.7	命令寄存器 (SDIOC_CMD)	1347
39.5.8	应答寄存器 0 (SDIOC_RESP0)	1347
39.5.9	应答寄存器 1 (SDIOC_RESP1)	1348
39.5.10	应答寄存器 2 (SDIOC_RESP2)	1348
39.5.11	应答寄存器 3 (SDIOC_RESP3)	1348
39.5.12	应答寄存器 4 (SDIOC_RESP4)	1349
39.5.13	应答寄存器 5 (SDIOC_RESP5)	1349
39.5.14	应答寄存器 6 (SDIOC_RESP6)	1349
39.5.15	应答寄存器 7 (SDIOC_RESP7)	1350
39.5.16	数据缓冲寄存器 0 (SDIOC_BUF0)	1350
39.5.17	数据缓冲寄存器 1 (SDIOC_BUF1)	1350
39.5.18	主机状态寄存器 (SDIOC_PSTAT)	1351
39.5.19	主机控制寄存器 (SDIOC_HOSTCON)	1352
39.5.20	电源控制寄存器 (SDIOC_PWRCON)	1352
39.5.21	数据块间隙控制寄存器 (SDIOC_BLKGPCON)	1353
39.5.22	时钟控制寄存器 (SDIOC_CLKCON)	1354
39.5.23	超时控制寄存器 (SDIOC_TOUTCON)	1355
39.5.24	软件复位寄存器 (SDIOC_SFTRST)	1356

39.5.25	普通中断状态寄存器 (SDIOC_NORINTST)	1357
39.5.26	错误中断状态寄存器 (SDIOC_ERRINTST)	1358
39.5.27	普通中断状态使能寄存器 (SDIOC_NORINTSTEN)	1359
39.5.28	错误中断状态使能寄存器 (SDIOC_ERRINTSTEN)	1360
39.5.29	普通中断信号使能寄存器 (SDIOC_NORINTSGEN)	1361
39.5.30	错误中断信号使能寄存器 (SDIOC_ERRINTSGEN)	1362
39.5.31	自动命令错误状态寄存器 (SDIOC_ATCERRST)	1363
39.5.32	强制自动命令错误状态控制寄存器 (SDIOC_FEA)	1364
39.5.33	强制错误状态控制寄存器 (SDIOC_FEE)	1365
39.5.34	系统控制寄存器 (SDIOC_SYCTLREG)	1366
40	外部存储器控制器 (EXMC)	1367
40.1	简介	1367
40.2	主要特性	1367
40.3	功能说明	1368
40.3.1	控制器架构	1368
40.3.2	基本访问规范	1368
40.3.3	地址映射	1370
40.3.4	协议接口	1372
40.3.5	SMC-SRAM/PSRAM/NOR Flash 控制器	1375
40.3.6	DMC-SDRAM 控制器	1395
40.3.7	NFC-NAND Flash 控制器	1402
40.3.8	中断说明	1405
40.4	寄存器	1406
40.4.1	寄存器总表	1406
40.4.2	EXMC 公共寄存器	1408
40.4.3	SMC-SRAM/PSRAM/NOR Flash 寄存器	1409
40.4.4	DMC-SDRAM 寄存器	1418
40.4.5	NFC-NAND Flash 寄存器	1431
40.5	注意事项	1446
41	数字视频接口 (DVP)	1447
41.1	简介	1447
41.2	主要特性	1447
41.3	功能说明	1448
41.3.1	系统框图	1448
41.3.2	视频数据格式	1449
41.3.3	并口存储格式	1449
41.3.4	模式选择	1451
41.3.5	同步控制	1452
41.3.6	窗口裁剪	1453
41.3.7	FIFO 控制	1454
41.3.8	DMA 控制	1454
41.3.9	中断和事件	1454
41.4	寄存器	1456
41.4.1	寄存器总表	1456
41.4.2	控制寄存器 (DVP_CTR)	1457
41.4.3	数据寄存器 (DVP_DTR)	1458
41.4.4	状态寄存器 (DVP_STR)	1458
41.4.5	中断寄存器 (DVP_IER)	1459
41.4.6	DMA 数据传输寄存器 (DVP_DMR)	1459
41.4.7	软件同步数据寄存器 (DVP_SSYNDR)	1460
41.4.8	软件同步屏蔽寄存器 (DVP_SSYNMR)	1460

41.4.9 窗口裁剪偏移寄存器 (DVP_CPSFTR)	1461
41.4.10 窗口裁剪尺寸寄存器 (DVP_CPSZER)	1461
42 加密协处理模块 (CPM)	1462
42.1 简介.....	1462
42.2 主要特性.....	1462
42.2.1 公钥引擎 (PKE) 主要特性.....	1462
42.2.2 对称加密算法模块 (SKE) 主要特性.....	1462
42.3 功能说明.....	1463
42.3.1 安全散列算法 (HASH)	1463
42.3.2 真随机数发生器 (TRNG)	1468
42.3.3 公钥引擎 (PKE)	1469
42.3.4 对称加密算法模块 (SKE)	1472
42.4 寄存器.....	1477
42.4.1 寄存器总表.....	1477
42.4.2 HASH 寄存器.....	1479
42.4.3 TRNG 寄存器.....	1482
42.4.4 PKE 寄存器.....	1484
42.4.5 SKE 寄存器.....	1497
43 CRC 运算 (CRC)	1506
43.1 简介.....	1506
43.2 功能说明.....	1506
43.2.1 功能框图.....	1506
43.2.2 CRC 参数模型.....	1506
43.2.3 CRC16 编码模式.....	1507
43.2.4 CRC16 校验模式.....	1507
43.2.5 CRC32 编码模式.....	1507
43.2.6 CRC32 校验模式.....	1508
43.2.7 CRC64 编码模式.....	1508
43.2.8 CRC64 校验模式.....	1508
43.3 寄存器.....	1509
43.3.1 寄存器总表.....	1509
43.3.2 控制寄存器 (CRC_CR)	1510
43.3.3 结果寄存器 1 (CRC_RESULT1)	1511
43.3.4 结果寄存器 2 (CRC_RESULT2)	1511
43.3.5 数据寄存器 (CRC_DAT)	1512
44 数据计算单元 (DCU)	1513
44.1 简介.....	1513
44.2 主要特性.....	1513
44.3 功能说明.....	1513
44.3.1 加法模式.....	1513
44.3.2 减法模式.....	1513
44.3.3 硬件触发启动模式.....	1514
44.3.4 比较模式.....	1514
44.3.5 三角波输出模式.....	1515
44.3.6 递增锯齿波输出模式.....	1516
44.3.7 递减锯齿波输出模式.....	1517
44.3.8 中断和事件.....	1518
44.4 寄存器.....	1519
44.4.1 寄存器总表.....	1519
44.4.2 DCU 控制寄存器 (DCU_CTL)	1520
44.4.3 DCU 标志寄存器 (DCU_FLAG) (DCU1~DCU4)	1521

44.4.4	DCU 标志寄存器 (DCU_FLAG) (DCU5~DCU8)	1522
44.4.5	DCU 数据寄存器 (DCU_DATAx) (x=0~2)	1523
44.4.6	DCU 标志复位寄存器 (DCU_FLAGCLR) (DCU1~DCU4)	1526
44.4.7	DCU 标志复位寄存器 (DCU_FLAGCLR) (DCU5~DCU8)	1527
44.4.8	DCU 中断和事件寄存器 (DCU_INTEVTSEL) (DCU1~DCU4)	1528
44.4.9	DCU 中断和事件寄存器 (DCU_INTEVTSEL) (DCU5~DCU8)	1530
45	数学运算单元 (MAU)	1532
45.1	简介	1532
45.2	功能说明	1532
45.2.1	开方运算	1532
45.2.2	正弦运算	1532
45.2.3	中断及事件输出	1533
45.3	寄存器	1534
45.3.1	寄存器总表	1534
45.3.2	控制状态寄存器 (MAU_CSR)	1535
45.3.3	数据输入寄存器 0 (MAU_DTR0)	1535
45.3.4	结果输出寄存器 0 (MAU_RTR0)	1536
45.3.5	数据输入寄存器 1 (MAU_DTR1)	1536
45.3.6	结果输出寄存器 1 (MAU_RTR1)	1536
46	滤波数学加速器 (FMAC)	1537
46.1	简介	1537
46.2	主要特性	1537
46.3	功能说明	1537
46.3.1	基本框图	1537
46.3.2	操作流程	1538
46.3.3	模块使能	1538
46.3.4	系数归一化	1538
46.3.5	中断和事件	1538
46.4	寄存器	1539
46.4.1	寄存器总表	1539
46.4.2	模块使能寄存器 (FMAC_ENR)	1540
46.4.3	基本控制寄存器 (FMAC_CTR)	1541
46.4.4	中断控制寄存器 (FMAC_IER)	1542
46.4.5	数据输入寄存器 (FMAC_DTR)	1542
46.4.6	滤波系数寄存器 n (FMAC_CORn) (n=0~16)	1542
46.4.7	结果输出寄存器 0 (FMAC_RTR0)	1543
46.4.8	结果输出寄存器 1 (FMAC_RTR1)	1543
46.4.9	运算状态寄存器 (FMAC_STR)	1543
47	错误管理单元 (ERMU)	1544
47.1	简介	1544
47.2	主要特性	1544
47.3	功能说明	1545
47.3.1	功能框图	1545
47.3.2	错误输出通道	1546
47.3.3	等待定时器	1548
47.3.4	复位请求	1548
47.3.5	中断请求	1548
47.3.6	伪错误触发器	1548
47.3.7	中断	1548
47.3.8	错误源	1548
47.4	操作示例	1551

47.4.1	ERUM 错误管理示例.....	1551
47.5	寄存器.....	1552
47.5.1	寄存器总表.....	1552
47.5.2	错误输出 y 控制寄存器 (ERUM_EOyC) (y=0~3)	1553
47.5.3	错误输出 y 状态寄存器 (ERUM_EOyS) (y=0~3)	1554
47.5.4	错误输出 y 清除定时器计数寄存器 (ERUM_EOyCTCNT) (y=0~3)	1554
47.5.5	错误输出 y 清除定时器比较寄存器 (ERUM_EOyCTCMP) (y=0~3)	1555
47.5.6	错误输出 y 翻转定时器控制寄存器 (ERUM_EOyTTC) (y=0~3)	1555
47.5.7	错误输出 y 翻转定时器比较寄存器 (ERUM_EOyTTCMP) (y=0~3)	1556
47.5.8	错误输出 y 屏蔽 0 寄存器 (ERUM_EOyOM0) (y=0~3)	1556
47.5.9	错误输出 y 屏蔽 1 寄存器 (ERUM_EOyOM1) (y=0~3)	1557
47.5.10	等待定时器 z 控制寄存器 (ERUM_WTzC) (z=0~3)	1557
47.5.11	等待定时器 z 状态寄存器 (ERUM_WTzS) (z=0~3)	1558
47.5.12	等待定时器 z 计数寄存器 (ERUM_WTzCNT) (z=0~3)	1558
47.5.13	等待定时器 z 比较寄存器 (ERUM_WTzCMP) (z=0~3)	1559
47.5.14	等待定时器 z 启动使能寄存器 (ERUM_WTzSE) (z=0~3)	1559
47.5.15	错误源状态 j 寄存器 (ERUM_ESSj) (j=0~1)	1560
47.5.16	错误源状态清除 j 寄存器 (ERUM_ESSCj) (j=0~1)	1560
47.5.17	伪错误源触发 j 寄存器 (ERUM_PETj) (j=0~1)	1561
47.5.18	复位使能 j 寄存器 (ERUM_REj) (j=0~1)	1561
47.5.19	低优先级中断使能 j 寄存器 (ERUM_LPIEj) (j=0~1)	1562
47.5.20	高优先级中断使能 j 寄存器 (ERUM_HPIEj) (j=0~1)	1562
47.5.21	计数时钟预分频寄存器 (ERUM_CCPS)	1563
48	调试控制器 (DBG)	1564
48.1	简介.....	1564
48.2	功能说明.....	1564
48.2.1	功能框图.....	1564
48.2.2	参考文档.....	1565
48.2.3	SWJ-DP 调试端口 (SWD 和 JTAG)	1565
48.2.4	引脚排列和调试端口引脚.....	1566
48.2.5	SW 调试端口-SW 协议简介.....	1567
48.2.6	TPIU (跟踪端口接口单元)	1567
48.3	寄存器.....	1570
48.3.1	寄存器总表.....	1570
48.3.2	芯片专属标志寄存器 (CHIPID)	1571
48.3.3	DBG 状态寄存器 (MCUDBGSTAT)	1571
48.3.4	外设调试暂停寄存器 (MCUSTPCTL)	1572
48.3.5	调试组件配置寄存器 (MCUTRACECTL)	1574
48.3.6	外设调试暂停寄存器 2 (MCUSTPCTL2)	1575
A	术语表.....	1576

图目录

图 2-1 QSPI 地址空间分配.....	6
图 3-1 总线架构图.....	12
图 4-1 上电复位时序图.....	16
图 4-2 NRST 复位时序图.....	17
图 4-3 欠压复位时序图.....	17
图 4-4 可编程电压检测 1 复位时序图.....	18
图 4-5 可编程电压检测 2 复位时序图.....	18
图 4-6 看门狗和专用看门狗复位时序图.....	19
图 4-7 掉电唤醒复位时序图.....	19
图 4-8 软件复位时序图.....	19
图 4-9 MPU 错误复位时序图.....	20
图 4-10 RAMECC 复位时序图.....	20
图 4-11 FLASHECC 复位时序图.....	20
图 4-12 ERMU 复位时序图.....	20
图 4-13 时钟频率异常复位时序图.....	21
图 4-14 外部高速振荡异常复位时序图.....	21
图 4-15 Cortex-M4 Lockup 复位时序图.....	21
图 4-16 模块功能复位.....	22
图 5-1 时钟系统框图.....	40
图 5-2 时钟频率测量框图.....	41
图 5-3 外部高速振荡器连接示例.....	44
图 5-4 外部时钟输入的连接示例.....	44
图 5-5 外部高速振荡器故障检测示例.....	45
图 5-6 系统时钟选择 XTAL，检测到 XTAL 振荡故障示例.....	45
图 5-7 外部低速振荡器连接示例.....	46
图 5-8 时钟源切换流程图.....	49
图 5-9 时钟分频切换流程图.....	50
图 5-10 时钟频率测量时序图.....	51
图 6-1 电源构成图.....	94
图 6-2 电池备用域电源切换图.....	95
图 6-3 上电复位、掉电复位波形.....	96
图 6-4 欠压复位波形.....	96
图 6-5 PVD1 中断/复位框图.....	97
图 6-6 PVD2 中断/复位框图.....	97
图 6-7 电源监视 1 中断时序图.....	98
图 6-8 电源监视 1 复位时序图.....	99
图 6-9 电源监视 2 中断时序图.....	99
图 6-10 电源监视 2 复位时序图.....	100
图 6-11 内部电压采样示意图.....	100
图 6-12 PTWK _n 结构框图.....	108
图 8-1 Flash 地址分布.....	154
图 8-2 2MBytes 产品 Flash 地址分布.....	161
图 8-3 启动引导交换 1.....	162
图 8-4 启动引导交换 2.....	162
图 9-1 SRAM 39-bit 错误注入功能示意图.....	198
图 9-2 CACHE RAM 137-bit 错注入功能示意图.....	199
图 10-1 端口基本结构示意图.....	227
图 11-1 中断系统功能框图.....	264
图 11-2 中断请求/事件输入选择.....	266
图 11-3 数字滤波器工作示意图（以 NFSEL[7:0]=0x02 为例）.....	266

图 12-1 AOS 模块框图.....	282
图 12-2 PLU 框图.....	285
图 13-1 只允许部分空间访问示意图.....	308
图 13-2 只禁止部分空间访问示意图.....	308
图 14-1 KEYSKAN 系统框图.....	338
图 14-2 键盘扫描功能示意图.....	339
图 15-1 CTC 功能框图.....	344
图 15-2 CTC 校准示意图.....	346
图 16-1 DMA 结构图.....	354
图 16-2 连锁传输示意图.....	357
图 16-3 不连续地址传输示意图（以源地址为例）	358
图 16-4 不连续式重置示意图.....	360
图 16-5 应用例 1：存储器到存储器传输.....	362
图 16-6 应用例 2：存储器到外围电路的传输.....	363
图 17-1 CMP 功能框图.....	394
图 17-2 电压比较示意图.....	396
图 17-3 窗口比较示意图.....	397
图 17-4 空白窗口输出示意图.....	398
图 18-1 ADC 框图.....	409
图 18-2 通道映射示意图.....	411
图 18-3 内部模拟信号选择.....	411
图 18-4 序列 A 单次扫描模式.....	412
图 18-5 连续扫描.....	413
图 18-6 数据缓冲模式.....	414
图 18-7 双序列扫描模式（序列 A 从被中断通道重新启动）	415
图 18-8 双序列扫描模式（序列 A 从第一个通道重新启动）	415
图 18-9 模拟看门狗保护区（比较模式）	416
图 18-10 ADC 转换时间示意图.....	417
图 18-11 平均功能有效时的转换动作.....	418
图 18-12 ADC1、ADC2 的 PGA 与 SH 通道示意图.....	419
图 18-13 通道专用采样保持电路有效.....	420
图 18-14 单次并行触发模式（三个 ADC）	421
图 18-15 单次延迟触发模式（三个 ADC）	422
图 18-16 循环并行触发模式（三个 ADC）	423
图 18-17 循环延迟触发模式（两个 ADC）	423
图 18-18 循环延迟触发模式（三个 ADC）	424
图 18-19 ADC 中断和事件输出时序.....	425
图 19-1 DAC 转换通道框图.....	457
图 19-2 DAC 转换示意图.....	458
图 19-3 ADC 转换优先模式动作示意图.....	459
图 20-1 OTS 功能框图.....	466
图 21-1 Timer6 基本框图.....	474
图 21-2 锯齿波波形（递加计数）	475
图 21-3 三角波波形.....	475
图 21-4 比较输出动作.....	476
图 21-5 捕获输入动作.....	476
图 21-6 硬件刷新动作.....	478
图 21-7 捕获输入端口的滤波功能.....	478
图 21-8 软件同步动作.....	479
图 21-9 硬件同步动作.....	481
图 21-10 脉宽测量.....	481
图 21-11 周期测量.....	482

图 21-12 单缓存方式比较输出时序.....	483
图 21-13 双缓存方式捕获输入时序.....	484
图 21-14 锯齿波模式时计数缓存动作.....	485
图 21-15 三角波模式时计数缓存动作 1.....	486
图 21-16 三角波模式时计数缓存动作 2.....	486
图 21-17 单边对齐独立 PWM.....	487
图 21-18 双边对称独立 PWM.....	488
图 21-19 软件设定 GCMBR 互补 PWM 波输出.....	488
图 21-20 硬件设定 GCMBR 互补 PWM 波输出.....	489
图 21-21 双边非对称 PWM 输出.....	489
图 21-22 单边对齐独立 PWM.....	490
图 21-23 带死区时间双边对称互补 PWM.....	491
图 21-24 位置模式-基本计数.....	492
图 21-25 位置计数模式-相位差计数 (1 倍计数)	492
图 21-26 位置计数模式-相位差计数 (2 倍计数)	492
图 21-27 位置计数模式-相位差计数 (4 倍计数)	493
图 21-28 位置计数模式-方向计数.....	493
图 21-29 公转计数模式-Z 相计数.....	493
图 21-30 公转计数模式-位置溢出计数.....	494
图 21-31 公转计数模式-混合计数.....	494
图 21-32 公转计数模式-混合计数 Z 相屏蔽动作例 1.....	494
图 21-33 公转计数模式-混合计数 Z 相屏蔽动作例 2.....	495
图 21-34 周期间隔有效请求信号动作.....	495
图 21-35 正转 AB 输出时序.....	497
图 21-36 反转 AB 输出时序.....	497
图 21-37 正转切反转.....	498
图 21-38 正转 Z 输出 (PTZOSPR<PTZOEPR)	499
图 21-39 反转 Z 输出 (PTZOSPR>PTZOEPR)	499
图 21-40 正转切反转 Z 输出 (PTZOSPR=PTZOEPR)	500
图 21-41 正转 Z 相输出压缩.....	500
图 21-42 反转 Z 相输出压缩.....	501
图 21-43 正转切反转 Z 相输出压缩.....	501
图 21-44 AB 输出 (无余数补偿)	502
图 21-45 AB 输出 (余数补偿)	502
图 21-46 锯齿波模式时中断&事件输出例.....	505
图 22-1 Timer4 基本框图.....	564
图 22-2 Timer4 锯齿波波形.....	565
图 22-3 Timer4 三角波波形.....	565
图 22-4 Timer4 锯齿波模式计数动作.....	565
图 22-5 Timer4 三角波模式计数动作.....	566
图 22-6 锯齿波模式波形输出例.....	567
图 22-7 三角波模式波形输出例.....	568
图 22-8 缓存无效时修改锯齿波计数周期.....	569
图 22-9 缓存使能时修改锯齿波计数周期.....	569
图 22-10 缓存使能时修改三角波计数周期.....	569
图 22-11 OCCR 缓冲数据传输 (周期间隔响应链接禁止时)	570
图 22-12 OCCR 缓冲数据传输 (周期间隔响应链接使能)	571
图 22-13 输出比较缓冲数据传输 (OCMR 缓冲使能)	571
图 22-14 SCCR 缓冲传输操作 (周期间隔响应链接传输禁止时)	572
图 22-15 SCCR 缓冲传输操作 (周期间隔响应链接传输使能时)	572
图 22-16 锯齿波独立 PWM 输出例.....	573
图 22-17 三角波独立 PWM 输出例.....	574

图 22-18 锯齿波扩展 PWM 输出.....	574
图 22-19 软件实现互补 PWM 输出.....	575
图 22-20 死区定时器模式下的互补 PWM 输出.....	576
图 22-21 脉宽异常时的死区定时器模式下的波形输出.....	576
图 22-22 死区定时器滤波模式下的互补 PWM 输出.....	577
图 22-23 周期间隔响应时序图.....	578
图 22-24 专用事件输出信号周期间隔响应输出.....	579
图 22-25 PWM 周期输出监测示例.....	580
图 22-26 比较启动模式下专用事件输出 TMR4_<t>_ADSM 时序.....	580
图 22-27 延时启动模式下专用事件输出信号的输出时序.....	581
图 23-1 EMB 通道功能框图.....	611
图 23-2 EMB 结构框图.....	612
图 24-1 TimerA 基本框图.....	630
图 24-2 锯齿波波形（递加计数）.....	631
图 24-3 三角波波形.....	631
图 24-4 比较输出动作.....	631
图 24-5 捕获输入动作.....	633
图 24-6 时钟输入端口的滤波功能.....	634
图 24-7 软件同步动作.....	634
图 24-8 锯齿波模式时缓存动作.....	635
图 24-9 锯齿波模式时缓存与特俗缓存动作.....	636
图 24-10 32 位级联计数动作.....	636
图 24-11 单边对齐 PWM 输出例.....	637
图 24-12 双边对称 PWM 输出例.....	637
图 24-13 位置模式-基本计数.....	638
图 24-14 位置计数模式-相位差计数（1 倍计数）.....	638
图 24-15 位置计数模式-相位差计数（2 倍计数）.....	639
图 24-16 位置计数模式-相位差计数（4 倍计数）.....	639
图 24-17 位置计数模式-方向计数.....	639
图 24-18 公转计数模式-Z 相计数.....	640
图 24-19 公转计数模式-位置溢出计数.....	640
图 24-20 公转计数模式-混合计数.....	640
图 25-1 Timer2 基本框图.....	663
图 25-2 比较输出动作.....	665
图 25-3 硬件启动、清零动作.....	665
图 25-4 捕获输入动作.....	666
图 25-5 脉宽测量.....	666
图 25-6 周期测量.....	667
图 25-7 TRIG 输入端口的数字滤波.....	667
图 26-1 Timer0 基本框图.....	682
图 26-2 Timer0 计数时序图.....	683
图 27-1 RTC 基本框图.....	699
图 28-1 硬件启动例.....	723
图 28-2 软件启动例.....	723
图 28-3 各种刷新动作时序示例（动作确认，刷新要求信号的下降沿等）.....	724
图 28-4 计数器下溢动作例.....	725
图 28-5 计数器刷新动作例.....	726
图 29-1 USART 系统框图.....	733
图 29-2 UART 数据格式.....	734
图 29-3 UART 发送数据图例 1.....	736
图 29-4 UART 发送数据图例 2.....	736
图 29-5 UART 内部同步和采样时序.....	737

图 29-6 UART 接收数据图例 1.....	738
图 29-7 UART 接收数据图例 2.....	738
图 29-8 半双工工作时序.....	741
图 29-9 OVER8=1 DE 断言.....	741
图 29-10 OVER8=0 DE 断言.....	741
图 29-11 OVER8=1 DE 去断言.....	742
图 29-12 OVER8=0 DE 去断言.....	742
图 29-13 多处理器通信图例.....	743
图 29-14 多处理器模式数据格式.....	743
图 29-15 多处理器模式发送数据图例.....	744
图 29-16 多处理器模式接收数据图例 1.....	745
图 29-17 多处理器模式接收数据图例 2.....	745
图 29-18 LIN 总线数据行为.....	746
图 29-19 智能卡连接示意图.....	748
图 29-20 智能卡模式同步时序和采样时序图.....	748
图 29-21 智能卡模式数据格式.....	749
图 29-22 智能卡模式发送数据图例.....	750
图 29-23 智能卡模式接收数据图例.....	750
图 29-24 时钟同步模式数据格式.....	751
图 29-25 时钟同步模式发送数据图例 1 (LSB/无 CTS 功能)	752
图 29-26 时钟同步模式发送数据图例 2 (LSB/CTS 功能)	753
图 29-27 时钟同步模式接收数据图例 1 (LSB/无 RTS 功能)	753
图 29-28 时钟同步模式接收数据图例 2 (LSB/RTS 功能)	754
图 29-29 数字滤波功能图例.....	755
图 30-1 I2C 系统框图.....	776
图 30-2 I2C 总线的结构例.....	777
图 30-3 I2C 总线的时序图.....	778
图 30-4 I2C 总线的格式.....	778
图 30-5 7 位地址格式的主机发送数据时序图 (例)	779
图 30-6 7 位地址格式的主机接收数据的时序图 (例)	780
图 30-7 7 位地址格式的从机发送模式时序图 (例)	780
图 30-8 7 位地址格式从机接收模式时序图 (例)	781
图 30-9 SCL 同步时序.....	782
图 30-10 从机发送时序图 (1)	782
图 30-11 从机发送时序图 (2)	783
图 30-12 从机接收时序图.....	783
图 30-13 快速 ACK/NACK 时序图.....	784
图 30-14 选择 7 位地址格式时的时序.....	784
图 30-15 选择 10 位地址格式时的时序.....	785
图 30-16 广播地址匹配时序图.....	785
图 30-17 SMBus 主机地址匹配时序图.....	786
图 30-18 SMBus 报警响应地址匹配时序图.....	786
图 30-19 SMBus 默认地址时序图.....	786
图 30-20 数字滤波电路框图.....	788
图 31-1 SPI 系统框图.....	814
图 31-2 主机模式结构.....	816
图 31-3 三线式时钟同步运行.....	817
图 31-4 数据格式.....	818
图 31-5 MSB 先传, 奇偶校验无效.....	818
图 31-6 LSB 先传, 奇偶校验无效.....	818
图 31-7 MSB 先传, 奇偶校验有效.....	819
图 31-8 LSB 先传, 奇偶校验有效.....	819

图 31-9 数据传送格式图 (CPHA=0)	820
图 31-10 数据传送格式 (CPHA=1)	820
图 31-11 全双工同步串行通信	821
图 31-12 只进行发送通信	821
图 31-13 常规通信	822
图 31-14 连续通信	822
图 31-15 奇偶校验流程	828
图 31-16 过载错误处理	830
图 31-17 启用时钟自动停止功能时的动作示意图 (CPHA=1)	830
图 31-18 启用时钟自动停止功能时的动作示意图 (CPHA=0)	831
图 31-19 奇偶校验错误	831
图 32-1 默认区域设定及 AHB 总线空间内存映射关系图	843
图 32-2 QSPI-ROM 空间内存映像图	844
图 32-3 QSPI 的模块构成图	845
图 32-4 扩展式 SPI 协议动作示意图 1 (快速读模式)	846
图 32-5 扩展式 SPI 协议动作示意图 2 (四线式输入输出快速读模式)	846
图 32-6 二线式 SPI 协议动作示意图 (快速读模式)	846
图 32-7 四线式 SPI 协议动作示意图 (快速读模式)	847
图 32-8 串行接口的基本时序图	847
图 32-9 基准时钟选择 HCLK 三分频时输出时钟占空比补正示意图	848
图 32-10 QSSN 建立时间配置示意图	848
图 32-11 QSSN 保持时间配置示意图	849
图 32-12 数据接收延迟示意图	849
图 32-13 标准读总线周期示意图	850
图 32-14 快速读总线周期示意图	851
图 32-15 选择 XIP 模式的快速读总线周期示意图	851
图 32-16 二线式输出快速读总线周期示意图	852
图 32-17 选择 XIP 模式的二线式输出快速读总线周期示意图	852
图 32-18 二线式输入输出快速读总线周期示意图	853
图 32-19 选择 XIP 模式的二线式输入输出快速读总线周期示意图	853
图 32-20 四线式输出快速读总线周期示意图	854
图 32-21 选择 XIP 模式的四线式输出快速读总线周期示意图	854
图 32-22 四线式输入输出快速读总线周期示意图	855
图 32-23 选择 XIP 模式的四线式输入输出快速读总线周期示意图	855
图 32-24 进入 4-Bytes 模式指令总线周期示意图	856
图 32-25 退出 4-Bytes 模式指令总线周期示意图	856
图 32-26 写许可指令总线周期示意图	857
图 32-27 独立转换的单一闪存数据读取操作示意图	857
图 32-28 预读取功能有效时数据读取操作示意图	858
图 32-29 使用 QSPI 总线周期延长功能的数据读取操作示意图	858
图 32-30 XIP 模式控制示意图	859
图 33-1 I2S 系统框图	871
图 33-2 I2S Philips 协议波形 (16/32 位全精度)	872
图 33-3 I2S Philips 协议波形 (16 位数据封装在 32 位帧中)	873
图 33-4 I2S Philips 协议波形 (24 位数据封装在 32 位帧中)	873
图 33-5 I2S MSB 协议波形 (16/32 位全精度)	873
图 33-6 I2S MSB 协议波形 (16 位数据封装在 32 位帧中)	874
图 33-7 I2S MSB 协议波形 (24 位数据封装在 32 位帧中)	874
图 33-8 I2S LSB 协议波形 (16/32 位全精度)	874
图 33-9 I2S LSB 协议波形 (16 位数据封装在 32 位帧中)	874
图 33-10 I2S LSB 协议波形 (24 位数据封装在 32 位帧中)	875
图 33-11 I2S PCM 协议波形 (16/32 位全精度)	875

图 33-12 I2S PCM 协议波形 (16 位数据封装在 32 位帧中)	875
图 33-13 I2S PCM 协议波形 (24 位数据封装在 32 位帧中)	876
图 33-14 音频采样频率定义	876
图 33-15 时钟发生器架构	876
图 33-16 主机只接收暂时停止接收	887
图 33-17 PCM 短帧主机发送暂停后重新发送方式一	888
图 33-18 PCM 短帧主机发送暂停后重新发送方式二	888
图 34-1 USBHS 系统框图	892
图 34-2 USBHS 主机模式系统构建图	894
图 34-3 USBHS 设备模式系统构建图	896
图 34-4 USBHS 动态更新 USBHS_HFIR 寄存器示意图	899
图 34-5 主机模式下 FIFO 架构示意图	900
图 34-6 USBHS 设备模式下 FIFO 架构示意图	901
图 34-7 USBHS 控制状态寄存器存储器映射	920
图 35-1 USBFS 系统框图	997
图 35-2 USBFS 主机模式系统构建图	998
图 35-3 USBFS 设备模式系统构建图	1001
图 35-4 USBFS 动态更新 USBFS_HFIR 寄存器示意图	1004
图 35-5 主机模式下 FIFO 架构示意图	1004
图 35-6 USBFS 设备模式下 FIFO 架构示意图	1006
图 35-7 USBFS 控制状态寄存器存储器映射	1025
图 36-1 CAN FD 系统框图	1093
图 36-2 CANFD LBMI 和 LBME 示意图	1094
图 36-3 CAN 位时间定义图	1095
图 36-4 CANFD TBUF 寄存器写发送缓冲器和示意图	1097
图 36-5 CAN RBUF 寄存器读接收缓冲器示意图	1098
图 36-6 CAN ACF 寄存器访问筛选器组示意图	1099
图 36-7 TDC 功能示意图	1102
图 37-1 MCAN 系统框图	1137
图 37-2 收发器延迟测量	1142
图 37-3 总线监听模式下的引脚控制	1143
图 37-4 外部回环模式的引脚控制	1144
图 37-5 内部回环模式的引脚控制	1145
图 37-6 标准消息 ID 过滤器路径	1147
图 37-7 扩展消息 ID 过滤器路径	1148
图 37-8 接收 FIFO 状态	1149
图 37-9 接收 FIFO 溢出处理	1150
图 37-10 混合配置专用发送缓冲区和发送 FIFO 示例	1154
图 37-11 混合配置专用发送缓冲区和发送队列示例	1154
图 37-12 消息 RAM 配置	1156
图 38-1 以太网 MAC 控制器架构图	1220
图 38-2 MII 接口连接图	1221
图 38-3 RMII 接口连接图	1223
图 38-4 SMI 接口连接图	1223
图 38-5 MAC 帧结构	1225
图 38-6 MII/RMII 发送位序	1230
图 38-7 无冲突发送图	1230
图 38-8 有冲突发送图	1230
图 38-9 MII/RMII 模式下的发送图	1231
图 38-10 MII/RMII 接收位序	1233
图 38-11 无错误发送图	1233
图 38-12 有错误发送图	1233

图 38-13 假载波指示下的接收图.....	1233
图 38-14 远程唤醒帧过滤寄存器.....	1237
图 38-15 时钟同步图.....	1239
图 38-16 系统时间校准.....	1241
图 38-17 描述符结构.....	1243
图 38-18 TxDMA 动作流程（非 OSF 模式）.....	1246
图 38-19 TxDMA 动作流程（OSF 模式）.....	1247
图 38-20 RxDMA 动作流程.....	1254
图 38-21 ETHMAC 中断方案.....	1261
图 38-22 DMA 中断组成.....	1262
图 40-1 EXMC 架构图.....	1368
图 40-2 外部空间地址分配.....	1370
图 40-3 SMC 的地址空间划分.....	1371
图 40-4 DMC 的地址空间划分.....	1371
图 40-5 NFC 的地址空间划分.....	1372
图 40-6 SMC 状态图.....	1376
图 40-7 SMC 初始设定流程.....	1376
图 40-8 单次读动作基本时序（异步方式（RSYN=0b0）&8 位位宽（MW=0b00））.....	1378
图 40-9 单次读动作基本时序（异步方式（RSYN=0b0）&16 位位宽（MW=0b01））.....	1379
图 40-10 单次读动作基本时序（异步方式（RSYN=0b0）&32 位位宽（MW=0b10））.....	1380
图 40-11 单次读动作基本时序（同步方式（RSYN=0b1）&8 位位宽（MW=0b00））.....	1381
图 40-12 单次读动作基本时序（同步方式（RSYN=0b1）&16 位位宽（MW=0b01））.....	1382
图 40-13 单次读动作基本时序（同步方式（RSYN=0b1）&32 位位宽（MW=0b10））.....	1383
图 40-14 地址数据线复用单次读动作基本时序（8 位位宽（MW=0b00））.....	1384
图 40-15 地址数据线复用单次读动作基本时序（16 位位宽（MW=0b01））.....	1385
图 40-16 地址数据线复用单次读动作基本时序（32 位位宽（MW=0b10））.....	1386
图 40-17 单次写动作基本时序（异步方式（WSYN=0b0）&8 位位宽（MW=0b00）&BLSS=0b0）.....	1387
图 40-18 单次写动作基本时序（异步方式（WSYN=0b0）&16 位位宽（MW=0b01）&BLSS=0b1）.....	1388
图 40-19 单次写动作基本时序（异步方式（WSYN=0b0）&32 位位宽（MW=0b10）&BLSS=0b1）.....	1389
图 40-20 单次写动作基本时序（同步方式（WSYN=0b1）&8 位位宽（MW=0b00）&BLSS=0b1）.....	1390
图 40-21 单次写动作基本时序（同步方式（WSYN=0b1）&16 位位宽（MW=0b01）&BLSS=0b0）.....	1391
图 40-22 单次写动作基本时序（同步方式（WSYN=0b1）&32 位位宽（MW=0b10）&BLSS=0b0）.....	1392
图 40-23 地址数据线复用单次写动作基本时序（异步方式（WSYN=0b0）&16 位位宽（MW=0b01））.....	1393
图 40-24 地址数据线复用单次写动作基本时序（同步方式（WSYN=0b1）&16 位位宽（MW=0b01））.....	1394
图 40-25 行激活到读写操作时序.....	1395
图 40-26 命令方式预充电.....	1396
图 40-27 读写后自动预充电.....	1396
图 40-28 自刷新动作的进入和退出时序.....	1396
图 40-29 DMC 状态图.....	1398
图 40-30 DMC 初始设定流程.....	1398
图 40-31 单次读动作基本时序（16 位位宽（DMCMW=0b00））.....	1399
图 40-32 单次写动作基本时序（32 位位宽（DMCMW=0b01））.....	1400
图 40-33 突发读动作基本时序（32 位位宽（DMCMW=0b10）&BURST=0b010）.....	1401
图 40-34 突发写动作基本时序（32 位位宽（DMCMW=0b10）&BURST=0b011）.....	1401
图 40-35 DMC_CKE 端口输出控制时序.....	1402
图 40-36 NFC 时序控制图 1.....	1440
图 40-37 NFC 时序控制图 2.....	1441
图 40-38 外部反馈时钟访问.....	1446
图 41-1 DVP 基本框图.....	1448
图 41-2 DVP 信号波形示意图.....	1450
图 41-3 单帧模式数据采集动作.....	1451
图 41-4 连续模式数据采集动作.....	1452

图 41-5 帧采集频率控制..... 1452

图 41-6 DVP 窗口设定图.....1453

图 41-7 窗口裁剪数据采集图..... 1454

图 42-1 HASH 算法流程图.....1463

图 42-2 TRNG 系统框图.....1468

图 42-3 PKE 系统框图.....1470

图 42-4 SKE 系统框图.....1473

图 43-1 CRC 应用示意图..... 1506

图 44-1 硬件触发启动示意图..... 1514

图 44-2 三角波输出模式波形..... 1515

图 44-3 三角波输出模式..... 1516

图 44-4 递增锯齿波输出模式波形..... 1517

图 44-5 递增锯齿波输出模式..... 1517

图 44-6 递减锯齿波输出模式波形..... 1518

图 44-7 递减锯齿波输出模式..... 1518

图 45-1 正弦运算角度值示意图.....1533

图 46-1 FMAC 基本框图..... 1537

图 47-1 ERMU 框图.....1545

图 48-1 调试控制系统框图.....1564

图 48-2 调试控制系统.....1565

图 48-3 JTAG-DP 到 SW-DP 切换时序.....1566

图 48-4 TPIU 框图.....1568

表目录

表 2-1 存储器映射.....	2
表 2-2 目标地址配置主闪存示例 (MMF_REMCR0.EN=1 或 MMF_REMCR1.EN=1)	7
表 2-3 目标地址配置 SRAMH 示例 (MMF_REMCR0.EN=1 或 MMF_REMCR1.EN=1)	7
表 2-4 存储器映射 (Memory Mapping) 寄存器偏移地址.....	8
表 4-1 复位方式及产生条件.....	14
表 4-2 复位方式及复位标志.....	15
表 4-3 各模块复位条件.....	22
表 4-4 复位控制 (RMU) 寄存器偏移地址.....	25
表 5-1 时钟源主要特性.....	38
表 5-2 各个内部时钟的规格.....	41
表 5-3 时钟频率测量 (FCM) 寄存器偏移地址.....	53
表 5-4 时钟控制器 (CMU) 寄存器偏移地址.....	53
表 6-1 BOR 配置.....	96
表 6-2 PVD1/PVD2 特性.....	97
表 6-4 运行模式.....	101
表 6-5 低功耗模式.....	101
表 6-6 低功耗模式的运行条件及各模块在低功耗模式下的状态.....	102
表 6-7 运行模式说明.....	104
表 6-8 掉电模式子模式.....	106
表 6-10 RAM 模块与 RAM 掉电控制位.....	109
表 6-11 寄存器保护列表.....	111
表 6-12 电源控制 (PWC) 寄存器偏移地址.....	111
表 7-1 初始化配置 (ICG) 寄存器偏移地址.....	148
表 8-1 CPU 时钟频率和 Flash 读等待周期对照表.....	155
表 8-2 Flash 实际读周期数.....	156
表 8-3 OTP 地址分布表.....	160
表 8-4 嵌入式 Flash (EFM) 寄存器偏移地址.....	165
表 9-1 SRAM 读写访问的等待周期设定和 CPU 时钟频率的关系.....	197
表 9-2 SRAM 空间分配.....	197
表 9-3 内置 SRAM (SRAM) 寄存器偏移地址.....	200
表 10-1 32-bit 访问时 PORT 寄存器一览.....	230
表 10-2 通用 IO (GPIO) 寄存器偏移地址.....	230
表 10-3 Event Port 寄存器偏移地址.....	231
表 11-1 外部中断输入管脚.....	247
表 11-2 中断向量表.....	248
表 11-4 中断控制器 (INTC) 寄存器偏移地址.....	268
表 12-1 AOS 目标列表.....	283
表 12-2 自动运行系统 (AOS) 寄存器偏移地址.....	286
表 13-1 MPU 模块简介.....	306
表 13-2 存储保护单元 (MPU) 寄存器偏移地址.....	309
表 14-1 KEYSCAN 管脚说明.....	338
表 14-2 键盘扫描控制模块 (KEYSCAN) 寄存器偏移地址.....	340
表 15-1 HRC 目标频率为 20MHz 时的测量误差.....	345
表 15-2 HRC 目标频率为 16MHz 时的测量误差.....	345
表 15-3 内部时钟校准器 (CTC) 寄存器偏移地址.....	348
表 16-1 通道重置说明.....	359
表 16-2 DMA 控制器 (DMA) 寄存器偏移地址.....	365
表 17-1 CMP 引脚一览表.....	395
表 17-2 CMP 正端输入一览表.....	395
表 17-3 CMP 负端输入一览表.....	396

表 17-4 电压比较器 (CMP) 寄存器偏移地址.....	399
表 17-17 空白窗口 PWM 一览表.....	405
表 18-1 各 ADC 单元规格.....	409
表 18-2 序列 A 和 B 的各种竞争.....	415
表 18-3 ADC 转换时间.....	417
表 18-4 模数转换模块 (ADC) 寄存器偏移地址.....	426
表 18-30 不同的 ADC 单元 CHx 映射关系表.....	441
表 19-1 DAC 引脚.....	457
表 19-2 数模转换器 (DAC) 寄存器偏移地址.....	460
表 19-11 DAC 转换和模拟输出控制.....	463
表 20-1 OTS 预置温度数据.....	467
表 20-2 E _{hrc} 使用及设定方法.....	467
表 20-3 OTS 预置数据寄存器偏移地址.....	468
表 20-4 温度传感器 (OTS) 寄存器偏移地址.....	468
表 21-1 Timer6 的基本功能及特性.....	473
表 21-2 Timer6 端口列表.....	474
表 21-3 不同模式下的功能对比表.....	503
表 21-4 高级控制定时器 (Timer6) 寄存器偏移地址.....	511
表 21-5 Timer6 公共寄存器偏移地址.....	513
表 21-90 计数器 (CNTNR) 控制优先级.....	561
表 21-91 PWMA 端口输出控制优先级.....	561
表 21-92 PWMB 端口输出控制优先级.....	562
表 22-1 Timer4 的基本功能及特性.....	563
表 22-2 Timer4 端口列表.....	564
表 22-3 通用比较器输出的内部信号.....	566
表 22-4 通用比较器输出的内部信号和端口输出控制的对应关系.....	573
表 22-5 通用控制定时器 (Timer4) 寄存器偏移地址.....	582
表 22-21 OCCR*h 计数匹配一览表.....	593
表 22-24 OCCR*h/l 计数匹配一览表.....	595
表 23-1 EMB 端口分配.....	613
表 23-2 EMB Group 对照表.....	614
表 23-3 紧急刹车模块 (EMB) 寄存器偏移地址.....	616
表 24-1 TimerA 的基本功能及特性.....	629
表 24-2 TimerA 端口列表.....	630
表 24-3 内部触发事件 TRGSEL 选择关系对应表.....	632
表 24-4 通用定时器 (TimerA) 寄存器偏移地址.....	642
表 25-1 Timer2 的基本功能及特性.....	662
表 25-2 Timer2 端口列表.....	663
表 25-3 通用定时器 (Timer2) 寄存器偏移地址.....	669
表 26-1 计数器与 PWM 同步启动状态.....	684
表 26-2 通用定时器 (Timer0) 寄存器偏移地址.....	685
表 26-3 TMR_SYNENR 寄存器偏移地址.....	685
表 27-1 RTC 主要特性.....	698
表 27-2 实时时钟 (RTC) 寄存器偏移地址.....	703
表 28-1 看门狗计数器的基本特性.....	722
表 28-2 看门狗计数器 (WDT) 寄存器偏移地址.....	727
表 28-3 看门狗计数器 (SWDT) 寄存器偏移地址.....	727
表 29-2 USART 管脚说明.....	733
表 29-3 UART 接收器的容差.....	737
表 29-5 UART 中断/事件表.....	742
表 29-6 多处理器模式中断/事件表.....	745
表 29-7 LIN 中断/事件表.....	747

表 29-8 智能卡模式中断/事件表.....	750
表 29-9 时钟同步模式中断/事件表.....	754
表 29-10 USART 整体中断一览表.....	756
表 29-11 USART1 滤波控制寄存器偏移地址.....	757
表 29-12 通用同步异步收发器 (USART) 寄存器偏移地址.....	757
表 29-21 波特率计算公式 (小数波特率无效 FBME=0)	763
表 29-22 波特率计算公式 (小数波特率有效 FBME=1)	763
表 30-1 输入/输出引脚.....	777
表 30-2 中断一览表.....	788
表 30-3 事件信号输出一览表.....	789
表 30-4 集成电路总线 (I2C) 寄存器偏移地址.....	791
表 31-1 SPI 主要特性.....	813
表 31-2 管脚说明.....	814
表 31-3 主机模式时 SPI 管脚状态说明.....	815
表 31-4 从机模式时 SPI 管脚状态说明.....	815
表 31-5 部分设定值的位速.....	817
表 31-6 SPI 模式和寄存器设定关系.....	823
表 31-7 错误检测对应表.....	828
表 31-8 SPI 中断源和触发源说明.....	832
表 31-9 串行外设接口 (SPI) 寄存器偏移地址.....	833
表 32-1 QSPI 主要规格.....	842
表 32-2 QSPI 管脚.....	845
表 32-3 参考指令一览表.....	849
表 32-4 QSI02 和 QSI03 的管脚状态.....	860
表 32-5 四线式串行外设接口 (QSPI) 寄存器偏移地址.....	862
表 33-1 I2S 主要特性.....	870
表 33-2 I2S 管脚说明.....	871
表 33-3 音频频率精度 (针对 VCO 输入频率=1MHz)	877
表 33-4 I2S 中断请求.....	879
表 33-5 集成电路内置音频总线模块 (I2S) 寄存器偏移地址.....	880
表 34-1 USBHS 管脚说明.....	892
表 34-2 USBHS_GLB 中断事件表.....	903
表 34-3 USB 系统控制寄存器偏移地址.....	921
表 34-4 USB2.0 高速模块 USBHS 寄存器偏移地址.....	921
表 35-1 USBFS 管脚说明.....	997
表 35-2 USBFS_GLB 中断事件表.....	1008
表 35-3 USB 系统控制寄存器偏移地址.....	1026
表 35-4 USB2.0 全速模块 USBFS 寄存器偏移地址.....	1026
表 36-1 CAN 管脚说明.....	1093
表 36-2 CAN 位时间设定规则.....	1095
表 36-3 20MHz 通信时钟时波特率设定建议.....	1096
表 36-4 40MHz 通信时钟时波特率设定建议.....	1096
表 36-5 80MHz 通信时钟时波特率设定建议.....	1096
表 36-6 CAN RAM 地址映射.....	1102
表 36-7 软件复位范围表.....	1103
表 36-8 CAN 中断表.....	1105
表 36-9 CAN 全局控制寄存器偏移地址.....	1106
表 36-10 CAN FD 控制器 (CAN FD) 寄存器偏移地址.....	1106
表 36-11 标准格式 CAN 接收邮箱格式.....	1108
表 36-12 扩展格式 CAN 接收邮箱格式.....	1108
表 36-14 DLC 控制位.....	1109
表 36-15 标准格式 CAN 发送邮箱格式.....	1110

表 36-16 扩展格式 CAN 发送邮箱格式.....	1110
表 37-1 MCAN 引脚说明.....	1138
表 37-2 MCAN 配置寄存器列表.....	1138
表 37-3 MCAN 中的 DLC 编码.....	1140
表 37-4 接收缓冲器和 FIFO 元素大小.....	1149
表 37-5 接收缓冲区过滤器配置示例.....	1151
表 37-6 调试消息过滤器配置示例.....	1151
表 37-7 发送帧的可能配置.....	1152
表 37-8 发送缓冲区、FIFO（队列）元素大小.....	1153
表 37-9 接收缓冲区和 FIFO 元素.....	1157
表 37-10 接收缓冲区和 FIFO 元素说明.....	1157
表 37-11 发送缓冲区元素.....	1159
表 37-12 发送缓冲区元素说明.....	1159
表 37-13 发送事件 FIFO 元素.....	1161
表 37-14 发送事件 FIFO 元素说明.....	1162
表 37-15 标准消息 ID 过滤器元素.....	1163
表 37-16 标准消息 ID 过滤器元素说明.....	1163
表 37-17 扩展消息 ID 过滤器元素.....	1164
表 37-18 扩展消息 ID 过滤器元素说明.....	1164
表 37-19 控制器局域网（MCAN）寄存器偏移地址.....	1166
表 37-20 MCANRAM ECC 寄存器偏移地址.....	1167
表 38-1 MII 接口信号说明.....	1221
表 38-2 发送数据时信号状态说明.....	1222
表 38-3 接收数据时信号状态说明.....	1222
表 38-4 SMI 接口信号说明.....	1223
表 38-5 SMI 帧格式.....	1223
表 38-6 ETHMAC 端口功能分配.....	1224
表 38-7 时间戳快照目标报文.....	1240
表 38-8 常规型 Tx 描述符.....	1248
表 38-13 增强型 Tx 描述符.....	1252
表 38-16 常规型 Rx 描述符.....	1256
表 38-21 增强型 Rx 描述符.....	1259
表 38-25 ETHMAC 公共寄存器偏移地址.....	1263
表 38-26 以太网 MAC 控制器（ETHMAC）寄存器偏移地址.....	1263
表 39-1 端口分配表.....	1336
表 39-2 SDIOC 系统控制寄存器偏移地址.....	1343
表 39-3 SDIO 控制器（SDIOC）寄存器偏移地址.....	1343
表 40-1 AHB 访问宽度和存储器位宽对应访问方式表.....	1369
表 40-2 SMC 协议接口.....	1372
表 40-3 DMC 协议接口.....	1373
表 40-4 NFC 协议接口.....	1373
表 40-5 EXMC 端口功能分配 1.....	1374
表 40-6 EXMC 端口功能分配 2.....	1374
表 40-7 单次读动作基本设定例.....	1377
表 40-8 地址数据线复用单次读动作基本设定例.....	1384
表 40-9 单次写动作基本设定例.....	1387
表 40-10 地址数据线复用异步方式单次写动作基本设定例.....	1393
表 40-11 地址数据线复用同步方式单次写动作基本设定例.....	1394
表 40-12 DMC 的命令真值表.....	1396
表 40-13 ONFI 访问命令.....	1403
表 40-14 EXMC 公共寄存器偏移地址.....	1406
表 40-15 外部存储器控制器（EXMC）SMC 寄存器偏移地址.....	1406

表 40-16 外部存储器控制器 (EXMC) DMC 寄存器偏移地址.....	1406
表 40-17 外部存储器控制器 (EXMC) NFC 寄存器偏移地址.....	1407
表 40-34 CS0 设定和访问地址对应关系表.....	1413
表 40-55 CS0 设定和访问地址对应关系表.....	1422
表 40-94 命令参数列表.....	1433
表 40-99 索引寄存器值与 MEM 地址对应表.....	1435
表 40-116 4bitECC 模式时的 ECCR 寄存器列表.....	1442
表 40-117 4bitECC 模式时每段 ECCR 寄存器.....	1443
表 41-1 DVP 基本功能及特性.....	1447
表 41-2 DVP 端口列表.....	1448
表 41-3 单色格式视频数据存储方式.....	1449
表 41-4 YCbCr 格式视频数据存储方式.....	1449
表 41-5 RGB565 格式视频数据存储方式.....	1449
表 41-6 8-bit 数据 DVP 存储方式.....	1450
表 41-7 10-bit 数据 DVP 存储方式.....	1450
表 41-8 12-bit 数据 DVP 存储方式.....	1450
表 41-9 14-bit 数据 DVP 存储方式.....	1451
表 41-10 帧传送中断.....	1454
表 41-11 同步码错误序列.....	1455
表 41-12 数字视频接口 (DVP) 寄存器偏移地址.....	1456
表 42-2 运算速度.....	1469
表 42-3 RAM 操作数分布.....	1471
表 42-4 编程入口地址.....	1472
表 42-5 AES 硬件运算周期个数 (分组数量:128-bit)	1473
表 42-6 SM4 硬件运算周期个数 (分组数量:128-bit)	1473
表 42-7 DES 硬件运算周期个数 (分组数量:64-bit)	1473
表 42-12 128 位 AES ECB 模式操作寄存器示例.....	1476
表 42-13 安全散列算法 (HASH) 寄存器偏移地址.....	1477
表 42-14 真随机数发生器 (TRNG) 寄存器偏移地址.....	1477
表 42-15 公钥引擎 (PKE) 寄存器偏移地址.....	1477
表 42-16 对称加密算法模块 (SKE) 寄存器偏移地址.....	1478
表 43-1 CRC 参数模型.....	1506
表 43-2 CRC 运算 (CRC) 寄存器偏移地址.....	1509
表 44-1 数据计算单元 (DCU) 寄存器偏移地址.....	1519
表 45-1 数学运算单元 (MAU) 寄存器偏移地址.....	1534
表 46-1 滤波数学加速器 (FMAC) 寄存器偏移地址.....	1539
表 47-1 ERMU 参数定义表.....	1545
表 47-2 ERMU 输入/输出端口.....	1545
表 47-3 错误输出操作.....	1546
表 47-4 ERMU_EOUT0M 输出情况说明.....	1546
表 47-5 ERMU_EOUTyM (y=1~3)和 ERMU_EOUTyC (y=0~3)输出端口说明.....	1547
表 47-7 ERMU 错误源列表.....	1549
表 47-8 错误管理单元 (ERMU) 寄存器偏移地址.....	1552
表 48-1 SWJ 调试端口引脚.....	1566
表 48-2 灵活的 SWJ-DP 引脚分配.....	1566
表 48-6 调试控制器 (DBG) 寄存器偏移地址.....	1570

1 产品简介

HC32A4A8 系列是基于 32 位 ARM® Cortex®-M4 微控制器，最高工作频率 240MHz 的高性能 MCU。Cortex-M4 内核集成了浮点运算单元（FPU）和 DSP，实现单精度浮点算术运算，支持所有 ARM 单精度数据处理指令和数据类型，支持完整 DSP 指令集。内核集成了 MPU 单元，同时叠加 DMAC 专用 MPU 单元，保障系统运行的安全性。

HC32A4A8 系列集成了高速片上存储器，包括最大 2MB 的 Flash，最大 512KB 的 SRAM，Flash 和 SRAM 均支持 ECC 功能。集成了 Flash 访问加速单元，实现 CPU 在 Flash 上的单周期程序执行。轮询式总线矩阵支持多个总线主机同时访问存储器和外设，提高运行性能。总线主机包括 CPU，DMA，USB 专用 DMA，ETHMAC 专用 DMA。除总线矩阵外，支持外设间数据传递，基本算术运算和事件相互触发，可以显著降低 CPU 的事务处理负荷。

HC32A4A8 系列集成了丰富的外设功能。包括 3 个独立的 12-bit 2.5Msps ADC，4 个增益可调 PGA，4 个 12 位 DAC，4 个高速电压比较器（CMP），8 个多功能 PWM Timer（Timer6），支持 16 路互补 PWM 输出，3 个电机 PWM Timer（Timer4）支持 18 路互补 PWM 输出，12 个 16bit 通用 Timer（TimerA）支持 6 路 3 相正交编码输入及 48 路 Duty 独立可设 PWM 输出，22 个串行通信接口（I2C/USART/SPI），1 个 QSPI 接口，4 路 CAN，4 个 I2S 支持音频 PLL，2 个 SDIO，1 个 ETHMAC，内置 USBFS PHY 的 USBFS Controller 和 USBHS Controller，1 个外部扩展总线控制器，包括 NFC 控制器，SMC 控制器和 DMC 控制器，1 个数字视频接口 DVP，1 个数学运算单元（MAU）和 4 个滤波数学加速器（FMAC），1 个对称加密算法单元（SKE），1 个非对称加密算法单元（PKE），1 个错误管理单元（ERUM），1 个硬件安全模块单元（HSM Light-Mid）。

HC32A4A8 系列支持宽电压范围（1.8-3.63V），宽温度范围（-40-125℃）和各种低功耗模式。支持低功耗模式的快速唤醒，STOP 模式唤醒最快至 2μs，Power Down 模式唤醒最快至 25μs。

典型应用

HC32A4A8 系列提供 100pin、144pin、176pin 的 LQFP 封装，适用于汽车电子领域，例如 T-Box、BMS 控制器、网关、车灯控制等应用。

关于本手册

本手册主要介绍芯片的功能、操作事项和使用方法。关于芯片的规格，请参阅对应的《数据手册》。

2 存储器映射 (Memory Mapping)

2.1 存储器映射

该 MCU 支持 4GB 的线性地址空间，地址从 0x00000000 到 0xFFFFFFFF，其中包含程序和数据。详细存储器映射请参阅下表。

表 2-1 存储器映射

存储器分类 ⁽²⁾		开始地址 ⁽¹⁾	结束地址 ⁽¹⁾	空间大小	模块 ⁽³⁾	保护 ⁽⁴⁾	说明
系统	-	0xE0100000	0xFFFFFFFF	511MB	Reserved	-	自定义空间
	私有外设 外部总线	0xE00FF000	0xE00FFFFF	4KB	ROMTABLE	-	-
		0xE0042400	0xE00FEFFF	755KB	BLANK	-	-
		0xE0042000	0xE00423FF	1KB	DBGC	-	调试控制寄存器区域
		0xE0041000	0xE0041FFF	4KB	ETM	-	-
		0xE0040000	0xE0040FFF	4KB	TPIU	-	-
	私有外设 内部总线	0xE000F000	0xE003FFFF	196KB	BLANK	-	-
		0xE000E000	0xE000EFFF	4KB	SCS	-	系统控制空间 NVIC/ MPU 等
		0xE0003000	0xE000DFFF	44KB	BLANK	-	-
		0xE0002000	0xE0002FFF	4KB	FPB	-	-
		0xE0001000	0xE0001FFF	4KB	DWT	-	-
		0xE0000000	0xE0000FFF	4KB	ITM	-	-
外部设备	-	0xA0000000	0xDFFFFFFF	1024MB	Reserved	-	-
外部存储	AHB5 时钟: HCLK	0x98000000	0x9FFFFFFF	128MB	QSPI	-	-
		0x88200000	0x97FFFFFFF	254MB	Reserved	-	-
	AHB5 时钟: BCLK	0x88100000	0x881FFFFF	1MB	NFC	-	-
		0x88000800	0x880FFFFF	1022KB	BLANK	-	-
		0x88000400	0x880007FF	1KB	DMCR	-	-
		0x88000000	0x880003FF	1KB	SMCR	-	-
		0x80000000	0x87FFFFFFF	128MB	DMC	-	-
		0x60000000	0x7FFFFFFF	512MB	SMC	-	-
外设	-	0x44000000	0x5FFFFFFF	448MB	Reserved	-	-
		0x42000000	0x43FFFFFFF	32MB	PeriBitBand	-	-
		0x40100000	0x41FFFFFFF	31MB	Reserved	-	-
	AHB4 时钟: PCLK1	0x400C0000	0x400FFFFF	256KB	USBHS	-	-
		0x40080000	0x400BFFFF	256KB	USBFS	-	-
	AHB2 时钟: PCLK1	0x40078800	0x4007FFFF	30KB	BLANK	-	-
		0x40078400	0x400787FF	1KB	SDIOC_2	-	-
		0x40078000	0x400783FF	1KB	CAN_2	-	-
		0x40070400	0x40077FFF	31KB	BLANK	-	-
	AHB3 时钟: PCLK1	0x40070000	0x400703FF	1KB	SDIOC_1	-	-
		0x40060000	0x4006FFFF	64KB	ETHMAC	-	-
		0x40059000	0x4005FFFF	28KB	BLANK	-	-
	AHB1 时钟: HCLK	0x40058C00	0x40058FFF	1KB	FMAC_4	-	-
		0x40058800	0x40058BFF	1KB	FMAC_3	-	-
		0x40058400	0x400587FF	1KB	FMAC_2	-	-
		0x40058000	0x400583FF	1KB	FMAC_1	-	-

存储器分类 ⁽²⁾		开始地址 ⁽¹⁾	结束地址 ⁽¹⁾	空间大小	模块 ⁽³⁾	保护 ⁽⁴⁾	说明
外设	AHB1 时钟: HCLK	0x40057C00	0x40057FFF	1KB	DCU_8	-	-
		0x40057800	0x40057BFF	1KB	DCU_7	-	-
		0x40057400	0x400577FF	1KB	DCU_6	-	-
		0x40057000	0x400573FF	1KB	DCU_5	-	-
		0x40056C00	0x40056FFF	1KB	DCU_4	-	-
		0x40056800	0x40056BFF	1KB	DCU_3	-	-
		0x40056400	0x400567FF	1KB	DCU_2	-	-
		0x40056000	0x400563FF	1KB	DCU_1	-	-
		0x40055800	0x40055BFF	1KB	BLANK	-	-
		0x40055800	0x40055BFF	1KB	DVP	-	-
		0x40055400	0x400557FF	1KB	PERIC	-	周边模块控制寄存器
		0x40055000	0x400553FF	1KB	MAU	-	-
		0x40054400	0x40054FFF	3KB	BLANK	-	-
		0x40054000	0x400543FF	1KB	CMU	带保护	-
		0x40053800	0x40053FFF	2KB	GPIO	-	-
		0x40053400	0x400537FF	1KB	DMA_2	-	-
		0x40053000	0x400533FF	1KB	DMA_1	-	-
		0x40052000	0x40052FFF	4KB	BLANK	-	-
		0x40051000	0x40051FFF	4KB	INTC	带保护	-
		0x40050C00	0x40050FFF	1KB	KEYSCAN	-	-
		0x40050800	0x40050BFF	1KB	RAMIF	带保护	-
		0x40050400	0x400507FF	1KB	BLANK	-	-
		0x40050000	0x400503FF	1KB	MPU	带保护	-
	APB4 时钟: PCLK3	0x4004F800	0x4004FFFF	2KB	BLANK	-	-
		0x4004F400	0x4004F7FF	1KB	I2C_6	-	-
		0x4004F000	0x4004F3FF	1KB	I2C_5	-	-
		0x4004EC00	0x4004EFFF	1KB	I2C_4	-	-
		0x4004E800	0x4004EBFF	1KB	I2C_3	-	-
		0x4004E400	0x4004E7FF	1KB	I2C_2	-	-
		0x4004E000	0x4004E3FF	1KB	I2C_1	-	-
		0x4004D000	0x4004DFFF	1KB	BLANK	-	-
		0x4004CC00	0x4004CFFF	1KB	PWC, CMU	带保护	-
		0x4004C800	0x4004CBFF	1KB	BLANK	-	-
		0x4004C400	0x4004C7FF	1KB	VBAT	带保护	-
		0x4004C000	0x4004C3FF	1KB	RTC	带保护	-
		0x4004AC00	0x4004BFFF	5KB	BLANK	-	-
		0x4004A800	0x4004ABFF	1KB	OTS	-	-
		0x4004A400	0x4004A7FF	1KB	CMP_3,4	-	-
		0x4004A000	0x4004A3FF	1KB	CMP_1,2	-	-
		0x40049C00	0x40049FFF	1KB	CTC	-	-
		0x40049800	0x40049BFF	1KB	BLANK	-	-
		0x40049400	0x400497FF	1KB	SWDT	带保护	-
		0x40049000	0x400493FF	1KB	WDT	带保护	-
		0x400488000	0x40048FFF	2KB	ERMU	带保护	-
		0x40048400	0x400487FF	1KB	FCM	-	-
		0x40048000	0x400483FF	1KB	FCG, CMU	带保护	-

存储器分类 ⁽²⁾		开始地址 ⁽¹⁾	结束地址 ⁽¹⁾	空间大小	模块 ⁽³⁾	保护 ⁽⁴⁾	说明
外设	APB3 时钟: PCLK4	0x40042400	0x40047FFF	23KB	BLANK	-	-
		0x40042000	0x400423FF	1KB	TRNG	带保护	-
		0x40041800	0x40041FFF	2KB	BLANK	-	-
		0x40041400	0x400417FF	1KB	DAC_2	-	-
		0x40041000	0x400413FF	1KB	DAC_1	-	-
		0x40040C00	0x40040FFF	1KB	BLANK	-	-
		0x40040800	0x40040BFF	1KB	ADC_3	-	-
		0x40040400	0x400407FF	1KB	ADC_2	-	-
		0x40040000	0x400403FF	1KB	ADC_1	-	-
	APB5 时钟: PCLK0	0x4003B000	0x4003FFFF	20KB	BLANK	-	-
		0x4003AC00	0x4003AFFF	1KB	TimerA_4	-	-
		0x4003A800	0x4003ABFF	1KB	TimerA_3	-	-
		0x4003A400	0x4003A7FF	1KB	TimerA_2	-	-
		0x4003A000	0x4003A3FF	1KB	TimerA_1	-	-
		0x40038C00	0x40039FFF	5KB	BLANK	-	-
		0x40038800	0x40038BFF	1KB	Timer4_3	-	-
		0x40038400	0x400387FF	1KB	Timer4_2	-	-
		0x40038000	0x400383FF	1KB	Timer4_1	-	-
	APB2 时钟: PCLK1	0x4002D400	0x40037FFF	43KB	BLANK	-	-
		0x4002D000	0x4002D3FF	1KB	MCAN_RAM CC	-	-
		0x4002C000	0x4002CFFF	4KB	MCAN_RAM	-	-
		0x4002B800	0x4002BFFF	2KB	BLANK	-	-
		0x4002B400	0x4002B7FF	1KB	MCAN_2	-	-
		0x4002B000	0x4002B3FF	1KB	MCAN_1	-	-
		0x40029C00	0x4002AFFF	5KB	BLANK	-	-
		0x40029800	0x40029BFF	1KB	Timer0_5	-	-
		0x40029400	0x400297FF	1KB	Timer0_4	-	-
		0x40029000	0x400293FF	1KB	Timer0_3	-	-
		0x40028000	0x40028FFF	4KB	BLANK	-	-
		0x40027C00	0x40027FFF	1KB	TimerA_12	-	-
		0x40027800	0x40027BFF	1KB	TimerA_11	-	-
		0x40027400	0x400277FF	1KB	TimerA_10	-	-
		0x40027000	0x400273FF	1KB	TimerA_9	-	-
		0x40026C00	0x40026FFF	1KB	TimerA_8	-	-
		0x40026800	0x40026BFF	1KB	TimerA_7	-	-
		0x40026400	0x400267FF	1KB	TimerA_6	-	-
		0x40026000	0x400263FF	1KB	TimerA_5	-	-
		0x40025800	0x40025FFF	2KB	BLANK	-	-
		0x40025400	0x400257FF	1KB	Timer2_4	-	-
		0x40025000	0x400253FF	1KB	Timer2_3	-	-
		0x40024C00	0x40024FFF	1KB	Timer2_2	-	-
		0x40024800	0x40024BFF	1KB	Timer2_1	-	-
		0x40024400	0x400247FF	1KB	Timer0_2	-	-
		0x40024000	0x400243FF	1KB	Timer0_1	-	-
		0x40022800	0x40023FFF	6KB	BLANK	-	-

存储器分类 ⁽²⁾		开始地址 ⁽¹⁾	结束地址 ⁽¹⁾	空间大小	模块 ⁽³⁾	保护 ⁽⁴⁾	说明
外设	APB2 时钟: PCLK1	0x40022400	0x400227FF	1KB	I2S_4	-	-
		0x40022000	0x400223FF	1KB	I2S_3	-	-
		0x40021C00	0x40021FFF	1KB	USART_10	-	-
		0x40021800	0x40021BFF	1KB	USART_9	-	-
		0x40021400	0x400217FF	1KB	USART_8	-	-
		0x40021000	0x400213FF	1KB	USART_7	-	-
		0x40020C00	0x40020FFF	1KB	USART_6	-	-
		0x40020800	0x40020BFF	1KB	SPI_6	-	-
		0x40020400	0x400207FF	1KB	SPI_5	-	-
		0x40020000	0x400203FF	1KB	SPI_4	-	-
	APB1 时钟: PCLK1	0x4001E800	0x4001FFFF	6KB	BLANK	-	-
		0x4001E400	0x4001E7FF	1KB	I2S_2	-	-
		0x4001E000	0x4001E3FF	1KB	I2S_1	-	-
		0x4001DC00	0x4001DFFF	1KB	USART_5	-	-
		0x4001D800	0x4001DBFF	1KB	USART_4	-	-
		0x4001D400	0x4001D7FF	1KB	USART_3	-	-
		0x4001D000	0x4001D3FF	1KB	USART_2	-	-
		0x4001CC00	0x4001CFFF	1KB	USART_1	-	-
		0x4001C800	0x4001CBFF	1KB	SPI_3	-	-
		0x4001C400	0x4001C7FF	1KB	SPI_2	-	-
		0x4001C000	0x4001C3FF	1KB	SPI_1	-	-
		0x40018000	0x4001BFFF	16KB	Timer6	-	计数时钟: PCLK0
		0x40017C00	0x40017FFF	1KB	EMB	-	-
		0x40010C00	0x40017BFF	28KB	BLANK	-	-
		0x40010800	0x40010BFF	1KB	AOS	-	内部触发事件寄存器区域
		0x40010400	0x400107FF	1KB	EFM	带保护	-
		0x40010000	0x400103FF	1KB	BLANK	-	-
	AHB4 时钟: PCLK1	0x4000F000	0x4000FFFF	4KB	BLANK	-	-
		0x4000E000	0x4000EFFF	4KB	SKE	带保护	-
		0x4000C000	0x4000DFFF	8KB	PKE	带保护	-
		0x4000B000	0x4000BFFF	4KB	PKE_RAM	-	-
		0x40009400	0x4000AFFF	7KB	BLANK	-	-
		0x40009000	0x400093FF	1KB	CAN_1	-	-
		0x40008C00	0x40008FFF	1KB	CRC	带保护	-
		0x40008800	0x40008BFF	1KB	BLANK	-	-
		0x40008400	0x400087FF	1KB	HASH	带保护	-
		0x40008000	0x400083FF	1KB	BLANK	-	-
		0x40000000	0x40007FFF	32KB	Reserved	-	-
SRAM	SRAM 时钟: HCLK	0x24000000	0x3FFFFFFF	448MB	Reserved	-	-
		0x22000000	0x23FFFFFF	32MB	SRAMBitBand	-	-
		0x20100000	0x21FFFFFF	31MB	Reserved	-	-
		0x200F1000	0x200FFFFF	60KB	BLANK	-	-
		0x200F0000	0x200F0FFF	4KB	RetSRAM	-	ECC RAM
		0x20060000	0x200EFFFF	576KB	BLANK	-	-

存储器分类 ⁽²⁾		开始地址 ⁽¹⁾	结束地址 ⁽¹⁾	空间大小	模块 ⁽³⁾	保护 ⁽⁴⁾	说明
SRAM	SRAM 时钟: HCLK	0x20058000	0x2005FFFF	32KB	SRAM4	-	ECC RAM
		0x20040000	0x20057FFF	96KB	SRAM3	-	ECC RAM
		0x20020000	0x2003FFFF	128KB	SRAM2	-	ECC RAM
		0x20000000	0x2001FFFF	128KB	SRAM1	-	ECC RAM
CODE	SRAM 时钟: HCLK	0x1FFE0000	0x1FFFFFFF	128KB	SRAMH	-	ECC RAM
		0x03004000	0x1FFDFFFF	463.86MB	BLANK	-	-
	OTP,Flash 时 钟: HCLK	0x0300400C	0x03005FFF	8180B	BLANK	-	-
		0x03004000	0x0300400B	12B	数据安全保 护	-	用于配置数据安全保 护
		0x03002004	0x03003FFF	8187B	BLANK	-	-
		0x03002000	0x03002003	4B	引导交换	-	仅 CPU 可以访问
		0x03001FF0	0x03001FFF	16B	OTP	-	仅 CPU 可以访问
		0x03001F70	0x03001FEF	128B	BLANK	-	-
		0x03000000	0x03001F6F	8048B	OTP	-	仅 CPU 可以访问
		-	0x02100000	0x02FFFFFF	15M	BLANK	-
	REMAP 时钟: HCLK	0x02080000	0x020FFFFF	512KB	REMAP1	-	地址重映射区域 1
		0x02000000	0x0207FFFF	512KB	REMAP0	-	地址重映射区域 0
	-	0x00200000	0x01FFFFFF	30M	BLANK	-	-
	Flash 时钟: HCLK	0x00100000	0x001FFFFF	1MB	EmbeddedFI ash1	-	-
		0x00000000	0x000FFFFF	1MB	EmbeddedFI ash0	-	-



说明

- 1. 请参考《ARM Cortex-M4 Processor Technical Reference Manual Revision:r0p1》。
- 2. 总线说明请参见《总线架构 (BUS)》章节相关描述。
- 3. Reserved：访问总线会引起总线错误；BLANK：写访问无效，读访问时读到 0。
- 4. 带保护功能的模块，在保护功能有效时只支持 CPU 特权模式访问。具体寄存器及说明请参见《存储保护单元 (MPU)》章节相关描述。

2.2 外部空间映射

QSPI 空间被区分为 2 段空间，包括 QSPI 寄存器空间 64MB 和外部 QSPI 设备空间 64MB。分配关系请参考下图。

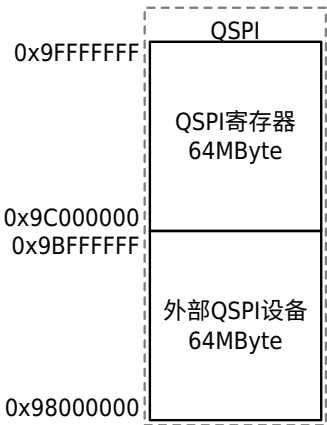


图 2-1 QSPI 地址空间分配

2.3 位段空间

Cortex-M4 存储器映射包括两个位段区域。这些区域将存储器别名区域中的每个字映射到存储器位段区域中的相应位。在别名区域写入字时，相当于对位段区域的目标位执行读-修改-写操作。

在该 MCU 中，外设寄存器和 SRAM 均映射到一个位段区域，这样可实现单个位段的读写操作。这些操作仅适用于 Cortex-M4 访问，对于其它总线主接口（如 DMA）无效。

2.4 地址重映射

本 MCU 提供了个存储器地址重映射功能，有 2 个重映射地址供自由配置,分别使用 MMF_REMCR0 和 MMF_REMCR1 来配置重映射目标地址，目标地址可以是主闪存地址也可以是 SRAMH 地址。

- 重映射地址 0：
0x02000000~0x02080000（视设定重映射空间大小而定 MMF_REMCR0.RMSIZE[4:0]）
- 重映射地址 1：
0x02080000~0x02100000（视设定重映射空间大小而定 MMF_REMCR1.RMSIZE[4:0]）

当使用重映射功能有效（MMF_REMCR0.EN=1 或 MMF_REMCR1.EN=1）时，地址对应表如下：

表 2-2 目标地址配置主闪存示例（MMF_REMCR0.EN=1 或 MMF_REMCR1.EN=1）

寄存器设定	重映射地址（CPU 地址- CPUADDR[31:0]）	重映射目标地址（主闪存地址）		
		高 12 位	中 6 位	低 14 位
REMCR0.RMSIZE[4:0]=0b01110 （重映射空间：16KB）	0x02000000~0x02003FFF	全 0	REMCR0.RMTADDR[5:0]	CPUADDR[13:0]
REMCR0.RMSIZE[4:0]=0b01111 （重映射空间：32KB）	0x02000000~0x02007FFF	全 0	REMCR0.RMTADDR[4:0] CPUADDR[14]	CPUADDR[13:0]
REMCR1.RMSIZE[4:0]=0b10000 （重映射空间：64KB）	0x02080000~0x02087FFF	全 0	REMCR1.RMTADDR[3:0] CPUADDR[15:14]	CPUADDR[13:0]
REMCR1.RMSIZE[4:0]=0b10001 （重映射空间：128KB）	0x02080000~0x0208FFFF	全 0	REMCR1.RMTADDR[2:0] CPUADDR[16:14]	CPUADDR[13:0]

表 2-3 目标地址配置 SRAMH 示例（MMF_REMCR0.EN=1 或 MMF_REMCR1.EN=1）

寄存器设定	重映射地址（CPU 地址- CPUADDR[31:0]）	重映射目标地址（SRAM 地址）		
		高 3 位	中 17 位	低 12 位
REMCR0.RMSIZE[4:0]=0b01100 （重映射空间：4KB）	0x02000000~0x02000FFF	全 0	REMCR0.RMTADDR[16:0]	CPUADDR[11:0]
REMCR0.RMSIZE[4:0]=0b01101 （重映射空间：8KB）	0x02000000~0x02001FFF	全 0	REMCR0.RMTADDR[15:0] CPUADDR[12]	CPUADDR[11:0]
REMCR1.RMSIZE[4:0]=0b01100 （重映射空间：4KB）	0x02080000~0x02080FFF	全 0	REMCR1.RMTADDR[16:0]	CPUADDR[11:0]
REMCR1.RMSIZE[4:0]=0b01101 （重映射空间：8KB）	0x02080000~0x02081FFF	全 0	REMCR1.RMTADDR[15:0] CPUADDR[12]	CPUADDR[11:0]

举例：

- 使用重映射地址 0 功能，设定源地址为主闪存 FLASH 地址 0x00008000，重映射空间 32K，寄存器 MMF_REMCR0 需设定为 0x8000800F。
- 使用重映射地址 1 功能，设定源地址为 SRAMH 地址 0x1FFFE000，重映射空间为 16K，寄存器 MMF_REMCR1 需设定为 0x9FFFE00E。

 **注意**
源地址的起始地址应设为重映射空间的整数倍。

2.5 寄存器

2.5.1 寄存器总表

基地址：0x40010500

表 2-4 存储器映射 (Memory Mapping) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	MMF_REMPRT	访问保护寄存器	32
0x0004+0x4*x	MMF_REMCrx	重映射寄存器 x (x=0~1)	32

2.5.2 访问保护寄存器 (MMF_REMPRT)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																MMF_REMPRT															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	MMF_REMPRT	保护寄存器	寄存器 MMF_REMCRO 和 MMF_REMCR1 写保护： 先对 MMF_REMPRT[15:0]写入 0x0123 接着写入 0x3210 解除保护。  说明 ● 寄存器 MMF_REMCRO 和 MMF_REMCR1 写保护状态时，读寄存器为 0； ● 寄存器 MMF_REMCRO 和 MMF_REMCR1 解除写保护状态时，读寄存器为 1。	RW

2.5.3 重映射寄存器 (MMF_REMCrx) (x=0~1)

偏移地址 (Offset) : 0x0004+0x4*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name		Reserved	RMTADDR																			Reserved						RMSIZE				
Access	RW		RW																									RW				

位/位域	标记	位名	功能描述	读写
31	EN	重映射有效位	0b0: 重映射无效 0b1: 重映射有效	RW
30:29	Reserved	保留	-	-
28:12	RMTADDR	重映射目标地址	有效位数和 RMSIZE[4:0]设定有关。 设定可参考 地址重映射 。	RW
11:5	Reserved	保留	-	-
4:0	RMSIZE	重映射空间大小	设定重映射区域的大小： 0b01100: 4KB 0b01101: 8KB 0b01110: 16KB 0b01111: 32KB 0b10000: 64KB 0b10001: 128KB 0b10010: 256KB 0b10011: 512KB - 其他: 禁止设置	RW

3 总线架构 (BUS)

3.1 简介

主系统由 32 位多层 AHB 总线矩阵构成，可实现以下主机总线和从机总线的互连：

- 主机总线
 - ▶ Cortex-M4 内核 CPU-I 总线，CPU-D 总线，CPU-S 总线
 - ▶ 系统 DMA_1 总线，系统 DMA_2 总线
 - ▶ USBFS_DMA 总线
 - ▶ USBHS_DMA 总线
 - ▶ ETHMAC_DMA 总线
- 从机总线
 - ▶ Flash ICODE 总线
 - ▶ Flash DCODE 总线
 - ▶ Flash MCODE 总线（CPU 以外其他主机访问 Flash 的总线）
 - ▶ 高速 SRAMH（SRAMH 128KB）总线
 - ▶ 系统 SRAM（SRAM1 128KB）总线
 - ▶ 系统 SRAM（SRAM2 128KB）总线
 - ▶ 系统 SRAM（SRAM3 96KB, SRAM4 32KB, Ret SRAM 4KB）总线
 - ▶ APB1 外设总线（AOS/EMB/Timer6/SPI/USART/I2S/EFM）
 - ▶ APB2 外设总线（Timer0/Timer2/TimerA/SPI/USART/I2S/CAN/MCAN）
 - ▶ APB3 外设总线（ADC/DAC/TRNG）
 - ▶ APB4 外设总线（FCG/CMU/FCM/ERMU/WDT/SWDT/CMP/PWC/CTC/OTS/RTC/VBAT/WKTM/I2C）
 - ▶ APB5 外设总线（Timer4/TimerA）
 - ▶ AHB1 外设总线（MPU/RAMIF/KEYSCAN/INTC/DCU/GPIO/DMA/CMU/DVP/MAU/FMAC）
 - ▶ AHB2 外设总线（CAN/SDIOC/USBFS）
 - ▶ AHB3 外设总线（SDIOC/ETHMAC）
 - ▶ AHB4 外设总线（HASH/CRC/CAN/USBHS/PKE/SKE）
 - ▶ AHB5 外设总线（SMC/DMC/SMCR/DMCR/NFC/QSPI）

借助总线矩阵，可以实现主机总线到从机总线高效率的并发访问。

3.2 总线架构

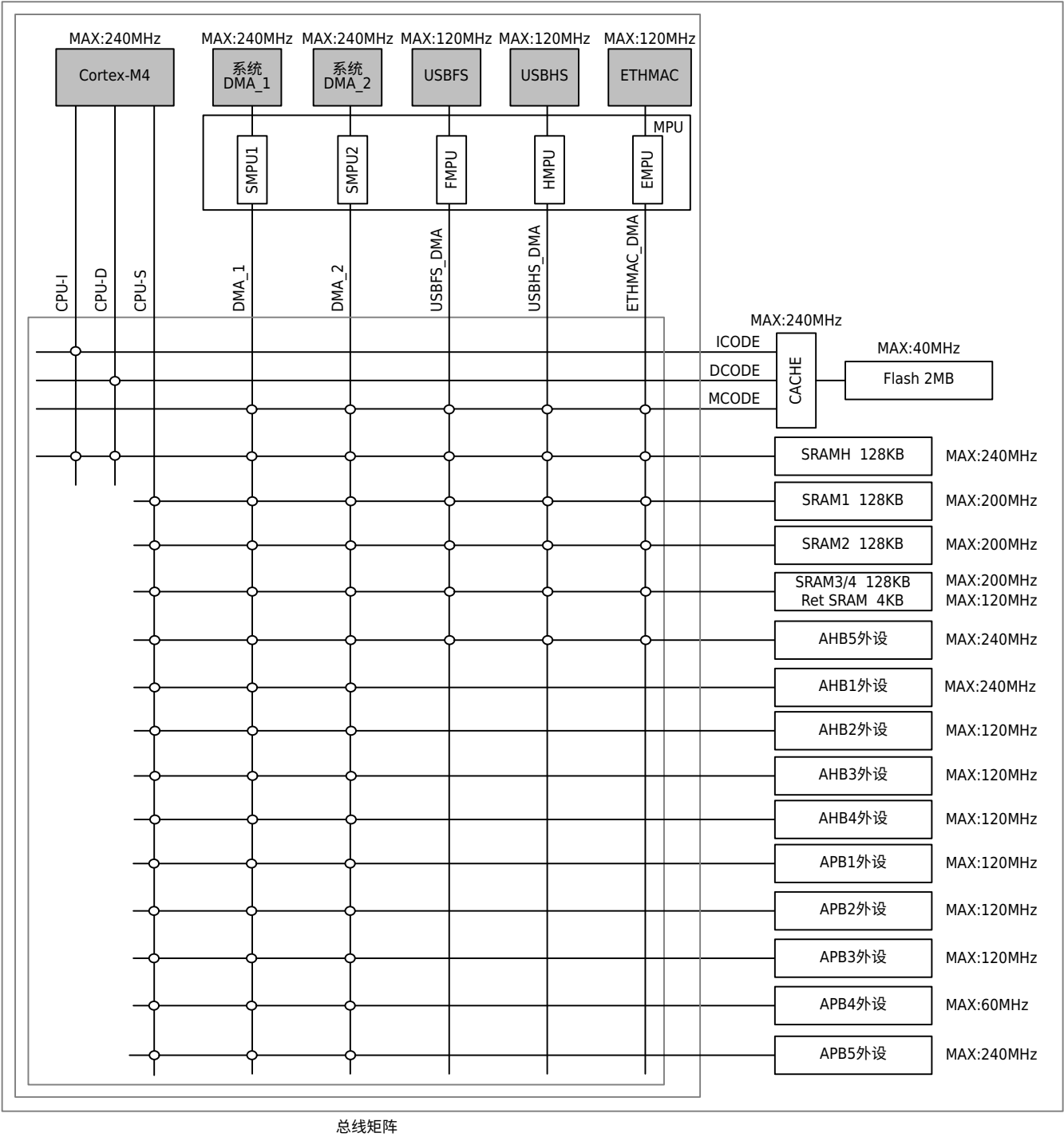


图 3-1 总线架构图

总线矩阵用于各主机总线之间的访问仲裁管理。仲裁采用循环调度算法。

● CPU-I 总线

Cortex-M4 内核的指令总线，CPU 通过此总线获取指令。访问对象是包含代码的 Flash 和 SRAMH。

● CPU-D 总线

Cortex-M4 内核的数据总线，CPU 通过此总线进行立即数加载和调试访问。访问对象是包含代码或数据的 Flash 和 SRAMH。

- CPU-S 总线

Cortex-M4 内核的系统总线，CPU 通过此总线访问外设或者系统 SRAM，也可以通过此总线获取指令和立即数（效率低于通过 CPU-I 总线和 CPU-D 总线）。访问对象为 SRAM1/SRAM2/SRAM3/SRAM4/Ret SRAM/所有外设以及外部扩展空间。

- DMA_1 总线, DMA_2 总线

系统 DMA_1/系统 DMA_2 专用总线，DMA_1/DMA_2 通过此总线访问数据存储器和外设，访问对象为 Flash/SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/Ret SRAM/所有外设以及外部扩展空间。

- USBFS_DMA 总线

USBFS 的 DMA 专用总线，USBFS 通过此总线访问所有存储器空间。访问对象为 Flash/ SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/Ret SRAM 以及外部扩展空间。

- USBHS_DMA 总线

USBHS 的 DMA 专用总线，USBHS 通过此总线访问所有存储器空间。访问对象为 Flash/ SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/Ret SRAM 以及外部扩展空间。

- ETHMAC_DMA 总线

ETHMAC 的 DMA 专用总线，ETHMAC 通过此总线访问所有存储器空间。访问对象为 Flash/ SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/Ret SRAM 以及外部扩展空间。

3.3 总线功能

总线负责实现主机对从机的读写访问。主机模块工作频率比从机模块高（如 CPU-S 访问 RTC）时，总线自动进行降频同步处理。主机模块工作频率比从机模块低（如 USBFS_DMA 访问 SRAMH）时，总线自动进行升频同步处理。

通过总线矩阵，不同主机总线的访问目标不冲突时，各访问能够同时进行。例如 CPU-I 访问 Flash，CPU-D 访问 SRAMH，CPU-S 访问 APB 外设，DMA_1 访问 SRAM1，DMA_2 访问 SRAM2，USBFS_DMA 访问 AHB5 的外部扩展空间，这些访问可以同时进行。

4 复位控制 (RMU)

4.1 简介

芯片配置了 16 种复位方式：

- 上电复位 (POR)
- NRST 引脚复位 (NRST)
- 欠压复位 (BOR)
- 可编程电压检测 1 复位 (PVD1R)
- 可编程电压检测 2 复位 (PVD2R)
- 看门狗复位 (WDTR)
- 专用看门狗复位 (SWDTR)
- 掉电唤醒复位 (PDRST)
- 软件复位 (SRST)
- MPU 错误复位 (MPUR)
- RAMECC 复位 (RAMECCR)
- FLASHECC 复位 (FLASHECCR)
- ERMU 复位 (ERMUR)
- 时钟异常复位 (CKFER)
- 外部高速振荡器异常停振复位 (XTALER)
- Cortex-M4 Lockup 复位 (LKUPR)

4.2 功能说明

4.2.1 复位方式和复位标志位

复位方式及产生条件如下表所示。

表 4-1 复位方式及产生条件

复位方式	产生条件
上电复位	VCC 上电
NRST 引脚复位	NRST 引脚输入低电平
欠压复位	VCC 电压降至低于 VBOR 电压
可编程电压检测 1 复位	VCC 电压降至低于 PVD1 电压
可编程电压检测 2 复位	VCC 电压降到低于 PVD2 电压
看门狗复位	看门狗定时器产生刷新错误或者溢出错误
专用看门狗复位	专用看门狗发生刷新错误或者溢出错误
掉电唤醒复位	通过设置掉电模式产生的复位，内核在掉电唤醒事件发生后从复位状态唤醒
软件复位	设置复位寄存器位 (ARM 寄存器 AIRCR.SYSRESETREQ 位)
MPU 错误复位	MPU 访问错误产生的复位

复位方式	产生条件
RAM ECC 错误复位	RAM 发生 ECC 错误时产生的复位
FLASH ECC 错误复位	FLASH 发生 ECC 错误时产生的复位
ERMU 复位	ERMU 发生复位请求时
时钟频率异常复位	时钟频率监视功能 (FCM) 检测到时钟周期错误时
外部高速振荡器异常停振复位	外部高速振荡器异常停振时产生的复位
Cortex-M4 Lockup 复位 (LKUPR)	当 Cortex-M4 遇到严重的异常时, 它会将自己的 PC 指针停在当前地址处并锁死, 在几个时钟周期延时之后复位整个芯片

发生复位时, 芯片根据复位方式将对应的复位标志位进行置位, 复位标志位如下表所示。比如发生引脚复位, 引脚复位标志位 PINRF 被置 1, 在 PINRF 被置位后通过写 CLRf 可以将 PINRF 清零。

表 4-2 复位方式及复位标志

复位标志	复位方式															
	上电复位	引脚复位	欠压复位	电压检测 1 复位	电压检测 2 复位	看门狗复位	专用看门狗复位	掉电唤醒复位	软件复位	MPU 错误复位	RAM ECC 错误复位	FLASH ECC 错误复位	ERMU 复位	时钟频率异常复位	外部高速振荡器异常停振复位	Cortex-M4 Lockup 复位
上电复位标志 (RMU_RSTF0.PORF)	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
引脚复位标志 (RMU_RSTF0.PINRF)	×	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-
欠压复位标志 (RMU_RSTF0.BORF)	×	×	✓	-	-	-	-	-	-	-	-	-	-	-	-	-
可编程电压检测 1 复位标志 (RMU_RSTF0.PVD1RF)	×	×	×	✓	-	-	-	-	-	-	-	-	-	-	-	-
可编程电压检测 2 复位标志 (RMU_RSTF0.PVD2RF)	×	×	×	-	✓	-	-	-	-	-	-	-	-	-	-	-
看门狗复位标志 (RMU_RSTF0.WDRF)	×	×	×	-	-	✓	-	×	-	-	-	-	-	-	-	-
专用看门狗复位标 (RMU_RSTF0.SWDRF)	×	×	×	-	-	-	✓	×	-	-	-	-	-	-	-	-
掉电唤醒复位标志 (RMU_RSTF0.PDRF)	×	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	-
软件复位标志 (RMU_RSTF0.SWRF)	×	×	×	-	-	-	-	×	✓	-	-	-	-	-	-	-
MPU 错误复位(RMU_RSTF0.MPUERF)	×	×	×	-	-	-	-	×	-	✓	-	-	-	-	-	-
RAM ECC 复位(RMU_RSTF0.RAECRF)	×	×	×	-	-	-	-	×	-	-	✓	-	-	-	-	-

复位标志	复位方式														
	上电复位	引脚复位	欠压复位	电压检测 1 复位	电压检测 2 复位	看门狗复位	专用看门狗复位	掉电唤醒复位	软件复位	MPU 错误复位	RAM ECC 错误复位	FLASH ECC 错误复位	ERUM 复位	时钟频率异常复位	外部高速振荡器异常停振复位
FLASH ECC 复位(RMU_RSTF0.FLECRF)	x	x	x	-	-	-	-	x	-	-	-	√	-	-	-
复位(RMU_RSTF0.ERMURF)	x	x	x	-	-	-	-	x	-	-	-	-	√	-	-
时钟频率异常复位 (RMU_RSTF0.CKFERF)	x	x	x	-	-	-	-	x	-	-	-	-	-	√	-
外部高速振荡器异常停振复位 (RMU_RSTF0.XTALERF)	x	x	x	-	-	-	-	x	-	-	-	-	-	-	√
Cortex-M4 Lockup 复位 (RSTF0.LKUPRF)	x	x	x	-	-	-	-	x	-	-	-	-	-	-	√

说明

“√”表示置位；“x”表示清零；“-”表示不变。

4.2.2 复位时序

4.2.2.1 上电复位

上电复位是上电复位电路引起的内部复位，时序如下所示。在将 NRST 引脚置为高电平的状态下接通电源，就产生上电复位。VCC 电压高于上电复位电压 V_{POR} 后经过一定的时间 (T_{RSTPOR}) 之后芯片内部复位解除，CPU 开始执行代码。产生上电复位时，上电复位标志 RMU_RSTF0.PORF 被置位。上电复位的详细说明请参见《电源控制 (PWC) 》章节的“上电复位/掉电复位动作说明”。

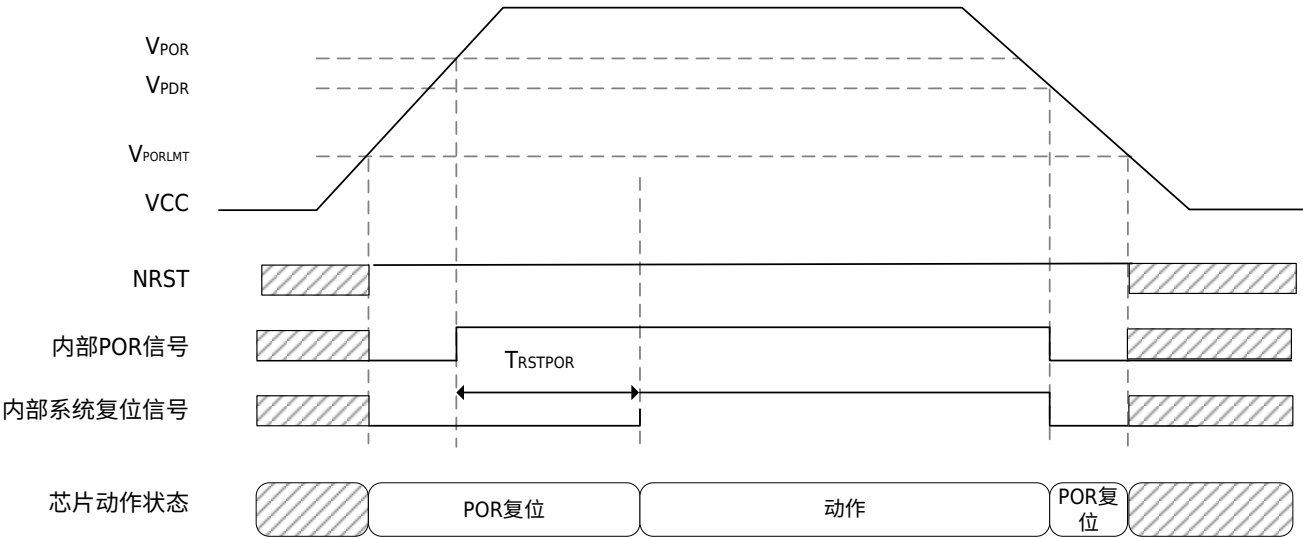


图 4-1 上电复位时序图

4.2.2.2 NRST 引脚复位

引脚复位是 NRST 引脚被驱动为低电平引起的复位，复位时序如下所示。NRST 管脚维持 T_{NRST} 宽度以上的低电平后，经过一定的内部复位时间 (T_{INRST})，解除内部复位。
产生 NRST 引脚复位时，引脚复位标志 `RMU_RSTF0.PINRF` 被置位。

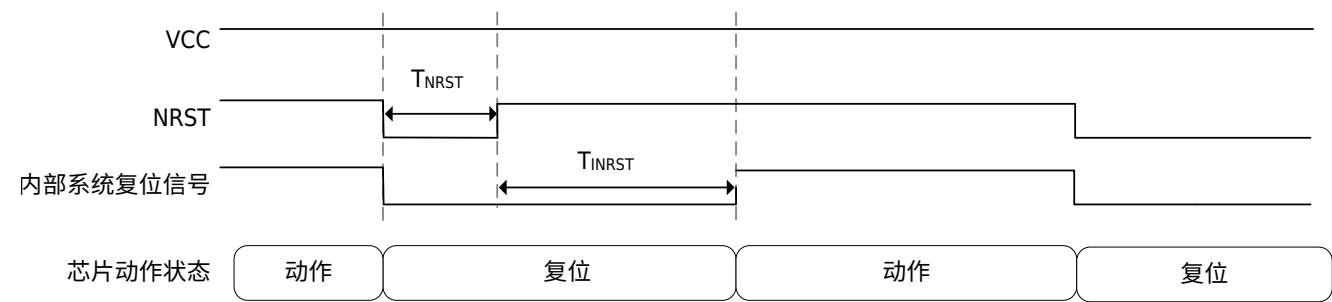


图 4-2 NRST 复位时序图

4.2.2.3 欠压复位

欠压复位是电压监测电路引起的内部复位，时序如下所示。欠压通过 ICG 寄存器设置为复位使能后，如果 VCC 电压低于监测电压 V_{BOR} ，`RMU_RSTF0.BORF` 被置位。当 VCC 电压高于监测电压 V_{BOR} 经过 V_{BOR} 的复位时间 (T_{RSTBOR}) 后解除复位。
欠压的复位设定请参见《电源控制 (PWC)》章节的“欠压复位 (BOR) 说明”。

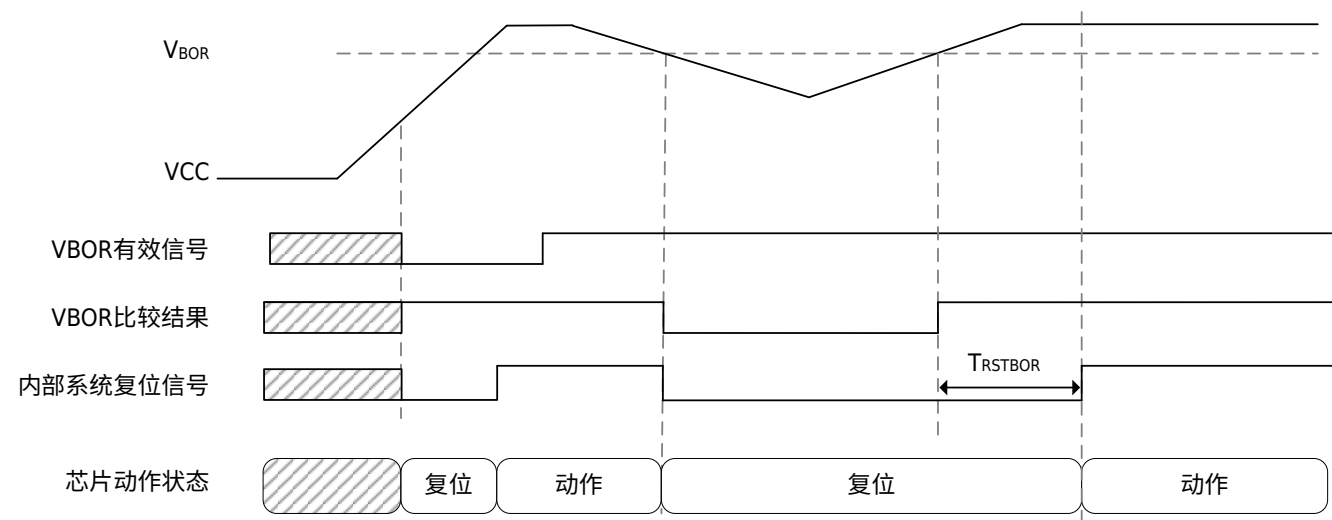


图 4-3 欠压复位时序图

4.2.2.4 可编程电压检测 1 复位、可编程电压检测 2 复位

可编程电压检测 1 和可编程电压检测 2 复位是电压监测电路引起的复位。
当 VCC 电压低于可编程电压检测 1 的监测电压时，根据设定，可产生可编程电压检测 1 复位，`RMU_RSTF0.PVD1RF` 置位。当 VCC 电压高于可编程电压检测 1 的监测电压后，经过 PVD1 的复位时间 (T_{IPVD1}) 后解除复位。

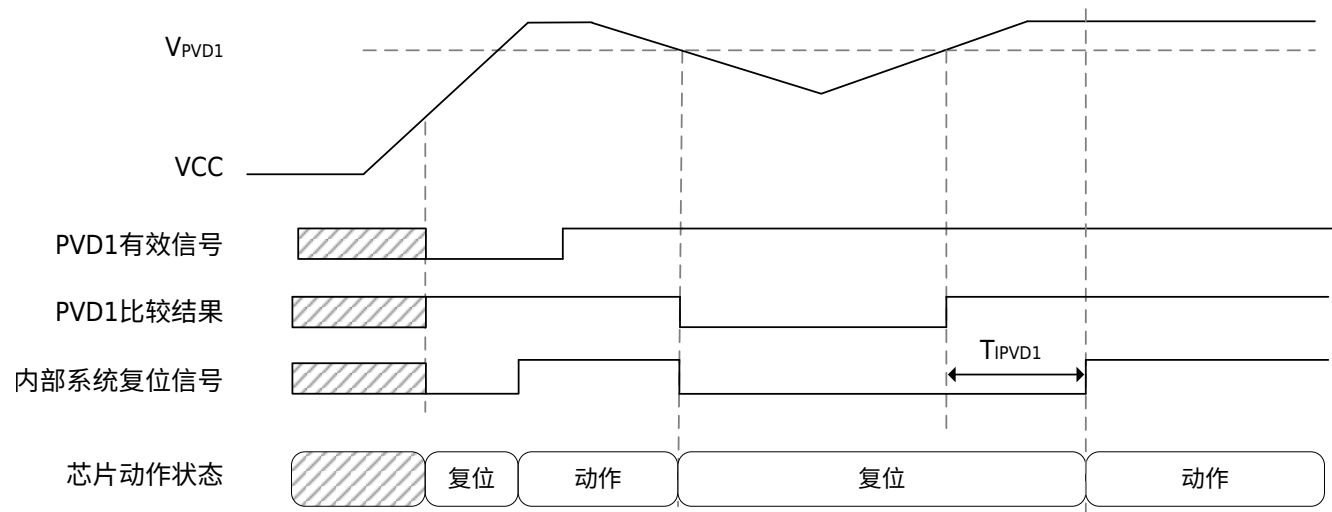


图 4-4 可编程电压检测 1 复位时序图

当 VCC 电压低于可编程电压检测 2 的监测电压时，根据设定，可产生可编程电压检测 2 复位，RMU_RSTF0.PVD2RF 置位。当 VCC 电压高于可编程电压检测 2 的监测电压后，经过 PVD2 的复位时间 (T_{IPVD2}) 后解除复位。

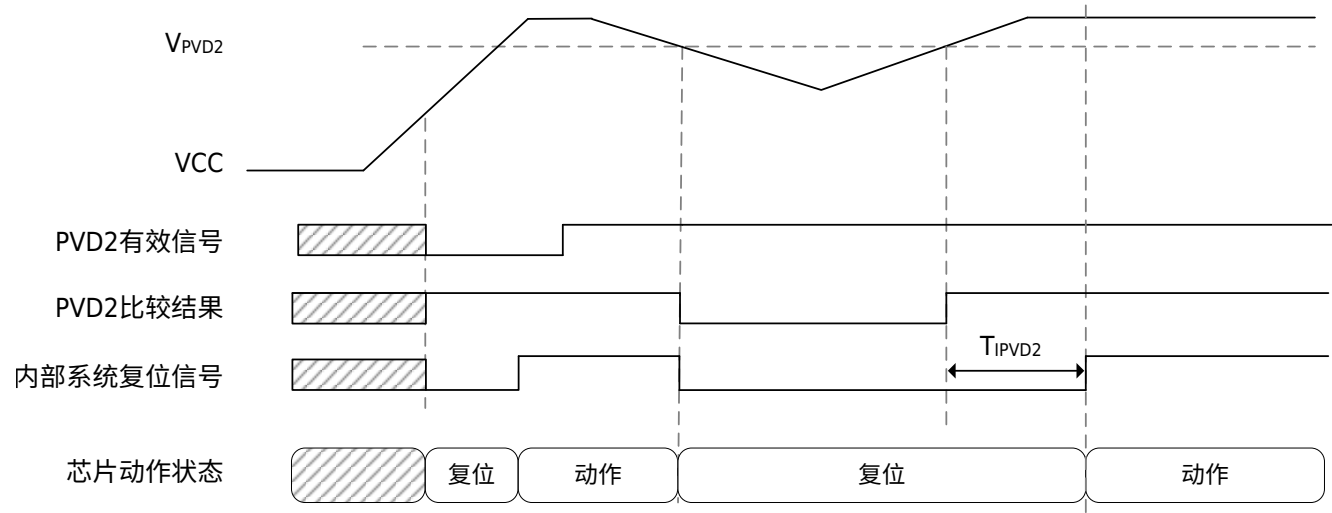


图 4-5 可编程电压检测 2 复位时序图

可编程电压检测 1、可编程电压检测 2 的复位设定请参见《电源控制 (PWC)》章节的“可编程电压检测 1 (PVD1)、可编程电压检测 2 (PVD2)”。

4.2.2.5 看门狗复位、专用看门狗复位

看门狗复位是看门狗定时器产生的内部复位，专用看门狗复位是专用看门狗定时器产生的内部复位，复位时序如下所示。

设定看门狗复位有效后，在看门狗定时器产生下溢或者不在刷新允许期间进行写操作时产生看门狗复位。看门狗复位将 RMU_RSTF0.WDRF 置位。产生看门狗复位后，经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

设定专用看门狗复位有效后，在专用看门狗定时器产生下溢或者不在刷新允许期间进行写操作时产生看门狗复位。专用看门狗复位将 RMU_RSTF0.SWDRF 置位。产生专用看门狗复位后，经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

有关看门狗复位和专用看门狗复位的详细内容请参见《看门狗计数器 (WDT/SWDT)》章节相关描述。

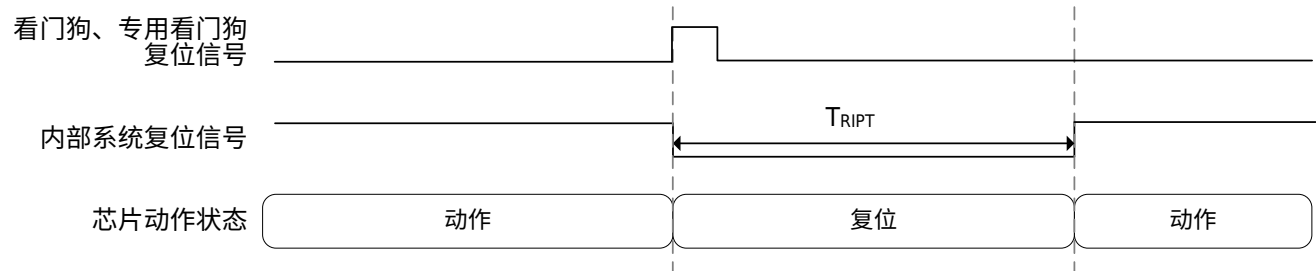


图 4-6 看门狗和专用看门狗复位时序图

4.2.2.6 掉电唤醒复位

掉电唤醒复位是芯片在设置 PWC_PWRC0.PWDN 为 1 时执行 WFI 命令，进入掉电模式后，通过掉电模式唤醒事件解除掉电模式时产生的内部复位，时序如下所示。在解除掉电模式并经过返回时间 T_{IPDx} (x=1、2、3、4) 后，解除掉电唤醒复位。返回时间根据设定的具体掉电模式有所不同，在掉电模式 1 时最小，掉电模式 3 时最大。

有关掉电唤醒复位的详细内容，请参见《电源控制 (PWC) 》章节的“掉电模式”。

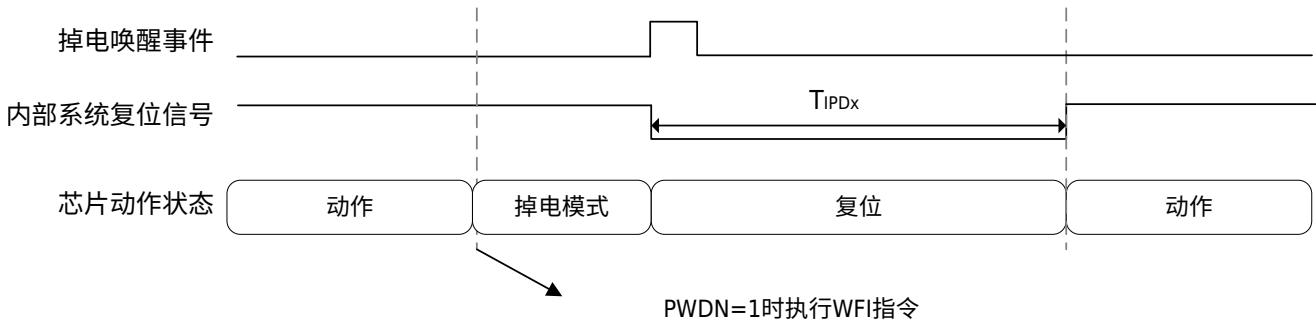


图 4-7 掉电唤醒复位时序图

4.2.2.7 软件复位

通过写 ARM 寄存器 AIRCR 的 SYSRESETREQ 位产生软件复位。产生软件复位时，RMU_RSTF0.SWRF 位被置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

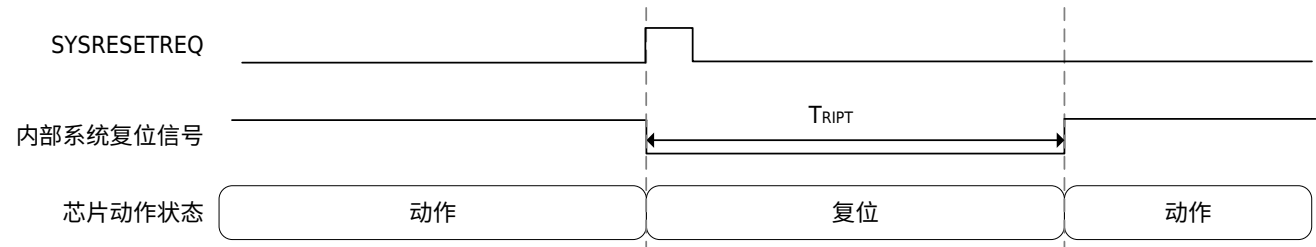


图 4-8 软件复位时序图

4.2.2.8 MPU 错误复位

MPU 错误复位将 RMU_RSTF0.MPUERF 置位，时序如下所示。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。有关 MPU 错误复位的设定，请参见《存储保护单元 (MPU) 》章节相关描述。

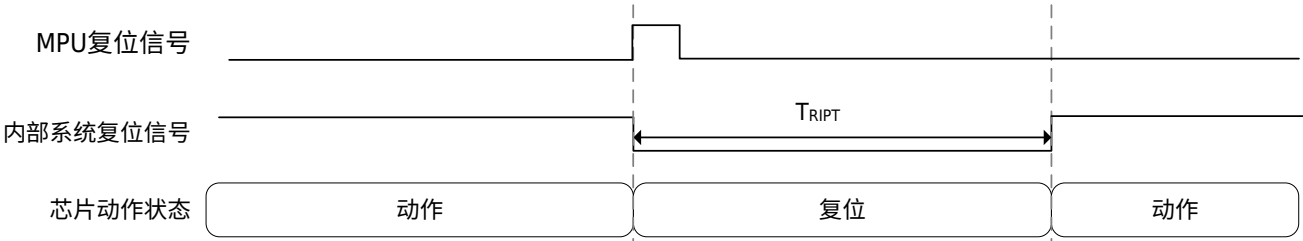


图 4-9 MPU 错误复位时序图

4.2.2.9 RAM ECC 复位

RAMECC 校验发生错误时，根据设定，可产生 RAMECC 复位，其复位时序如下所示。RAMECC 复位将 RMU_RSTF0.RAECRF 置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

RAMECC 复位的设定，请参见《内置 SRAM (SRAM) 》章节相关描述。

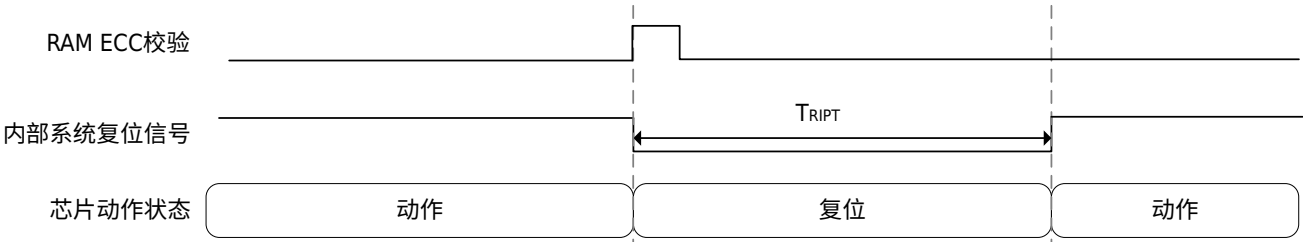


图 4-10 RAMECC 复位时序图

4.2.2.10 FLASH ECC 复位

FLASHECC 校验发生错误时，产生 FLASHECC 复位，时序如下所示。FLASHECC 复位将 RMU_RSTF0.FLECRF 置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

FLASHECC 复位的设定，请参见《嵌入式 FLASH (EFM) 》章节相关描述。

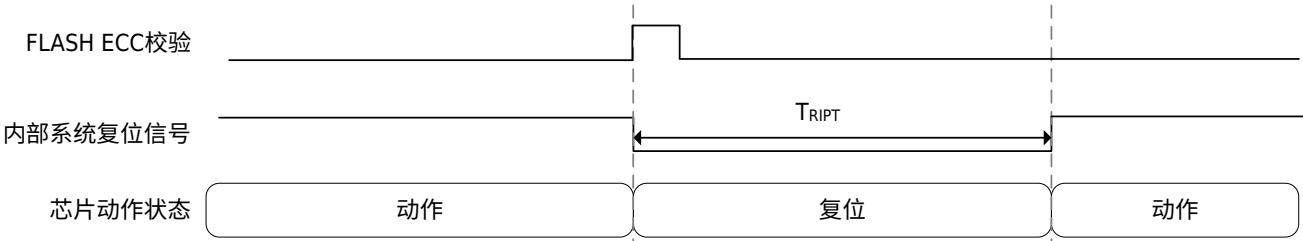


图 4-11 FLASHECC 复位时序图

4.2.2.11 ERMU 复位

ERMU 发生复位时，产生 ERMU 复位，时序如下所示。ERMU 复位将 RMU_RSTF0.ERMURF 置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。

ERMU 复位的设定，请参见《错误管理单元 (ERMU) 》章节相关描述。

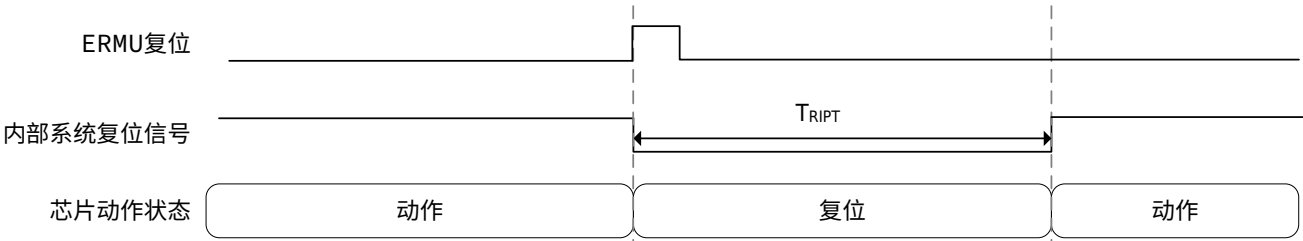


图 4-12 ERMU 复位时序图

4.2.2.12 时钟频率异常复位

芯片内置的 FCM 模块在监测到时钟频率发生异常时，根据设定，可产生时钟频率异常复位，其复位时序如下所示。时钟频率异常复位将 RMU_RSTF0.CKFERF 置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。时钟频率异常复位的设定，请参见《时钟控制器 (CMU) 》章节的“[时钟频率测量](#)”。

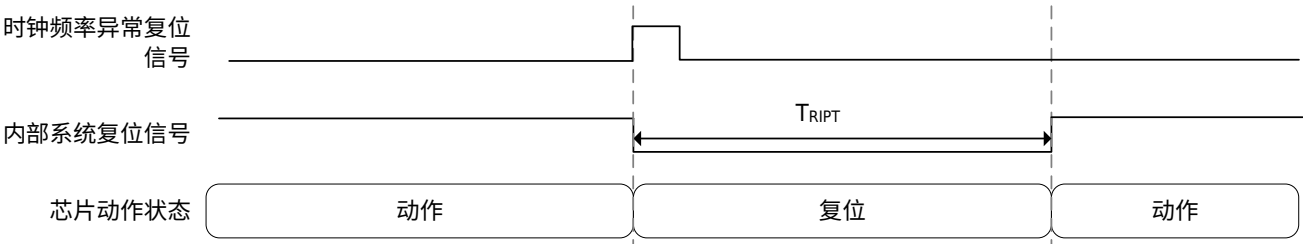


图 4-13 时钟频率异常复位时序图

4.2.2.13 外部高速振荡器异常停振复位

当芯片的振荡停止检测模块有效并且复位使能后，如果发生外部高速振荡器异常停振时，产生外部高速振荡器异常停振复位，RMU_RSTF0.XTALERF 被置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。外部高速振荡器异常停振复位的设定，请参见《时钟控制器 (CMU) 》章节的“[外部高速振荡器故障检测](#)”。

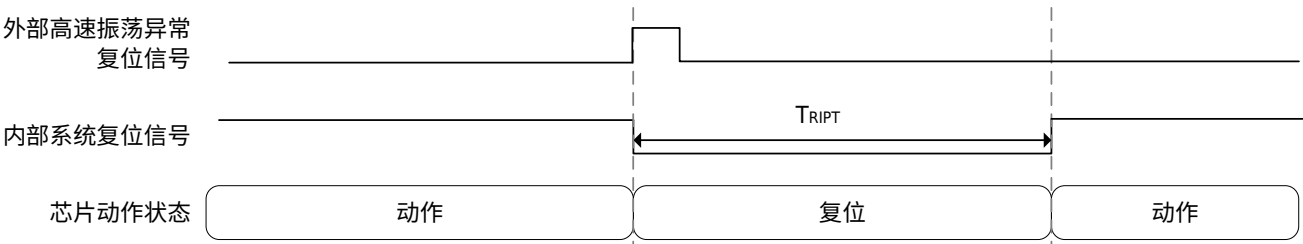


图 4-14 外部高速振荡异常复位时序图

4.2.2.14 Cortex-M4 Lockup 复位

当 Cortex-M4 遇到严重的异常时，它会将自己的 PC 指针停在当前地址处，并锁死，并在几个时钟周期延时之后复位整个芯片，时序如下所示。Cortex-M4 Lockup 复位将 RSTF0.LKUPRF 置位。经过内部复位时间 T_{RIPT} 之后，芯片解除复位。Cortex-M4 Lockup 复位必须在 RMU_PRSTCR0.LKUPREN 置位后才能使用。

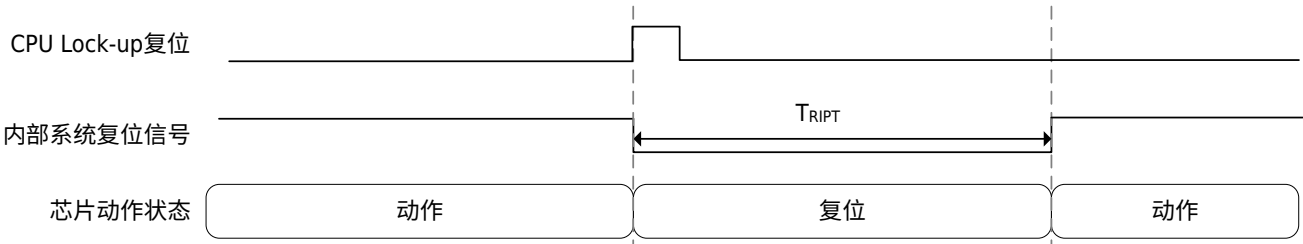


图 4-15 Cortex-M4 Lockup 复位时序图

4.2.3 复位方式判断

根据 RMU_RSTF0 的复位标志可以判断复位方式。当两个或者两个以上复位同时产生时，可能会产生多个复位标志。RMU_RSTF0 中的 MULTIRF 位为 1 时表示发生多个复位。在读 RMU_RSTF0 之后，通过置位 CLRF 位能够将所有的复位标志清 0。在置位将 RMU_RSTF0 清零后，至少需要等待 6 个 CPU 时钟周期后，才能再次读取 RMU_RSTF0 寄存器。

4.2.4 模块功能复位

除上述多种芯片复位方式以外，产品还为各外设模块配置了功能复位寄存器 RMU_FRST0~3。对 RMU_FRST 的对应位写 0，可以让对应模块的功能复位。RMU_FRST 写 0 后经过 2 个外设模块的时钟上升沿后会自动置 1。

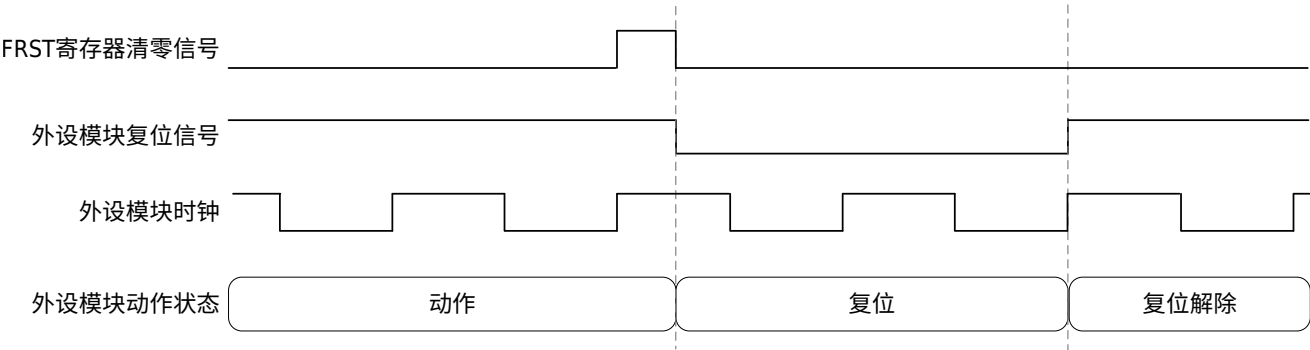


图 4-16 模块功能复位

4.2.5 复位条件

表 4-3 各模块复位条件

模块	寄存器	复位源
调试控制器 (DBGC)	MCUSTPCTL MCUSTPCTL2 MCUTRACECTL MCUDBGSTAT	1. 上电复位 2. 掉电唤醒复位
实时时钟 (RTC)	RTC 内部寄存器	模块软件复位控制位：RTC_CR0.RESET
ERMU	ERMU_ESSj	1. 上电复位 2. 掉电唤醒复位
	ERMU_EOyC ERMU_EOyS ERMU_EOyCTCNT ERMU_EOyCTCMP ERMU_EOyOM0 ERMU_EOyOM1	ERMU 功能复位 (RMU_FRST0.ERMU) 以外的所有复位源
	上記以外的其他寄存器	所有复位源
MCAN	MCANRAM_CKSR, MCANRAM_ECCERRADDR	1. 上电复位 2. 掉电唤醒复位
	上記以外的其他寄存器	所有复位源
CAN1/CAN2	CAN1_RAM_CKSR CAN1_RAM_ECCERRADDR CAN2_RAM_CKSR CAN2_RAM_ECCERRADDR	1. 上电复位 2. 掉电唤醒复位

模块	寄存器	复位源
CAN1/CAN2	上記以外の其他寄存器	所有复位源
PKE	PKE_RAM0_CKSR PKE_RAM0_ECCERRADDR PKE_RAM1_CKSR PKE_RAM1_ECCERRADDR	1. 上电复位 2. 掉电唤醒复位
	上記以外の其他寄存器	所有复位源
嵌入式 FLASH (EFM)	EFM_CKSR, EFM_F0ERRn (n=0~1) , EFM_F1ERRn (n=0~1)	1. 上电复位 2. 掉电唤醒复位
	上記以外の其他寄存器	所有复位源
内置 SRAM (SRAM)	SRAMC_CKSR SRAMC_SRAM1_ECCERRADDR SRAMC_SRAM2_ECCERRADDR SRAMC_SRAM3_ECCERRADDR SRAMC_SRAM4_ECCERRADDR SRAMC_SRAMH_ECCERRADDR SRAMC_SRAMB_ECCERRADDR SRAMC_CACHE_ECCERRADDR	1. 上电复位 2. 掉电唤醒复位
	上記以外の其他寄存器	所有复位源
电源控制 (PWC) 时钟控制 (CMU)	PWC_PWRC0 PWC_PWRC1 PWC_PDWKE0 PWC_PDWKE1 PWC_PDWKE2 PWC_PDWKES PWC_PWRC1.PDTS CMU_XTALCFGR	下记复位以外的所有复位源: ● 掉电模式 1 唤醒复位 ● 掉电模式 2 唤醒复位 ● 掉电模式 4 唤醒复位
	PWC_PDWKF0 PWC_PDWKF1	掉电唤醒复位以外的所有复位源
	PWC_PVDLCR PWC_PVDCCR1 PWC_PVDFCR PWC_PVDCCR0 PWC_PWRC1.PDTS 以外的位 PWR_PWRC4	1. 上电复位 2. 引脚复位 3. 欠压复位 4. 看门狗复位 5. 专用看门狗复位 6. 掉电模式 3 唤醒复位

模块	寄存器	复位源
电源控制 (PWC) 时钟控制 (CMU)	PWC_PVDICR[2:0] PWC_PVDDSR.PVD1DETFLG PWC_PVDICR[6:4] PWC_PVDDSR.PVD2DETFLG	1. 上电复位 2. 引脚复位 3. 欠压复位 4. 看门狗复位 5. 专用看门狗复位 6. 掉电唤醒复位
	PWC_VBATCR CMU_XTAL32CFGR CMU_XTAL32NFR CMU_XTAL32CR CMU_RTCLRCTRM PWC_WKTC0 PWC_WKTC1 PWC_WKTC2	VBAT 域复位，方法：PWC_VBATRSTR 写入 0xA5
	上記以外的其他寄存器	所有复位源
以上模块以外的寄存器		所有复位源

4.3 寄存器

4.3.1 寄存器总表

基地址：0x4004CC00

表 4-4 复位控制 (RMU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0xC0	RMU_FRST0	功能复位控制寄存器 0	32
0xC4	RMU_FRST1	功能复位控制寄存器 1	32
0xC8	RMU_FRST2	功能复位控制寄存器 2	32
0xCC	RMU_FRST3	功能复位控制寄存器 3	32
0xF8	RMU_PRSTCR0	复位控制寄存器 0	8
0xFC	RMU_RSTF0	复位标志寄存器 0	32

4.3.2 功能复位控制寄存器 0 (RMU_FRST0)

Offset	Bit Position																																
0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	DCU8	DCU7	DCU6	DCU5	DCU4	DCU3	DCU2	DCU1	CRC	TRNG	HASH	SKE	MAU	CTC	AOS	FCM	DMA2	DMA1	KEYSCAN	PKE	Reserved				ERMU	Reserved							
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW					RW								

位/位域	标记	位名	功能描述	读写
31	DCU8	DCU8 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
30	DCU7	DCU7 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
29	DCU6	DCU6 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
28	DCU5	DCU5 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
27	DCU4	DCU4 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
26	DCU3	DCU3 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
25	DCU2	DCU2 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
24	DCU1	DCU1 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
23	CRC	CRC 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
22	TRNG	加密协处理模块 CPM 中 TRNG 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
21	HASH	加密协处理模块 CPM 的安全散列算法模块 HASH 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
20	SKE	加密协处理模块 CPM 的加解密算法处理器 SKE 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
19	MAU	MAU 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
18	CTC	CTC 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW

位/位域	标记	位名	功能描述	读写
17	AOS	AOS 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
16	FCM	时钟控制器 CMU 中 FCM 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
15	DMA2	DMA2 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
14	DMA1	DMA1 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
13	KEYSCAN	KEYSCAN 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
12	PKE	加密协处理模块 CPM 中 PKE 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
11:9	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
8	ERMU	ERMU 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
7:0	Reserved	保留	所有位读出为 1, 写入时必须写 1	-

4.3.3 功能复位控制寄存器 1 (RMU_FRST1)

Offset	Bit Position																															
0xC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved	MCAN2	MCAN1	FMAC4	FMAC3	FMAC2	FMAC1	USBHS	USBFS	SPI6	SPI5	SPI4	SPI3	SPI2	SPI1	I2S4	I2S3	I2S2	I2S1	SDIOC2	SDIOC1	I2C6	I2C5	I2C4	I2C3	I2C2	I2C1	QSPI	ETHMAC	CAN2	CAN1	
Access		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	所有位读出的 1，写入时必须写 1	-
29	MCAN2	MCAN 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
28	MCAN1	MCAN 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
27	FMAC4	FMAC 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
26	FMAC3	FMAC 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
25	FMAC2	FMAC 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
24	FMAC1	FMAC 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
23	USBHS	USBHS 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
22	USBFS	USBFS 复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
21	SPI6	SPI 单元 6 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
20	SPI5	SPI 单元 5 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
19	SPI4	SPI 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
18	SPI3	SPI 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
17	SPI2	SPI 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
16	SPI1	SPI 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
15	I2S4	I2S 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW

位/位域	标记	位名	功能描述	读写
14	I2S3	I2S 单元 3 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
13	I2S2	I2S 单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
12	I2S1	I2S 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
11	SDIOC2	SDIOC 模块单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
10	SDIOC1	SDIOC 模块单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
9	I2C6	I2C 单元 6 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
8	I2C5	I2C 单元 5 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
7	I2C4	I2C 单元 4 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
6	I2C3	I2C 单元 3 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
5	I2C2	I2C 单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
4	I2C1	I2C 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
3	QSPI	QSPI 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
2	ETHMAC	ETHMAC 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
1	CAN2	CAN 单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
0	CAN1	CAN 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW

4.3.4 功能复位控制寄存器 2 (RMU_FRST2)

Offset	Bit Position																															
0xC8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	TMRA_12	TMRA_11	TMRA_10	TMRA_9	TMRA_8	TMRA_7	TMRA_6	TMRA_5	TMRA_4	TMRA_3	TMRA_2	TMRA_1	TMR2_4	TMR2_3	TMR2_2	TMR2_1	EMB	TMR0_3	TMR0_2	TMR0_1	Reserved	TMR4_3	TMR4_2	TMR4_1	Reserved						TMR6	
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW							RW	

位/位域	标记	位名	功能描述	读写
31	TMRA_12	TimerA 单元 12 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
30	TMRA_11	TimerA 单元 11 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
29	TMRA_10	TimerA 单元 10 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
28	TMRA_9	TimerA 单元 9 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
27	TMRA_8	TimerA 单元 8 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
26	TMRA_7	TimerA 单元 7 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
25	TMRA_6	TimerA 单元 6 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
24	TMRA_5	TimerA 单元 5 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
23	TMRA_4	TimerA 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
22	TMRA_3	TimerA 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
21	TMRA_2	TimerA 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
20	TMRA_1	TimerA 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
19	TMR2_4	Timer2 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
18	TMR2_3	Timer2 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
17	TMR2_2	Timer2 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW

位/位域	标记	位名	功能描述	读写
16	TMR2_1	Timer2 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
15	EMB	EMB 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
14	TMR0_3	Timer0 单元 3 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
13	TMR0_2	Timer0 单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
12	TMR0_1	Timer0 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
11	Reserved	保留	读出为 1, 写入时必须写 1	-
10	TMR4_3	Timer4 单元 3 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
9	TMR4_2	Timer4 单元 2 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
8	TMR4_1	Timer4 单元 1 功能复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW
7:1	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
0	TMR6	Timer6 复位控制	● 0b0: 功能复位有效 ● 0b1: 功能复位解除	RW

4.3.5 功能复位控制寄存器 3 (RMU_FRST3)

Offset	Bit Position																															
0xCC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved	USART10	USART9	USART8	USART7	USART6	USART5	USART4	USART3	USART2	USART1	Reserved	NFC	DMC	SMC	DVP	Reserved	OTS	TMR0_5	TMR0_4	CMP34	CMP12	Reserved	DAC2	DAC1	Reserved	ADC3	ADC2	ADC1			
Access		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW		RW	RW	RW			

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	所有位读数为 1，写入时必须写 1	-
29	USART10	USART 单元 10 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
28	USART9	USART 单元 9 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
27	USART8	USART 单元 8 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
26	USART7	USART 单元 7 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
25	USART6	USART 单元 6 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
24	USART5	USART 单元 5 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
23	USART4	USART 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
22	USART3	USART 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
21	USART2	USART 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
20	USART1	USART 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
19	Reserved	保留	读数为 1，写入时必须写 1	-
18	NFC	外部存储控制器的 NFC 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
17	DMC	外部存储控制器的 DMC 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
16	SMC	外部存储控制器的 SMC 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
15	DVP	DVP 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW

位/位域	标记	位名	功能描述	读写
14:13	Reserved	保留	所有位读出于 1, 写入时必须写 1	-
12	OTS	温度传感器 OTS 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
11	TMR0_5	Timer0 单元 5 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
10	TMR0_4	Timer0 单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
9	CMP34	电压比较器 CMP 单元 3 和单元 4 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
8	CMP12	电压比较器 CMP 单元 1 和单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
7:6	Reserved	保留	所有位读出于 1, 写入时必须写 1	-
5	DAC2	数模转换器 DAC2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
4	DAC1	数模转换器 DAC1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
3	Reserved	保留	读出于 1, 写入时必须写 1	-
2	ADC3	模数转换模块 ADC 单元 3 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
1	ADC2	模数转换模块 ADC 单元 2 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW
0	ADC1	模数转换模块 ADC 单元 1 功能复位控制	0b0: 功能复位有效 0b1: 功能复位解除	RW



说明

以下寄存器的复位不受功能复位控制寄存器 RMU_FRST0/1/2/3 控制:

USB_SYCTLREG, SDIOC_SYCTRLREG, NFC_STCR, NFC_STSR, EXMC_ENAR, USART1_NFC, TMR_SYNNR1, TMR_SYNNR2, CAN_SYCTRLREG, ETH_MAC_IFCONFR

4.3.6 复位控制寄存器 0 (RMU_PRSTCR0)

Offset	Bit Position							
0xF8	7	6	5	4	3	2	1	0
Reset	0x40							
Name	Reserved		LKUPREN	Reserved				
Access			RW					

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6	Reserved	保留	读出为 1，写入时必须写 1	-
5	LKUPREN	Cortex-M4 Lockup 复位使能	0b0: Cortex-M4 Lockup 复位无效 0b1: Cortex-M4 Lockup 复位使能	RW
4:0	Reserved	保留	-	-

4.3.7 复位标志寄存器 0 (RMU_RSTF0)

Offset	Bit Position																																	
0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0xFFFFFFFF（根据复位方式不同，复位值不同）																																	
Name	CLRf	MULTIRf	Reserved													ERMURf	FLECRf	LKUPRf	XTALERf	CKFERf	RAECRf	Reserved	MPUERf	SWRf	PDRf	SWDRf	WDRf	PVD2Rf	PVD1Rf	BORf	PINRf	PORf		
Access	RW	RW														RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	CLRf	清除复位标志	写 1 清零复位标志, 写 0 无效; 读出恒为 0  说明 置位动作必须在读取 RMU_RSTF0 之后进行。	RW
30	MULTIRf	2 个以上复位发生标志	发生两个及两个以上复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生两个及两个以上复位 ● 0b1: 发生两个及两个以上复位	RW
29:17	Reserved	保留	-	-
16	ERMURf	ERMU 复位标志	发生 ERMU 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
15	FLECRf	FLASHECC 复位标志	发生 FLASHECC 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
14	LKUPRf	Cortex-M4 Lockup 复位标志	发生 Cortex-M4 Lockup 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
13	XTALERf	外部高速振荡器异常停振复位标志	发生外部高速振荡器异常停振复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
12	CKFERf	时钟频率异常复位标志	发生时钟频率异常复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW

位/位域	标记	位名	功能描述	读写
11	RAECRF	RAMECC 复位标志	发生 RAMECC 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
10	Reserved	保留	-	-
9	MPUERF	MPU 错误复位标志	发生 MPU 错误复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
8	SWRF	软件复位标志	发生软件复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
7	PDRF	掉电模式复位	发生掉电模式复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
6	SWDRF	专用看门狗复位标志	发生专用看门狗复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
5	WDRF	看门狗复位标志	发生看门狗复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
4	PVD2RF	可编程电压检测 2 复位标志	发生可编程电压检测 2 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
3	PVD1RF	可编程电压检测 1 复位标志	发生可编程电压检测 1 复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW
2	BORF	欠压复位标志	发生欠压复位时, 由硬件置位。 通过置位 CLRf 清零。 读取: ● 0b0: 未发生复位 ● 0b1: 发生复位	RW

位/位域	标记	位名	功能描述	读写
1	PINRF	NRST 引脚复位标志	发生引脚复位时，由硬件置位。 通过置位 CLR_F 清零。 读取： ● 0b0：未发生复位 ● 0b1：发生复位	RW
0	PORF	上电复位标志	发生上电复位时，由硬件置位。 通过置位 CLR_F 清零。 读取： ● 0b0：未发生复位 ● 0b1：发生复位	RW

5 时钟控制器 (CMU)

5.1 简介

时钟控制单元提供了一系列频率的时钟功能，包括：一个外部高速振荡器、一个外部低速振荡器、两个 PLL 时钟、一个内部高速振荡器、一个内部中速振荡器、一个内部低速振荡器、一个 RTC 专用内部低速振荡器、一个 SWDT 专用内部低速振荡器、时钟预分频器、时钟多路复用和时钟门控电路。

时钟控制单元还提供时钟频率测量功能。时钟频率测量电路 (FCM) 使用测定基准时钟对测定对象时钟进行监视测定。在超出设定范围时发生中断或者复位。

AHB、APB 和 Cortex-M4 时钟都源自系统时钟。系统时钟的最大运行时钟频率可以达到 240MHz，有 6 个可选择的时钟源：

- 1. 外部高速振荡器 (XTAL)
- 2. 外部低速振荡器 (XTAL32)
- 3. PLLH 时钟 (PLLH)
- 4. 内部高速振荡器 (HRC)
- 5. 内部中速振荡器 (MRC)
- 6. 内部低速振荡器 (LRC)

对于每一个时钟源，在未使用时都可以单独打开和关闭，以降低功耗。SWDT 有独立的时钟源：SWDT 专用内部低速振荡器 (SWDTLRC)。实时时钟 (RTC) 使用外部低速振荡器、RTC 专用内部低速振荡器或者 XTAL 小数分频时钟作为时钟源。USBFS 的 48MHz 时钟可以选择系统时钟、PLLH、PLLA 作为时钟源。

5.2 主要特性

表 5-1 时钟源主要特性

时钟源	规格
外部高速振荡器 XTAL	晶振的频率范围：4~25MHz 外部时钟输入：最高 25MHz 振荡器故障检测功能
外部低速振荡器 XTAL32	晶振的频率：32.768kHz
PLLH 时钟	输入时钟：外部高速振荡器或者内部高速振荡器 PLLH 输入时钟分频：1~4 任意分频可选 PFD 输入时钟频率=输入时钟/PLLH 输入时钟分频，频率范围 8MHz~25MHz PLLH 倍频系数：25~150 倍 VCO 振荡频率：600MHz~1200MHz PLLHP 输出分频比：2~16 任意分频 PLLHQ 输出分频比：2~16 任意分频 PLLHR 输出分频比：2~16 任意分频 PLLHP 输出频率=（输入时钟/PLLH 输入时钟分频）*PLLH 倍频系数/PLLHP 输出分频比 PLLHQ 输出频率=（输入时钟/PLLH 输入时钟分频）*PLLH 倍频系数/PLLHQ 输出分频比 PLLHR 输出频率=（输入时钟/PLLH 输入时钟分频）*PLLH 倍频系数/PLLHR 输出分频比

时钟源	规格
PLLA 时钟	输入时钟：外部高速振荡器或者内部高速振荡器 PLLA 输入时钟分频：1~25 任意分频可选 $\text{PFD 输入时钟频率} = \text{输入时钟} / \text{PLLA 输入时钟分频}$ ，频率范围 1MHz~25MHz PLLA 倍频系数：20~480 倍 VCO 振荡频率：240MHz~480Hz PLLAP 输出分频比：2~16 任意分频 PLLAQ 输出分频比：2~16 任意分频 PLLAR 输出分频比：2~16 任意分频 $\text{PLLAP 输出频率} = (\text{输入时钟} / \text{PLLA 输入时钟分频}) * \text{PLLA 倍频系数} / \text{PLLAP 输出分频比}$ $\text{PLLAQ 输出频率} = (\text{输入时钟} / \text{PLLA 输入时钟分频}) * \text{PLLA 倍频系数} / \text{PLLAQ 输出分频比}$ $\text{PLLAR 输出频率} = (\text{输入时钟} / \text{PLLA 输入时钟分频}) * \text{PLLA 倍频系数} / \text{PLLAR 输出分频比}$
内部高速振荡器 HRC	频率：16MHz 或者 20MHz 用户可写寄存器对频率微调
内部中速振荡器 MRC	频率：8MHz 用户可写寄存器对频率微调
内部低速振荡器 LRC	频率：32.768kHz 用户可写寄存器对频率微调
RTC 专用内部低速振荡器 RTCLRC	频率：32.768kHz 用户可写寄存器对频率微调
SWDT 专用内部低速振荡器 SWDTLRC	频率：10kHz

5.3 功能说明

5.3.1 系统框图

5.3.1.1 时钟系统框图

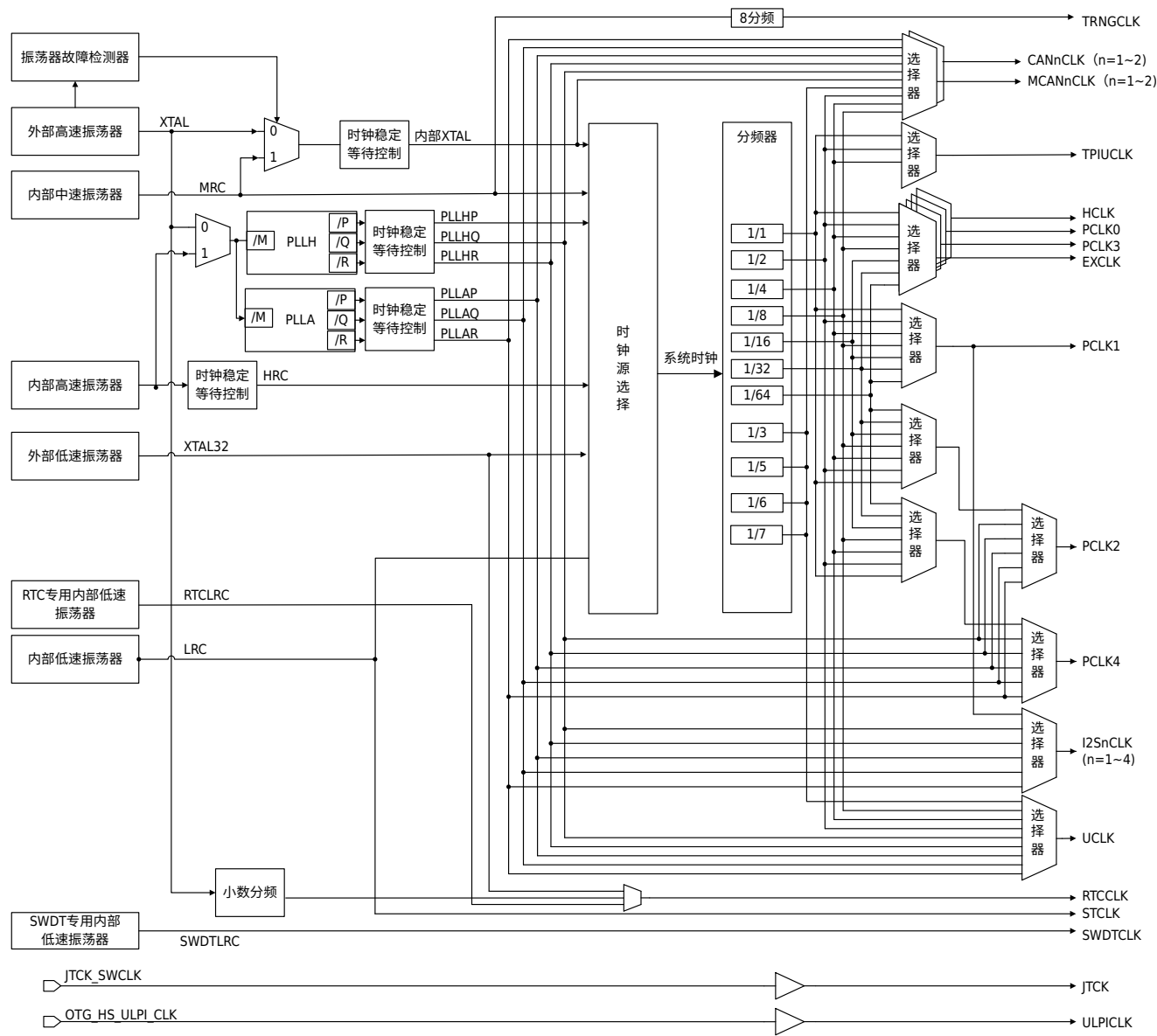


图 5-1 时钟系统框图

5.3.1.2 时钟频率测量框图

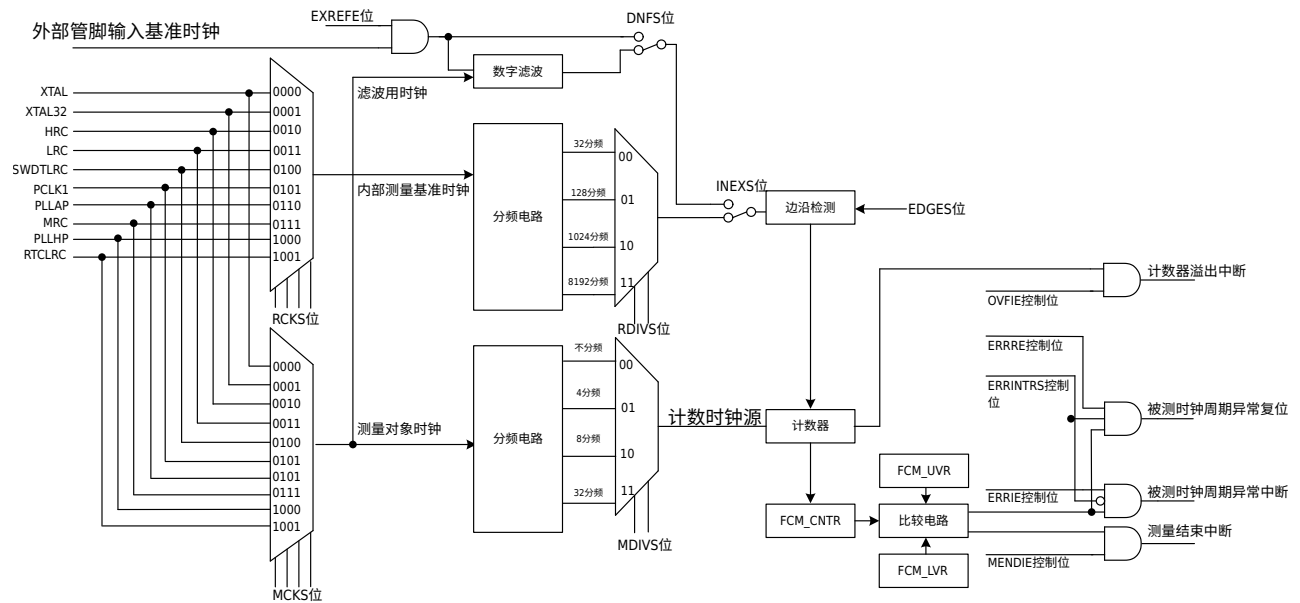


图 5-2 时钟频率测量框图

5.3.2 工作时钟规格

表 5-2 各个内部时钟的规格

时钟	作用范围	规格
HCLK	CPU、DMA _n (n=1~2)、EFM、SRAM _n (n=1~4)、SRAMH、Ret SRAM、MPU、GPIO、DCUn (n=1~8)、INTC、QSPI、DVP、KEYSCAN、FMAC _n (n=1~4)、MAU	最高频率 240MHz。 由 CMU_SCFGR 寄存器 HCLKS 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。
PCLK0	Timer6 _n (n=1~8) 计数器、Timer4 _n (n=1~3)、TimerA _n (n=1~4)	最高频率 240MHz。 由 CMU_SCFGR 寄存器 PCLK0S 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。
PCLK1	USBHS/USBFS 控制逻辑、USART _n (n=1~10)、SPI _n (n=1~6)、Timer0 _n (n=1~5)、Timer2 _n (n=1~4)、TimerA _n (n=5~12)、Timer6 _n (n=1~8) 控制逻辑、EMB、CRC、HASH、I2S _n (n=1~4)、SDIO _{Cn} (n=1~2)、ETHMAC、CAN _n (n=1~2) 控制逻辑、MCAN _n (n=1~2) 控制逻辑、SKE、PKE	最高频率 120MHz。 由 CMU_SCFGR 寄存器 PCLK1S 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。

时钟	作用范围	规格
PCLK2	ADC 转换时钟	最高频率 60MHz。 由 CMU_SCFGR 寄存器的 PCLK2S 位和 CMU_PERICKSEL 寄存器 PERICKSEL 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。 可选独立时钟源：PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR。
PCLK3	RTC (控制逻辑)、I2Cn (n=1~6)、CMPn (n=1~2)、WDT、SWDT 控制逻辑、WKTm、OTS、FCM、VBAT 备份寄存器、CTC、ERMU	电源电压高于等于 2.0V 时，最高频率 60MHz。 电源电压低于 2.0V 时，最高频率 30MHz。 由 CMU_SCFGR 寄存器 PCLK3S 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。
PCLK4	ADCn (n=1~3) (控制逻辑)、DACn (n=1~2) (控制逻辑)、TRNG	最高频率 120MHz。 由 CMU_SCFGR 寄存器的 PCLK4S 位和 CMU_PERICKSEL 寄存器 PERICKSEL 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。 可选独立时钟源：PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR。
EXCLK	NFC、DMC、SMC	最高频率 120MHz。 由 CMU_SCFGR 寄存器的 EXCKS 位配置。 可选系统时钟的分频：1, 2, 4, 8, 16, 32, 64。
UCLK	USBFS/USBHS FullSpeed 通信用时钟	频率 48MHz。 由 CMU_USBCKCFGR 寄存器的 USBCKS 位配置。 可选系统时钟分频：2, 3, 4, 5, 6, 7, 8。 可选独立时钟源：PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR。
CANnCLK	CANn (n=1~2) 通信时钟	最高频率 80MHz。 由 CMU_CANCKCFGR 寄存器的 CANnCKS (n=1~2) 位配置。 可选系统时钟分频 2, 3, 4, 5, 6, 7, 8。 可选独立时钟源：PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR, XTAL。
MCANnCLK	MCANn (n=1~2) 通信时钟	最高频率 80MHz。 由 CMU_CANCKCFGR 寄存器的 MCANnCKS (n=1~2) 位配置。 可选系统时钟分频 2, 3, 4, 5, 6, 7, 8。 可选独立时钟源：PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR, XTAL。

时钟	作用范围	规格
STCLK	SYSTICK 定时器外部参考时钟	内部低速振荡器时钟。 CPU 的 SYSTICK 控制和状态寄存器的 CLKSOURCE 位配置 SYSTICK 定时器时钟。
SWDTCLK	SWDT 计数器用时钟	频率 10kHz。
TCK	JTAG 用时钟	最高频率 25MHz。
TPIUCLK	Cortex-M4 调试跟踪器用时钟	最高频率 60MHz。 由 CMU_TPIUCKCFGR 寄存器的 TPIUCKS 位配置。 可选系统时钟的分频：1，2，4。
ULPICKL	USBHS HighSpeed 通信用时钟	频率 60MHz
I2SnCLK	I2Sn (n=1~4)	最高频率 240MHz。 由 CMU_I2SCKSEL 寄存器配置。 可选择时钟：PLLAP，PLLAQ，PLLAR，PLLHQ，PLLHR，PCLK1。
TRNGCLK	TRNG 动作时钟	频率为 1MHz。 固定为 MRC 的 8 分频，使用 TRNG 功能时禁止停止 MRC 时钟。

各时钟之间需遵守下列规则：

1. HCLK 频率≥PCLK2、PCLK3、PCLK4 频率
2. ETHMAC：
 - ETHMAC MII 模式：
 - ▶ PCLK1 频率>2 倍 MII_TX_CLK/MII_RX_CLK 频率
 - ▶ HCLK 频率>PCLK1 频率
 - ETHMAC RMII 模式：
 - ▶ PCLK1 频率>RMII_REF_CLK 频率
 - ▶ HCLK 频率>PCLK1 频率
 - ETHMAC 未使用时：
 - ▶ HCLK 频率≥PCLK1 频率
3. NFC：
 - NFC 使用时：HCLK 频率>PCLK1 频率
 - NFC 未使用时：HCLK 频率≥PCLK1 频率
4. PCLK0 频率≥PCLK1 频率，PCLK0 频率≥PCLK3 频率
5. HCLK 频率：PCLK0 频率=N:1，1:N
6. PCLK2 频率：PCLK4 频率=1:8，1:4，1:2，1:1，2:1，4:1，8:1

5.3.3 晶振电路

5.3.3.1 外部高速振荡器

5.3.3.1.1 振荡器模式

外部高速振荡器可为系统时钟提供更为精确时钟源。

XTAL 通过 CMU_XTALCR 的 XTALSTP 位打开和关闭。

CMU_OSCSTBSR 的 XTALSTBF 标志位指示外部高速振荡器是否稳定，稳定时间通过寄存器 CMU_XTALSTBCR 配置。CMU_XTALSTBCR 设定的稳定时间一定要大于等于晶振厂商要求的稳定时间。

晶振的电路常数因晶振和安装电路的寄生电容而不同，因此必须和晶振厂商仔细商谈后决定。振荡器的各种特性与用户的电路板设计密接相关，晶振和负载电容必须尽可能地靠近振荡器的引脚，以尽量减小输出失真和起振稳定时间。负载电容值必须根据所选振荡器的不同做适当调整。在振荡电路附近不能通过信号线，否则就可能因电感而不能正常振荡。

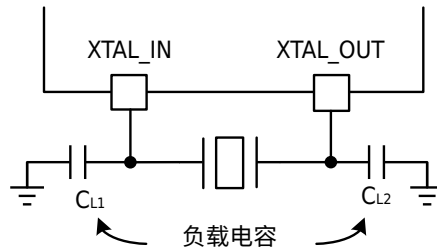


图 5-3 外部高速振荡器连接示例

5.3.3.1.2 时钟输入模式

时钟输入模式下，必须提供外部时钟源。此模式通过 CMU_XTALCFGR 的 XTALMS 位置 1 和 CMU_XTALCR 的 XTALSTP 位置 0 进行选择。必须使用占空比约为 50% 的外部时钟信号来驱动 XTAL_OUT/XTAL_EXT 引脚。此时 XTAL_IN 引脚可根据寄存器设定配置成 GPIO。

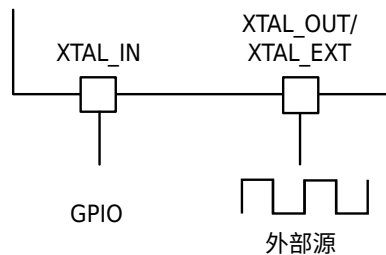


图 5-4 外部时钟输入的连接示例

5.3.3.2 外部高速振荡器故障检测

振荡器故障检测是检测外部高速振荡器 (XTAL) 振荡是否正常。

通过寄存器 CMU_XTALSTDCR 的 XTALSTDE 位打开或关闭。

复位解除后，外部高速振荡器停止振荡，外部高速振荡器故障检测功能无效。要将外部高速振荡器故障检测功能置为有效时，必须使外部高速振荡器振荡，并且在等到外部高速振荡器稳定即 CMU_OSCSTBSR.XTALSTBF 为 1，通过寄存器 CMU_XTALSTDCR 的 XTALSTDE 位打开。

PLLH、PLLA 选择 XTAL 时钟作为输入源时，只能选择 XTAL 振荡故障产生复位功能。

因为振荡器故障检测是检测外部因素导致的振荡器异常振荡，所以要通过软件使外部高速振荡器停止振荡或者转移到停止模式和掉电模式时，将振荡器振荡故障检测功能无效。

如果外部高速振荡器发生故障，动作波形如下图所示。操作流程请参见“检测到 XTAL 故障检测动作”。

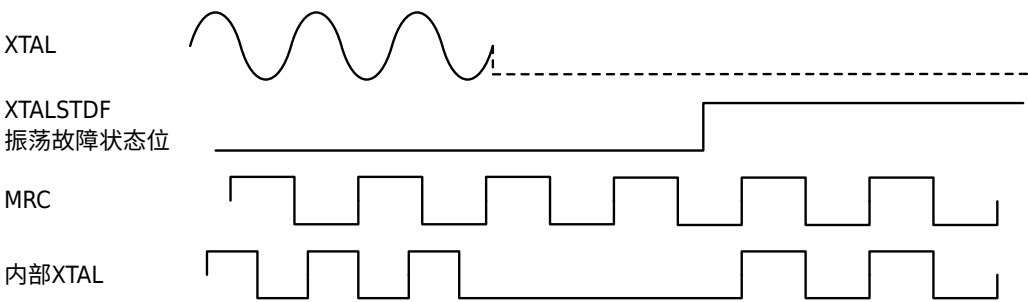


图 5-5 外部高速振荡器故障检测示例

5.3.3.2.1 检测到 XTAL 故障检测动作

当检测到外部高速振荡器振荡故障时，如果系统时钟选择外部高速振荡器作为系统时钟，系统时钟会自动切换到 MRC。

当检测到外部高速振荡器振荡故障时，可触发 EMB，将 Timer6/Timer4 的 PWM 输出置成 Hi-Z 输出。详细信息请参见《紧急刹车模块 (EMB) 》章节相关描述。

系统时钟选择成 XTAL，检测到 XTAL 故障时，动作例如下所示：

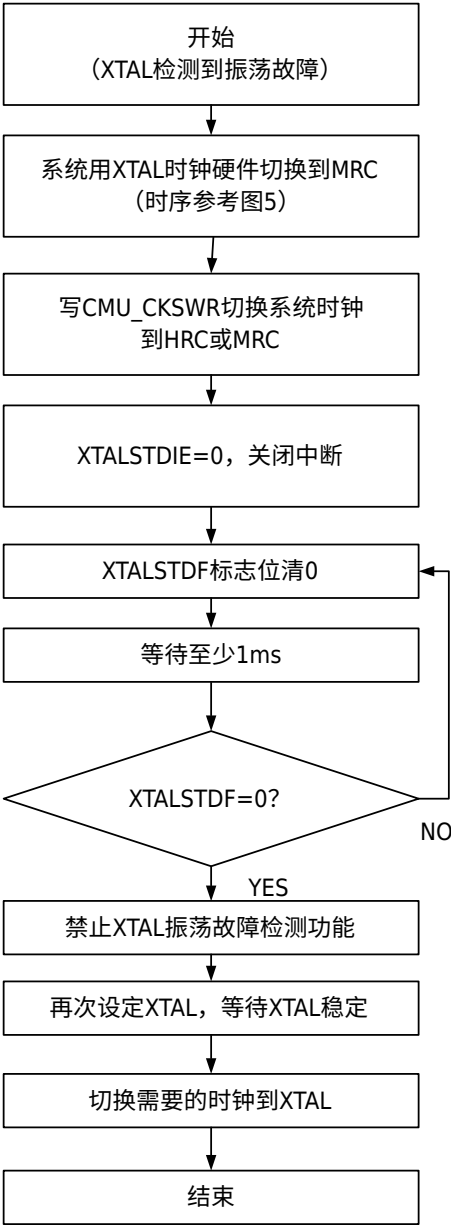


图 5-6 系统时钟选择 XTAL，检测到 XTAL 振荡故障示例

5.3.3.2 检测到 XTAL 振荡故障产生中断复位

XTAL 振荡故障中断可配置成可屏蔽中断或不可屏蔽中断，详细信息请参见《中断控制器 (INTC)》章节。
XTAL 振荡故障配置成复位时，检测到 XTAL 振荡故障，芯片产生复位，复位动作请参见《复位控制 (RMU)》章节。

5.3.3.3 外部高速振荡器小数分频

通过设定 CMU_XTALDIVR 寄存器，外部高速振荡器时钟可以分频到 32.768kHz 供 RTC 使用。

外部高速振荡器 10MHz 分频到 32.768kHz 设定例：

$$\text{分频数} = \frac{10000000\text{Hz}}{32768\text{Hz}} = \frac{78125 \text{ (10进制)}}{256 \text{ (10进制)}} = \frac{0x1312D \text{ (16进制)}}{0x100 \text{ (16进制)}}$$

NUMER[16:0]寄存器设定为 0x1312D，DENOM[10:0]寄存器设定为 0x100，即寄存器 CMU_XTALDIVR 设定为 0x1312D100，再通过 CMU_XTALDIVCR 设定开始。

外部高速振荡器 8MHz 分频到 32.768kHz 设定例：

$$\text{分频数} = \frac{8000000\text{Hz}}{32768\text{Hz}} = \frac{31250 \text{ (10进制)}}{128 \text{ (10进制)}} = \frac{0x7A12 \text{ (16进制)}}{0x80 \text{ (16进制)}}$$

NUMER[16:0]寄存器设定为 0x07A12，DENOM[10:0]寄存器设定为 0x080，即寄存器 CMU_XTALDIVR 设定为 0x07A12080，再通过 CMU_XTALDIVCR 设定开始。

5.3.3.4 外部低速振荡器

32.768kHz 的外部低速振荡器可为系统时钟、实时时钟电路 (RTC) 提供更为精确时钟源。具有功耗低且精度高的优点。

XTAL32 通过 CMU_XTAL32CR 的 XTAL32STP 位打开和关闭。

晶振的电路常数因晶振和安装电路的寄生电容而不同，因此必须和晶振厂商仔细商谈后决定。振荡器的各种特性与用户的电路板设计密接相关，晶振和负载电容必须尽可能地靠近振荡器的引脚，以尽量减小输出失真和起振稳定时间。负载电容值必须根据所选驱动能力不同做适当调整。在振荡电路附近不能通过信号线，否则就可能因电感而不能正常振荡。

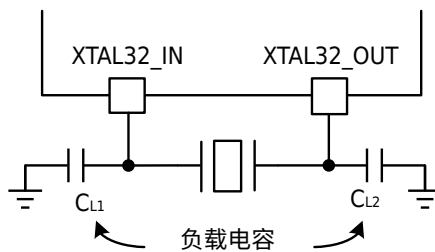


图 5-7 外部低速振荡器连接示例

XTAL32 初次上电的初始化流程如下所示：

1. CMU_XTAL32CR.XTAL32STP 位写 1，停止 XTAL32；
2. 通过 CMU_XTAL32FGR 设定匹配的 XTAL32 驱动能力；
3. 通过 CMU_XTAL32NFR 设定滤波功能；
4. CMU_XTAL32CR.XTAL32STP 位写 0，XTAL32 振荡；
5. 软件等待 XTAL32 稳定，稳定时间请参见《数据手册》“电气特性”章节。

如果不使用外部低速振荡器，将 CMU_XTAL32CR 的 XTAL32STP 位设 1，关闭外部低速振荡器。

5.3.4 内部 RC 时钟

5.3.4.1 HRC 时钟

HRC 时钟信号由内部高速振荡器生成，可直接用作系统时钟，或者用作 PLLH、PLLA 输入。HRC 的频率可由 ICG1.HRCFREQSEL 配置成 16MHz 或者 20MHz。

HRC 振荡器的优点是成本较低（无需使用外部组件）。此外，其启动速度也要比 XTAL 晶振块，但即使校准后，其精度也不及外部晶振。

频率校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此会对每个器件进行出厂校准，保证精度请参见《数据手册》“电气特性-工作条件-内部时钟源特性-内部高速（HRC）振荡器”章节。

如果应用受到温度变化影响，则这可能也会影响到 RC 振荡器的速度。用户可通过寄存器对 HRC 频率进行微调。

CMU_OSCSTBSR 中的 HRCSTBF 标志指示 HRC 是否稳定。在启动时，硬件将此位置 1 后，HRC 才可以使用。

HRC 可通过 CMU_HRCCR 控制寄存器中的 HRCSTP 位打开或关闭。

5.3.4.2 MRC 时钟

MRC 时钟信号由内部 8MHz 中速振荡器生成，可直接用作系统时钟。

MRC 振荡器的优点是启动速度快。

频率校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此会对每个器件进行出厂校准，保证精度请参见《数据手册》“电气特性-工作条件-内部时钟源特性-内部中速（MRC）振荡器”章节。

如果应用受到温度变化影响，则这可能也会影响到 RC 振荡器的速度。用户可通过寄存器对 MRC 频率进行微调。

MRC 可通过 CMU_MRCCR 控制寄存器中的 MRCSTP 位打开或关闭。

MRC 时钟还可作为备份时钟源使用，以防 XTAL 晶振发生故障。请参见“检测到 XTAL 故障检测动作”。



注意

使用 TRNG 模块时禁止关闭 MRC 时钟。

5.3.4.3 LRC 时钟

LRC 时钟信号由内部 32.768kHz 低速振荡器生成，可直接用作系统时钟。LRC 可作为低功耗时钟源在掉电模式和停止模式下保持运行，供 Timer0、Timer2 和 KEYSKAN 使用。

LRC 振荡器的启动速度快。

频率校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此会对每个器件进行出厂校准，保证精度请参见《数据手册》“电气特性-工作条件-内部时钟源特性-内部低速（LRC）振荡器”章节。

如果应用受到电压或温度变化影响，则这可能也会影响到 RC 振荡器的速度。用户可通过寄存器对 LRC 频率进行微调。

LRC 可通过 CMU_LRCCR 控制寄存器中的 LRCSTP 位打开或关闭。

5.3.4.4 SWDTLRC 时钟

SWDTLRC 时钟信号由内部 10kHz 低速振荡器生成，SWDT 专属时钟。SWDT 已通过 ICG 设置的方式启动，则 SWDT 专用内部低速振荡器将强制打开且不可禁止。

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此会对每个器件进行出厂校准，保证精度请参见《数据手册》“电气特性-工作条件-内部时钟源特性-SWDT 专用内部低速（SWDTLRC）振荡器”章节。

5.3.4.5 RTCLRC 时钟

RTCLRC 时钟信号由内部 32.768kHz 低速振荡器生成。RTCLRC 可作为低功耗时钟源在掉电模式和停止模式下保持运行，供 RTC、WKTM 使用。

RTCLRC 振荡器的启动速度快。

频率校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同，因此会对每个器件进行出厂校准，保证精度请参见《数据手册》“电气特性-工作条件-内部时钟源特性-RTC 专用内部低速 (RTCLRC) 振荡器”章节。

如果应用受到电压或温度变化影响，则这可能也会影响到 RC 振荡器的速度。用户可通过寄存器对 RTCLRC 频率进行微调。

RTCLRC 可通过 RTC_CR3 控制寄存器中的 LRCEN 位打开或关闭。

5.3.5 PLL 时钟

本产品配备了两个 PLL (PLLH 和 PLLA)：

- PLLH 和 PLLA 均可选择 XTAL 或 HRC 振荡器作为输入时钟，由 CMU_PLLHCFGR.PLLSRC 位统一配置；
- PLLH 有三个输出时钟 PLLHP、PLLHQ、PLLHR，分频系数可独立配置；
- PLLA 有三个输出时钟 PLLAP、PLLAQ、PLLAR，分频系数可独立配置；
- PLLHP 时钟可作为系统时钟；
- PLLHQ、PLLHR、PLLAP、PLLAQ、PLLAR 时钟可用于生成 USB、ADC、DAC、I2S、CAN、MCAN 时钟。



注意

1. 在 HRC 或 XTAL 振荡器稳定后，再对 PLL 进行配置。
2. 由于在 PLL 使能后 PLL 配置参数便不可更改，所以建议先对 PLL 进行配置，然后再使能。
3. 当进入掉电和停止模式后，PLL 将由硬件禁止。

5.3.6 时钟切换步骤

在系统复位后，默认系统时钟为 MRC。通过设定寄存器 CMU_CKSW 切换时钟源，切换步骤请参见“[时钟源切换](#)”。只有在目标时钟源已稳定的状态下，才可以从一个时钟源切换到另一个时钟源。

时钟切换时需要正确配置 Flash/SRAM 的等待周期，防止系统时钟频率大于 Flash/SRAM 的最大动作频率。详细配置请参见《嵌入式 FLASH (EFM)》的“CPU 时钟和 Flash 读取时间之间的关系”章节、《内置 SRAM (SRAM)》章节。

时钟源切换

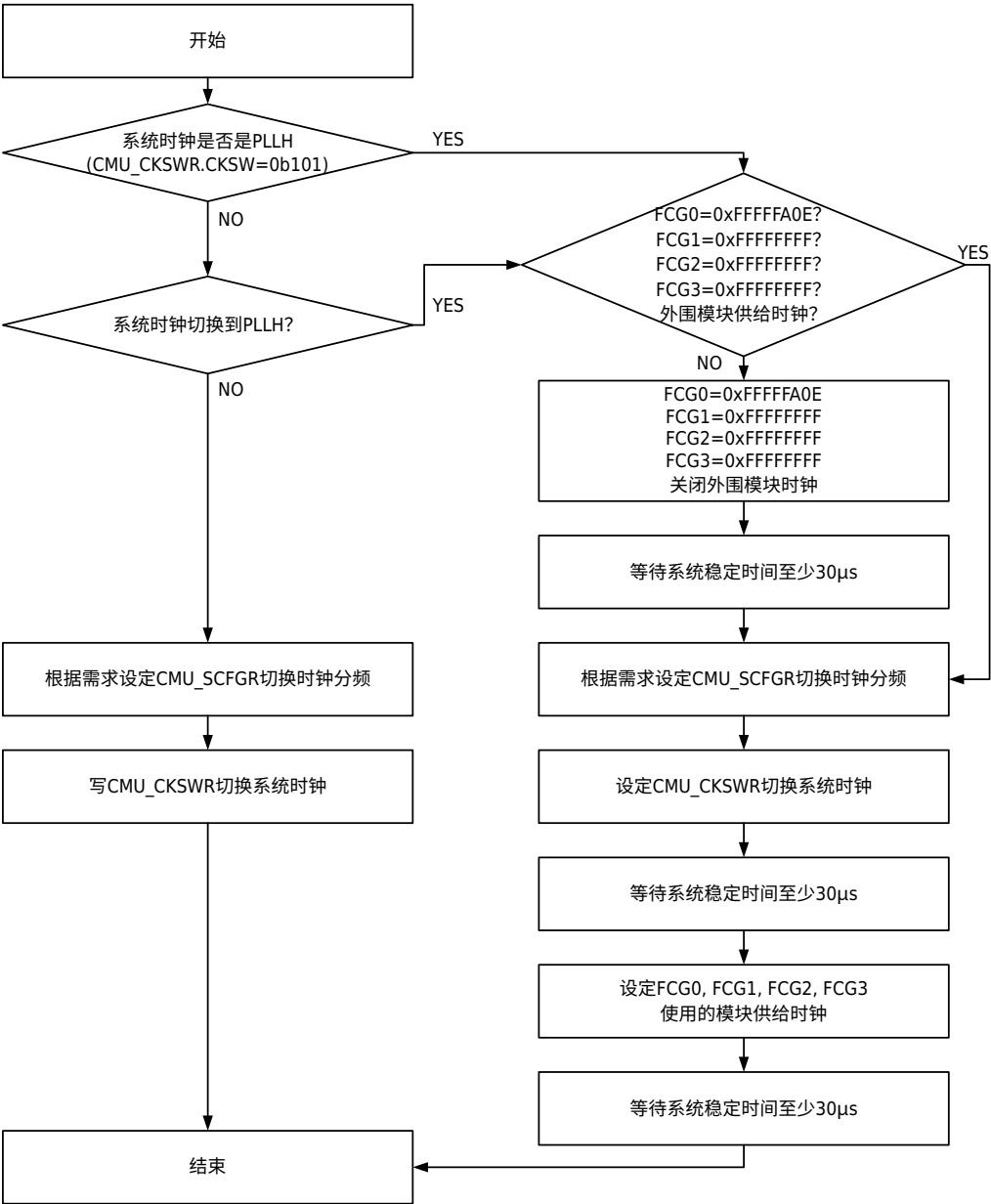


图 5-8 时钟源切换流程图

时钟分频切换

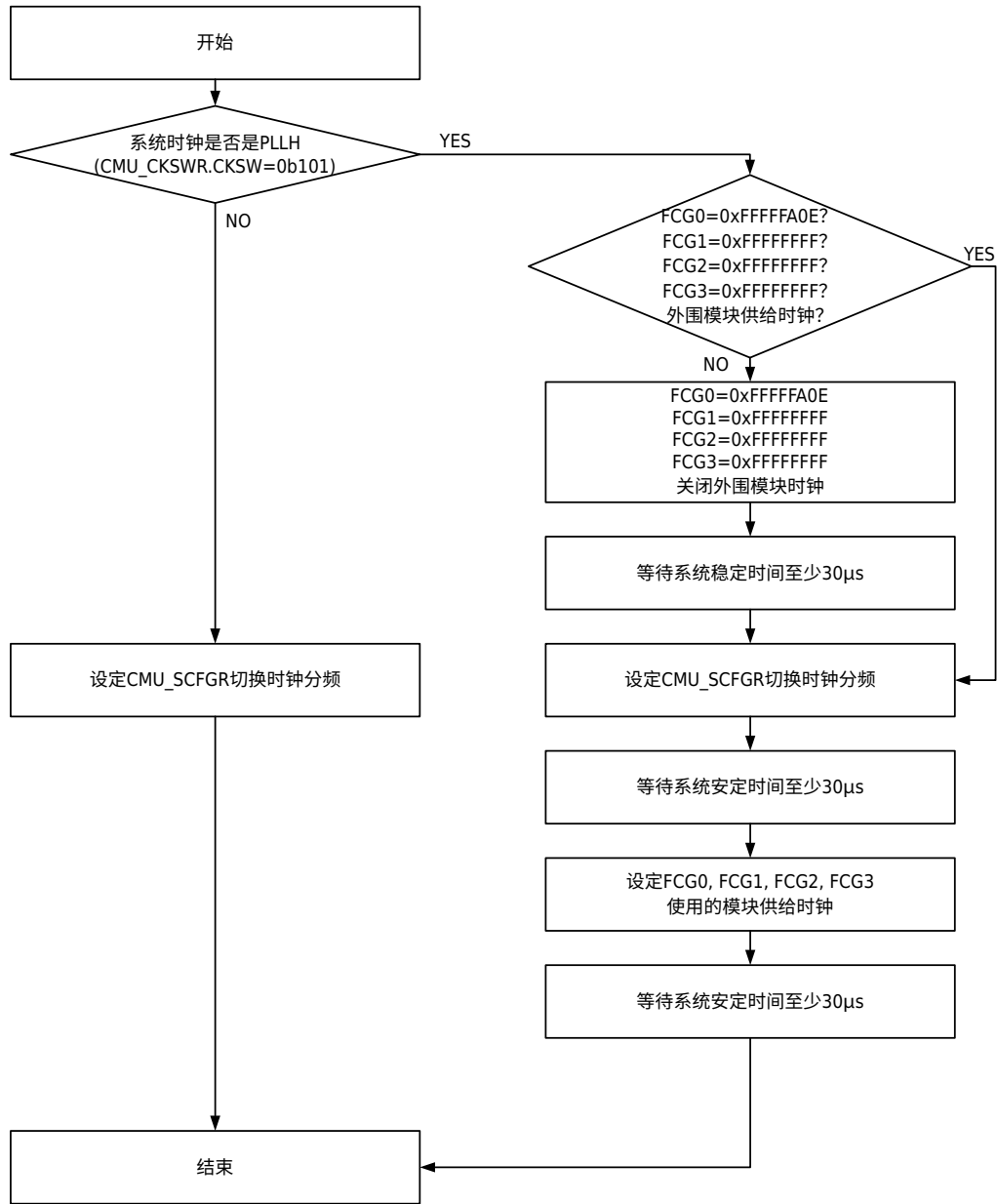


图 5-9 时钟分频切换流程图

5.3.7 时钟输出功能

共有两路时钟输出：

● MCO_1

用户可通过可配置的预分配器（1、2、4、8、...、128 分频）向 MCO_1 引脚输出不同的时钟源：

- ▶ HRC 时钟
- ▶ MRC 时钟
- ▶ LRC 时钟
- ▶ XTAL 时钟
- ▶ XTAL32 时钟
- ▶ PLLHP/PLLHQ 时钟
- ▶ PLLAP/PLLAQ/PLLAR 时钟

► 系统时钟

所需的时钟源通过 CMU_MCO1CFGR.MCO1SEL 位选择。

● MCO_2

用户可通过可配置的预分配器 (1、2、4、8、...、128 分频) 向 MCO_2 引脚输出不同的时钟源：

- HRC 时钟
- MRC 时钟
- LRC 时钟
- XTAL 时钟
- XTAL32 时钟
- PLLHP/PLLHQ 时钟
- PLLAP/PLLAQ/PLLAR 时钟
- 系统时钟

所需的时钟源通过 CMU_MCO2CFGR.MCO2SEL 位选择。

MCO_1/MCO_2 输出时钟不得超过 100MHz (最大 I/O 速度)。

5.3.8 时钟频率测量

5.3.8.1 时钟频率测量功能

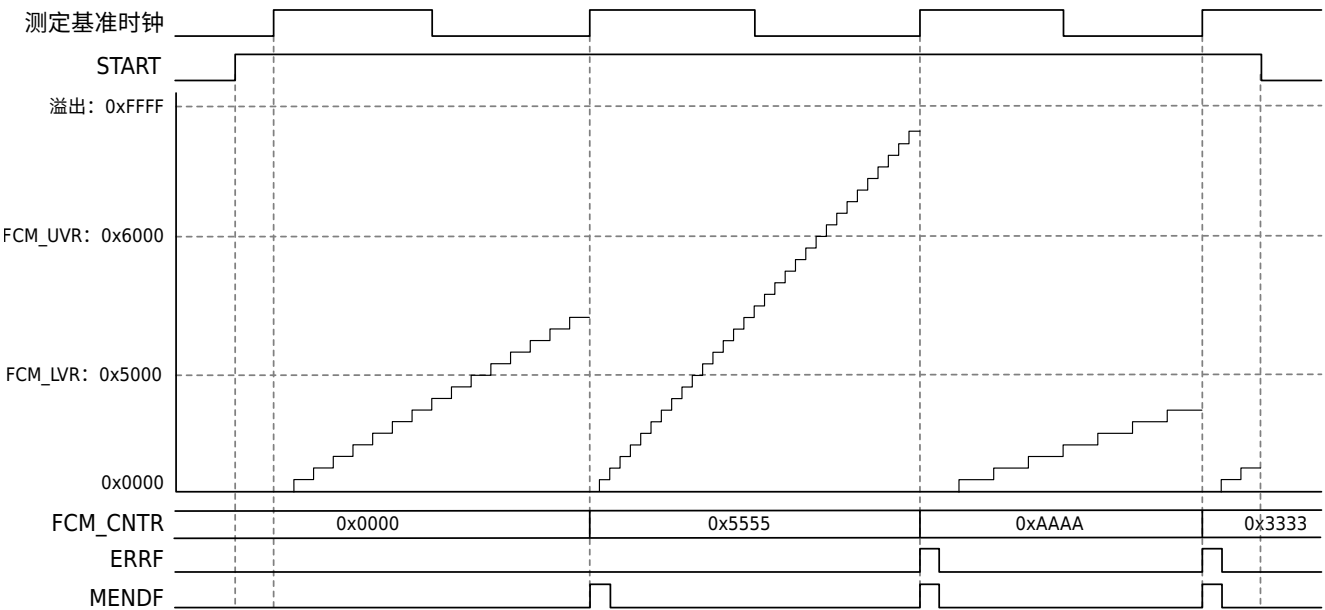


图 5-10 时钟频率测量时序图

- 用 FCM_MCCR/FCM_RCCR 选择基准时钟被测量时钟，时钟的分频以及选择基准时钟的有效边沿。
- FCM_STR 的 START 位写 1 后，检测到 EDGES 位选择的有效边沿，计数器就开始递增计数。
- 在检测到基准时钟下一个 EDGES 位选择的有效边沿时，将计数器的值保存到 FCM_CNTR 寄存器，并且与 FCM_LVR/FCM_UVR 的设定值进行比较。当 $FCM\_LVR \leq FCM\_CNTR \leq FCM\_UVR$ 时，被测时钟频率测量正常。当 $FCM\_LVR > FCM\_CNTR$ 或者 $FCM\_CNTR > FCM\_UVR$ 时，被测时钟频率异常，根据 ERRINTRS/ERRRE/ERRIE 设定可以发生中断或者复位。
- FCM_STR 的 START 位写 0 后，计数器计数停止并清零。
- 计数器共分为 4 组计数，各计数器之间相互独立不影响。
- 当 $FCM\_LVR > FCM\_CNTR$ 或者 $FCM\_CNTR > FCM\_UVR$ 时，被测时钟频率异常，其异常源为 FCM_FERR。

5.3.8.2 数字滤波功能

外部管脚输入参考时钟 FCMREF 具有数字滤波功能。数字滤波器功能根据 DNFS 位选择的采样时钟进行 3 次采样，3 次采样的电平相同时，将此电平送到内部。

数字滤波功能可以设定数字滤波功能有效无效以及采样时钟。

5.3.8.3 中断/复位功能

时钟频率测量电路支持三种中断请求：

- 频率异常中断
- 频率测量结束中断
- 计数器溢出中断

时钟频率测量电路支持一种复位请求：

- 频率异常复位

5.4 寄存器

5.4.1 寄存器总表

FCM 基地址: 0x40048400

表 5-3 时钟频率测量 (FCM) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00+0x40*(n-1)	FCMn_LVR	FCM 下限比较值寄存器 (n=1~4)	32
0x04+0x40*(n-1)	FCMn_UVR	FCM 上限比较值寄存器 (n=1~4)	32
0x08+0x40*(n-1)	FCMn_CNTR	FCM 计数器值寄存器 (n=1~4)	32
0x0C	FCM_STR	FCM 开始停止寄存器	32
0x10+0x40*(n-1)	FCMn_MCCR	FCM 测量对象控制寄存器 (n=1~4)	32
0x14+0x40*(n-1)	FCMn_RCCR	FCM 测量基准控制寄存器 (n=1~4)	32
0x18+0x40*(n-1)	FCMn_RIER	FCM 中断复位控制寄存器 (n=1~4)	32
0x1C	FCM_SR	FCM 标志寄存器	32
0x20+0x40*(n-1)	FCMn_CLR	FCM 标志位清除寄存器 (n=1~4)	32

CMU 基地址: 0x40048000

表 5-4 时钟控制器 (CMU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x4400	CMU_XTAL32CR	CMU XTAL32 控制寄存器	8
0x4404	CMU_XTAL32CFGR	CMU XTAL32 配置寄存器	8
0x4414	CMU_XTAL32NFR	CMU XTAL32 滤波寄存器	8
0x441C	CMU_LRCCR	CMU LRC 控制寄存器	8
0x4424	CMU_LRCTRM	CMU LRC 校准寄存器	8
0x442C	CMU_RTCLRCTRM	CMU RTCLRC 校准寄存器	8
0x4C78	CMU_XTALCFGR	CMU XTAL 配置寄存器	8
0x0018	CMU_XTALDIVR	CMU XTAL 小数分频配置寄存器	32
0x001C	CMU_XTALDIVCR	CMU XTAL 小数分频控制寄存器	32
0xC0A2	CMU_XTALSTBCR	CMU XTAL 稳定配置寄存器	8
0xC032	CMU_XTALCR	CMU XTAL 控制寄存器	8
0xC040	CMU_XTALSTDCR	CMU XTAL 振荡故障控制寄存器	8
0xC041	CMU_XTALSTDSR	CMU XTAL 振荡故障状态寄存器	8
0xC036	CMU_HRCCR	CMU HRC 控制寄存器	8

偏移地址	寄存器	描述	位宽
0xC062	CMU_HRCTRM	CMU HRC 校准寄存器	8
0xC038	CMU_MRCCR	CMU MRC 控制寄存器	8
0xC061	CMU_MRCTRM	CMU MRC 校准寄存器	8
0xC100	CMU_PLLHCFGR	CMU PLLH 配置寄存器	32
0xC02A	CMU_PLLHCR	CMU PLLH 控制寄存器	8
0xC104	CMU_PLLACFGR	CMU PLLA 配置寄存器	32
0xC02E	CMU_PLLACR	CMU PLLA 控制寄存器	8
0xC03C	CMU_OSCSTBSR	CMU 时钟源稳定状态寄存器	8
0xC026	CMU_CKSWR	CMU 系统时钟源切换寄存器	8
0xC020	CMU_SCFGR	CMU 时钟分频配置寄存器	32
0xC024	CMU_USBCKCFGR	CMU USB 时钟配置寄存器	8
0xC018	CMU_CANCKCFGR	CMU CAN 时钟配置寄存器	16
0xC010	CMU_PERICKSEL	CMU ADC/DAC 时钟配置寄存器	16
0xC03F	CMU_TPIUCKCFGR	CMU 调试时钟配置寄存器	8
0xC012	CMU_I2SCKCFGR	CMU I2S 时钟配置寄存器	16
0xC03D	CMU_MCO1CFGR	CMU MCO1 时钟输出配置寄存器	8
0xC03E	CMU_MCO2CFGR	CMU MCO2 时钟输出配置寄存器	8

5.4.2 CMU XTAL 配置寄存器 (CMU_XTALCFGR)

Offset	Bit Position							
0x4C78	7	6	5	4	3	2	1	0
Reset	0x80							
Name	Reserved	XTALMS	XTALDRV		Reserved			
Access		RW	RW					

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	读出为 1，写入时必须写 1	-
6	XTALMS	XTAL 模式选择位	0b0：振荡器模式 0b1：外部时钟输入模式	RW
5:4	XTALDRV	XTAL 驱动能力选择	0b00：高驱动能力（20MHz<晶振≤25MHz） 0b01：中驱动能力（16MHz<晶振≤20MHz） 0b10：低驱动能力（8MHz<晶振≤16MHz） 0b11：超低驱动能力（4MHz≤晶振≤8MHz）	RW
3:0	Reserved	保留	-	-

5.4.3 CMU XTAL 稳定配置寄存器 (CMU_XTALSTBCR)

Offset	Bit Position							
0xC0A2	7	6	5	4	3	2	1	0
Reset	0x05							
Name	Reserved				XTALSTB			
Access					RW			

位/位域	标记	位名	功能描述	读写
7:4	Reserved	保留	-	-
3:0	XTALSTB	XTAL 稳定时间选择	● 0b0001: 稳定计数器 35 周期 ● 0b0010: 稳定计数器 67 周期 ● 0b0011: 稳定计数器 131 周期 ● 0b0100: 稳定计数器 259 周期 ● 0b0101: 稳定计数器 547 周期 ● 0b0110: 稳定计数器 1059 周期 ● 0b0111: 稳定计数器 2147 周期 ● 0b1000: 稳定计数器 4291 周期 ● 0b1001: 稳定计数器 8163 周期 ● 其他: 禁止设置  说明 稳定计数器的一个计数周期=LRC 周期/8 仅支持在 CMU_XTALCR.XTALSTP=1 且 CMU_OSCSTBSR.XTALSTBF=0 的状态下配置此寄存器。	RW

5.4.4 CMU XTAL 小数分频配置寄存器 (CMU_XTALDIVR)

Offset	Bit Position																																					
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x03D09040																																					
Name	Reserved				RW NUMER																Reserved				RW DENOM													
Access					RW																				RW													

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:12	NUMER	XTAL 小数分频分子设定值	XTAL 小数分频分子设定值，使用方法请参见“ 外部高速振荡器小数分频 ”	RW
11	Reserved	保留	-	-
10:0	DENOM	XTAL 小数分频分母设定值	XTAL 小数分频分母设定值，使用方法请参见“ 外部高速振荡器小数分频 ”  说明 请将此值设定为 2^N。例如，0x100 表示分母为 256；0x80 表示分母为 128。	RW

5.4.5 CMU XTAL 小数分频控制寄存器 (CMU_XTALDIVCR)

Offset	Bit Position																															
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															FRADIVEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	FRADIVEN	XTAL 小数分频启停控制	0b0: XTAL 小数分频停止 0b1: XTAL 小数分频开始	RW

5.4.6 CMU XTAL 控制寄存器 (CMU_XTALCR)

Offset	Bit Position							
0xC032	7	6	5	4	3	2	1	0
Reset	0x01							
Name	Reserved							XTALSTP
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	XTALSTP	XTAL 振荡器启停控制	0b0: XTAL 振荡器振荡 0b1: XTAL 振荡器停止	RW

 **注意**

1. XTAL 选作系统时钟或者 PLLH/PLLA 时钟源时，禁止通过 XTALSTP 写 1 停止 XTAL 振荡器。

2. 软件设定 XTAL 振荡器振荡，通过 XTALSTBF 位确认 XTAL 振荡器稳定后，才可以进入停止模式、掉电模式或者软件设定 XTAL 振荡器停止。

3. 软件设定 XTAL 振荡器停止，通过 XTALSTBF 位确认 XTAL 振荡器停止后，才可以进入停止模式、掉电模式或者再次启动 XTAL 振荡器。

5.4.7 CMU XTAL 振荡故障控制寄存器 (CMU_XTALSTDCR)

Offset	Bit Position							
0xC040	7	6	5	4	3	2	1	0
Reset	0x00							
Name	XTALSTDE	Reserved				XTALSTDRIS	XTALSTDRE	XTALSTDIE
Access	RW					RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	XTALSTDE	XTAL 振荡故障检测功能允许	0b0: 禁止 XTAL 振荡故障检测 0b1: 允许 XTAL 振荡故障检测  说明 振荡器故障检测是检测外部因素导致的振荡器异常振荡，在进入停止模式或者掉电模式前，请将振荡器振荡故障检测功能无效。	RW
6:3	Reserved	保留	-	-
2	XTALSTDRIS	XTAL 振荡故障复位中断选择	0b0: XTAL 振荡故障产生中断 0b1: XTAL 振荡故障产生复位  说明 PLLH、PLLA 选择 XTAL 时钟作为输入源时，只能选择 XTAL 振荡故障产生复位功能。	RW
1	XTALSTDRE	XTAL 振荡故障复位允许	0b0: 禁止 XTAL 振荡故障复位 0b1: 允许 XTAL 振荡故障复位	RW
0	XTALSTDIE	XTAL 振荡故障中断允许	0b0: 禁止 XTAL 振荡故障中断 0b1: 允许 XTAL 振荡故障中断  说明 通过 EMB 将 Timer6/Timer4 的 PWM 输出置成 Hi-Z 输出，XTALSTDIE 位需要设置成 1。	RW

**注意**

XTAL 选作系统时钟或者 PLLH/PLLA 时钟源时，禁止通过 XTALSTP 写 1 停止 XTAL 振荡器。

5.4.8 CMU XTAL 振荡故障状态寄存器 (CMU_XTALSTDSR)

Offset	Bit Position							
0xC041	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							XTALSTDF
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	XTALSTDF	XTAL 振荡故障状态位	0b0: 未检测到 XTAL 振荡故障 0b1: 检测到 XTAL 振荡故障 置位条件: - XTALSTDE=1 的条件下, XTAL 振荡故障 清零条件: - 系统时钟选择 XTAL 以外时钟时, 读 1 写 0	RW

5.4.9 CMU XTAL32 配置寄存器 (CMU_XTAL32CFGR)

Offset	Bit Position							
0x4404	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved					XTAL32DRV		
Access						RW		

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2:0	XTAL32DRV	XTAL32 驱动能力选择	0b000: 中驱动能力 0b001: 高驱动能力 0b010: 低驱动能力 - 其他: 禁止设置  说明 使用方法请参见《数据手册》“电气特性-工作条件-外部时钟源特性-晶振/陶瓷谐振器产生的低速外部时钟”。	RW

5.4.10 CMU XTAL32 滤波寄存器 (CMU_XTAL32NFR)

Offset	Bit Position							
0x4414	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved						XTAL32NF	
Access							RW	

位/位域	标记	位名	功能描述	读写
7:2	Reserved	保留	-	-
1:0	XTAL32NF	XTAL32 振荡器滤波选择	0b00: RUN 模式/停止模式/掉电模式, XTAL32 的 3μs 滤波有效 0b01: RUN 模式 XTAL32 的 3μs 滤波有效, 停止模式或掉电模式 XTAL32 的 3μs 滤波无效 0b10: 禁止设置 0b11: RUN 模式/停止模式/掉电模式, XTAL32 的 3μs 滤波无效	RW

5.4.11 CMU XTAL32 控制寄存器 (CMU_XTAL32CR)

Offset	Bit Position							
0x4400	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved						XTAL32STP	
Access							RW	

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	XTAL32STP	XTAL32 振荡器开启停止位	0b0: XTAL32 振荡器振荡 0b1: XTAL32 振荡器停止	RW



注意

1. XTAL32 选作系统时钟源时, 禁止通过 XTAL32STP 写 1 停止 XTAL32 振荡器。
2. 软件设定 XTAL32 振荡, 等待 5 个 XTAL32 周期后, 才可以再次停止 XTAL32。
3. 软件设定 XTAL32 停止, 等待 5 个 XTAL32 周期后, 才可以再次启动 XTAL32。

5.4.12 CMU HRC 校准寄存器 (CMU_HRCTRM)

Offset	Bit Position							
0xC062	7	6	5	4	3	2	1	0
Reset	0x00							
Name	HRCTRM							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	HRCTRM	HRC 频率校准	● 0b10000000: -128 ● 0b10000001: -127 ● ● 0b11111111: -1 ● 0b00000000: 中心 Code ● 0b00000001: +1 ● ● 0b01111110: +126 ● 0b01111111: +127	RW

 **说明**

频率校准需在 HRC 频率保证范围内。

5.4.13 CMU HRC 控制寄存器 (CMU_HRCCR)

Offset	Bit Position							
0xC036	7	6	5	4	3	2	1	0
Reset	由 ICG1.HRCSTOP 值决定							
Name	Reserved							HRCSTP
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	HRCSTP	HRC 振荡器开启 停止位	● 0b0: HRC 振荡器振荡 ● 0b1: HRC 振荡器停止  说明 根据 ICG1.HRCSTOP 的配置，复位后 HRC 振荡或停止。	RW

 **注意**

1. HRC 选作系统时钟源或 PLLH/PLLA 时钟源时，禁止通过 HRCSTP 写 1 停止 HRC 时钟。

2. 软件设定 HRC 振荡，通过 HRCSTBF 位确认 HRC 稳定后，才可以进入停止模式、掉电模式或者停止 HRC。

3. 软件设定 HRC 停止，通过 HRCSTBF 位确认 HRC 停止后，才可以进入停止模式、掉电模式或者再次启动 HRC。

5.4.14 CMU MRC 校准寄存器 (CMU_MRCTRM)

Offset	Bit Position							
0xC061	7	6	5	4	3	2	1	0
Reset	0x00							
Name	MRCTRM							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	MRCTRM	MRC 频率校准	● 0b10000000: -128 ● 0b10000001: -127 ● ● 0b11111111: -1 ● 0b00000000: 中心 Code ● 0b00000001: +1 ● ● 0b01111110: +126 ● 0b01111111: +127	RW

 说明

频率校准需在 MRC 频率保证范围内。

5.4.15 CMU MRC 控制寄存器 (CMU_MRCCR)

Offset	Bit Position							
0xC038	7	6	5	4	3	2	1	0
Reset	0x80							
Name	Reserved							MRCSTP
Access								RW

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	读出为 1，写入时必须写 1	-
6:1	Reserved	保留	-	-
0	MRCSTP	MRC 振荡器开启 停止位	● 0b0：MRC 振荡器振荡 ● 0b1：MRC 振荡器停止  说明 ● XTAL 振荡故障功能有效时，本位同时清零，MRC 振荡。 ● PWC_STPMCR.CKSMRC 位为 1 时的停止模式唤醒动作，在 MRC 振荡器处于振荡状态下设定。	RW

 **注意**

1. MRC 选作系统时钟源时，禁止通过 MRCSTP 写 1 停止 MRC 时钟。

2. 软件设定 MRC 振荡，等待 5 个 MRC 周期后，才可以进入停止模式、掉电模式或者停止 MRC。

3. 软件设定 MRC 停止，等待 5 个 MRC 周期后，才可以进入停止模式、掉电模式或者再次启动 MRC。

4. MRC 用作 RTC 校准时钟。RTC 未初始化时，MRC 有振荡的可能性。RTC 校准功能有效时，无视 MRCSTP 位设定，MRC 振荡。

5.4.16 CMU LRC 校准寄存器 (CMU_LRCTRM)

Offset	Bit Position							
0x4424	7	6	5	4	3	2	1	0
Reset	0x00							
Name	LRCTRM							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	LRCTRM	LRC 频率校准	● 0b10000000: -128 ● 0b10000001: -127 ● ● 0b11111111: -1 ● 0b00000000: 中心 Code ● 0b00000001: +1 ● ● 0b01111110: +126 ● 0b01111111: +127	RW



说明

频率校准需在 LRC 频率保证范围内。

5.4.17 CMU LRC 控制寄存器 (CMU_LRCCR)

Offset	Bit Position							
0x441C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							LRCSTP
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	LRCSTP	LRC 振荡器开启 停止位	● 0b0: LRC 振荡器振荡 ● 0b1: LRC 振荡器停止	RW



注意

- LRC 选作系统时钟源时，禁止通过 LRCSTP 写 1 停止 LRC 时钟。
- 软件设定 LRC 振荡，等待 5 个 LRC 周期后，才可以进入停止模式、掉电模式或者停止 LRC。
- 软件设定 LRC 停止，等待 5 个 LRC 周期后，才可以进入停止模式、掉电模式或者再次启动 LRC。
- 等待 XTAL 振荡器、HRC、PLLH、PLLA 时钟稳定时，LRCSTP 位设定无视，LRC 强制振荡。

5.4.18 CMU RTCLRC 校准寄存器 (CMU_RTCLRCTRM)

Offset	Bit Position							
0x442C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	RTCLRCTRM							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	RTCLRCTRM	RTCLRC 频率校准	● 0b10000000: -128 ● 0b10000001: -127 ● ● 0b11111111: -1 ● 0b00000000: 中心 Code ● 0b00000001: +1 ● ● 0b01111110: +126 ● 0b01111111: +127	RW

 **说明**
频率校准需在 RTCLRC 频率保证范围内。

5.4.19 CMU PLLH 配置寄存器 (CMU_PLLHCFGR)

Offset	Bit Position																															
0xC100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x11101300																															
Name	PLLHP				PLLHQ				PLLHR				Reserved				PLLHN				PLLSRC				Reserved				PLLHM			
Access	RW				RW				RW								RW				RW								RW			

位/位域	标记	位名	功能描述	读写
31:28	PLLHP	PLLHP 时钟分频系数	PLLHP 输出时钟频率=PLLH 的 VCO 频率/PLLHP ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLH 停止条件下写 PLLHP，设置 PLLHP 时钟的频率。	RW
27:24	PLLHQ	PLLHQ 时钟分频系数	PLLHQ 输出时钟频率=PLLH 的 VCO 频率/PLLHQ ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLH 停止条件下写 PLLHQ，设置 PLLHQ 时钟的频率。	RW
23:20	PLLHR	PLLHR 时钟分频系数	PLLHR 输出时钟频率=PLLH 的 VCO 频率/PLLHR ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLH 停止条件下写 PLLHR，设置 PLLHR 时钟的频率。	RW
19:16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15:8	PLLHN	PLLH 倍频系数	PLLH 的 VCO 频率=PLLH 的 VCO 输入频率*PLLHN ● 0b00011000: 25 ● 0b00011001: 26 ● 0b00011010: 27 ● ● 0b10010100: 149 ● 0b10010101: 150 ● 其他: 禁止设置  说明 用于控制 PLLH 的 VCO 的倍频系数，在 PLLH 停止条件下写 PLLHN。确保 PLLH 的 VCO 频率介于 600MHz 到 1200MHz 之间。	RW
7	PLLSRC	PLLH/PLLA 输入时钟源选择	● 0b0: 选择外部高速振荡器 ● 0b1: 选择内部高速振荡器	RW
6:2	<i>Reserved</i>	<i>保留</i>	-	-
1:0	PLLHM	PLLH 输入时钟分频系数	● 0b00: 1 分频 ● 0b01: 2 分频 ● 0b10: 3 分频 ● 0b11: 4 分频  说明 用于在 PLLH 的 VCO 之前对 PLLH 输入时钟进行分频，在 PLLH 停止条件下写 PLLHM。确保 PLLH 的 PFD 输入时钟频率介于 8MHz 到 25MHz 之间。	RW

5.4.20 CMU PLLH 控制寄存器 (CMU_PLLHCR)

Offset	Bit Position							
0xC02A	7	6	5	4	3	2	1	0
Reset	0x01							
Name	Reserved							PLLHOFF
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	PLLHOFF	PLLH 使能	● 0b0: PLLH 动作开始 ● 0b1: PLLH 停止  说明 如果 PLLH 时钟用作系统时钟时，请不要将此位设成 1。	RW

 **注意**

- PLLH 选作系统时钟源时，禁止通过 PLLHOFF 写 1 停止 PLLH 时钟。
- 软件设定 PLLH 动作开始，通过 PLLHSTBF 位确认 PLLH 稳定后，才可以进入停止模式，掉电模式或者软件设定停止 PLLH。
- 软件设定 PLLH 停止，通过 PLLHSTBF 位确认 PLLH 停止后，才可以进入停止模式，掉电模式或者再次启动 PLLH。
- PLLH 选择 XTAL 振荡器作时钟源时，通过 XTALSTBF 位确认 XTAL 振荡器稳定后，才可以设定 PLLH 动作开始。PLLH 选择 HRC 作时钟源时，通过 HRCSTBF 位确认 HRC 稳定后，才可以设定 PLLH 动作开始。

5.4.21 CMU PLLA 配置寄存器 (CMU_PLLACFGR)

Offset	Bit Position																															
0xC104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x11101300																															
Name	PLLAP				PLLAQ				PLLAR				Reserved				PLLAN				Reserved				PLLAM							
Access	RW				RW				RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:28	PLLAP	PLLAP 时钟分频系数	PLLAP 输出时钟频率=PLLA 的 VCO 频率/PLLAP ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLA 停止条件下写 PLLAP，设置 PLLAP 时钟的频率。	RW
27:24	PLLAQ	PLLAQ 时钟分频系数	PLLAQ 输出时钟频率=PLLA 的 VCO 频率/PLLAQ ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLA 停止条件下写 PLLAQ，设置 PLLAQ 时钟的频率。	RW
23:20	PLLAR	PLLAR 时钟分频系数	PLLAR 输出时钟频率=PLLA 的 VCO 频率/PLLAR ● 0b0000: 禁止设置 ● 0b0001: 2 分频 ● 0b0010: 3 分频 ● ● 0b1110: 15 分频 ● 0b1111: 16 分频  说明 在 PLLA 停止条件下写 PLLAR，设置 PLLAR 时钟的频率。	RW
19:17	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
16:8	PLLAN	PLLA 倍频系数	PLLA 的 VCO 频率=PLLA 的 VCO 输入频率*PLLAN ● 0b000010011: 20 ● 0b000010100: 21 ● 0b000010101: 22 ● ● 0b111011110: 479 ● 0b111011111: 480 ● 其他: 禁止设置  说明 用于控制 PLLA 的 VCO 的倍频系数，在 PLLA 停止条件下写 PLLAN。确保 PLLA 的 VCO 频率介于 240MHz 到 480Hz 之间。	RW
7:5	Reserved	保留	-	-
4:0	PLLAM	PLLA 输入时钟分频系数	● 0b00000: 1 分频 ● 0b00001: 2 分频 ● 0b00010: 3 分频 ● ● 0b11000: 25 分频 ● 其他: 禁止设置  说明 用于在 PLLA 的 VCO 之前对 PLLA 输入时钟进行分频，在 PLLA 停止条件下写 PLLAM。确保 PLLA 的 PFD 输入时钟频率介于 1MHz 到 25MHz 之间。	RW

5.4.22 CMU PLLA 控制寄存器 (CMU_PLLACR)

Offset	Bit Position							
0xC02E	7	6	5	4	3	2	1	0
Reset	0x01							
Name	Reserved							PLLAOFF
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	PLLAOFF	PLLA 使能	0b0: PLLA 动作开始 0b1: PLLA 停止  说明 如果 PLLA 时钟用作系统时钟时，请不要将此位设成 1。	RW

 **注意**

- PLLA 选作 I2S/ ADC/DAC/USB/CAN/MCAN 时钟源时，禁止通过 PLLAOFF 写 1 停止 PLLA 时钟。
- 软件设定 PLLA 动作开始，通过 PLLASTBF 位确认 PLLA 稳定后，才可以进入停止模式，掉电模式或者软件设定停止 PLLA。
- 软件设定 PLLA 停止，通过 PLLASTBF 位确认 PLLA 停止后，才可以进入停止模式，掉电模式或者再次启动 PLLA。
- PLLA 选择 XTAL 振荡器作时钟源时，通过 XTALSTBF 位确认 XTAL 振荡器稳定后，才可以设定 PLLA 动作开始。PLL A 选择 HRC 作时钟源时，通过 HRCSTBF 位确认 HRC 稳定后，才可以设定 PLLA 动作开始。

5.4.23 CMU 时钟源稳定状态寄存器 (CMU_OSCSTBSR)

Offset	Bit Position							
0xC03C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	PLLASTBF	PLLHSTBF	Reserved	XTALSTBF	Reserved		HRCSTBF
Access		RO	RO		RO			RO

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6	PLLASTBF	PLLA 稳定标志位	0b0: PLLA 停止或者未稳定 0b1: PLLA 稳定	RO
5	PLLHSTBF	PLLH 稳定标志位	0b0: PLLH 停止或者未稳定 0b1: PLLH 稳定	RO
4	Reserved	保留	-	-
3	XTALSTBF	XTAL 稳定标志位	0b0: XTAL 停止或者未稳定 0b1: XTAL 稳定	RO
2:1	Reserved	保留	-	-
0	HRCSTBF	HRC 稳定标志位	0b0: HRC 停止或者未稳定 0b1: HRC 稳定	RO

5.4.24 CMU 系统时钟源切换寄存器 (CMU_CKSWR)

Offset	Bit Position							
0xC026	7	6	5	4	3	2	1	0
Reset	0x01							
Name	Reserved					CKSW		
Access						RW		

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2:0	CKSW	系统时钟源切换	● 0b000: 选择 HRC 时钟作为系统时钟 ● 0b001: 选择 MRC 时钟作为系统时钟 ● 0b010: 选择 LRC 时钟作为系统时钟 ● 0b011: 选择 XTAL 时钟作为系统时钟 ● 0b100: 选择 XTAL32 时钟作为系统时钟 ● 0b101: 选择 PLLH 作为系统时钟 ● 其他: 禁止设置 说明 ● 切换的目标时钟源, 需保证在时钟稳定状态。操作步骤请参见“时钟源切换”。 ● PWC_STPMCRC.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, 系统时钟源选择 MRC 时钟。	RW

5.4.25 CMU 时钟分频配置寄存器 (CMU_SCFGR)

Offset	Bit Position																																					
0xC020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved				HCLKS				Reserved	EXCKS				Reserved	PCLK4S				Reserved	PCLK3S				Reserved	PCLK2S				Reserved	PCLK1S				Reserved	PCLK0S			
Access					RW					RW					RW					RW					RW					RW					RW			

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26:24	HCLKS	HCLK 时钟分频选择位	0b000: 系统时钟的 1 分频 0b001: 系统时钟的 2 分频 0b010: 系统时钟的 4 分频 0b011: 系统时钟的 8 分频 0b100: 系统时钟的 16 分频 0b101: 系统时钟的 32 分频 0b110: 系统时钟的 64 分频 0b111: 禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, HCLK 为系统时钟的 1 分频。	RW
23	Reserved	保留	-	-
22:20	EXCKS	EXMC 时钟分频选择位	0b000: 系统时钟的 1 分频 0b001: 系统时钟的 2 分频 0b010: 系统时钟的 4 分频 0b011: 系统时钟的 8 分频 0b100: 系统时钟的 16 分频 0b101: 系统时钟的 32 分频 0b110: 系统时钟的 64 分频 0b111: 禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, EXCLK 为系统时钟的 1 分频。	RW
19	Reserved	保留	-	-
18:16	PCLK4S	PCLK4 时钟分频选择位	0b000: 系统时钟的 1 分频 0b001: 系统时钟的 2 分频 0b010: 系统时钟的 4 分频 0b011: 系统时钟的 8 分频 0b100: 系统时钟的 16 分频 0b101: 系统时钟的 32 分频 0b110: 系统时钟的 64 分频 0b111: 禁止设置	RW

位/位域	标记	位名	功能描述	读写
18:16	PCLK4S	PCLK4 时钟分频选择位	 说明 PWC_STPMCR.CKSMRC 位为 1 时，停止模式唤醒后，此寄存器初始化，PCLK4 为系统时钟的 1 分频。	RW
15	Reserved	保留	-	-
14:12	PCLK3S	PCLK3 时钟分频选择位	● 0b000：系统时钟的 1 分频 ● 0b001：系统时钟的 2 分频 ● 0b010：系统时钟的 4 分频 ● 0b011：系统时钟的 8 分频 ● 0b100：系统时钟的 16 分频 ● 0b101：系统时钟的 32 分频 ● 0b110：系统时钟的 64 分频 ● 0b111：禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时，停止模式唤醒后，此寄存器初始化，PCLK3 为系统时钟的 1 分频。	RW
11	Reserved	保留	-	-
10:8	PCLK2S	PCLK2 时钟分频选择位	● 0b000：系统时钟的 1 分频 ● 0b001：系统时钟的 2 分频 ● 0b010：系统时钟的 4 分频 ● 0b011：系统时钟的 8 分频 ● 0b100：系统时钟的 16 分频 ● 0b101：系统时钟的 32 分频 ● 0b110：系统时钟的 64 分频 ● 0b111：禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时，停止模式唤醒后，此寄存器初始化，PCLK2 为系统时钟的 1 分频。	RW
7	Reserved	保留	-	-
6:4	PCLK1S	PCLK1 时钟分频选择位	● 0b000：系统时钟的 1 分频 ● 0b001：系统时钟的 2 分频 ● 0b010：系统时钟的 4 分频 ● 0b011：系统时钟的 8 分频 ● 0b100：系统时钟的 16 分频 ● 0b101：系统时钟的 32 分频 ● 0b110：系统时钟的 64 分频 ● 0b111：禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时，停止模式唤醒后，此寄存器初始化，PCLK1 为系统时钟的 1 分频。	RW
3	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
2:0	PCLK0S	PCLK0 时钟分频选择位	● 0b000: 系统时钟的 1 分频 ● 0b001: 系统时钟的 2 分频 ● 0b010: 系统时钟的 4 分频 ● 0b011: 系统时钟的 8 分频 ● 0b100: 系统时钟的 16 分频 ● 0b101: 系统时钟的 32 分频 ● 0b110: 系统时钟的 64 分频 ● 0b111: 禁止设置  说明 PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, PCLK0 为系统时钟的 1 分频。	RW

5.4.26 CMU USB 时钟配置寄存器 (CMU_USBCKCFGR)

Offset	Bit Position							
0xC024	7	6	5	4	3	2	1	0
Reset	0x40							
Name	USBCKS				Reserved			
Access	RW							

位/位域	标记	位名	功能描述	读写
7:4	USBCKS	USBFS/USBHS 的 48MHz 时钟源选择	● 0b0001: 系统时钟 2 分频 ● 0b0010: 系统时钟 3 分频 ● 0b0011: 系统时钟 4 分频 ● 0b0100: 系统时钟 5 分频 ● 0b0101: 系统时钟 6 分频 ● 0b0110: 系统时钟 7 分频 ● 0b0111: 系统时钟 8 分频 ● 0b1000: PLLHQ ● 0b1001: PLLHR ● 0b1010: PLLAP ● 0b1011: PLLAQ ● 0b1100: PLLAR ● 其他: 禁止设置  说明 ● 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。 ● PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, USBCLK 为系统时钟的 5 分频。	RW
3:0	Reserved	保留	-	-

5.4.27 CMU CAN 时钟配置寄存器 (CMU_CANCKCFGR)

Offset	Bit Position															
0xC018	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xDDDD															
Name	MCAN2CKS				MCAN1CKS				CAN2CKS				CAN1CKS			
Access	RW				RW				RW				RW			

位/位域	标记	位名	功能描述	读写
15:12	MCAN2CKS	MCAN2 通讯时钟选择	0b0001: 系统时钟 2 分频 0b0010: 系统时钟 3 分频 0b0011: 系统时钟 4 分频 0b0100: 系统时钟 5 分频 0b0101: 系统时钟 6 分频 0b0110: 系统时钟 7 分频 0b0111: 系统时钟 8 分频 0b1000: PLLHQ 0b1001: PLLHR 0b1010: PLLAP 0b1011: PLLAQ 0b1100: PLLAR 0b1101: XTAL - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。 - 系统时钟选择 PLLH 时, 需要关闭各外围模块时钟, 再写 CMU_SCFGR 寄存器切换时钟分频。写完 CMU_CANCKCFGR 寄存器后, 软件等待系统稳定时间至少 30μs。 - PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, MCAN2CLK 为 XTAL 时钟源。	RW
11:8	MCAN1CKS	MCAN1 通讯时钟选择	0b0001: 系统时钟 2 分频 0b0010: 系统时钟 3 分频 0b0011: 系统时钟 4 分频 0b0100: 系统时钟 5 分频 0b0101: 系统时钟 6 分频 0b0110: 系统时钟 7 分频 0b0111: 系统时钟 8 分频 0b1000: PLLHQ 0b1001: PLLHR 0b1010: PLLAP 0b1011: PLLAQ 0b1100: PLLAR	RW

位/位域	标记	位名	功能描述	读写
11:8	MCAN1CKS	MCAN1 通讯时钟选择	● 0b1101: XTAL ● 其他: 禁止设置  说明 ● 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。 ● 系统时钟选择 PLLH 时, 需要关闭各外围模块时钟, 再写 CMU_SCFGR 寄存器切换时钟分频。写完 CMU_CANCKCFGR 寄存器后, 软件等待系统稳定时间至少 30μs。 ● PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, MCAN1CLK 为 XTAL 时钟源。	RW
7:4	CAN2CKS	CAN2 通讯时钟选择	● 0b0001: 系统时钟 2 分频 ● 0b0010: 系统时钟 3 分频 ● 0b0011: 系统时钟 4 分频 ● 0b0100: 系统时钟 5 分频 ● 0b0101: 系统时钟 6 分频 ● 0b0110: 系统时钟 7 分频 ● 0b0111: 系统时钟 8 分频 ● 0b1000: PLLHQ ● 0b1001: PLLHR ● 0b1010: PLLAP ● 0b1011: PLLAQ ● 0b1100: PLLAR ● 0b1101: XTAL ● 其他: 禁止设置  说明 ● 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。 ● 系统时钟选择 PLLH 时, 需要关闭各外围模块时钟, 再写 CMU_SCFGR 寄存器切换时钟分频。写完 CMU_CANCKCFGR 寄存器后, 软件等待系统稳定时间至少 30μs。 ● PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, CAN2CLK 为 XTAL 时钟源。	RW
3:0	CAN1CKS	CAN1 通讯时钟选择	● 0b0001: 系统时钟 2 分频 ● 0b0010: 系统时钟 3 分频 ● 0b0011: 系统时钟 4 分频 ● 0b0100: 系统时钟 5 分频 ● 0b0101: 系统时钟 6 分频 ● 0b0110: 系统时钟 7 分频 ● 0b0111: 系统时钟 8 分频 ● 0b1000: PLLHQ ● 0b1001: PLLHR ● 0b1010: PLLAP ● 0b1011: PLLAQ ● 0b1100: PLLAR	RW

位/位域	标记	位名	功能描述	读写
3:0	CAN1CKS	CAN1 通讯时钟选择	0b1101: XTAL - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。 - 系统时钟选择 PLLH 时, 需要关闭各外围模块时钟, 再写 CMU_SCFGR 寄存器切换时钟分频。写完 CMU_CANCKCFGR 寄存器后, 软件等待系统稳定时间至少 30μs。 - PWC_STPMCR.CKSMRC 位为 1 时, 停止模式唤醒后, 此寄存器初始化, CAN1CLK 为 XTAL 时钟源。	RW

5.4.28 CMU ADC/DAC 时钟配置寄存器 (CMU_PERICKSEL)

Offset	Bit Position															
0xC010	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												PERICKSEL			
Access													RW			

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3:0	PERICKSEL	ADC/DAC 时钟源选择	0b0000: CMU_SCFGR 设定的 PCLK2/ PCLK4 0b1000: PCLK2/PCLK4 配置为 PLLHQ 0b1001: PCLK2/PCLK4 配置为 PLLHR 0b1010: PCLK2/PCLK4 配置为 PLLAP 0b1011: PCLK2/PCLK4 配置为 PLLAQ 0b1100: PCLK2/PCLK4 配置为 PLLAR - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。	RW

5.4.29 CMU 调试时钟配置寄存器 (CMU_TPIUCKCFGR)

Offset	Bit Position							
0xC03F	7	6	5	4	3	2	1	0
Reset	0x00							
Name	TPIUCKOE	Reserved					TPIUCKS	
Access	RW						RW	

位/位域	标记	位名	功能描述	读写
7	TPIUCKOE	TPIU 时钟供给允许位	● 0b0：禁止 ● 0b1：允许	RW
6:2	Reserved	保留	-	-
1:0	TPIUCKS	TPIU 时钟分频选择位	● 0b00：1 分频 ● 0b01：2 分频 ● 0b10：4 分频 ● 0b11：禁止设置	RW

5.4.30 CMU I2S 时钟配置寄存器 (CMU_I2SCKCFGR)

Offset	Bit Position															
0xC012	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xB BBBB															
Name	I2S4CKSEK				I2S3CKSEK				I2S2CKSEK				I2S1CKSEK			
Access	RW				RW				RW				RW			

位/位域	标记	位名	功能描述	读写
15:12	I2S4CKSEK	I2S4 时钟选择	0b0000: CMU_SCFGR 设定的 PCLK1 0b1000: PLLHQ 0b1001: PLLHR 0b1010: PLLAP 0b1011: PLLAQ 0b1100: PLLAR - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。	RW
11:8	I2S3CKSEK	I2S3 时钟选择	0b0000: CMU_SCFGR 设定的 PCLK1 0b1000: PLLHQ 0b1001: PLLHR 0b1010: PLLAP 0b1011: PLLAQ 0b1100: PLLAR - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。	RW
7:4	I2S2CKSEK	I2S2 时钟选择	0b0000: CMU_SCFGR 设定的 PCLK1 0b1000: PLLHQ 0b1001: PLLHR 0b1010: PLLAP 0b1011: PLLAQ 0b1100: PLLAR - 其他: 禁止设置  说明 - 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。	RW
3:0	I2S1CKSEK	I2S1 时钟选择	0b0000: CMU_SCFGR 设定的 PCLK1 0b1000: PLLHQ 0b1001: PLLHR	RW

位/位域	标记	位名	功能描述	读写
3:0	I2S1CKSEK	I2S1 时钟选择	● 0b1010: PLLAP ● 0b1011: PLLAQ ● 0b1100: PLLAR ● 其他: 禁止设置  说明 ● 切换的目标时钟源为 PLLH/PLLA 时, 需保证在 PLLH/PLLA 时钟处于稳定状态。	RW

5.4.31 CMU MCO1 时钟输出配置寄存器 (CMU_MCO1CFGR)

Offset	Bit Position							
0xC03D	7	6	5	4	3	2	1	0
Reset	0x00							
Name	MCO1EN	MCO1DIV			MCO1SEL			
Access	RW	RW			RW			

位/位域	标记	位名	功能描述	读写
7	MCO1EN	MCO_1 输出许可	● 0b0: 禁止 ● 0b1: 允许	RW
6:4	MCO1DIV	MCO_1 分频选择	● 0b000: 1 分频 ● 0b001: 2 分频 ● 0b010: 4 分频 ● 0b011: 8 分频 ● 0b100: 16 分频 ● 0b101: 32 分频 ● 0b110: 64 分频 ● 0b111: 128 分频	RW
3:0	MCO1SEL	MCO_1 时钟源选择	● 0b0000: HRC 时钟 ● 0b0001: MRC 时钟 ● 0b0010: LRC 时钟 ● 0b0011: XTAL 时钟 ● 0b0100: XTAL32 时钟 ● 0b0110: PLLHP ● 0b0111: PLLAP ● 0b1000: PLLHQ ● 0b1001: PLLAQ ● 0b1010: PLLAR ● 0b1011: 系统时钟 ● 其他: 禁止设置	RW

5.4.32 CMU MCO2 时钟输出配置寄存器 (CMU_MCO2CFGR)

Offset	Bit Position							
0xC03E	7	6	5	4	3	2	1	0
Reset	0x00							
Name	MCO2EN	MCO2DIV			MCO2SEL			
Access	RW	RW			RW			

位/位域	标记	位名	功能描述	读写
7	MCO2EN	MCO_2 输出许可	0b0: 禁止 0b1: 允许	RW
6:4	MCO2DIV	MCO_2 分频选择	0b000: 1 分频 0b001: 2 分频 0b010: 4 分频 0b011: 8 分频 0b100: 16 分频 0b101: 32 分频 0b110: 64 分频 0b111: 128 分频	RW
3:0	MCO2SEL	MCO_2 时钟源选择	0b0000: HRC 时钟 0b0001: MRC 时钟 0b0010: LRC 时钟 0b0011: XTAL 时钟 0b0100: XTAL32 时钟 0b0110: PLLHP 0b0111: PLLAP 0b1000: PLLHQ 0b1001: PLLAQ 0b1010: PLLAR 0b1011: 系统时钟 - 其他: 禁止设置	RW

5.4.33 FCM 下限比较值寄存器 (FCMn_LVR) (n=1~4)

偏移地址 (Offset) : 0x00+0x40*(n-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																LVR															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	LVR	下限比较值	START 位为 0 时配置此寄存器	RW

5.4.34 FCM 上限比较值寄存器 (FCMn_UVR) (n=1~4)

偏移地址 (Offset) : 0x04+0x40*(n-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																UVR															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	UVR	上限比较值	START 位为 0 时配置此寄存器	RW

5.4.35 FCM 计数器值寄存器 (FCMn_CNTR) (n=1~4)

偏移地址 (Offset) : 0x08+0x40*(n-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNTR															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNTR	计数器值	检测到基准时钟的 EDGES 位选择的有效边沿时，将计数器值保存到此寄存器 (START=1 后第一个有效边沿除外)	RO

5.4.36 FCM 开始停止寄存器 (FCM_STR)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																												START4	START3	START2	START1
Access																													RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:0	STARTn (n=4~1)	通道 n 频率测量 开始位	0b0: 频率测量停止 0b1: 频率测量开始	RW

5.4.37 FCM 测量对象控制寄存器 (FCMn_MCCR) (n=1~4)

偏移地址 (Offset) : 0x10+0x40*(n-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								MCKS		Reserved	MDIVS				
Access																									RW			RW				

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:4	MCKS	测量对象时钟选择位	0b0000: XTAL 0b0001: XTAL32 0b0010: HRC 0b0011: LRC 0b0100: SWDTLRC 0b0101: PCLK1 0b0110: PLLAP 0b0111: MRC 0b1000: PLLHP 0b1001: RTCLRC - 其他: 禁止设置	RW
3:2	Reserved	保留	-	-
1:0	MDIVS	测量对象分频选择	0b00: 不分频 0b01: 4 分频 0b10: 8 分频 0b11: 32 分频	RW

5.4.38 FCM 测量基准控制寄存器 (FCMn_RCCR) (n=1~4)

偏移地址 (Offset) : 0x14+0x40*(n-1)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																EXREFE	Reserved	EDGES	Reserved	DNFS	INEXS	RCKS				Reserved	RDIVS				
Access																	RW		RW		RW	RW	RW			RW					RW	

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	EXREFE	外部管脚输入参考时钟 FCMREF 允许位	0b0: 禁止外部管脚输入参考时钟 FCMREF 0b1: 允许外部管脚输入参考时钟 FCMREF	RW
14	Reserved	保留	-	-
13:12	EDGES	测量基准边沿选择位	0b00: 上升沿 0b01: 下降沿 0b10: 上升和下降沿 0b11: 禁止设置	RW
11:10	Reserved	保留	-	-
9:8	DNFS	数字滤波器功能选择位	0b00: 无滤波功能 0b01: MCKS 位选择的时钟作为滤波时钟 0b10: MCKS 位选择的时钟的 4 分频作为滤波时钟 0b11: MCKS 位选择的时钟的 16 分频作为滤波时钟	RW
7	INEXS	测量基准, 内部时钟和端子选择位	0b0: 外部管脚输入参考时钟 FCMREF 0b1: RCKS 选择位选择的时钟	RW
6:3	RCKS	测量基准时钟选择位	0b0000: XTAL 0b0001: XTAL32 0b0010: HRC 0b0011: LRC 0b0100: SWDTLRC 0b0101: PCLK1 0b0110: PLLAP 0b0111: MRC 0b1000: PLLHP 0b1001: RTCLRC - 其他: 禁止设置	RW
2	Reserved	保留	-	-
1:0	RDIVS	测量基准分频选择	0b00: 32 分频 0b01: 128 分频 0b10: 1024 分频 0b11: 8192 分频	RW

5.4.39 FCM 中断复位控制寄存器 (FCMn_RIER) (n=1~4)

偏移地址 (Offset) : 0x18+0x40*(n-1)

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																								ERRRE	Reserved		ERRINTRS	Reserved	OVFIE	MENDIE	ERRIE		
Access																									RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	ERRRE	频率异常复位允许位	● 0b0: 禁止 ● 0b1: 允许	RW
6:5	Reserved	保留	-	-
4	ERRINTRS	频率异常中断复位选择位	● 0b0: 频率异常发生中断 ● 0b1: 频率异常发生复位	RW
3	Reserved	保留	-	-
2	OVFIE	计数器溢出中断允许位	● 0b0: 禁止计数器溢出中断 ● 0b1: 允许计数器溢出中断	RW
1	MENDIE	测量结束中断允许位	● 0b0: 禁止测量结束发生中断 ● 0b1: 允许测量结束发生中断	RW
0	ERRIE	频率异常中断允许位	● 0b0: 禁止频率异常发生中断 ● 0b1: 允许频率异常发生中断	RW

5.4.40 FCM 标志寄存器 (FCM_SR)

Offset	Bit Position																																
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																				OVF4	MENDF4	ERRF4	OVF3	MENDF3	ERRF3	OVF2	MENDF2	ERRF2	OVF1	MENDF1	ERRF1	
Access																					RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	OVF4	通道 4 计数器溢出标志位	0b0: 计数器未溢出 0b1: 计数器溢出	RO
10	MENDF4	通道 4 测量结束标志位	0b0: 测量中 0b1: 测量结束	RO
9	ERRF4	通道 4 频率异常标志位	0b0: 无频率异常发生 0b1: 发生频率异常	RO
8	OVF3	通道 3 计数器溢出标志位	0b0: 计数器未溢出 0b1: 计数器溢出	RO
7	MENDF3	通道 3 测量结束标志位	0b0: 测量中 0b1: 测量结束	RO
6	ERRF3	通道 3 频率异常标志位	0b0: 无频率异常发生 0b1: 发生频率异常	RO
5	OVF2	通道 2 计数器溢出标志位	0b0: 计数器未溢出 0b1: 计数器溢出	RO
4	MENDF2	通道 2 测量结束标志位	0b0: 测量中 0b1: 测量结束	RO
3	ERRF2	通道 2 频率异常标志位	0b0: 无频率异常发生 0b1: 发生频率异常	RO
2	OVF1	通道 1 计数器溢出标志位	0b0: 计数器未溢出 0b1: 计数器溢出	RO
1	MENDF1	通道 1 测量结束标志位	0b0: 测量中 0b1: 测量结束	RO
0	ERRF1	通道 1 频率异常标志位	0b0: 无频率异常发生 0b1: 发生频率异常	RO

5.4.41 FCM 标志位清除寄存器 (FCMn_CLR) (n=1~4)

偏移地址 (Offset) : 0x20+0x40*(n-1)

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																												OVFCLR		MENDFCLR		ERRFCLR	
Access																													WO	WO	WO			

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	OVFCLR	计数器溢出标志清除	写 1 清除标志, 写 0 无效	WO
1	MENDFCLR	测量结束标志清除	写 1 清除标志, 写 0 无效	WO
0	ERRFCLR	频率异常标志清除	写 1 清除标志, 写 0 无效	WO

6 电源控制 (PWC)

6.1 简介

电源控制器用来控制芯片的多个电源域在多个运行模式和低功耗模式下的电源供给、切换、检测。电源控制器由功耗控制逻辑 (PWCL)、电源电压检测单元 (PVD)、自动切换 VCC 与电池供电的电池备份控制模块 (BATBKUP) 构成。

芯片的工作电压 (VCC) 为 1.8V 到 3.63V。电压调节器 (LDO) 为 VDD 域和 VDDR 域供电, VDDR 电压调节器 (RLDO) 在掉电模式或者 VCC 掉电情况下为 VDDR 域供电。芯片通过功耗控制逻辑 (PWCL) 提供了高速、超低速两种运行模式, 睡眠、停止和掉电等三种低功耗模式。

电源电压检测单元 (PVD) 提供了上电复位 (POR)、掉电复位 (PDR)、欠压复位 (BOR)、可编程电压检测 1 (PVD1)、可编程电压检测 2 (PVD2)、基准电压测量通路、VBAT 电压检测、VBAT 电压测量等功能, 其中 POR、PDR、BOR 通过检测 VCC 电压, 控制芯片复位动作。PVD1 通过检测 VCC 电压, 根据寄存器设定使芯片产生复位或者中断。PVD2 通过检测 VCC 电压或者外部输入检测电压, 根据寄存器选择产生复位或者中断。基准电压测量通路, 是使用 ADC 测量基准电压的功能。VBAT 电压检测, 是读取寄存器获得 VBAT 电压高于或者低于 VBAT 检测电压的功能, VBAT 检测电压可以使用寄存器选择 1.8V 或者 2.1V。VBAT 电压测量功能, 是指使用 ADC 测量 VBAT 的 1/2 分压, 从而获得 VBAT 电压值的功能。

电池备份域在 VCC 掉电情况下通过 VBAT 维持电源, 保证实时时钟模块 (RTC)、唤醒定时器 (WKTm) 能够继续动作, 并为 RLDO 提供电源。VDDR 区域在芯片进入掉电模式或者 VCC 掉电情况下可以通过 RLDO 维持电源, 保持 4KB 的 Ret SRAM 的数据。

模拟模块配备了专用供电引脚, 提高了模拟性能。

6.2 功能说明

6.2.1 电源分布

6.2.1.1 电源分布构成

芯片由 VCC 域、VDD 电源域、AVCC 电源域、电池备份域和 VDDR 域构成。

- VCC 域通过 VCC/VSS 引脚供电，由功耗控制逻辑 (PWCL)、电源电压检测单元 (PVD)、IO 电平保持电路、电压调节器 (LDO)、外部高速振荡器 (XTAL)、内部低速振荡器 (LRC) 等电路构成。
- VDD 电源域由 CPU、数字外设等数字逻辑、RAM、FLASH 等构成，通过 LDO 产生的 VDD 供电。在 VDD 域中的 RAM 被分为 5 组，由 11 个独立模块构成，可以通过寄存器独立控制每个模块的断电。
- AVCC 电源域为模拟电源域，主要由模数转换器 (ADC)、数模转换器 (DAC)、比较器 (CMP)、可编程增益放大器 (PGA)、模拟系的输入输出管脚构成，由 AVCC/AVSS 引脚供电。为了提高高精度的模拟性能，模拟区域配有独立电源。为了确保 ADC 具有更高的精度，ADC 的参考电压使用专用管脚 (VREFH, VREFL)。
- BATBKUP 电源域为电池备份域，该区域的电源经电源切换开关自动选择由 VCC 或 VBAT 供电。当 VCC 断电时，备份域的电源自动切换成 VBAT。电池备份域由实时时钟 (RTC)、唤醒定时器 (WKTM)、外部低速振荡器 (XTAL32)、RTC 用内部低速振荡器 (RTCLRC) 构成。
- VDDR 域由 4KB Ret SRAM 构成。Ret SRAM 通过 RLDO 供电。在掉电模式下，Ret SRAM 能够保持数据。不需要使用 Ret SRAM 时，可以置位 PWC_VBATCR.VBTRSD，使芯片进入掉电模式或者 VCC 掉电时切断 VDDR 域电源，进一步降低功耗。

下图是芯片的电源分布图。

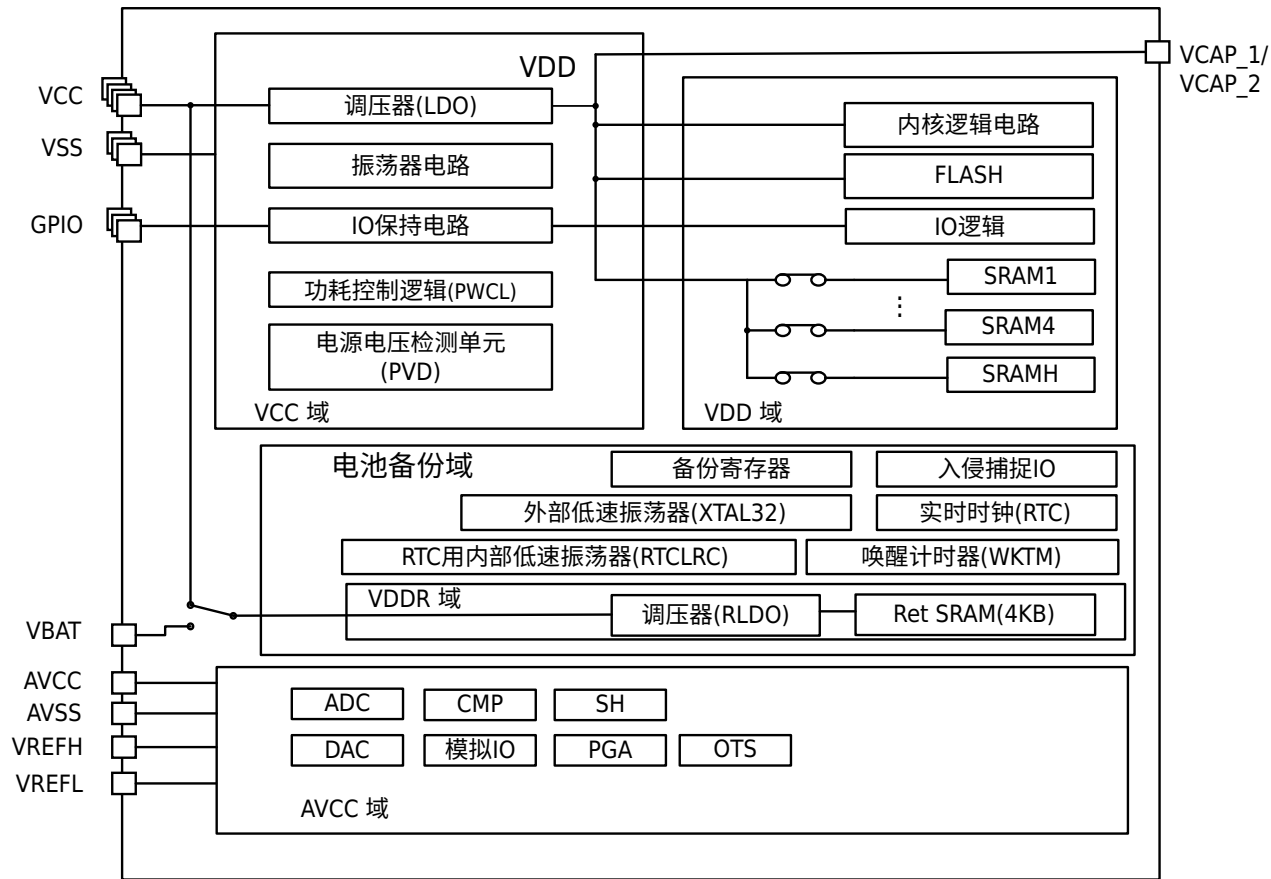


图 6-1 电源构成图

6.2.1.2 电池备份电源域

电池备份域由内部电源切换器来选择 VCC 供电或 VBAT（电池）供电，该备份域包含 RTC（实时时钟）、XTAL32（外部低速振荡器）、128 字节备份寄存器、4K 字节 Ret SRAM、内部低速振荡器、以及 RTCIC0-1 等 2 个管脚构成。当 VCC 引脚的电压降低时，由专用的电池备用电源引脚（VBAT 引脚）给实时时钟（RTC）和 RTC 用内部低速振荡器供电。对于没有外部电池的应用，建议将 VBAT 引脚通过 100nF 的外部陶瓷去耦电容连接到 VCC 引脚上。

电池备用功能的切换运行如下图所示。VCC 电压降低到 V_{PDR} 以下时，电源备份域切换成由 VBAT 供电；VCC 电压升高到 V_{POR} 以上后，电源备份域切换成由 VCC 供电。如果芯片使用掉电模式 3 或者掉电模式 4， V_{PDR} 的检测电平不能确定，在 VCC 电压下降时，不能保证芯片可以正确切换到 VBAT 电源，因此，芯片不能同时使用掉电模式 3 和电池备份功能，也不能同时使用掉电模式 4 和电池备份功能。

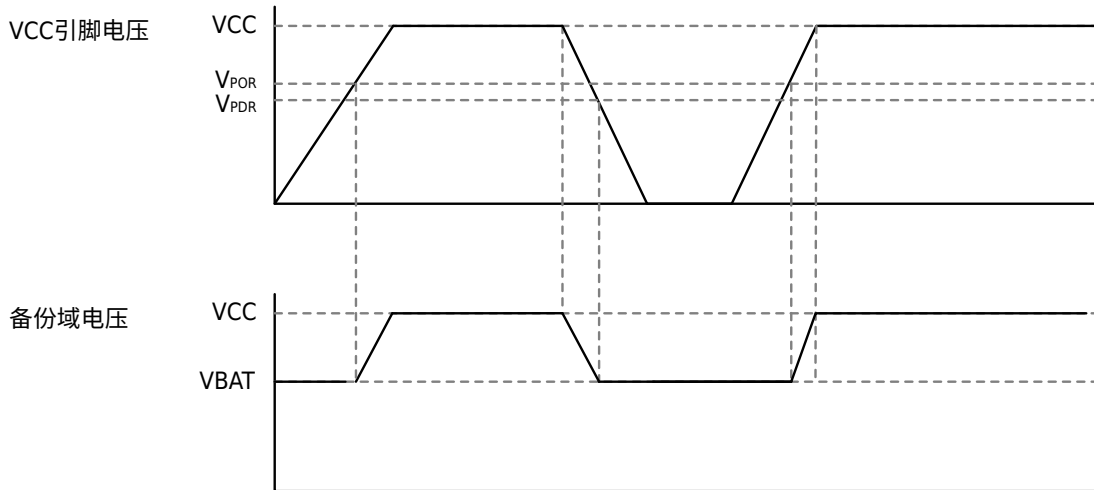


图 6-2 电池备用域电源切换图

RTC 的时钟源可以是 RTC 用内部低速振荡器（RTCLRC）或外部低速振荡器（XTAL32）。在通过 WFI/WFE 指令进入省电模式之前，Cortex-M4 需要通过 RTC 寄存器设置预期的唤醒时间并启用唤醒功能，以实现 RTC 定时器唤醒事件。

Ret SRAM 在掉电模式甚至 VCC 掉电后仍然可以保持数据，因此如果持续给 VBAT 供电时，可以将 Ret SRAM 视为内部 EEPROM。发生入侵事件时，根据软件设定可以选择 Ret SRAM 是否发生掉电动作，及备份寄存器是否需要初始化。Ret SRAM 可以通过置位寄存器 PWC_VBATCR 的 VBTRSD 位关闭。在 Ret SRAM 被关闭后，为 Ret SRAM 供电的调压器等电路也随之关闭，可以进一步降低功耗。Ret SRAM 在 VBTRSD 位置位后要等到 RAMVALID 标志位变成 1 后才可以进行读写操作。

使用电池备份域时，需要先写 PWC_VBATRSTR 寄存器将 VBAT 域初始化。PWC_VBATRSTR 写 0xA5 后，外部低速振荡器 XTAL32、Ret SRAM 用的 RLDO 启动、唤醒计时器 WKTIM 被初始化。

6.2.2 电源电压检测单元 (PVD) 说明

电源电压检测单元 (PVD) 包括上电复位 (POR)、掉电复位 (PDR)、欠压复位 (BOR)、可编程电压检测 1 (PVD1)、可编程电压检测 2 (PVD2)，基准电压测量通路、VBAT 电压检测、VBAT 电压测量等功能。

6.2.2.1 上电复位/掉电复位动作说明

芯片集成有上电复位、掉电复位电路。上电复位、掉电复位波形如下图所示，当 VCC 高于指定阈值 V_{POR} 之后，经过 T_{RSTPOR} 时间之后，芯片解除上电复位状态，CPU 开始执行代码。当 VCC 低于 V_{PDR} 时，芯片保持复位状态。使用上电复位时，复位管脚 NRST 必须为 1。如果复位管脚被下拉，芯片将通过管脚复位的方式复位启动。

V_{POR} 、 V_{PDR} 、 T_{RSTPOR} 等参数的详细信息，请参见《数据手册》“电气特性”章节。

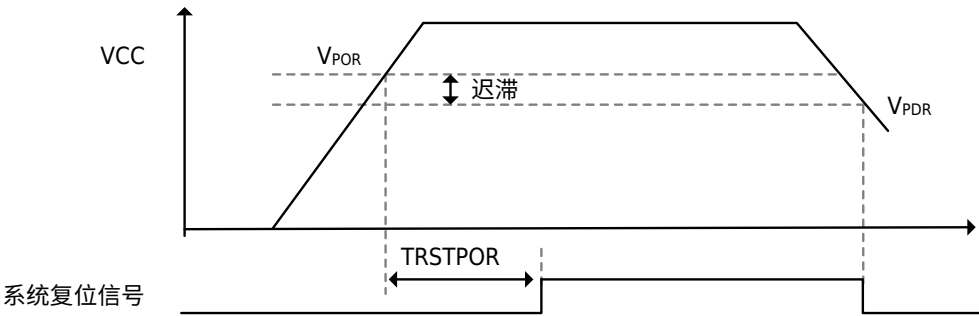


图 6-3 上电复位、掉电复位波形

6.2.2.2 欠压复位 (BOR) 说明

在上电期间，直到 VCC 高于 VBOR 为止，欠压复位 (BOR) 将使芯片处于复位状态。
VBOR 阈值通过初始化配置位 (ICG) 的 BOR_LEV、BORDIS 进行配置。BORDIS=0b0 时，BOR 检测电压可以从 4 个阈值中选择。BORDIS 被配置成 0b1 时，芯片通过上电复位、掉电复位进行复位控制。

表 6-1 BOR 配置

BORDIS	BOR_LEV	说明
0b1	0bXX	BOR 无效
0b0	0b00	BOR 有效、选择 BOR 阈值 0 (VBOR0)
0b0	0b01	BOR 有效、选择 BOR 阈值 1 (VBOR1)
0b0	0b10	BOR 有效、选择 BOR 阈值 2 (VBOR2)
0b0	0b11	BOR 有效、选择 BOR 阈值 3 (VBOR3)

BOR 阈值的电气特性，请参见《数据手册》“电气特性”章节。

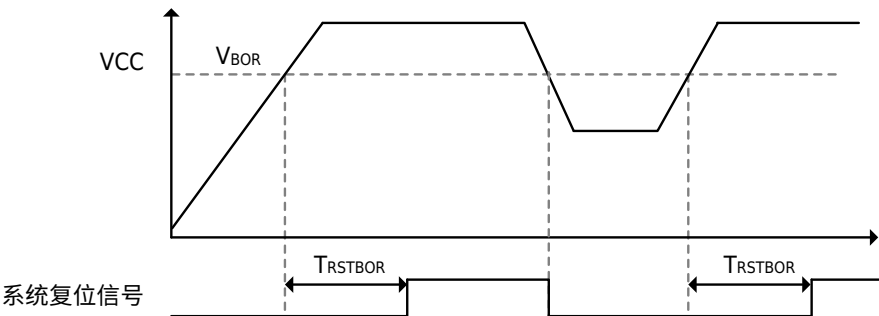


图 6-4 欠压复位波形

6.2.2.3 可编程电压检测 1 (PVD1)、可编程电压检测 2 (PVD2)

可编程电压检测 1 和可编程电压检测 2 通过检测 VCC 供电电压是否经过检测阈值，触发相应的复位或者中断动作。各检测电路分别可编程配置。

电源电压经过各个检测电路的阈值电压点时，可将该事件编程配置成复位/中断（可屏蔽/不可屏蔽）/AOS 触发功能。

可编程电压检测主要特性如下表所示。

表 6-2 PVD1/PVD2 特性

项目	PVD1	PVD2
检测对象	VCC 下降/上升过程中是否经过阈值电压点 (VPVD1)	VCC 下降/上升过程中是否经过阈值电压点 (VPVD2)、外部输入电压的上升下降是否经过阈值电压点 (VPVD2, PWC_PVDLCR.PVD2LVL[2:0]=0b111)
检测电压点	由 PVD1LVL[2:0]配置	由 PVD2LVL[2:0]配置
复位	复位: $VCC < VPVD1$; 复位解除: $VCC > VPVD1$ 经过一定复位处理时间。	复位: $VCC < VPVD2$; 复位解除: $VCC > VPVD2$ 经过一定复位处理时间。
中断	配置成电压检测 1 中断或者非可屏蔽中断 VCC 下降经过阈值电压点 (VPVD1) 或者 VCC 上升经过阈值电压点 (VPVD1) 或者 VCC 上升/下降经过阈值电压点 (VPVD1)	配置成电压检测 2 中断或者非可屏蔽中断 VCC 下降经过阈值电压点 (VPVD2) 或者 VCC 上升经过阈值电压点 (VPVD2) 或者 VCC 上升/下降经过阈值电压点 (VPVD2)
滤波功能	数字滤波	数字滤波
AOS 触发功能	VCC 下降经过阈值电压点 (VPVD1)	VCC 下降经过阈值电压点 (VPVD2)

6.2.2.4 PVD1、PVD2 中断/复位框图

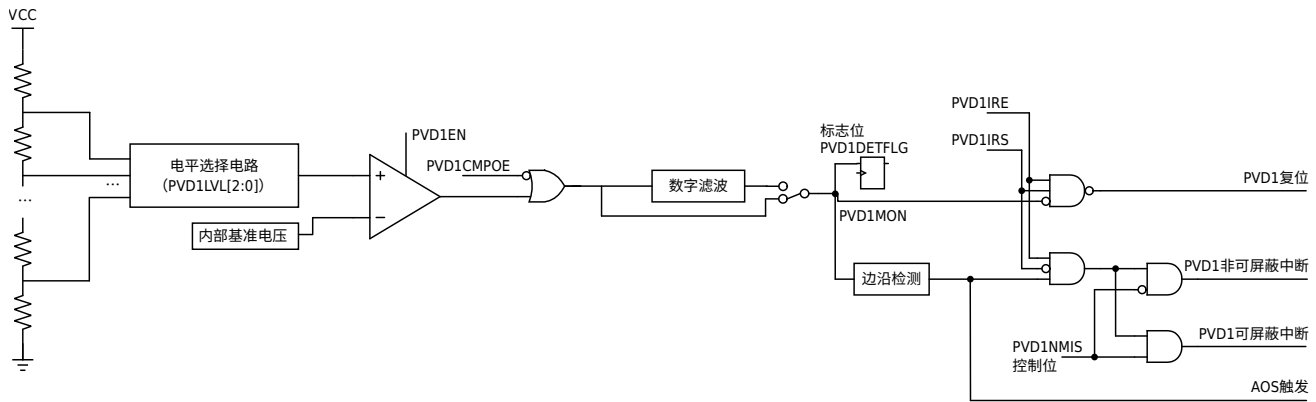


图 6-5 PVD1 中断/复位框图

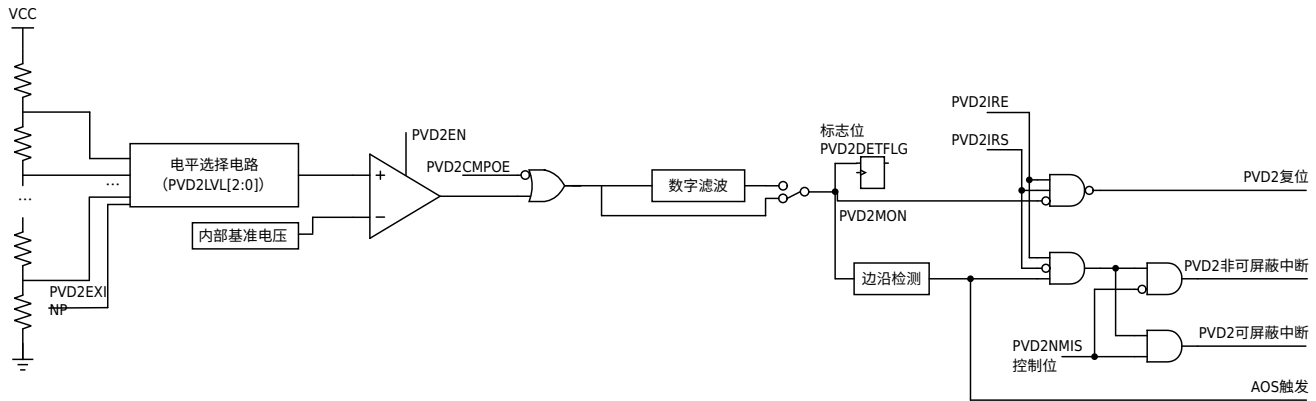


图 6-6 PVD2 中断/复位框图

6.2.2.5 输入/输出引脚

引脚名	输入/输出	功能
PVD2EXINP	输入	外部输入 PVD2 比较电压

6.2.2.6 PVD1 中断和复位

在停止模式或者掉电模式中使用 PVD1 电路时，请遵守如下注意事项：

1. 停止模式
- 必须将数字滤波器无效。
2. 掉电模式
- 必须将数字滤波器无效。
- PVD1IRS 置 0，选择 PVD1 产生中断；选择复位功能时，不能进入掉电模式。

下图是电压检测 1 中断的运行时序图，PVD1DETFLG 需要清零后才能再次发生中断。

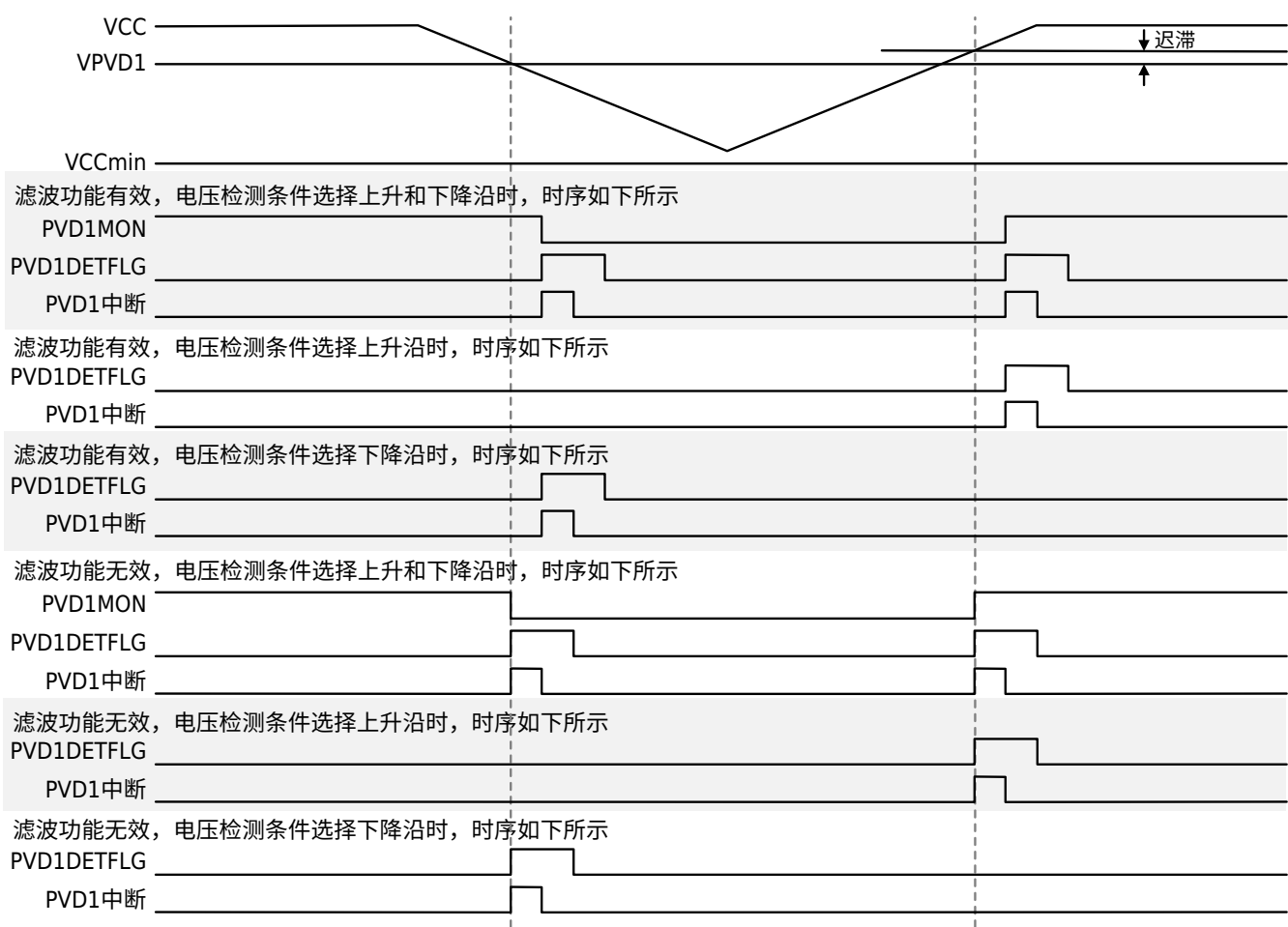


图 6-7 电源监视 1 中断时序图

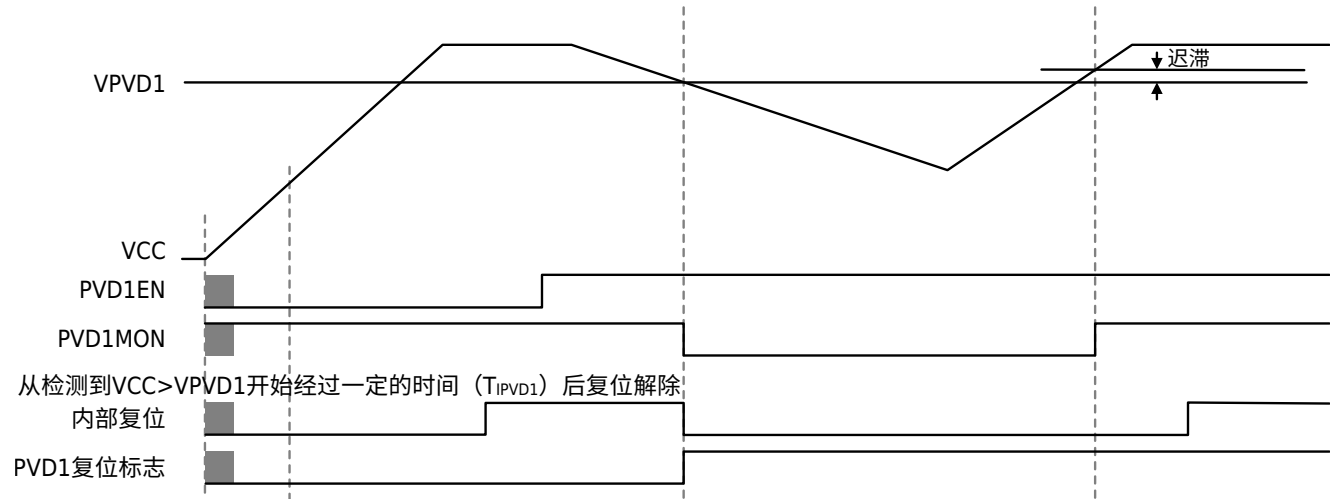


图 6-8 电源监视 1 复位时序图

6.2.2.7 PVD2 中断和复位

在停止模式或者掉电模式中使用 PVD2 电路时，请遵守如下注意事项：

1. 停止模式
 - 必须将数字滤波器无效。
2. 掉电模式
 - 必须将数字滤波器无效。
 - PVD2IRS 置 0，选择 PVD2 产生中断；选择复位功能时，不能进入掉电模式。

下图是电压检测 2 中断的运行时序图，PVD2DETFLG 需要清零后才能再次发生中断。

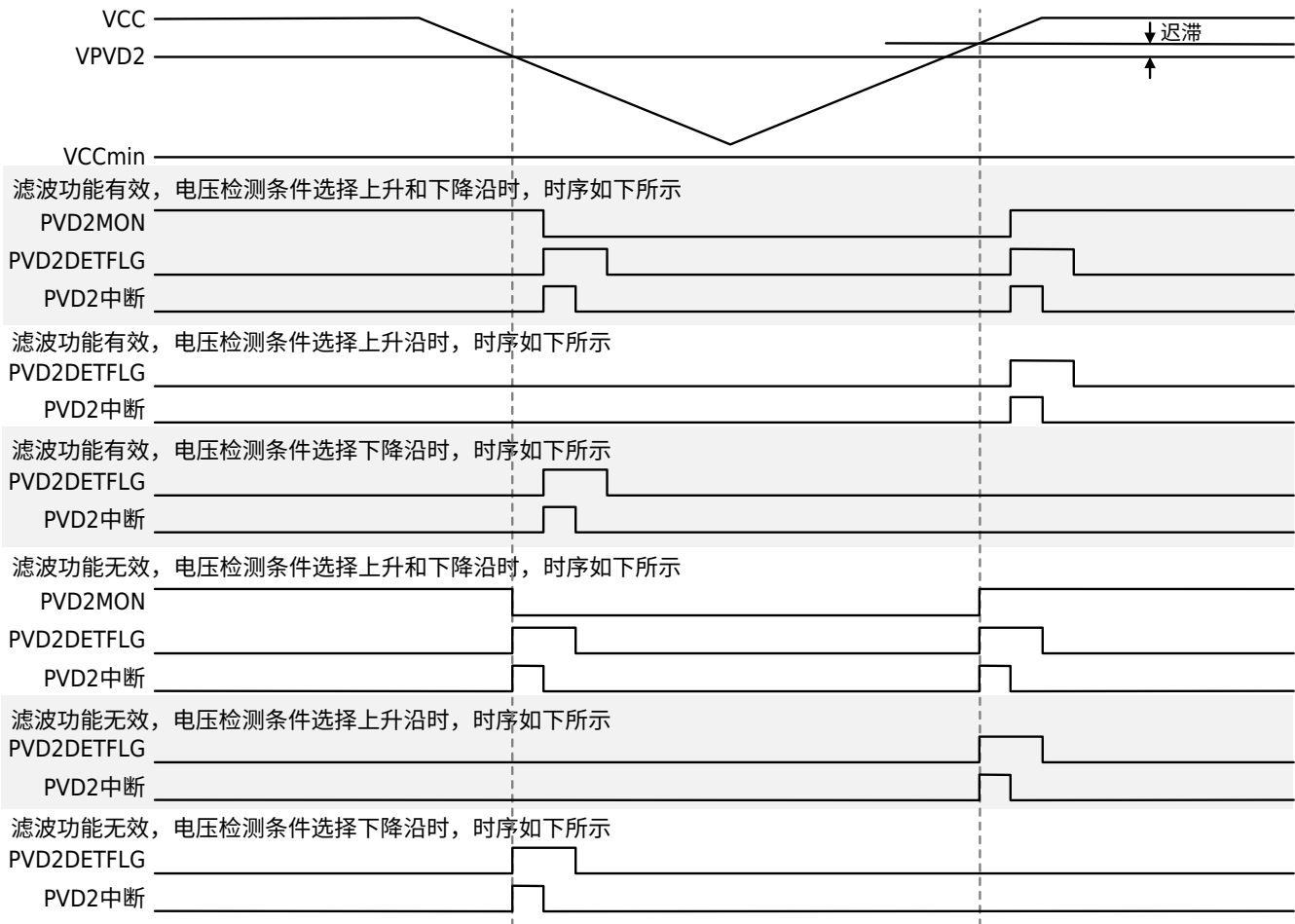


图 6-9 电源监视 2 中断时序图

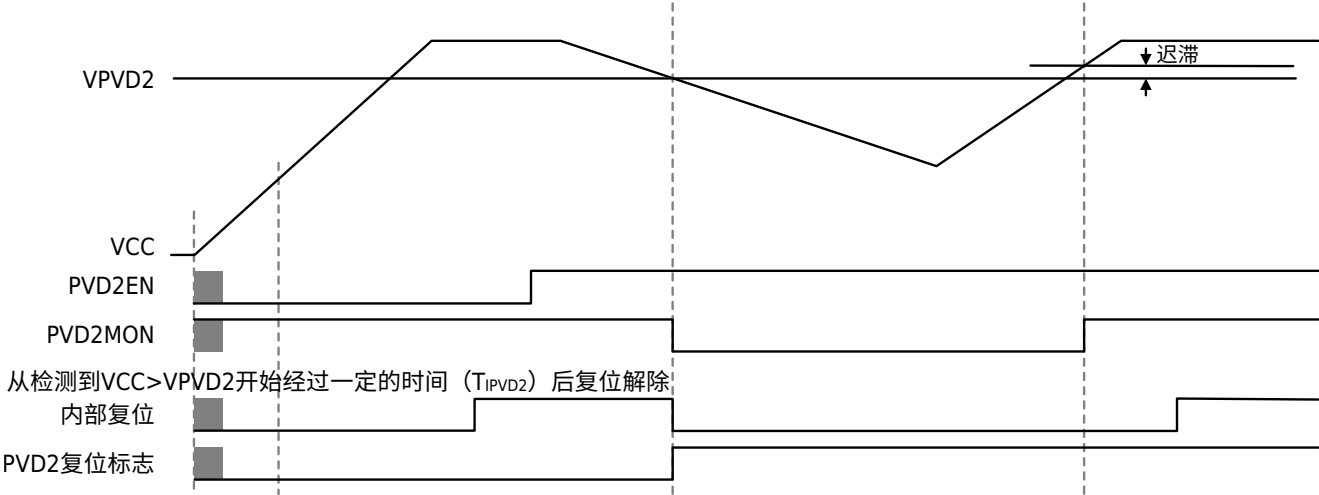


图 6-10 电源监视 2 复位时序图

6.2.2.8 内部电压采样和检测功能

芯片内部电压采样和检测功能包括基准电压测量，VBAT 电压检测，VBAT 电压测量等三个功能。基准电压测量通路，是使用 ADC 测量基准电压的功能。内部基准电压约为 1.15V。VBAT 电压检测，是读取寄存器获得 VBAT 电压高于或者低于 VBAT 检测电压的功能，VBAT 检测电压可以使用寄存器选择 1.8V 或者 2.1V。VBAT 电压测量功能，是指使用 ADC 测量 VBAT 的 1/2 分压，从而获得 VBAT 电压值的功能。

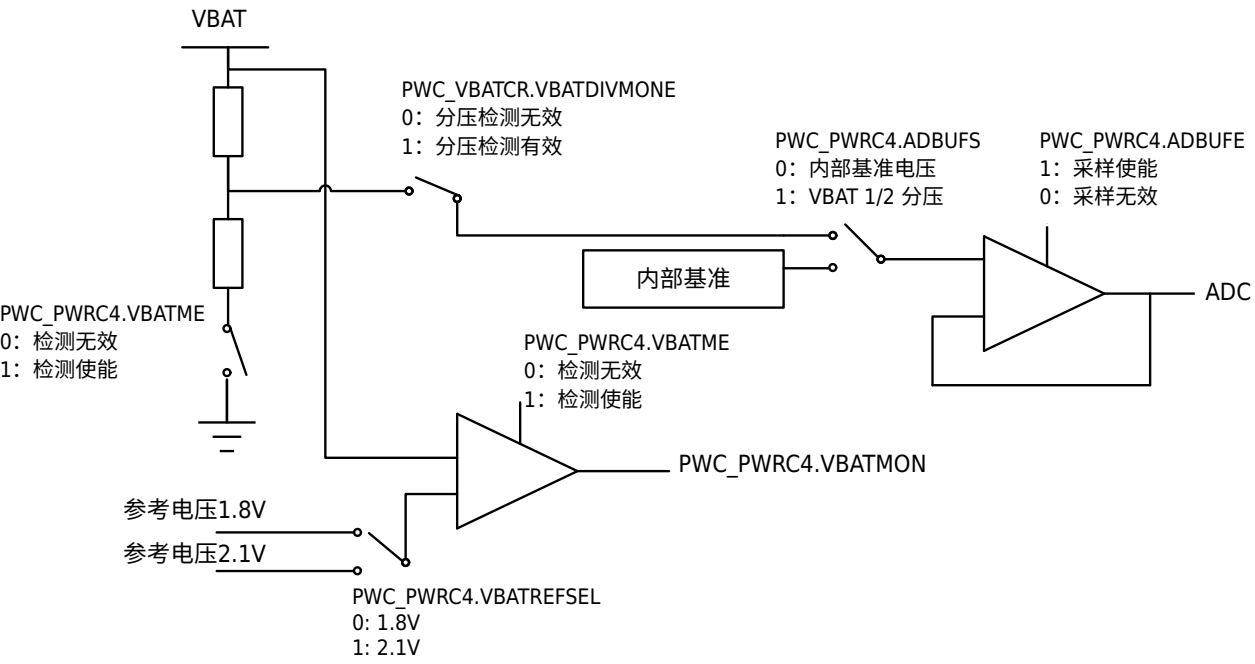


图 6-11 内部电压采样示意图

● 基准电压测量通路

使用基准电压测量通路时，需要按照下述步骤选中基准电压测量通路。

1. PWC_PWRC4.ADBUFS=0b0，选择内部基准电压；
2. PWC_PWRC4.ADBUFE=0b1，使能内部电压测量功能；
3. ADC_EXCHSELR.EXCHSEL=0b1，使 ADC 选通内部电压测量通道；
4. 等待 50μs 后使用 ADC 测量内部基准电压。

● VBAT 电压检测

使用 VBAT 电压检测通路时，需要按照下述步骤选中 VBAT 电压检测通路。

- 1. PWC_PWRC4.VBATREFSEL=0b1，选择 VBATREF=2.1V；
PWC_PWRC4.VBATREFSEL=0b0，选择 VBATREF=1.8V；
- 2. PWC_PWRC4.VBATME=0b1，使能 VBAT 分压电压电路；
- 3. 等待 50μs；
- 4. 读 PWC_PWRC4.VBATMON 位，读出为 0b0 时，VBAT>VBATREF。读出为 0b1 时，VBAT<VBATREF。

● VBAT 电压测量

使用 VBAT 电压测量通路时，需要按照下述步骤选中 VBAT 电压测量通路。

- 1. PWC_VBATCR.VBATDIVMONE=0b1，使能 VBAT 分压测量；
- 2. PWC_PWRC4.VBATME=0b1，使能 VBAT 分压电路；
- 3. PWC_PWRC4.ADBUFS=0b1，选择 VBAT 的 1/2 分压；
- 4. PWC_PWRC4.ADBUFE=0b1，使能内部电压测量功能；
- 5. ADC_EXCHSELR.EXCHSEL=0b1，使 ADC 选通内部电压测量通道；
- 6. 等待 50μs 后使用 ADC 测量 VBAT 电压；
- 7. PWC_VBATCR.VBATDIVMONE=0b0，关闭 VBAT 分压测量。

6.2.3 唤醒定时器

芯片内置了唤醒定时器 WKTM，该计数器可以选择 RTC 用内部低速振荡器 (RTCLRC)、外部低速振荡器作为时钟源，在 RTC 使用外部低速振荡器作为时钟动作时还可以选择 64Hz 的内部时钟信号作为时钟源。该计数器是加计数的计数器，在 PWC_WKTC2.WKTCE 置位后，计数器开始计数，当计数值等于 PWC_WKTC1 和 PWC_WKTC0 的 WKTCMP[11:0]设置值后，计数器计数清零后重新计数的同时，产生匹配一致事件，该事件可以用作普通中断，可以用于唤醒停止模式和掉电模式。

WKTM 寄存器值上电复位后不定，上电复位后，必须先设定 VBAT 区域复位寄存器后，设置 PWC_WKTC0/ PWC_WKTC1/PWC_WKTC2。在启动 WKTM 前，要确保计数时钟处于开启状态。

6.2.4 动作模式与低功耗模式

6.2.4.1 模式概述

系统复位或者上电复位之后，芯片的所有电源域都处于供电状态，芯片进入高速运行模式。在运行模式下，CPU 通过 HCLK 提供时钟，并执行程序代码。芯片提供了高速运行模式、超低速运行模式两种运行模式。芯片可以配置的运行模式如下表所示。

表 6-4 运行模式

运行模式	说明
高速运行模式 (High Speed Run Mode)	主频≤240MHz
超低速运行模式 (Ultra Low Speed Run Mode)	主频≤8MHz

为了节省 CPU 不需要运行时的功耗，系统提供了睡眠模式、停止模式、掉电模式等三种低功耗模式。芯片可以配置的低功耗模式如下表所示。在睡眠模式中，芯片的 Cortex-M4 内核停止动作，外设保持运行状态；在停止模式中，芯片的外设和 CPU 都停止动作；在掉电模式中，VDD 域的电 源被关闭，VDD 域的外设都停止动作。位于电池备份域中的实时时钟、唤醒定时器在低功耗模式中可以动作，Ret SRAM 可以保持数据。

表 6-5 低功耗模式

模式	说明
睡眠模式 (Sleep Mode)	CPU 时钟停止，外设保持运行
停止模式 (Stop Mode)	芯片外设和 CPU 的时钟都停止

模式		说明
掉电模式 (Power DownMode)	掉电模式 1 (PDMD1)	VDD 域掉电
	掉电模式 2 (PDMD2)	VDD 域掉电之外, 电压检测单元无效
	掉电模式 3 (PDMD3)	VDD 域掉电之外, 上电复位电路进入低功耗模式, 电压检测单元 (PVD) 无效, 与掉电模式 4 相比, 掉电唤醒时除 PWC_PDWKFO/PWC_PDWKFI/RMU_RSTFO 外芯片彻底复位。
	掉电模式 4 (PDMD4)	VDD 域掉电之外, 上电复位电路进入低功耗模式, 电压检测单元 (PVD) 无效

用户可以根据应用选择运行模式和低功耗模式, 以在低功耗、短启动时间、可唤醒源、和系统执行效率之间寻求最佳平衡。

用户可以通过下列方法优化运行模式下的功耗:

1. 设定最适的运行模式
2. 降低系统时钟速度
3. 关闭不使用的时钟源
4. 设定功能时钟控制寄存器 PWC_FCGn (n=0~3) 关闭不需要使用的功能
5. 关闭 RAM 的电源

低功耗模式的运行条件及各模块在低功耗模式下的状态如下表所示。

表 6-6 低功耗模式的运行条件及各模块在低功耗模式下的状态

项目	睡眠模式	停止模式	掉电模式
进入	PWC_STPMCR.STOP=0b0, PWC_PWRC0.PWDN=0b0, WFI	PWC_STPMCR.STOP=0b1, PWC_PWRC0.PWDN=0b0, WFI	PWC_STPMCR.STOP=0b1, PWC_PWRC0.PWDN=0b1, WFI
解除	任意中断或者复位	停止模式下可以使用的中断或复位	掉电模式下可以使用的唤醒事件或者复位
外部高速振荡器	工作可设	停止	停止
外部低速振荡器	工作可设	工作可设	工作可设
内部高速振荡器	工作可设	停止	掉电
内部中速振荡器	工作可设	停止	掉电
内部低速振荡器	工作可设	工作可设	工作可设
WDT 专用时钟振荡器	工作可设	工作可设	掉电
PLLA	工作可设	停止	掉电
PLLH	工作可设	停止	掉电
CPU	停止 (保持)	停止 (保持)	掉电
RAM (Ret SRAM 以外)	工作可设 可以设置工作, 掉电	停止 (保持) 根据进入待机前的设定, 可以维持掉电或者睡眠	掉电
Ret SRAM	工作可设 可以设置工作, 掉电, 睡眠	停止 (保持) 可以设置掉电, 睡眠	可以设置掉电, 睡眠
Flash	工作可设	停止 (保持)	掉电, 内容保持
DMA	工作可设	停止 (保持)	掉电
调压器	工作 驱动可以调整	工作 驱动可以调整	停止

项目	睡眠模式	停止模式	掉电模式
上电复位电路	工作	工作	工作 掉电模式 1、掉电模式 2 复位电路精度可以保证、掉电模式 3 和掉电模式 4 下上电复位电压不保证
欠压复位 BOR	工作可设	工作可设	掉电模式 1 工作可设 掉电模式 2/3/4 中停止
电压检测模块 PVD	工作可设	工作可设	掉电模式 1 工作可设 掉电模式 2/3/4 中停止
WDT	工作可设	停止 (保持)	掉电
SWDT	工作可设	工作可设	掉电
RTC	工作可设	工作可设	工作可设
USBFS	工作可设	停止 (保持)	掉电
Timer0	工作可设	工作可设	掉电
Timer2	工作可设	工作可设	掉电
WKTm	工作可设	工作可设	工作可设
其他周边外设模块	工作可设	停止 (保持)	掉电
ADC	工作可设	停止	掉电
DAC	工作可设	工作可设	掉电
PGA	工作可设	工作可设	掉电
CMP	工作可设	工作可设	掉电
PA0-PA10, PB0-PB2, PB5-PB13, PC0-PC13, PD0-PD15, PE0-PE15, PH2-PH15, PI0-PI12, PI14-15	工作可设	保持	保持或者高阻
PC14-PC15	工作可设	作为外部低速振荡器的管脚使用时, 保持振荡器动作; 设为 GPIO 或者其他外设功能时, 请设定保持两个管脚为同样的电平	设为 GPIO 或者其他外设功能时, PC14 和 PC15 的状态可以设置为保持或者高阻, 请设定保持两个管脚为同样的电平
PH0-PH1	工作可设	作为外部高速振荡器使用时, 振荡器停止震荡, 管脚状态保持进入 STOP 模式前的状态; 设为 GPIO 或者其他外设功能时, 保持 STOP 之前的状态	作为外部高速振荡器使用时, 振荡器停止震荡, 管脚状态保持进入掉电模式前的状态; 设为 GPIO 或者其他外设功能时, 保持掉电模式之前的状态
NRST 复位管脚	芯片外面通过电路上拉到 VCC	芯片外面通过电阻上拉到 VCC	芯片外面通过电阻上拉到 VCC
PA11-PA12, PB14-PB15	工作可设	保持 由于该管脚的电平被拉高时会产生电流, 不使用 USB 的 STOP 模式唤醒功能时, 进入 STOP 模式时禁止上拉	保持或者高阻 由于该管脚的电平被拉高时会产生冗余电流, 进入掉电模式时禁止上拉
PI13/MD	工作可设	保持	保持 芯片外面通过电阻接到地
PA13-PA15, PB3, PB4	工作可设 作为 JTAG 功能时, 内置上拉电路有效	保持 作为 JTAG 功能时, 内置上拉电路有效	保持 作为 JTAG 功能时, 内置上拉电路有效

6.2.4.2 运行模式

芯片有高速、超低速等两种运行模式。根据系统的速度要求, 设置最佳的运行模式, 使内核电压、驱动能力适配于系统时钟频率, 从而达到降低功耗的目的。如下表所示, 根据 PWC_PWRC2 的 DVS 和 DDAS 位, 可以让芯片工作在相应的运行模式。芯片选择动作在超低速模式时, FLASH 和 RAM 也需要设定工作在低内核电压下, 因此需要设置寄存器位 FRMC.LVM=0b1、PWC_RAMOPM=0x9062。

表 6-7 运行模式说明

运行模式	频率范围	寄存器设置		
		PWC_PWRC2.DVS	PWC_PWRC2.DDAS	PWC_PWRC3.DDAS
高速运行模式	≤240MHz	0b11	0b1111	0b11111111
超低速运行模式*	≤8MHz	0b10	0b0000	0b00000000



说明

*: 超低速模式只支持 $T_A = -40^{\circ}\text{C} \sim 85^{\circ}\text{C}$ 。

高速运行模式与超低速运行模式之间的切换需要遵循如下流程 1 和流程 2。

● **高速模式向超低速模式切换**

1. 设置在超低速模式下使用的时钟源，确保时钟源符合超低速模式下的频率要求；
2. 关闭在超低速模式下不需要使用的时钟源及模块，确认 Flash 不处于编程或者擦除状态；
3. 设置 FLASH 的 FRMC.LVM=0b1，RAM 动作模式寄存器 PWC_RAMOPM 设置成 0x9062；
4. 确认 FRMC.LVM=0b1，PWC_RAMOPM=0x9062；
5. 设置 PWC_PWRC2.DDAS[3:0]为 0b0000，设置 PWC_PWRC3.DDAS[7:0]为 0x00，PWC_PWRC2.DVS[1:0]为 0b10；
6. 等待 TSWMD1 (30μs) ；
7. 芯片在超低速模式下动作。

● **超低速模式向高速模式切换**

1. 设置 PWC_PWRC2.DDAS[3:0]为 0b1111，PWC_PWRC3.DDAS[7:0]为 0xFF，PWC_PWRC2.DVS[1:0]根据系统频率要求设置为 0b11；
2. 等待 TSWMD2 (30μs) ；
3. 设置 FLASH 的 FRMC.LVM=0b0，RAM 动作模式寄存器 PWC_RAMOPM 设置成 0x8043；
4. 确认 FRMC.LVM=0b0，PWC_RAMOPM=0x8043；
5. 芯片在高速模式下动作。

6.2.4.3 睡眠模式

在睡眠模式中，CPU 停止运行，它的内部寄存器保持进入睡眠模式之前的状态。看门狗和专用看门狗以外的外设和其他系统模块的动作状态不会改变。

通过 ICG 设置为自动启动时，如果 ICG0 的 WDTSLPOFF 位为 0b1，看门狗在睡眠模式下停止计数；如果 WDTSLPOFF 位为 0b0，看门狗在睡眠模式下继续计数。如果 ICG 未设置为自动启动，通过软件启动的方式启动看门狗，则如果 WDT_CR.SLPOFF 位为 0b1，看门狗在睡眠模式下停止计数；如果 WDT_CR.SLPOFF 位为 0b0，看门狗在睡眠模式下不停止计数。

通过 ICG 设置为自动启动时，如果 ICG0 的 SWDTSLPOFF 位为 0b1，专用看门狗在睡眠模式下停止计数；如果 SWDTSLPOFF 位为 0b0，专用看门狗在睡眠模式下继续计数。

● **进入睡眠模式**

在 PWC_STPMCR.STOP=0b0 时执行 WFI 指令即可进入睡眠模式。

● **退出睡眠模式**

任意中断、复位都可以将芯片从睡眠模式唤醒。通过中断唤醒时，芯片进入中断处理程序；通过复位退出睡眠模式时，芯片进入复位状态。

6.2.4.4 停止模式

在停止模式中，CPU、大部分外设和时钟源都停止动作。芯片保持 CPU 内部寄存器和 SRAM 数据，外设状态和管脚状态。在停止模式下，由于大部分时钟源停止工作，调压器也降低了驱动能力，因此芯片功耗会显著降低。

通过 ICG 设置为自动启动时，如果 ICG0 的 SWDTSLP0FF 位为 0b1，专用看门狗在停止模式下停止计数；如果 SWDTSLP0FF 位为 0b0，专用看门狗在停止模式下继续计数。

执行 WFI 指令进入停止模式之前，需要确保 FLASH 不处于编程或者擦除状态，并且振荡停止检测功能无效，否则芯片会进入睡眠模式而非停止模式。

执行 WFI 指令进入停止模式之前，必须确保 PWC_PWRC0.PDMDS[1:0]配置为 0b00。

在停止模式下，ADC 和 DAC 也会发生功耗，除非在进入停止模式前将其禁止。要禁止 DAC，需要将 DACR.DAE，DACR.DA1E，DACR.DA2E 清零。要禁止 ADC，需要 ADC_STR.STRT 位清零，置位 PWC_FCG3.ADC3、PWC_FCG3.ADC2、PWC_FCG3.ADC1 后，执行 WFI 指令进入停止模式。

STOP 模式唤醒时，通过 PWC_STPMCR 寄存器的位 CKSMRC 和 FLNWT 来选择唤醒后的时钟及是否需要等待 Flash 稳定。CKSMRC 用于控制唤醒后的时钟源，CKSMRC=0b1 时，唤醒后的系统时钟源选择为 MRC；CKSMRC=0b0 时，唤醒后的系统时钟维持进入 STOP 之前的时钟源不变。FLNWT 用于控制唤醒后是否等待 Flash 稳定，FLNWT=0b0 时，唤醒时需要等待 Flash 稳定；FLNWT=0b1 时，唤醒是不需要等待 Flash 稳定。FLNWT 必须在程序运行在 RAM 上时才能置位，否则芯片从 STOP 唤醒后的动作不可保证。在程序运行在 RAM 上进入 STOP 模式时，选择 CKSMRC=0b1，FLNWT=0b1 将可以在最短的时间内唤醒系统。

执行 WFI 指令进入停止模式之前，需要确保 DMA 处于停止状态，否则芯片可能会出现不可保证的动作。

执行 WFI 指令进入停止模式之前，需要确保 HCLK 与 PCLK1/2/3/4、EXCLK 的分频比例控制在 1:1 或 1:2 或 1:4。若 HCLK 与任一外设时钟分频比例超过 1:4，需要配置 PCLK1=PCLK3 且 PCLK1 与 PCLK3 为最慢时钟。

STOP 模式下的漏电流在不同的电压温度是不同的，设置的驱动能力必须满足芯片的漏电需要。

执行 WFI 指令进入停止模式之前，需要将 EIRQ 的数字滤波设置为无效，否则该中断不能用于 STOP 唤醒。

芯片在超低速模式下需要设置 PWC_PWRC1.STPDAS=0b11 后再进入 STOP 模式；如果 PWC_PWRC1.STPDAS=0b00 时进入 STOP 模式，在 STOP 模式中芯片会消耗更多的电流。

通过非可屏蔽中断解除停止模式时，需要设置 INTC_NMIER 的相应位使能该中断；通过可屏蔽中断解除停止模式时，需要设置 INTC_WKEN 寄存器的相应位使能该中断的唤醒许可。

● 进入停止模式

步骤 1. 清除 PWC_LPMCSR.LPMF；

步骤 2. 在 PWC_STPMCR.STOP=0b1，PWC_PWRC0.PWDN=0b0 时执行 WFI 指令即可进入停止模式；

步骤 3. 读 PWC_LPMCSR.LPMF，如果该位为 1，芯片正常进入了 STOP 模式，如果该位为 0，返回步骤 2。

芯片的外设和时钟源在停止模式下的状态请参见表“[低功耗模式的运行条件及各模块在低功耗模式下的状态](#)”。

● 解除停止模式

停止模式可以通过复位和中断解除。可以用于解除停止模式的复位方式有管脚复位，上电复位，欠压复位（BOR），可编程电压检测 1/2 复位，专用看门狗复位。可以用于解除停止模式的中断事件：管脚中断 EIRQ0-15、电压检测 1 中断、电压检测 2 中断、专用看门狗下溢中断、实时时钟的周期中断、闹钟中断、唤醒定时器中断、比较器中断、USART1 RX 中断、Timer0 比较匹配中断、Timer2 比较匹配中断。芯片通过中断解除停止模式时，首先启动进入停止模式之前的使用到的时钟源。在所有的时钟源稳定之后，芯片解除停止模式。

执行 WFI 指令进入停止模式的同时产生了不用于解除停止模式的中断事件，芯片会优先响应该中断而不进入 STOP 模式。如果需要优先进入 STOP 模式而不执行该中断，在执行 WFI 指令前需要关闭不用于解除停止模式的中断。

6.2.4.5 掉电模式

在掉电模式中，VDD 域的所有模块的电源被切断，功耗可以达到最低。

通过 ICG 设置为 SWDT 自动启动时：

- 如果 ICG0.SWDTSLPOFF 为 0b1，专用看门狗将同 VDD 域的其他模块一样，电源被切断，不再计数；
- 如果 ICG0.SWDTSLPOFF 为 0b0，芯片将进入停止模式而非掉电模式，专用看门狗的振荡器和专用看门狗将继续运行。

通过软件刷新寄存器设置 SWDT 启动时：

- 如果 SWDT_CR.SLPOFF 为 0b1，专用看门狗将同 VDD 域的其他模块一样，电源被切断，不再计数；
- 如果 SWDT_CR.SLPOFF 为 0b0，芯片将进入停止模式而非掉电模式，专用看门狗的振荡器和专用看门狗将继续运行。

当电压检测 1 和电压检测 2 的复位使能时，芯片将进入停止模式而非掉电模式。

执行 WFI 指令进入掉电模式之前，需要确保 FLASH 不处于编程或者擦除状态，并且振荡停止检测功能无效，否则芯片会进入睡眠模式而非掉电模式。

芯片的 VCAP_1/VCAP_2 管脚使用的电容如下：

- 同时存在 VCAP_1 和 VCAP_2 管脚的芯片，每个管脚可以使用 0.047μF 或者 0.1μF 电容（总容量为 0.094μF 或者 0.2μF）。
- 只有 VCAP_1 管脚的芯片，可以使用 0.1μF 或者 0.22μF 电容。

从掉电模式唤醒时，内核电压建立过程中需要给 VCAP_1/VCAP_2 充电。一方面，较小的 VCAP_1/VCAP_2 总容量能够缩短充电时间，为应用带来快速响应能力；另一方面，较大的 VCAP_1/VCAP_2 总容量会延长充电时间，但是也提供更强的电磁兼容性（EMC）。用户可以根据电磁兼容性和系统响应速度的要求，选择较小或者较大的电容。芯片的 VCAP_1/VCAP_2 总容量必须与 PWC_PWRC1.PDTS 位的赋值相匹配。VCAP_1/VCAP_2 的总容量为 0.2μF 或者 0.22μF 时，需要在进入掉电模式之前确保 PWC_PWRC1.PDTS 位清零。VCAP_1/VCAP_2 的总容量为 0.094μF 或者 0.1μF 时，需要在进入掉电模式之前确保 PWC_PWRC1.PDTS 位置位。

通过设置 PWC_PWRC0.PDMDS[1:0]，可以进一步降低掉电模式的功耗。掉电模式的子模式如下表所示。

- 掉电模式 1 中，电压检测电路可以使用，上电复位检测电路处于动作状态，由于唤醒时不需要等待 VCC 域参考电压、电压检测电路和上电复位检测电路的稳定，因此在实现低功耗的同时，唤醒时间最短。
- 掉电模式 2 中，VCC 域参考电压电路，电压检测电路停止工作，上电复位检测电路处于动作状态，唤醒时需要等待 VCC 域参考电压电路、电压检测电路的稳定时间。
- 掉电模式 3 中，VCC 域参考电压电路、电压检测电路、上电复位检测电路都停止工作，唤醒时需要等待这些电路的稳定，因此在实现最低功耗的同时，唤醒时间比掉电模式 2 和掉电模式 1 长。
- 掉电模式 4 与掉电模式 3 中停止工作的电路相同，因此掉电模式 4 与掉电模式 3 具有相同的功耗。

具体功耗数值和唤醒时间，请参见《数据手册》的“电气特性”章节。

表 6-8 掉电模式子模式

掉电模式	PDMDS[1:0]	功耗	唤醒时间	说明
掉电模式 1	0b00	I_{PD1}	T_{PD1}	VCC 域电源电压检测单元有效
掉电模式 2	0b01	I_{PD2}	T_{PD2}	VCC 域 POR、PDR 检测电路有效，BOR、PVD1、PVD2 无效
掉电模式 3	0b10	I_{PD3}	T_{PD3}	VCC 域 POR、PDR，BOR、PVD1、PVD2 无效
掉电模式 4	0b11	I_{PD4}	T_{PD4}	VCC 域 POR、PDR，BOR、PVD1、PVD2 无效

功耗和唤醒时间的关系： $I_{PD1} > I_{PD2} > I_{PD3} = I_{PD4}$ ， $T_{PD1} < T_{PD2} < T_{PD3} < T_{PD4}$

● 进入掉电模式

- 步骤 1. 清除 PWC_LPMCSR.LPMF;
- 步骤 2. 在 PWC_STPMCR.STOP=0b1, PWC_PWRC0.PWDN=0b1 时执行 WFI 指令即可进入掉电模式;
- 步骤 3. 读 PWC_LPMCSR.LPMF, 如果该位为 0, 返回步骤 2; 如果正常产生掉电复位, 步骤 3 不会被执行。

● 解除掉电模式

掉电模式可以通过掉电模式唤醒事件或者复位解除。能够用于唤醒掉电模式的复位有管脚复位、上电复位和电压检测 0 复位。能够用于掉电模式唤醒的事件包括: WKUPn_0/1/2/3 (n=0/1/2/3) 唤醒事件、实时时钟的闹钟和定时事件、电压检测 1 唤醒事件、电压检测 2 唤醒事件、唤醒定时器唤醒事件。

从掉电模式 1、掉电模式 2 唤醒后, 芯片复位后重新执行程序。唤醒事件可以通过掉电唤醒标志位查询, 复位标志位可以通过 RSTF0.PDRF 查询。

在掉电模式 3 中 POR、PDR, BOR、PVD1、PVD2 电路都处于无效状态, 从掉电模式 3 唤醒后, 除 PWC_PDWKFO/PWC_PDWKF1/RSTF0 外的所有寄存器都被复位, 芯片按照类似上电复位的方式工作; 复位标志位可以通过 RMU_RSTF0.PDRF 查询。

在掉电模式 4 中 POR、PDR, BOR、PVD1、PVD2 电路都处于无效状态, 芯片复位后重新执行程序; 复位标志位可以通过 RMU_RSTF0.PDRF 查询。

掉电唤醒事件通过掉电唤醒使能寄存器 (PWC_PDWKE0-PDWKE3)、掉电唤醒事件边沿选择寄存器 (PWC_PDWKES) 进行控制。当发生掉电唤醒事件时, 该事件相对应的掉电唤醒标志 (PWC_PDWKFO-PWC_PDWKF1) 被置位。掉电唤醒后, 如果不清除掉电唤醒标志, 芯片不能再次进入掉电模式。掉电唤醒事件的边沿可以通过 PWC_PDWKES 选择。

掉电模式唤醒时, VDD 域将重新供电, 系统执行掉电唤醒复位, 工作时钟内部中速振荡器。掉电模式唤醒时不被复位的寄存器如下表。

掉电模式	不被复位的寄存器
掉电模式 1、掉电模式 2、掉电模式 4	PWC_PWRC0、PWC_PWRC1、PWC_PWRC3、PWC_PDWKE0、PWC_PDWKE1、PWC_PDWKE2、PWC_PDWKES、PWC_PDWKFO、PWC_PDWKF1、PWC_PVDCR0、PWC_PVDCR1、PWC_PVDFCR、PWC_PVDLCR、PWC_PVDICR、PWC_PVDDSR
掉电模式 3	PWC_PDWKFO、PWC_PDWKF1、RMU_RSTF0

● 解除掉电模式后的管脚状态

在掉电模式中, 根据寄存器设定值, 芯片管脚将保持进入掉电模式之前的状态或者高阻状态。如果 PWC_PWRC0.IORTN[1:0]=0b10 或者 0b11, 管脚状态在掉电模式中为高阻状态, 掉电模式解除后管脚初始成高阻状态。如果 PWC_PWRC0.IORTN[1:0]=0b00, 管脚状态在掉电模式中保持掉电模式前的状态, 唤醒后管脚初始成高阻状态。如果 PWC_PWRC0.IORTN[1:0]=0b01, 芯片管脚将保持进入掉电模式之前的状态, 唤醒后即使对外设或者管脚的寄存器进行设定, 芯片管脚的状态也不会改变。通过软件将 PWC_PWRC0.IORTN 清零后, 管脚状态才受外设或者管脚的寄存器设定所控制。

● WKTM 掉电模式唤醒事件

芯片内置了用于掉电唤醒的计数器 WKTM, 该计数器可以选择内部低速振荡器、外部低速振荡器, 在 RTC 使用外部低速振荡器作为时钟动作时, WKTM 还可以选择 64Hz 的内部时钟信号作为时钟源。该计数器是累加计数器, 在 WKTC0.WKTCE 置位后, 计数器开始计数, 当计数值等于 WKTCMP[11:0]设定值的一个周期后, 计数停止并产生唤醒事件, 使芯片从掉电模式唤醒。再次使用 WKTM 时需要将 WKTC0.WKTCE 复位后再次置位。

● PTWK 掉电模式唤醒事件

芯片具有 4 个用于掉电模式唤醒的 PTWK 事件, PTWK0、PTWK1、PTWK2、PTWK3。通过软件设定 PWC_PDWKE0/PWC_PDWKE1 可以使能 WKUPn_0、WKUPn_1、WKUPn_2、WKUPn_3 等 4 个管脚中的一个用于触发 PTWK 事件。每个 PTWK 事件可以选择触发管脚的上升或者下降触发边沿, 并具有独立的标志位。

PTWK 掉电唤醒事件配置流程如下:

1. 配置 PWC_PDWKES 的相应位选择边沿;
2. 配置 PWC_PDWKE0 和 PWC_PDWKE1 使能相应的管脚;
3. 清除 PWC_PDWKFO 中的相应标志。

配置 PTWK_n 的结构框图如下图所示。

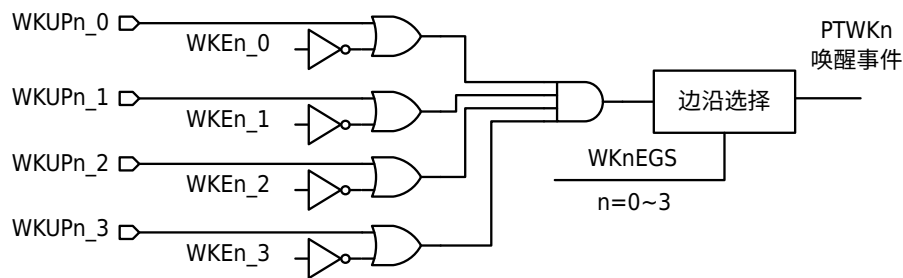


图 6-12 PTWK_n 结构框图

● 电池备份域在掉电模式时的动作说明

电池备份域在芯片进入掉电模式后, Ret SRAM 通过 RLDO 继续供电, RTC/WKTM/备份寄存器通过 VCC 继续供电, 因此 RTC/WKTM/Ret SRAM 都能继续动作或者保持数据。掉电唤醒后 VDDR 域由 LDO 供电。进入掉电模式前, 需要关闭 VBAT 电压测量功能。

● 降低功耗的方法

可以通过下列方法优化运行模式下的功耗。

1. 设定最适的运行模式;
2. 降低系统时钟速度;
3. 关闭不使用的时钟源;
4. 设定功能时钟控制寄存器 PWC_FCG_n (n=0/1/2/3) 关闭不需要使用的功能;
5. 关闭 RAM 的电源。

6.2.4.6 降低系统时钟速度

在运行模式下, 可通过对预分频寄存器编程来降低系统时钟 (HCLK)、外部总线时钟 (EXCLK)、周边外设时钟 PCLK0/PCLK1/PCLK2/PCLK3/PCLK4 的速度。进入睡眠模式前, 也可以使用这些预分频器降低外设速度。详细信息请参见《时钟控制器 (CMU) 》。

6.2.4.7 关闭不使用的时钟源

芯片的系统时钟有 6 个时钟源:

- 外部高速振荡器 (XTAL)
- 外部低速振荡器 (XTAL32)
- PLLH 时钟 (PLLH)
- 内部高速振荡器 (HRC)
- 内部中速振荡器 (MRC)
- 内部低速振荡器 (LRC)

SWDT 具有独立的专用内部低速振荡器 (SWDTLRC)；RTC 可以选择外部低速振荡器或者 RTC 用内部低速振荡器 (RTCLRC) 作为时钟源。HRC 和 PLL 都配有独立的电源电路，在 HRC 关闭后，可以通过置位 PWC_PWRC1.VHRCSD 位关闭 HRC 的电源，进一步降低功耗；在 PLLA 和 PLLH 都被关闭后，可以通过设置 PWC_PWRC1.VPLLSD=0b11 关闭 PLLA 和 PLLH 的电源。

详细信息请参见《时钟控制器 (CMU)》。

6.2.4.8 功能时钟停止

芯片的外设模块设有功能时钟停止功能，通过将寄存器对应的位置位，可以将不需要使用的模块停止运行，相应模块的时钟也停止供给，降低功耗。在模块停止状态下，模块内部的寄存器将维持停止之前的状态。

6.2.4.9 关闭不使用的 RAM

芯片里的每个 RAM 模块都配置了功能时钟停止位，掉电控制位，通过置位模块停止位停止给不需要使用的 RAM 提供时钟，从而降低功耗。通过置位模块的掉电控制位，可以让相应的 RAM 模块掉电，从而降低功耗。下表是 RAM 模块与掉电控制位的对应关系，通过设置 PWC_RAMPC0 寄存器的相应 RAMPDCn (n=0-10)、PWC_VBATCR.VBTRSD 位可以使对应的 RAM 掉电。

表 6-10 RAM 模块与 RAM 掉电控制位

RAM 模块	说明	掉电控制位
SRAM1	0x20000000~0x20000FFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC0
	0x20010000~0x20001FFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC1
SRAM2	0x20020000~0x2002FFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC2
	0x20030000~0x2003FFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC3
SRAM3	0x20040000~0x2004FFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC4
	0x20050000~0x20057FFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC5
SRAM4	0x20058000~0x2005FFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC6
SRAMH	0x1FFE0000~0x1FFE7FFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC7
	0x1FFE8000~0x1FFEFFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC8
	0x1FFF0000~0x1FFF7FFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC9
	0x1FFF8000~0x1FFFFFFF 地址空间用 RAM	PWC_RAMPC0.RAMPDC10

RAM 模块	说明	掉电控制位
Ret SRAM	0x200F0000~0x200F0FFF 地址空间用 RAM	PWC_VBATCR.VBTRSD
CAN_1 RAM	CAN 单元 1 用 RAM	PWC_PRAMLPC.PRAMDC0
CAN_2 RAM	CAN 单元 2 用 RAM	PWC_PRAMLPC.PRAMDC1
CACHERAM	Cache 用 RAM	PWC_PRAMLPC.PRAMDC2
USBFS RAM	USBFS 用 RAM	PWC_PRAMLPC.PRAMDC3
USBHS RAM	USBHS 用 RAM	PWC_PRAMLPC.PRAMDC4
ETHMAC TX RAM	ETHMAC TX 用 RAM	PWC_PRAMLPC.PRAMDC5
ETHMAC RX RAM	ETHMAC RX 用 RAM	PWC_PRAMLPC.PRAMDC6
SDIOC1 RAM	SDIOC1 用 RAM	PWC_PRAMLPC.PRAMDC7
SDIOC2 RAM	SDIOC2 用 RAM	PWC_PRAMLPC.PRAMDC8
NFCRAM	NFC 用 RAM	PWC_PRAMLPC.PRAMDC9
MCAN RAM	MCAN 用 RAM	PWC_PRAMLPC.PRAMDC10
PKE RAM	PKE 用 RAM	PWC_PRAMLPC.PRAMDC11

6.3 寄存器

寄存器保护功能用于将寄存器的写操作无效，以保护寄存器被意外改写。

表 6-11 寄存器保护列表

保护寄存器位	被保护寄存器
PWC_FPRC.FPRCB0	CMU_XTALCFGR、CMU_XTALSTBCR、CMU_XTALCR、CMU_XTALSTDCR、CMU_XTALSTDSR、CMU_HRCTRM、CMU_HRCCR、CMU_MRCTRM、CMU_MRCCR、CMU_PLLHCFGR、CMU_PLLHCR、CMU_PLLACFGR、CMU_PLLACR、CMU_OSCSTBSR、CMU_CKSWR、CMU_SCFGR、CMU_USBCKCFGR、CMU_CANCKCFGR、CMU_TPIUCKCFGR、CMU_MCO1CFGR、CMU_MCO2CFGR、CMU_XTAL32CR、CMU_XTALC32CFGR、CMU_XTAL32NFR、CMU_LRCCR、CMU_LRCTRM、CMU_RTCLRCTRM
PWC_FPRC.FPRCB1	PWC_PWRC0、PWC_PWRC1、PWC_PWRC2、PWC_PWRC3、PWC_PWRC4、PWC_PDWKE0、PWC_PDWKE1、PWC_PDWKE2、PWC_PDWKES、PWC_PDWKF0、PWC_PDWKF1、CMU_PERICKSEL、CMU_I2SCKSEL、PWC_STPMCR、PWC_RAMPC0、PWC_RAMOPM、PWC_PRAMLPC、RMU_RSTF0、RMU_FRST0、RMU_FRSTR1、RMU_FRST2、RMU_FRST3、RMU_PRSTCR0、PWC_VBATRSTR、PWC_VBATCR、PWC_WKTC0、PWC_WKTC1、PWC_WKTC2
PWC_FPRC.FPRCB3	PWC_PVDCR0、PWC_PVDCR1、PWC_PVDFCR、PWC_PVDLCR、PWC_PVDICR、PWC_PVDDSR
PWC_FCG0PC.PRT0	PWC_FCG0

6.3.1 寄存器总表

PWC 基地址：0x40048000

表 6-12 电源控制 (PWC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	PWC_FCG0	功能时钟控制 0	32
0x0004	PWC_FCG1	功能时钟控制 1	32
0x0008	PWC_FCG2	功能时钟控制 2	32
0x000C	PWC_FCG3	功能时钟控制 3	32
0x0010	PWC_FCG0PC	PWC_FCG0 保护控制	32
0x4C00	PWC_PWRC0	电源模式控制寄存器 0	8
0x4C04	PWC_PWRC1	电源模式控制寄存器 1	8
0x4C08	PWC_PWRC2	电源模式控制寄存器 2	8
0x4C0C	PWC_PWRC3	电源模式控制寄存器 3	8
0x4C10	PWC_PWRC4	电源模式控制寄存器 4	8
0x4C14	PWC_PVDCR0	PVD 控制寄存器 0	8

偏移地址	寄存器	描述	位宽
0x4C18	PWC_PVDCR1	PVD 控制寄存器 1	8
0x4C1C	PWC_PVDFCR	PVD 滤波控制寄存器	8
0x4C20	PWC_PVDLCR	PVD 电平控制寄存器	8
0x4C28	PWC_PDWKE0	掉电唤醒使能寄存器 0	8
0x4C2C	PWC_PDWKE1	掉电唤醒使能寄存器 1	8
0x4C30	PWC_PDWKE2	掉电唤醒使能寄存器 2	8
0x4C34	PWC_PDWKES	掉电唤醒事件边沿选择寄存器	8
0x4C38	PWC_PDWKFO	掉电唤醒标志寄存器 0	8
0x4C3C	PWC_PDWKF1	掉电唤醒标志寄存器 1	8
0x4CE0	PWC_RAMPC0	RAM 功耗控制寄存器 0	32
0x4CE4	PWC_RAMOPM	RAM 运行条件寄存器	16
0x4CE8	PWC_PRAMLPC	外设 RAM 低功耗控制寄存器	32
0x4CEC	PWC_LPMCSR	低功耗状态寄存器	16
0x4CEE	PWC_LPMCSCR	低功耗状态清除寄存器	16
0x4CF0	PWC_PVDICR	PVD 中断控制寄存器	8
0x4CF4	PWC_PVDDSR	PVD 检测状态寄存器	8
0xC00C	PWC_STPMCR	STOP 模式控制寄存器	16
0xC3FE	PWC_FPRC	功能保护控制寄存器	16
0x4430	PWC_VBATRSTR	备份域复位寄存器	8
0x4440	PWC_VBATCR	备份域控制寄存器	8
0x4448	PWC_VBATPORF	备份域上电复位标志寄存器	8
0x444C	PWC_VBATPORFCLR	备份域上电复位标志清除寄存器	8
0x4450	PWC_WKTC0	唤醒计时器控制寄存器 0	8
0x4454	PWC_WKTC1	唤醒计时器控制寄存器 1	8
0x4458	PWC_WKTC2	唤醒计时器控制寄存器 2	8
0x4600+0x4*n	PWC_BKRn	VBAT 备份寄存器 n (n=0~127)	8

6.3.2 电源模式控制寄存器 0 (PWC_PWRC0)

Offset	Bit Position							
0x4C00	7	6	5	4	3	2	1	0
Reset	0x00							
Name	PWDN	Reserved	IORTN		Reserved		PDMDS	
Access	RW		RW				RW	

位/位域	标记	位名	功能描述	读写
7	PWDN	掉电模式控制位	0b0: 掉电模式无效 0b1: 掉电模式使能	RW
6	Reserved	保留	-	-
5:4	IORTN	掉电模式下 IO 保持控制	0b00: 掉电模式中 IO 保持状态, 掉电唤醒后硬件解除 IO 保持状态 0b01: 掉电模式中 IO 保持状态, 掉电唤醒后将 IORTN[1:0]设为 0b00, 解除 IO 保持状态 0b10/0b11: 掉电模式中及掉电唤醒后 IO 为高阻	RW
3:2	Reserved	保留	-	-
1:0	PDMDS	掉电模式选择控制	0b00: 掉电模式 1 0b01: 掉电模式 2 0b10: 掉电模式 3 0b11: 掉电模式 4	RW

6.3.3 电源模式控制寄存器 1 (PWC_PWRC1)

Offset	Bit Position							
0x4C04	7	6	5	4	3	2	1	0
Reset	0x00							
Name	STPDAS		Reserved		PDTS	VHRCSD	VPLLSD	
Access	RW				RW	RW	RW	

位/位域	标记	位名	功能描述	读写
7:6	STPDAS	STOP 模式 LDO 驱动选择	0b00: 高速模式模式进入 STOP 模式时设定的驱动能力 0b11: 超低速模式进入 STOP 模式时设定的驱动能力 - 其他: 禁止设置  说明 设置 STPDAS=0b11 时只支持 $T_A=-40^{\circ}\text{C}\sim 105^{\circ}\text{C}$。	RW
5:4	Reserved	保留	-	-
3	PDTS	掉电唤醒时间控制位	0b0: 当 VCAP_1/VCAP_2 总电容是 2 个 0.1μF 时或者 1 个 0.22μF 时, 进入掉电模式之前, PDTS 位必须清零 0b1: VCAP_1/VCAP_2 总电容是 2 个 0.047μF 时或者 1 个 0.1μF 时, 进入掉电模式之前, PDTS 位必须置位	RW
2	VHRCSD	HRC 电源关闭	0b0: HRC 电源使能 0b1: HRC 电源关闭  说明 在 HRC 不使用时, 置位 VHRCSD 后关闭 HRC 用电源, 进一步降低功耗。VHRCSD 清零后, 需要等待 25μs 后再启动 HRC 模块。	RW
1:0	VPLLSD	PLL 电源关闭	0b00: PLL 电源使能 0b11: PLL 电源关闭 - 其他: 禁止设置  说明 在 PLLA 关闭, PLLH 关闭并等待 50μs 后, 设置 VPLLSD=0b11 后关闭 PLL 用电源, 进一步降低功耗。设置 VPLLSD=0b00 后, 需要等待 25μs 后再启动 PLLA 或者 PLLH 模块。	RW

6.3.4 电源模式控制寄存器 2 (PWC_PWRC2)

Offset	Bit Position							
0x4C08	7	6	5	4	3	2	1	0
Reset	0xFF							
Name	Reserved		DVS		DDAS[11:8]			
Access			RW		RW			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	所有位读出的 1，写入时必须写 1	-
5:4	DVS	动作模式下电压选择	0b00：选择超低速动作电压 0b11：选择高速动作模式电压 - 其他：禁止设置	RW
3:0	DDAS[11:8]	电源驱动选择	与 PWC_PWRC3 中的 DDAS[7:0]联合，用以调整驱动能力。DDAS[11:0]取值： 0x000：超低速驱动模式 0xFFF：高速动作模式 - 其他：禁止设置	RW

6.3.5 电源模式控制寄存器 3 (PWC_PWRC3)

Offset	Bit Position							
0x4C0C	7	6	5	4	3	2	1	0
Reset	0xFF							
Name	DDAS[7:0]							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	DDAS[7:0]	电源驱动选择	与 PWC_PWRC2 中的 DDAS[11:8]联合，用以调整驱动能力。DDAS[11:0]取值： 0x000：超低速驱动模式 0xFFF：高速动作模式 - 其他：禁止设置	RW

6.3.6 电源模式控制寄存器 4 (PWC_PWRC4)

Offset	Bit Position							
0x4C10	7	6	5	4	3	2	1	0
Reset	0x00							
Name	ADBUFE	ADBUFS	Reserved	VBATMON	Reserved		VBATME	VBATREFSEL
Access	RW	RW		RO			RW	RW

位/位域	标记	位名	功能描述	读写
7	ADBUFE	内部电压采样使能	0b0: 无效 0b1: 有效  说明 使用 ADC 采样芯片内部电压时，该位需要设置为 1。	RW
6	ADBUFS	采样内部电压选择位	0b0: 内部基准电压 0b1: VBAT 的 1/2 分压	RW
5	Reserved	保留	-	-
4	VBATMON	VBAT 电压检测状态	0b0: VBAT>VBATREF 0b1: VBAT<VBATREF	RO
3:2	Reserved	保留	该位禁止写 1	-
1	VBATME	VBAT 检测使能	0b0: 检测无效 0b1: 检测使能	RW
0	VBATREFSEL	VBAT 检测用参考电压 (VBATREF) 选择	0b0: 1.8V 0b1: 2.1V	RW

6.3.7 掉电唤醒使能寄存器 0 (PWC_PDWKE0)

Offset	Bit Position							
0x4C28	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKE13	WKE12	WKE11	WKE10	WKE03	WKE02	WKE01	WKE00
Access	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	WKE13	WKUP1_3 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
6	WKE12	WKUP1_2 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
5	WKE11	WKUP1_1 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
4	WKE10	WKUP1_0 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
3	WKE03	WKUP0_3 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
2	WKE02	WKUP0_2 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
1	WKE01	WKUP0_1 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
0	WKE00	WKUP0_0 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW

6.3.8 掉电唤醒使能寄存器 1 (PWC_PDWKE1)

Offset	Bit Position							
0x4C2C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKE33	WKE32	WKE31	WKE30	WKE23	WKE22	WKE21	WKE20
Access	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	WKE33	WKUP3_3 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
6	WKE32	WKUP3_2 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
5	WKE31	WKUP3_1 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
4	WKE30	WKUP3_0 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
3	WKE23	WKUP2_3 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
2	WKE22	WKUP2_2 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
1	WKE21	WKUP2_1 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW
0	WKE20	WKUP2_0 唤醒事件使能	● 0b0: 禁止 ● 0b1: 使能	RW

6.3.9 掉电唤醒使能寄存器 2 (PWC_PDWKE2)

Offset	Bit Position							
0x4C30	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKTMWKE	Reserved	RTCALMWKE	RTCPRDWKE	Reserved		VD2WKE	VD1WKE
Access	RW		RW	RW			RW	RW

位/位域	标记	位名	功能描述	读写
7	WKTMWKE	WKT M 唤醒事件使能	0b0: 禁止 0b1: 使能	RW
6	Reserved	保留	-	-
5	RTCALMWKE	RTC 闹钟唤醒事件使能	0b0: 禁止 0b1: 使能	RW
4	RTCPRDWKE	RTC 周期唤醒事件使能	0b0: 禁止 0b1: 使能	RW
3:2	Reserved	保留	-	-
1	VD2WKE	PVD2 唤醒事件使能	0b0: 禁止 0b1: 使能	RW
0	VD1WKE	PVD1 唤醒事件使能	0b0: 禁止 0b1: 使能	RW

6.3.10 掉电唤醒事件边沿选择寄存器 (PWC_PDWKES)

Offset	Bit Position							
0x4C34	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved		VD2EGS	VD1EGS	WK3EGS	WK2EGS	WK1EGS	WK0EGS
Access			RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5	VD2EGS	VD2 边沿选择	0b0: VCC<VPVD2 0b1: VCC>VPVD2	RW
4	VD1EGS	VD1 边沿选择	0b0: VCC<VPVD1 0b1: VCC>VPVD1	RW
3	WK3EGS	PTWK3 边沿选择	0b0: 下降沿 0b1: 上升沿	RW
2	WK2EGS	PTWK2 边沿选择	0b0: 下降沿 0b1: 上升沿	RW
1	WK1EGS	PTWK1 边沿选择	0b0: 下降沿 0b1: 上升沿	RW
0	WK0EGS	PTWK0 边沿选择	0b0: 下降沿 0b1: 上升沿	RW

6.3.11 掉电唤醒标志寄存器 0 (PWC_PDWKFO)

Offset	Bit Position							
0x4C38	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved		VD2WKF	VD1WKF	PTWK3F	PTWK2F	PTWK1F	PTWK0F
Access			RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5	VD2WKF	PVD2 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW
4	VD1WKF	PVD1 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW
3	PTWK3F	PTWK3 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW
2	PTWK2F	PTWK2 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW
1	PTWK1F	PTWK1 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW
0	PTWK0F	PTWK0 唤醒标志位	掉电唤醒后, 需要写 0 清除标志 读取: ● 0b0: 未发生唤醒事件 ● 0b1: 发生唤醒事件	RW

6.3.12 掉电唤醒标志寄存器 1 (PWC_PDWKF1)

Offset	Bit Position							
0x4C3C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKTMWKF	Reserved	RTCALMWKF	RTCPRDWKF	Reserved			
Access	RW		RW	RW				

位/位域	标记	位名	功能描述	读写
7	WKTMWKF	WKT M 唤醒标志位	掉电唤醒后，需要写 0 清除标志 读取： ● 0b0：未发生唤醒事件 ● 0b1：发生唤醒事件	RW
6	Reserved	保留	-	-
5	RTCALMWKF	RTC 闹钟唤醒标志位	掉电唤醒后，需要写 0 清除标志 读取： ● 0b0：未发生唤醒事件 ● 0b1：发生唤醒事件	RW
4	RTCPRDWKF	RTC 周期唤醒标志位	掉电唤醒后，需要写 0 清除标志 读取： ● 0b0：未发生唤醒事件 ● 0b1：发生唤醒事件	RW
3:0	Reserved	保留	-	-

6.3.13 RAM 功耗控制寄存器 0 (PWC_RAMPC0)

Offset	Bit Position																															
0x4CE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																					RAMPDC10	RAMPDC9	RAMPDC8	RAMPDC7	RAMPDC6	RAMPDC5	RAMPDC4	RAMPDC3	RAMPDC2	RAMPDC1	RAMPDC0
Access																						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10	RAMPDC10	RAM 掉电控制位 10	0b0: 0x1FFF8000~0x1FFFFFFF 空间 RAM 动作 0b1: 0x1FFF8000~0x1FFFFFFF 空间 RAM 掉电	RW
9	RAMPDC9	RAM 掉电控制位 9	0b0: 0x1FFF0000~0x1FFF7FFF 空间 RAM 动作 0b1: 0x1FFF0000~0x1FFF7FFF 空间 RAM 掉电	RW
8	RAMPDC8	RAM 掉电控制位 8	0b0: 0x1FFE8000~0x1FFEFFFF 空间 RAM 动作 0b1: 0x1FFE8000~0x1FFEFFFF 空间 RAM 掉电	RW
7	RAMPDC7	RAM 掉电控制位 7	0b0: 0x1FFE0000~0x1FFE7FFF 空间 RAM 动作 0b1: 0x1FFE0000~0x1FFE7FFF 空间 RAM 掉电	RW
6	RAMPDC6	RAM 掉电控制位 6	0b0: 0x20058000~0x2005FFFF 空间 RAM 动作 0b1: 0x20058000~0x2005FFFF 空间 RAM 掉电	RW
5	RAMPDC5	RAM 掉电控制位 5	0b0: 0x20050000~0x20057FFF 空间 RAM 动作 0b1: 0x20050000~0x20057FFF 空间 RAM 掉电	RW
4	RAMPDC4	RAM 掉电控制位 4	0b0: 0x20040000~0x2004FFFF 空间 RAM 动作 0b1: 0x20040000~0x2004FFFF 空间 RAM 掉电	RW
3	RAMPDC3	RAM 掉电控制位 3	0b0: 0x20030000~0x2003FFFF 空间 RAM 动作 0b1: 0x20030000~0x2003FFFF 空间 RAM 掉电	RW
2	RAMPDC2	RAM 掉电控制位 2	0b0: 0x20020000~0x2002FFFF 空间 RAM 动作 0b1: 0x20020000~0x2002FFFF 空间 RAM 掉电	RW
1	RAMPDC1	RAM 掉电控制位 1	0b0: 0x20010000~0x2001FFFF 空间 RAM 动作 0b1: 0x20010000~0x2001FFFF 空间 RAM 掉电	RW
0	RAMPDC0	RAM 掉电控制位 0	0b0: 0x20000000~0x2000FFFF 空间 RAM 动作 0b1: 0x20000000~0x2000FFFF 空间 RAM 掉电	RW

6.3.14 RAM 运行条件寄存器 (PWC_RAMOPM)

Offset	Bit Position															
0x4CE4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x8043															
Name	PWC_RAMOPM															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PWC_RAMOPM	RAM 动作模式选择位	● 芯片工作在高速运行模式时，PWC_RAMOPM 设置成 0x8043 ● 芯片工作在超低速模式动作时，PWC_RAMOPM 设置成 0x9062	RW

6.3.15 外设 RAM 低功耗控制寄存器 (PWC_PRAMLPC)

Offset	Bit Position																															
0x4CE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved															PRAMPDC11	PRAMPDC10	PRAMPDC9	PRAMPDC8	PRAMPDC7	PRAMPDC6	PRAMPDC5	PRAMPDC4	PRAMPDC3	PRAMPDC2	PRAMPDC1	PRAMPDC0					
Access																RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	PRAMPDC11	PKE RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
10	PRAMPDC10	MCAN1/MCAN2 RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
9	PRAMPDC9	NFC_RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
8	PRAMPDC8	SDIO2_RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
7	PRAMPDC7	SDIO1_RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
6	PRAMPDC6	ETHER RX RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
5	PRAMPDC5	ETHER TX RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
4	PRAMPDC4	USBHS RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
3	PRAMPDC3	USBFS RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
2	PRAMPDC2	CACHE_RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
1	PRAMPDC1	CAN_2 RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW
0	PRAMPDC0	CAN_1 RAM 掉电控制	0b0: RAM 动作 0b1: RAM 掉电	RW

6.3.16 低功耗状态寄存器 (PWC_LPMCSR)

Offset	Bit Position															
0x4CEC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved															LPMF
Access																RO

位/位域	标记	位名	功能描述	读写
15:1	Reserved	保留	-	-
0	LPMF	低功耗模式标志位	0b0: 芯片未进入停止模式 0b1: 芯片进入停止模式 说明 芯片进入掉电模式后，该位清零。	RO

6.3.17 低功耗状态清除寄存器 (PWC_LPMCSCR)

Offset	Bit Position															
0x4CEE	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved															LPMFC
Access																RW

位/位域	标记	位名	功能描述	读写
15:1	Reserved	保留	-	-
0	LPMFC	LPMF 标志清除	写 1 清除标志，写 0 无效；读出恒为 0	RW

6.3.18 PVD 控制寄存器 0 (PWC_PVDCR0)

Offset	Bit Position							
0x4C14	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	PVD2EN	PVD1EN	Reserved				EXVCCINEN
Access		RW	RW					RW

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6	PVD2EN	电压检测 2 允许	0b0: 电压检测 2 电路无效 0b1: 电压检测 2 电路有效	RW
5	PVD1EN	电压检测 1 允许	0b0: 电压检测 1 电路无效 0b1: 电压检测 1 电路有效	RW
4:1	Reserved	保留	-	-
0	EXVCCINEN	外部 VCC 电压输入使能	0b0: 外部 VCC 电压输入无效 0b1: 外部 VCC 电压输入有效	RW

6.3.19 PVD 控制寄存器 1 (PWC_PVDCR1)

Offset	Bit Position							
0x4C18	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	PVD2CMPOE	PVD2IRS	PVD2IRE	Reserved	PVD1CMPOE	PVD1IRS	PVD1IRE
Access		RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6	PVD2CMPOE	PVD2 比较结果输出使能	0b0: 禁止输出比较器 2 的比较结果 0b1: 允许输出比较器 2 的比较结果	RW
5	PVD2IRS	PVD2 中断复位选择	0b0: 在 VCC 变化满足检测条件时产生 PVD2 中断 0b1: 在下降过程中经过 VPVD2 产生 PVD2 复位  说明 当 PVD1IRS 位为 1 或者 PVD2IRS 位为 1 时，不能进入掉电模式。要进入 PD 模式时，必须将 PVD1IRS 位和 PVD2IRS 位同时置 0。	RW
4	PVD2IRE	PVD2 中断复位使能	0b0: 禁止 0b1: 允许  说明 请在 PWC_PVDCR0.PVD2EN 位为 1 且 PVD2CMPOE 位为 1 的情况下将该位写 1。	RW
3	Reserved	保留	-	-
2	PVD1CMPOE	PVD1 比较结果输出使能	0b0: 禁止输出比较器 1 的比较结果 0b1: 允许输出比较器 1 的比较结果	RW
1	PVD1IRS	PVD1 中断复位选择	0b0: 在 VCC 变化满足检测条件时产生 PVD1 中断 0b1: 在下降过程中经过 VPVD1 产生 PVD1 复位  说明 当 PVD1IRS 位为 1 或者 PVD2IRS 位为 1 时，不能进入掉电模式。要进入 PD 模式时，必须将 PVD1IRS 位和 PVD2IRS 位同时置 0。	RW
0	PVD1IRE	PVD1 中断复位使能	0b0: 禁止 0b1: 允许  说明 请在 PWC_PVDCR0.PVD1EN 位为 1 且 PVD1CMPOE 位为 1 的情况下将该位写 1。	RW

6.3.20 PVD 滤波控制寄存器 (PWC_PVDFCR)

Offset	Bit Position							
0x4C1C	7	6	5	4	3	2	1	0
Reset	0x11							
Name	Reserved	PVD2NFCKS		PVD2NFDIS	Reserved	PVD1NFCKS		PVD1NFDIS
Access		RW		RW		RW		RW

位/位域	标记	位名	功能描述	读写
7	<i>Reserved</i>	<i>保留</i>	-	-
6:5	PVD2NFCKS	PVD2 数字滤波采样时钟选择	0b00: 2 个滤波时钟周期 0b01: 4 个滤波时钟周期 0b10: 8 个滤波时钟周期 0b11: 16 个滤波时钟周期 滤波时钟周期=LRC 周期/8  说明 只能在 PVD2NFDIS 位为 1 时改写该位。	RW
4	PVD2NFDIS	PVD2 数字滤波器屏蔽	0b0: 数字滤波器有效 0b1: 数字滤波器无效	RW
3	<i>Reserved</i>	<i>保留</i>	-	-
2:1	PVD1NFCKS	PVD1 数字滤波采样时钟选择	0b00: 2 个滤波时钟周期 0b01: 4 个滤波时钟周期 0b10: 8 个滤波时钟周期 0b11: 16 个滤波时钟周期 滤波时钟周期=LRC 周期/8  说明 只能在 PVD1NFDIS 位为 1 时改写该位。	RW
0	PVD1NFDIS	PVD1 数字滤波器屏蔽	0b0: 数字滤波器有效 0b1: 数字滤波器无效	RW

6.3.21 PVD 电平控制寄存器 (PWC_PVDLCR)

Offset	Bit Position							
0x4C20	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	PVD2LVL			Reserved	PVD1LVL		
Access		RW				RW		

位/位域	标记	位名	功能描述	读写
7	<i>Reserved</i>	<i>保留</i>	-	-
6:4	PVD2LVL	PVD2 阈值电压选择	● 0b000: 2.20V ● 0b001: 2.40V ● 0b010: 2.60V ● 0b011: 2.70V ● 0b100: 2.85V ● 0b101: 2.95V ● 0b110: 3.05V ● 0b111: 1.15V (仅在 PWC_PVDCR0.EXVCCINEN=0b1 时有效, 其它情况请不要设定该值)	RW
3	<i>Reserved</i>	<i>保留</i>	-	-
2:0	PVD1LVL	PVD1 阈值电压选择	● 0b000: 2.10V ● 0b001: 2.20V ● 0b010: 2.40V ● 0b011: 2.60V ● 0b100: 2.70V ● 0b101: 2.80V ● 0b110: 2.95V ● 0b111: 3.05V	RW

6.3.22 PVD 中断控制寄存器 (PWC_PVDICR)

Offset	Bit Position							
0x4CF0	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	PVD2EDGS		PVD2NMIS	Reserved	PVD1EDGS		PVD1NMIS
Access		RW		RW		RW		RW

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:5	PVD2EDGS	PVD2 检测条件选择	0b00: 当检测到 $VCC < VPVD2$ (下降) 时 0b01: 当检测到 $VCC \geq VPVD2$ (上升) 时 0b10: 当检测到 $VCC < VPVD2$ (下降) 时或者当检测到 $VCC \geq VPVD2$ (上升) 时 0b11: 禁止设置	RW
4	PVD2NMIS	PVD2 中断类型选择	0b0: PVD2 中断作为不可屏蔽中断 0b1: PVD2 中断作为可屏蔽中断	RW
3	Reserved	保留	-	-
2:1	PVD1EDGS	PVD1 检测条件选择	0b00: 当检测到 $VCC < VPVD1$ (下降) 时 0b01: 当检测到 $VCC \geq VPVD1$ (上升) 时 0b10: 当检测到 $VCC < VPVD1$ (下降) 时或者当检测到 $VCC \geq VPVD1$ (上升) 时 0b11: 禁止设置	RW
0	PVD1NMIS	PVD1 中断类型选择	0b0: PVD1 中断作为不可屏蔽中断 0b1: PVD1 中断作为可屏蔽中断	RW

6.3.23 PVD 检测状态寄存器 (PWC_PVDDSR)

Offset	Bit Position							
0x4CF4	7	6	5	4	3	2	1	0
Reset	0x11							
Name	Reserved		PVD2DETFLG	PVD2MON	Reserved		PVD1DETFLG	PVD1MON
Access			RW	RO			RW	RO

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5	PVD2DETFLG	PVD2 检测标志位	读取该位后写 0 清除本标志位；写 1 无效 读取： 0b0：PVD2 未检测到 VCC 经过 VPVD2 0b1：PVD2 检测到 VCC 经过 VPVD2  说明 当 PWC_PVDCR0.PVD2EN 位为 1 且 PWC_PVDCR1.PVD2CMPOE 位为 1 时，该位有效。	RW
4	PVD2MON	PVD2 监视位	0b0：VCC≤VPVD2 或者外部输入比较电压≤PVD2 内部基准电压 0b1：PVD2 无效时或者 VCC>VPVD2 或者外部输入比较电压>PVD2 内部基准电压	RO
3:2	Reserved	保留	-	-
1	PVD1DETFLG	PVD1 检测标志位	读取该位后写 0 清除本标志位；写 1 无效 读取： 0b0：PVD1 未检测到 VCC 经过 VPVD1 0b1：PVD1 检测到 VCC 经过 VPVD1  说明 当 PWC_PVDCR0.PVD1EN 位为 1 且 PWC_PVDCR1.PVD1CMPOE 位为 1 时，该位有效。	RW
0	PVD1MON	PVD1 监视位	0b0：VCC<VPVD1 0b1：PVD1 无效时或者 VCC>VPVD1	RO

6.3.24 功能时钟控制 0 (PWC_FCG0)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFA0E																															
Name	DCU8	DCU7	DCU6	DCU5	DCU4	DCU3	DCU2	DCU1	CRC	TRNG	HASH	SKE	MAU	CTC	AOS	FCM	DMA2	DMA1	KEYSCAN	PKE	Reserved	SRAMB	Reserved	ERMU	SRAM4	SRAM3	SRAM2	SRAM1	Reserved			SRAMH
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW			RW	

位/位域	标记	位名	功能描述	读写
31	DCU8	DCU8 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
30	DCU7	DCU7 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
29	DCU6	DCU6 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
28	DCU5	DCU5 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
27	DCU4	DCU4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
26	DCU3	DCU3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
25	DCU2	DCU2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
24	DCU1	DCU1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
23	CRC	CRC 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
22	TRNG	TRNG 功能控制	0b0: CPM 中的真随机发生器 TRNG 功能使能 0b1: CPM 中的真随机发生器 TRNG 功能禁止	RW
21	HASH	HASH 功能控制	0b0: CPM 中的安全散列算法模块 HASH 功能使能 0b1: CPM 中的安全散列算法模块 HASH 功能禁止	RW
20	SKE	SKE 功能控制	0b0: CPM 中的加解密算法处理器 SKE 功能使能 0b1: CPM 中的加解密算法处理器 SKE 功能禁止	RW
19	MAU	MAU 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
18	CTC	CTC 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
17	AOS	AOS 功能控制	0b0: 功能使能 0b1: 功能禁止	RW

位/位域	标记	位名	功能描述	读写
16	FCM	FCM 功能控制	● 0b0: CMU 中的时钟频率测量模块 FCM 功能使能 ● 0b1: CMU 中的时钟频率测量模块 FCM 功能禁止	RW
15	DMA2	DMA2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
14	DMA1	DMA1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
13	KEYSCAN	KEYSCAN 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
12	PKE	PKE 功能控制	● 0b0: CPM 中的公钥密码加速器 PKE 功能使能 ● 0b1: CPM 中的公钥密码加速器 PKE 功能禁止	RW
11	<i>Reserved</i>	<i>保留</i>	<i>读出为 1, 写入时必须写 1</i>	-
10	SRAMB	Ret SRAM 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
9	<i>Reserved</i>	<i>保留</i>	<i>读出为 1, 写入时必须写 1</i>	-
8	ERMU	ERMU 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
7	SRAM4	SRAM4 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
6	SRAM3	SRAM3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
5	SRAM2	SRAM2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
4	SRAM1	SRAM1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
3:1	<i>Reserved</i>	<i>保留</i>	<i>所有位读出为 1, 写入时必须写 1</i>	-
0	SRAMH	SRAMH 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW

6.3.25 功能时钟控制 1 (PWC_FCG1)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved	MCAN2	MCAN1	FMAC4	FMAC3	FMAC2	FMAC1	USBHS	USBFS	SPI6	SPI5	SPI4	SPI3	SPI2	SPI1	I2S4	I2S3	I2S2	I2S1	SDIOC2	SDIOC1	I2C6	I2C5	I2C4	I2C3	I2C2	I2C1	QSPI	ETHMAC	CAN2	CAN1	
Access		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	所有位读数为 1，写入时必须写 1	-
29	MCAN2	MCAN2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
28	MCAN1	MCAN1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
27	FMAC4	FMAC4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
26	FMAC3	FMAC3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
25	FMAC2	FMAC2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
24	FMAC1	FMAC1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
23	USBHS	USBHS 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
22	USBFS	USBFS 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
21	SPI6	SPI6 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
20	SPI5	SPI5 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
19	SPI4	SPI4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
18	SPI3	SPI3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
17	SPI2	SPI2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
16	SPI1	SPI1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
15	I2S4	I2S4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW

位/位域	标记	位名	功能描述	读写
14	I2S3	I2S3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
13	I2S2	I2S2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
12	I2S1	I2S1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
11	SDIOC2	SDIOC2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
10	SDIOC1	SDIOC1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
9	I2C6	I2C6 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
8	I2C5	I2C5 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
7	I2C4	I2C4 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
6	I2C3	I2C3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
5	I2C2	I2C2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
4	I2C1	I2C1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
3	QSPI	QSPI 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
2	ETHMAC	ETHMAC 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
1	CAN2	CAN2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
0	CAN1	CAN1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW

6.3.26 功能时钟控制 2 (PWC_FCG2)

Offset	Bit Position																																
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	TMRA_12	TMRA_11	TMRA_10	TMRA_9	TMRA_8	TMRA_7	TMRA_6	TMRA_5	TMRA_4	TMRA_3	TMRA_2	TMRA_1	TMR2_4	TMR2_3	TMR2_2	TMR2_1	EMB	TMR0_3	TMR0_2	TMR0_1	Reserved	TMR4_3	TMR4_2	TMR4_1	TMR6_8	TMR6_7	TMR6_6	TMR6_5	TMR6_4	TMR6_3	TMR6_2	TMR6_1	
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	TMRA_12	TimerA_12 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
30	TMRA_11	TimerA_11 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
29	TMRA_10	TimerA_10 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
28	TMRA_9	TimerA_9 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
27	TMRA_8	TimerA_8 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
26	TMRA_7	TimerA_7 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
25	TMRA_6	TimerA_6 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
24	TMRA_5	TimerA_5 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
23	TMRA_4	TimerA_4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
22	TMRA_3	TimerA_3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
21	TMRA_2	TimerA_2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
20	TMRA_1	TimerA_1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
19	TMR2_4	Timer2_4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
18	TMR2_3	Timer2_3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
17	TMR2_2	Timer2_2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW

位/位域	标记	位名	功能描述	读写
16	TMR2_1	Timer2_1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
15	EMB	EMB 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
14	TMR0_3	Timer0_3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
13	TMR0_2	Timer0_2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
12	TMR0_1	Timer0_1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
11	Reserved	保留	读出为 1, 写入时必须写 1	-
10	TMR4_3	Timer4_3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
9	TMR4_2	Timer4_2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
8	TMR4_1	Timer4_1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
7	TMR6_8	Timer6_8 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
6	TMR6_7	Timer6_7 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
5	TMR6_6	Timer6_6 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
4	TMR6_5	Timer6_5 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
3	TMR6_4	Timer6_4 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
2	TMR6_3	Timer6_3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
1	TMR6_2	Timer6_2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
0	TMR6_1	Timer6_1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW

6.3.27 功能时钟控制 3 (PWC_FCG3)

Offset	Bit Position																																
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	Reserved	USART10	USART9	USART8	USART7	USART6	USART5	USART4	USART3	USART2	USART1	Reserved	NFC	DMC	SMC	DVP	Reserved	OTS	TMR0_5	TMR0_4	CMP34	CMP12	Reserved	DAC2	DAC1	CMBIAS	ADC3	ADC2	ADC1				
Access		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	所有位读出为 1，写入时必须写 1	-
29	USART10	USART10 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
28	USART9	USART9 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
27	USART8	USART8 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
26	USART7	USART7 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
25	USART6	USART6 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
24	USART5	USART5 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
23	USART4	USART4 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
22	USART3	USART3 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
21	USART2	USART2 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
20	USART1	USART1 功能控制	0b0: 功能使能 0b1: 功能禁止	RW
19	Reserved	保留	读出为 1，写入时必须写 1	-
18	NFC	NFC 功能控制	0b0: 外部存储控制器 (EXMC) 的 NAND FLASH 控制器功能使能 0b1: 外部存储控制器 (EXMC) 的 NAND FLASH 控制器功能禁止	RW
17	DMC	DMC 功能控制	0b0: 外部存储控制器 (EXMC) 的 DMC 控制器功能使能 0b1: 外部存储控制器 (EXMC) 的 DMC 控制器功能禁止	RW

位/位域	标记	位名	功能描述	读写
16	SMC	SMC 功能控制	● 0b0: 外部存储控制器 (EXMC) 的 SMC 控制器功能使能 ● 0b1: 外部存储控制器 (EXMC) 的 SMC 控制器功能禁止	RW
15	DVP	DVP 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
14:13	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
12	OTS	OTS 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
11	TMR0_5	Timer0_5 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
10	TMR0_4	Timer0_4 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
9	CMP34	CMP34 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
8	CMP12	CMP12 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
7:6	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
5	DAC2	DAC2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
4	DAC1	DAC1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
3	CMBIAS	CMP/PGA/SH 基准电流源控制	● 0b0: CMP/PGA/SH 基准电流源有效 ● 0b1: CMP/PGA/SH 基准电流源无效	RW
2	ADC3	ADC3 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
1	ADC2	ADC2 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW
0	ADC1	ADC1 功能控制	● 0b0: 功能使能 ● 0b1: 功能禁止	RW

6.3.28 PWC_FCG0 保护控制 (PWC_FCG0PC)

Offset	Bit Position																																
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	FCG0PCWE																Reserved																PRT0
Access	RW																																RW

位/位域	标记	位名	功能描述	读写
31:16	FCG0PCWE	PWC_FCG0PC 写使能	在写入 PRT0 时，必须同时向 FCG0PCWE 写入 0xA5A5；读出恒为 0	RW
15:1	Reserved	保留	-	-
0	PRT0	PWC_FCG0 写使能控制位	0b0: PWC_FCG0 写无效 0b1: PWC_FCG0 写使能	RW

 说明

向本寄存器写入 0xA5A50001 可以将 PRT0 置 1；写入 0xA5A50000 可以将 PRT0 清 0；禁止写入其他值修改 PRT0。

6.3.29 功能保护控制寄存器 (PWC_FPRC)

Offset	Bit Position															
0xC3FE	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PWC_FPRCWE								FPRCB7	FPRCB6	FPRCB5	FPRCB4	FPRCB3	FPRCB2	FPRCB1	FPRCB0
Access	RW								RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:8	PWC_FPRCWE	PWC_FPRC 寄存器写使能	在写入 FPRCB7~FPRCB0 时，必须同时向 PWC_FPRCWE 写入 0xA5；读出恒为 0	RW
7	FPRCB7	FPRC 位 7	预留，读出为 0，写入时必须写 0	RW
6	FPRCB6	FPRC 位 6	预留，读出为 0，写入时必须写 0	RW
5	FPRCB5	FPRC 位 5	预留，读出为 0，写入时必须写 0	RW
4	FPRCB4	FPRC 位 4	预留，读出为 0，写入时必须写 0	RW
3	FPRCB3	FPRC 位 3	保护对象请参见“ 寄存器保护列表 ”： ● 0b0：对象写无效 ● 0b1：对象写使能	RW
2	FPRCB2	FPRC 位 2	预留，读出为 0，写入时必须写 0	RW
1	FPRCB1	FPRC 位 1	保护对象请参见“ 寄存器保护列表 ”： ● 0b0：对象写无效 ● 0b1：对象写使能	RW
0	FPRCB0	FPRC 位 0	保护对象请参见“ 寄存器保护列表 ”： ● 0b0：对象写无效 ● 0b1：对象写使能	RW

6.3.30 STOP 模式控制寄存器 (PWC_STPMCR)

Offset	Bit Position															
0xC00C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	STOP	EXBUSOE	Reserved												CKSMRC	FLNWT
Access	RW	RW													RW	RW

位/位域	标记	位名	功能描述	读写
15	STOP	STOP 模式选择位	0b0: STOP 模式无效 0b1: STOP 模式有效	RW
14	EXBUSOE	EXMC 地址总线和输出控制信号的输出使能	掉电模式和停止模式下外部总线的地址总线信号和输出控制信号的状态控制: 0b0: 高阻 0b1: 保持掉电模式和停止模式前的状态	RW
13:2	Reserved	保留	-	-
1	CKSMRC	时钟切换到 MRC 选项	0b0: 维持进入 STOP 模式之前的系统时钟及分频 0b1: STOP 模式唤醒时系统时钟切换到 MRC、SCKCFGR 寄存器被初始化	RW
0	FLNWT	FLASH 稳定等待控制	0b0: STOP 模式唤醒时等待 FLASH 稳定 0b1: STOP 模式唤醒时不等待 FLASH 稳定	RW

6.3.31 备份域复位寄存器 (PWC_VBATRSTR)

Offset	Bit Position							
0x4430	7	6	5	4	3	2	1	0
Reset	0x00							
Name	RST							
Access	WO							

位/位域	标记	位名	功能描述	读写
7:0	RST	备份域复位寄存器	写入 0xA5 复位备份域	WO

6.3.32 备份域控制寄存器 (PWC_VBATCR)

Offset	Bit Position							
0x4440	7	6	5	4	3	2	1	0
Reset	0x00							
Name	CSDIS	Reserved			VBATDIVMONE	RAMPDF	RAMVALID	VBTRSD
Access	RW				RW	RO	RO	RW

位/位域	标记	位名	功能描述	读写
7	CSDIS	VBAT 域电流源无效控制	0b0: VBAT 域电流源有效 0b1: VBAT 域电流源无效, 只有当 Ret SRAM、XTAL32、XTAL32NF 都不使用的时候, 这个位才可以设置成 1	RW
6:4	Reserved	保留	-	-
3	VBATDIVMONE	VBAT 分压电路使能位	0b0: 分压电路无效 0b1: 分压电路使能	RW
2	RAMPDF	RAM 掉电标志位	0b0: Ret SRAM 未掉电 0b1: Ret SRAM 掉电	RO
1	RAMVALID	RAM 有效标志	0b0: Ret SRAM 不可读写 0b1: Ret SRAM 可以读写	RO
0	VBTRSD	VBAT RLDO 关闭控制位	0b0: VBAT RLDO 启动 0b1: VBAT RLDO 关闭	RW

6.3.33 备份域上电标志寄存器 (PWC_VBATPORF)

Offset	Bit Position							
0x4448	7	6	5	4	3	2	1	0
Reset	0x01							
Name	Reserved							VBATPORF
Access								RO

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	VBATPORF	备份域上电标志位	0b0: 备份域未发生上电动作 0b1: 备份域发生上电动作 清零条件: - 通过置位 PWC_VBATPORFCLR.VBATPORFCLR 位清零。 置位条件: - 备份域发生上电时, 由硬件置位。	RO

6.3.34 备份域上电标志清除寄存器 (PWC_VBATPORFCLR)

Offset	Bit Position							
0x444C	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							VBATPORFCLR
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	VBATPORFCLR	备份域上电标志清除	写 1 清除 VBATPORF 标志 (备份域上电标志位), 写 0 无效; 读出恒为 0  说明 写 1 动作必须在 PWC_VBATPORF.VBATPORF 为 1 状态下进行。	RW

6.3.35 唤醒计时器控制寄存器 0 (PWC_WKTC0)

Offset	Bit Position							
0x4450	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKTMCMP[7:0]							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	WKTMCMP[7:0]	WKTMCMP 计数器的比较值	WKTMCMP 计数器的比较值的低 8 位	RW

6.3.36 唤醒计时器控制寄存器 1 (PWC_WKTC1)

Offset	Bit Position							
0x4454	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved				WKTMCMP[11:8]			
Access					RW			

位/位域	标记	位名	功能描述	读写
7:4	Reserved	保留	-	-
3:0	WKTMCMP[11:8]	WKTMCMP 计数器的比较值	WKTMCMP 计数器的比较值高 4 位	RW

6.3.37 唤醒计时器控制寄存器 2 (PWC_WKTC2)

Offset	Bit Position							
0x4458	7	6	5	4	3	2	1	0
Reset	0x00							
Name	WKTCE	WKCKS		WKOVF	Reserved			
Access	RW	RW		RW				

位/位域	标记	位名	功能描述	读写
7	WKTCE	WKTm 使能	0b0: WKTm 停止 0b1: WKTm 计数  说明 WKTm 计数时清零 WKTCE 位会将计数器清零，计数器停止动作。	RW
6:5	WKCKS	WKTm 时钟选择	0b00: 64Hz 时钟 0b01: XTAL32 0b10: RTCLRC 0b11: 禁止设置	RW
4	WKOVF	计时器比较结果标志	写 0 清除本标志位，写 1 无效 读取： 0b0: 计数器与 WKTMCMP 值不一致 0b1: 计数器与 WKTMCMP 值一致	RW
3:0	Reserved	保留	-	-

6.3.38 VBAT 备份寄存器 n (PWC_BKRn) (n=0~127)

偏移地址 (Offset) : 0x4600+0x4*n

Offset	Bit Position							
-	7	6	5	4	3	2	1	0
Reset	0xxx							
Name	BRKnB							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	BRKnB	备份寄存器	备份寄存器	RW

7 初始化配置 (ICG)

7.1 简介

芯片复位解除后，硬件电路会读取 FLASH 地址 0x00000400~0x0000046F 把数据加载到初始化配置寄存器。地址 0x00000408~0x0000040B，0x00000410~0x0000041F，0x00000438~0x0000045F 为保留地址，请写入全 1 保证芯片正常动作。FLASH 引导交换有效，且 OTP 不使能时，该区域位于 FLASH 块 1 扇区 0；否则该区域存在 FLASH 块 0 扇区 0。用户可通过编程或擦除扇区 0 来修改初始化配置 (ICG) 寄存器。地址 0x00000420~0x00000437 为数据安全保护使能区，详细内容请参见《嵌入式 FLASH (EFM) 》中“功能说明-FLASH 编程和擦除操作-数据安全保护”。寄存器复位值由 FLASH 数据确定。

7.2 寄存器

7.2.1 寄存器总表

基地址：0x00000400

表 7-1 初始化配置 (ICG) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x000	ICG0	初始化配置寄存器 0	32
0x004	ICG1	初始化配置寄存器 1	32
0x00C	ICG3	初始化配置寄存器 3	32
0x060	ICG8A	初始化配置寄存器 8A	32
0x064	ICG8B	初始化配置寄存器 8B	32
0x068	ICG8C	初始化配置寄存器 8C	32

7.2.2 初始化配置寄存器 0 (ICG0)

Offset	Bit Position																																			
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0xFFFFFFFF																																			
Name	Reserved		WDTSLPOFF		WDTWDPT				WDTCKS				WDTPERI		WDTITS		WDTAUS		Reserved		SWDTSLOFF		SWDTWDPT				SWDTCKS				SWDTPERI		SWDTITS		SWDTAUS	
Access			RO		RO				RO				RO		RO		RO				RO		RO				RO				RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	所有位读出的 1，写入时必须写 1	-
28	WDTSLPOFF	WDT 在 sleep 模式下计数停止	0b0: WDT 在睡眠模式下计数不停止 0b1: WDT 在睡眠模式下计数停止	RO
27:24	WDTWDPT	刷新允许区域计数值百分比	WDT 计数值刷新允许区间： 0b0000: 0%~100% 0b0001: 0%~25% 0b0010: 25%~50% 0b0011: 0%~50% 0b0100: 50%~75% 0b0101: 0%~25%, 50%~75% 0b0110: 25%~75% 0b0111: 0%~75% 0b1000: 75%~100% 0b1001: 0%~25%, 75%~100% 0b1010: 25%~50%, 75%~100% 0b1011: 0%~50%, 75%~100% 0b1100: 50%~100% 0b1101: 0%~25%, 50%~100% 0b1110: 25%~100% 0b1111: 0%~100%	RO
23:20	WDTCKS	WDT 计数时钟	0b0010: PCLK3/4 0b0110: PCLK3/64 0b0111: PCLK3/128 0b1000: PCLK3/256 0b1001: PCLK3/512 0b1010: PCLK3/1024 0b1011: PCLK3/2048 0b1101: PCLK3/8192 - 其他: Reserved	RO
19:18	WDTPERI	WDT 计数溢出周期	0b00: 256 周期 0b01: 4096 周期 0b10: 16384 周期 0b11: 65536 周期	RO

位/位域	标记	位名	功能描述	读写
17	WDTITS	WDT 中断选择	0b0: 中断请求 0b1: 复位请求	RO
16	WDTAUTS	WDT 自动启动	0b0: 复位后, WDT 自动启动 (硬件启动) 0b1: 复位后, WDT 停止状态	RO
15:13	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
12	SWDTSLPPOFF	SWDT 在 Sleep/Stop 模式下计数停止	0b0: SWDT 在睡眠模式和停止模式下计数不停止 0b1: SWDT 在睡眠模式和停止模式下计数停止	RO
11:8	SWDTWDPT	刷新允许区域计数值百分比	SWDT 计数值刷新允许区间: 0b0000: 0%~100% 0b0001: 0%~25% 0b0010: 25%~50% 0b0011: 0%~50% 0b0100: 50%~75% 0b0101: 0%~25%, 50%~75% 0b0110: 25%~75% 0b0111: 0%~75% 0b1000: 75%~100% 0b1001: 0%~25%, 75%~100% 0b1010: 25%~50%, 75%~100% 0b1011: 0%~50%, 75%~100% 0b1100: 50%~100% 0b1101: 0%~25%, 50%~100% 0b1110: 25%~100% 0b1111: 0%~100%	RO
7:4	SWDTCKS	SWDT 计数时钟	0b0000: SWDTCLK 0b0100: SWDTCLK/16 0b0101: SWDTCLK/32 0b0110: SWDTCLK/64 0b0111: SWDTCLK/128 0b1000: SWDTCLK/256 0b1011: SWDTCLK/2048 - 其他: Reserved	RO
3:2	SWDTPERI	SWDT 计数溢出周期	0b00: 256 周期 0b01: 4096 周期 0b10: 16384 周期 0b11: 65536 周期	RO
1	SWDTITS	SWDT 中断选择	0b0: 中断请求 0b1: 复位请求	RO
0	WDTAUTS	SWDT 自动启动	0b0: 复位后, SWDT 自动启动 (硬件启动) 0b1: 复位后, SWDT 停止状态	RO

7.2.3 初始化配置寄存器 1 (ICG1)

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved														BORDIS	BOR_LEV	Reserved						HRCSTOP	Reserved						HRCFREQSEL		
Access															RO	RO							RO							RO		

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	所有位读出的 1，写入时必须写 1	-
18	BORDIS	BOR 动作选择	0b0: 复位后允许 BOR 动作 0b1: 复位后禁止 BOR 动作	RO
17:16	BOR_LEV	BOR 阈值电压选择	0b00: 2.00V 0b01: 2.10V 0b10: 2.20V 0b11: 2.40V	RO
15:9	Reserved	保留	所有位读出的 1，写入时必须写 1	-
8	HRCSTOP	HRC 振荡停止位	0b0: HRC 振荡 0b1: HRC 停止	RO
7:1	Reserved	保留	所有位读出的 1，写入时必须写 1	-
0	HRCFREQSEL	HRC 频率选择	0b0: 20MHz 0b1: 16MHz	RO

7.2.4 初始化配置寄存器 3 (ICG3)

Offset	Bit Position																															
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																DBUSPRT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	所有位读出的 1，写入时必须写 1	-
15:0	DBUSPRT	对 128KBytes 区域 D-BUS 读保护功能	当 DBUSPRT[15:0]=0x4450 时，对 0x00000000~0x0001FFFF 区域的 D-BUS 读保护使能；否则 D-BUS 读保护无效。	RO

7.2.5 初始化配置寄存器 8x (ICG8x) (x=A/B/C)



说明

FLASH 地址 0x00000460-0x00000463, 0x00000464-0x00000467, 0x00000468-0x0000046B 请编程写入相同的数据。ICG8A/8B/8C 需要三个寄存器联合使用, 值要相同。

偏移地址 (Offset) : 0x060 (ICG8A) , 0x064 (ICG8B) , 0x068 (ICG8C)

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0xFFFFFFFF																																	
Name	Reserved																												ERMUOUTDIS		Reserved		ERMUOUTSEL	
Access																													RO				RO	

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	所有位读出于 1, 写入时必须写 1	-
4	ERMUOUTDIS	ERMU 功能端口输出控制	0b0: 输出有效 0b1: 输出无效	RO
3:2	Reserved	保留	所有位读出于 1, 写入时必须写 1	-
1:0	ERMUOUTSEL	ERMU 功能端口输出选择	0b00: ERMU_EOUT0M 由 PA5 输出 0b01: ERMU_EOUT0M 由 PH5 输出 0b10: ERMU_EOUT0M 由 PH13 输出 0b11: ERMU_EOUT0M 由 PC1 输出	RO

8 嵌入式 Flash (EFM)

8.1 简介

Flash 接口通过 Flash ICODE, DCODE 和 MCODE 总线对 Flash 进行访问。该接口可对 Flash 执行编程, 擦除和全擦除操作; 通过指令预取和缓存机制加速代码执行。

8.2 主要特性

- 两块独立 Flash 构成 dual bank, 可实现 BGO (BackGroud Operation) 功能
- 134KBytes 的 OTP 空间
- ICODE 总线 16Bytes 预取指
- 两个独立缓存区: ICODE 总线缓存空间 4KBytes (256x128) ; DCODE 总线缓存空间 1KBytes (64x128)
- 支持引导交换功能
- 支持数据安全保护
- 带有 ECC 校验 (Error Checking and Correcting) , ECC 校验可以纠正 128 位数据和 9 位 ECC 的一位错误, 检测 20 位地址相关信息的一位错误, 检测 128 位数据和 9 位 ECC 以及 20 位地址相关信息的两位错误

8.3 功能说明

8.3.1 嵌入式 Flash

Flash 具有以下主要特性：

- 容量高达 2MBytes，由两块 1MBytes 的 Flash 构成，共 256 个扇区，每个扇区为 8KBytes。块 0 中扇区 0~扇区 15 为可配置为 OTP 区域。
- OTP（One Time Program）区域共 134KBytes，其中 128KBytes 配置在块 0 地址 0x00000000~0x0001FFFF，6KBytes 配置在地址 0x03000000~0x030017FF。地址 0x03001800~0x03001F6F 为 OTP 块锁存区，0x03001F70~0x03001FEF 为 OTP 预留区域，0x03001FF0~0x03001FFF 为 OTP 使能地址。
- 128 位宽数据读取，读缓存 128 位宽缓冲，加速代码执行。
- 编程单位为 16Bytes。擦除单位为 8KBytes。

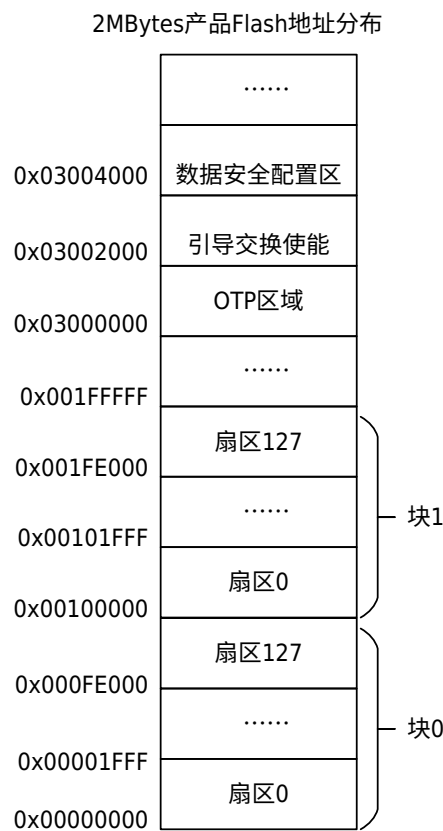


图 8-1 Flash 地址分布

OTP 区域，引导交换使能地址和数据安全配置区分别单独享有 1 个扇区，可进行编程，擦除，但全擦除禁止；这三个扇区物理上从属于 Flash 块 0，擦写标志位跟随 Flash 块 0，但对 Flash 块 0 全擦除这三个扇区数据不变。

OTP 区域 0x03000000~0x03001FFF，引导交换使能地址 0x03002000~0x0300200F，数据安全配置区 0x03004000~0x0300400F。

8.3.2 读接口

8.3.2.1 CPU 时钟和 Flash 读取时间之间的关系

要正确读取 Flash 数据，用户需要根据 CPU 动作频率在 Flash 读模式寄存器（EFM_FRMC）中正确设定等待周期数（FLWT[3:0]）。

系统复位后，CPU 时钟源为 MRC（8MHz），Flash 读等待周期为 0。建议用户按照以下步骤修改 CPU 主频和 Flash 读等待周期位。等待周期数如下表所示。

表 8-1 CPU 时钟频率和 Flash 读等待周期对照表

CPU 时钟频率 (HCLK)	读等待周期设定
200MHz<HCLK≤240MHz	FLWT[3:0]=0b0101
160MHz<HCLK≤200MHz	FLWT[3:0]=0b0100
120MHz<HCLK≤160MHz	FLWT[3:0]=0b0011
80MHz<HCLK≤120MHz	FLWT[3:0]=0b0010
40MHz<HCLK≤80MHz	FLWT[3:0]=0b0001
HCLK≤40MHz	FLWT[3:0]=0b0000

CPU 频率提高步骤：

- 步骤 1. 将新的读等待周期设定值（FLWT[3:0]）写入寄存器 EFM_FRMC。
- 步骤 2. 读取寄存器 EFM_FRMC，检查新的等待周期是否设定成功。
- 步骤 3. 通过设定系统时钟源切换寄存器 CMU_CKSWR（CKSW[2:0]）或者系统时钟配置寄存器 CMU_SCFGR（HCLKS[2:0]）来提升 CPU 时钟频率。
- 步骤 4. 读取寄存器 CMU_CKSWR 或者 CMU_SCFGR，检查新的设定是否成功。

CPU 频率降低步骤：

- 步骤 1. 通过设定系统时钟源切换寄存器 CMU_CKSWR（CKSW[2:0]）或者系统时钟配置寄存器 CMU_SCFGR（HCLKS[2:0]）来降低 CPU 时钟频率。
- 步骤 2. 读取寄存器 CMU_CKSWR 或者 CMU_SCFGR，检查新的设定是否成功。
- 步骤 3. 将新的读等待周期设定值（FLWT[3:0]）写入寄存器 EFM_FRMC。
- 步骤 4. 读取寄存器 EFM_FRMC，检查新的等待周期是否设定成功。

8.3.2.2 Normal 读

默认情况下，Flash 处于 Normal 读模式。它是客户最常用的。

8.3.2.3 Blank 读

Flash 扇区被擦除后，包括 ECC 数据在内的 Flash 数据全部为“1”的检查。

Blank 读模式设定步骤如下：

- 1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
 - 2. 设定 Blank 读模式。（EFM_FRMC2.BRM0/1=1）
 - 3. 读取要检查的 Flash 地址。
 - 4. 检查读取数据是否含“0”的标志（EFM_FSR.BRMER0/1）。
 - 如果读取数据（包括 ECC 数据在内）不是全部为“1”的 Blank 状态，EFM_FSR.BRMER0/1=1。
 - 5. 寄存器 EFM_FSCLR 对应标志清除位写 1，状态位清零。
 - 6. 设定回 Normal 读模式。（EFM_FRMC2.BRM0/1=0）
- 上述程序应在和检查地址不同的 Flash 块上执行，或者在其他存储器（如 RAM）上执行。

8.3.3 Flash 读加速缓存

每次 Flash 读操作为 128 位读取，数据送给 CPU 同时也存入缓冲寄存器，这 128 位数据可以是 4 行 32 位指令，也可以是 8 行 16 位指令，具体取决于烧写在 Flash 中的程序。

为了能快速读取 Flash 数据，Flash 控制器配置了读加速缓存，优化了读取等待周期。为了发挥处理器性能，该加速器将对 Flash 的 ICODE，DCODE 总线访问数据保存到缓存存储器中，从而提高了程序执行速度。系统提供 5KBytes 空间做为缓存存储器，可以有效地减少因指令跳转而产生的时间损耗。通过 EFM_FRMC 寄存器中的缓存使能（ICACHE/DCACHE）位置 1，来使缓存功能有效。每当出现指令或数据缺失（即请求的指令或数据未存在于当前使用的指令行或缓存存储器中）时，系统会将新读取的数据行（128 位）复制到缓存存储器中。如果 CPU 请求的指令或者数据已存在于缓存中，则无需任何延时即可立即获取。缓存存储器存满后，采用 LRU（最近最少使用）策略确定缓存存储器中待替换的数据。

CPU 读取指令或数据时，Flash 地址在缓冲，缓存命中时，读取 Flash 周期数会改变，具体请参考下表。

表 8-2 Flash 实际读周期数

EFM_FRMC. FLWT[3:0]设定	缓存不使能		缓存使能	
	缓冲命中	缓冲不命中	缓冲或缓存命中	缓冲/缓存都不命中
0	1	1	1	1
1	1	2	1	2
2	1	3	1	3
N (N>2)	1	N+1	1	N+1

8.3.4 Flash 编程和擦除操作

8.3.4.1 编程与擦除操作概述

Flash 支持编程，扇区擦除，全擦除操作。

Flash 编程，扇区/全擦除地址末位必须以 4 对齐（末位地址为：0x0，0x4，0x8，0xC）。编程单位是 16Bytes。扇区擦除单位为 8KBytes。全擦除根据寄存器设定可以是单个 Flash 块也可能是两个 Flash 块。Flash 编程方式分为单次编程无回读模式，单次编程回读模式，连续编程模式三种。Flash 编程，擦除期间，设定 EFM_FWMC.BUSHLCTL=0，则总线被占有，直至擦写结束；EFM_FWMC.BUSHLCTL=1，则总线被释放，总线可以继续访问另一块 Flash 地址。Flash 编程，擦除前，请把缓存使能及预取指无效。以下分别介绍编程和擦除操作的设定步骤。

8.3.4.2 解锁 EFM_KEY1 寄存器

复位解除后，Flash 编程，擦除模式寄存器（EFM_FWMC）处于写禁止状态，需要先解除 Flash 访问保护寄存器（EFM_FAPRT），然后再解除 EFM_KEY1 的保护。以下步骤用于解除上述保护：

步骤 1. 解除 Flash 寄存器访问写保护（EFM_FAPRT 先写 0x0123，再写 0x3210）

步骤 2. 解除 EFM_KEY1 锁定（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）

如对 EFM_KEY1 写入错误的序列，则会发生引发一次总线错误，并且自锁，EFM_FWMC 寄存器将写禁止直至发生复位。

对 EFM_FWMC 寄存器正确写入设定值后，如需回归到 EFM_FWMC 写禁止状态，可对 EFM_FWMC.KEY1LOCK 写入 1。

8.3.4.3 写保护功能

Flash 每个扇区有 1 个写保护位，存在于写保护寄存器 EFM_F0/1NWPRTx（x=0~3）中。每个写保护寄存器存在 1 个写保护锁定位（WLOCK[7:0]），写保护锁定位一旦设定为 1，写保护寄存器位只能从写许可设定为写禁止。当 Flash 扇区设定为写禁止时，对该扇区地址进行编程和擦除操作将不发生，并发生错误标志（EFM_FSR.PRTWERR0/1=1）。

在全擦除模式，若该 Flash 块存在一个及以上扇区设定为写禁止，则全擦除不发生，并发生错误标志（EFM_FSR.PRTWERR0/1=1）。

8.3.4.4 单次编程无回读功能

单编程无回读模式设定步骤如下：

- 步骤 1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
- 步骤 2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）
- 步骤 3. 设定单次编程模式。（EFM_FWMC.PEMODE[2:0]=0b001）
- 步骤 4. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) 对应位设定为 1)
- 步骤 5. 对编程地址写入 128 位数据，对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址连续依次分别写入 32 位数据（合计 128 位）。
- 步骤 6. 等待 Flash 操作结束并处于空闲状态。（EFM_FSR.OPTEND0/1=1, EFM_FSR.RDY0/1=1）
- 步骤 7. 读出编程地址值判断是否和写入值一致：
一致，表示编程成功；不一致，表示该 Flash 地址已遭破坏，永久废弃。
- 步骤 8. 清除编程结束标志位。（EFM_FSR.OPTEND0/1）

对已锁存的 OTP 地址发行单编程不回读写操作，编程不成功，标志位 EFM_FSR.OTPWERR0 置位。

对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址如果没有连续依次写入，编程动作就不会执行，标志位 EFM_FSR.PGSZERR0/1 置位。

8.3.4.5 单编程回读功能

单编程回读模式是指编程结束后硬件自动读取编程地址数据并和写入数据对比，判断标志位 EFM_FSR.MISMTCH0/1 验证写入数据正确。

单编程回读模式设定步骤如下：

- 步骤 1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
- 步骤 2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）
- 步骤 3. 设定单次编程回读模式。（EFM_FWMC.PEMODE[2:0]=0b010）
- 步骤 4. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) 对应位设定为 1)
- 步骤 5. 对编程地址写入 128 位数据，对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址连续依次分别写入 32 位数据（合计 128 位）。
- 步骤 6. 等待 Flash 操作结束并处于空闲状态。（EFM_FSR.OPTEND0/1=1, EFM_FSR.RDY0/1=1）
- 步骤 7. 判断编程自读取结果标志位（EFM_FSR.MISMTCH0/1）：
如为 0，则表示编程成功；为 1 表示该 Flash 地址已遭破坏，永久废弃。
- 步骤 8. 清除编程结束标志位（EFM_FSR.OPTEND0/1）。

对已锁存的 OTP 地址发行单编程回读写操作，编程不成功，标志位 EFM_FSR.OTPWERR0 置位。

对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址如果没有连续依次写入，编程动作就不会执行，标志位 EFM_FSR.PGSZERR0/1 置位。

8.3.4.6 连续编程功能

当连续对 Flash 地址进行编程时，推荐使用连续编程模式。连续编程模式比单编程模式可以节省时间 50% 以上。连续编程模式时，对 Flash 地址写命令间隔不能超过 16μs。连续编程操作设定步骤如下：

- 步骤 1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
- 步骤 2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）
- 步骤 3. 设定连续编程模式。（EFM_FWMC.PEMOD[2:0]=0b011）
- 步骤 4. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) 对应位设定为 1)
- 步骤 5. 对编程地址写入 128 位数据，对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址连续依次分别写入 32 位数据（合计 128 位）。
编程地址需要和执行程序分属不同 Flash 块或者程序在 Flash 以外区域运行。
- 步骤 6. 等待操作结束标志位（EFM_FSR.OPTEND0/1）置位。

步骤 7. 清除操作结束标志位，直至读到 EFM_FSR.OPTEND0/1 为 0。

步骤 8. 重复步骤 5~7，直至所有数据写完。

步骤 9. 修改连续编程模式为非连续编程模式。(EFM_FWMC.PEMOD[2:0]=0b000)

步骤 10. 等待 Flash 处于空闲状态。(EFM_FSR.RDY0/1=1)

步骤 11. 读取编程地址并判断是否和写入值一致：

一致，表示编程成功；不一致，表示该 Flash 地址已遭破坏，永久废弃。

对已锁存的 OTP 地址发行连续编程写操作，编程不成功，标志位 EFM_FSR.OTPWERR0 置位。

对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址如果没有连续依次写入，编程动作就不会执行，标志位 EFM_FSR.PGSZERR0/1 置位。

8.3.4.7 擦除功能

对 Flash 进行扇区擦除操作后，该扇区内地址（8KBytes 空间）数据刷新为全 1。扇区擦除操作设定步骤如下：

步骤 1. 解除 Flash 的寄存器写保护（EFM_FAPRT 先写 0x0123，再写 0x3210）。

步骤 2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）

步骤 3. 设定擦除模式（EFM_FWMC.PEMOD[2:0]=0b100）。

步骤 4. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) 对应位设定为 1)

步骤 5. 对需要擦除扇区内的任意地址（地址需以 4 对齐）写入 32 位任意值。

步骤 6. 等待 Flash 操作结束并处于空闲状态。（EFM_FSR.OPTEND0/1=1, EFM_FSR.RDY0/1=1）

步骤 7. 清除擦除结束标志位。（EFM_FSR.OPTEND0/1）

对已锁存的 OTP 地址发行擦除操作，擦除不成功，OTP 区域数据保留，标志位 EFM_FSR.OTPWERR0 置位。

8.3.4.8 全擦除功能

EFM 提供了单块 Flash 全擦除和两块 Flash 同时全擦除两种擦除方式。全擦除操作设定步骤如下：

步骤 1. 解除 Flash 的寄存器写保护（EFM_FAPRT 先写 0x0123，再写 0x3210）。

步骤 2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）

步骤 3. 单块 Flash 全擦除时，设定 EFM_FWMC.PEMOD[2:0]=0b101。

步骤 4. 两块 Flash 同时全擦除时，设定 EFM_FWMC.PEMOD[2:0]=0b110。

步骤 5. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) Flash 块对应所有位都设定为 1)

步骤 6. 单块 Flash 全擦除：对需要擦除 Flash 的任意地址（地址需以 4 对齐）写入 32 位任意值。

步骤 7. 两块 Flash 全擦除：对 Flash 块 0 或 Flash 块 1 地址（地址需以 4 对齐）写入 32 位任意值。

全擦除动作硬件会自动同步到另一块 Flash。

步骤 8. 等待 Flash 操作结束并处于空闲状态。（EFM_FSR.OPTEND0/1=1, EFM_FSR.RDY0/1=1）

步骤 9. 清除擦除结束标志位。（EFM_FSR.OPTEND0/1）

OTP 使能后，对 Flash 块 0 地址发行单块全擦除写操作，全擦除不成功，Flash 块 0 数据保留，标志位 EFM_FSR.OTPWERR0 置位；对 Flash 块 0 地址发行两块全擦除写操作，Flash 块 0 数据保留，Flash 块 1 数据被全擦，标志位 EFM_FSR.OTPWERR0 置位。

8.3.4.9 数据安全保护

本产品对 Flash 数据提供 4 个保护等级，以防不受信任的用户通过调试接口（JTAG 和 SWD 接口），ISP 接口（In System Program）和测试接口读取和篡改 Flash。

保护级别 0：无保护

调试接口，ISP 接口和测试接口可以访问（读和改写）MCU 资源，包括 Flash 数据。

保护级别 1

Flash 地址 0x00000430-0x00000433 编程写入数据 0xAF180402，同时地址 0x03004000~0x0300400B 编程写入 96 位密码后，保护级别 1 使能。

保护级别 1 使能后：

- 调试接口，ISP 接口和测试接口无法访问 MCU 资源。
- 用户程序无法对扇区 0 进行编程和扇区擦除。
- 地址 0x03004000~0x0300400B 的数据无法读出。

激活保护级别 1 后，可通过密码认证方式或全擦除方式复归到保护级别 0。若 OTP 使能，则全擦除无效。密码认证方式和全擦除方式请咨询销售窗口。

保护级别 2

Flash 地址 0x00000434-0x00000437 编程写入数据 0xA85173AE，保护级别 2 使能。

保护级别 2 使能后：

- 调试接口，ISP 接口和测试接口无法访问 MCU 资源。
- 用户程序无法对扇区 0 进行编程和扇区擦除。

激活保护级别 2 后，可通过全擦除方式复归到保护级别 0。若 OTP 使能，则全擦除无效。

保护级别 3

Flash 地址 0x00000420-0x00000423，0x00000424-0x00000427，0x00000428-0x0000042B 编程写入数据都为 0x42545048 时，保护级别 3 使能。

保护级别 3 使能后：

- 调试接口，ISP 接口和测试接口无法改写 Flash 数据。
- 用户程序无法对扇区 0 进行编程和扇区擦除。

激活保护级别 3 后，可通过全擦除方式复归到保护级别 0。若 OTP 使能，则全擦除无效。

保护级别 1，保护级别 2 和保护级别 3 可单独使能，也可同时使能。同时使能时，保护措施叠加生效。

8.3.4.10 D-BUS 读保护功能

对地址 0x00000000~0x0001FFFF (128KBytes) 空间提供了 D-BUS 读保护。使能 D-BUS 保护，需设定 ICG3 寄存器 DBUSPRT[15:0]为 0x4450，复位重启后，功能使能。CPU-PC 指针在此 128KB 空间时（即 CPU 在此区域执行程序），CPU 可以正常访问；当 CPU-PC 指针在此 128KB 空间以外时（即 CPU 在 128KB 以外区域执行程序），CPU 对 128KB 空间内进行 D-BUS 读则被禁止，并返回总线错误。D-BUS 是 CPU 的数据访问总线，包括操作数访问，堆栈数据访问，一般数据访问等。

8.3.4.11 BGO 功能

Flash 编程，擦除目标地址和执行程序地址不在同一个 Flash 块，或者程序在 RAM 上执行，可设定 EFM_FWMC.BUSHLCTL 为 1（编程，擦除期间，总线释放），实现程序在执行同时 Flash 在进行编程或擦除。以下为 BGO 功能设定步骤：

1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
2. 解除 EFM_KEY1 锁定。（EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98）
3. 设定编程，擦除模式。（EFM_FWMC.PEMODE[2:0]=0b001/0b010/0b011/0b100/0b101）
4. 解除写保护。（EFM_F0/1NWPRTx (x=0~3) 目标地址对应扇区位设定为 1)
5. 设定编程，擦除期间总线释放（EFM_FWMC.BUSHLCTL=1）
6. 对编程地址写入 128 位数据，对末位地址为 0x0，0x4，0x8，0xC 的编程地址连续依次分别写入 32 位数据（合计 128 位）。

（编程地址和执行程序分属不同块 Flash）

若执行程序在 RAM 上，可以对两块 Flash 同时编程，（全）擦除。

7. 等待擦写结束标志位置位。(EFM_FSR.OPTEND0/1=1)

8.3.5 一次性可编程字节 (OTP)

本 MCU 提供最大 134KBytes 的 OTP 领域，地址分布如下。

分为 16 个 8KBytes，1 个 2KBytes，1 个 1KBytes，1 个 512Bytes，4 个 256Bytes，96 个 16Bytes。

表 8-3 OTP 地址分布表

Sector	OTP 块数据地址	容量	OTP 块锁存地址
0	0x00000000~0x00001FFF	8KB	0x03001800~0x0300180F
1	0x00002000~0x00003FFF	8KB	0x03001810~0x0300181F
...	...	...	...
14	0x0001C000~0x0001DFFF	8KB	0x030018E0~0x030018EF
15	0x0001E000~0x0001FFFF	8KB	0x030018F0~0x030018FF
16	0x03000000~0x0300000F	16B	0x03001900~0x0300190F
...	...	...	...
111	0x030005F0~0x030005FF	16B	0x03001EF0~0x03001EFF
112	0x03000600~0x030007FF	512B	0x03001F00~0x03001F0F
113	0x03000800~0x03000FFF	2KB	0x03001F10~0x03001F1F
114	0x03001000~0x030010FF	256B	0x03001F20~0x03001F2F
115	0x03001100~0x030011FF	256B	0x03001F30~0x03001F3F
116	0x03001200~0x030012FF	256B	0x03001F40~0x03001F4F
117	0x03001300~0x030013FF	256B	0x03001F50~0x03001F5F
118	0x03001400~0x030017FF	1KB	0x03001F60~0x03001F6F

启用 OTP 功能，需要预先对 OTP 使能地址 (0x03001FF0~0x03001FFF) 编程写入 4 个 32 位分别包含 0 数据 (推荐写入 128 位全 0)。

当 OTP 块锁存地址 4 个 32 位数据分别包含 0 时，则锁存地址对应的块空间无法再次编程，擦除和全擦除。

OTP 的 sector0~15 与 Flash 块 0 地址 0x00000000~0x0001FFFF 共享 128KBytes 物理空间；其空间的锁存地址数据若为全 1，此区域为普通 Flash 空间，可多次编程和擦除；若对应的锁存地址数据包含 0，则对应的扇区为只读，Flash 块 0 将无法进行全擦除。

对 OTP 锁存地址编程需要解锁 EFM_KEY2 寄存器 (EFM_KEY2 先写 0x10325476，再写 0xEFCDAB89)。如对 EFM_KEY2 寄存器写入错误的序列，则会发生引发一次总线错误，并且 EFM_KEY2 寄存器会锁定直至发生复位。

对 OTP 区域操作请按以下步骤进行：

- 步骤 1.** 解除 Flash 的寄存器写保护。(EFM_FAPRT 先写 0x0123，再写 0x3210)。
- 步骤 2.** 解除 EFM_KEY1 锁定。(EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98)。
- 步骤 3.** 设定编程模式。(EFM_FWMC.PEMODE[2:0]=0b001, 0b010, 0b011)。
- 步骤 4.** 根据需要对 OTP 地址写入数据。
- 步骤 5.** 解除 EFM_KEY2 锁定。(EFM_KEY2 先写 0x10325476，再写 0xEFCDAB89)。

- 步骤 6.** 对 OTP 使能地址 0x03001FF0~0x03001FFF 的 4 个 32-bit 的每 32-bit 任意 Bit 写 0。（推荐写入 128-bit 全 0）
- 步骤 7.** 对 OTP 锁存地址的 4 个 32-bit 的每 32-bit 任意 Bit 写入 0。（推荐写入 128-bit 全 0）
- 芯片复位启动后，硬件会自动加载 OTP 锁存地址数据到电路中，若 OTP 使能地址（0x03001FF0~0x03001FFF），已编程写入过 0，则需省略步骤 6。
- 对已经锁存的 OTP 数据区域地址进行编程，擦除和全擦除，将会产生 OTPWERR0 错误标志。
- OTP 锁存地址写完后，如需回归到 EFM_KEY2 锁定状态，可对 EFM_FWMC.KEY2LOCK 写入 1。

8.3.6 引导交换

- 对引导交换使能地址（0x03002000）编程写入数据 0x5A5A5A，可实现引导交换功能。
- 当 OTP 功能未启用（0x03001FF0~0x03001FFF 全 1），引导交换使能地址编程写入复位后，Flash 块 0 和 Flash 块 1 全体地址互换；此时，OTP 使能地址无法进行编程。可通过 EFM_FSWP.FSWP 位查询引导交换是否使能。
- 当 OTP 功能启用后（0x03001FF0~0x03001FFF 的 4 个 32 位数据分别包含 0），引导交换使能地址编程写入复位后，Flash 块 0 和 Flash 块 1 地址部分互换。此时，EFM_FSWP.FSWP 为 0，通过读取地址 0x03002000 数据来判断地址交换功能。
- 交换规则如下图所示：

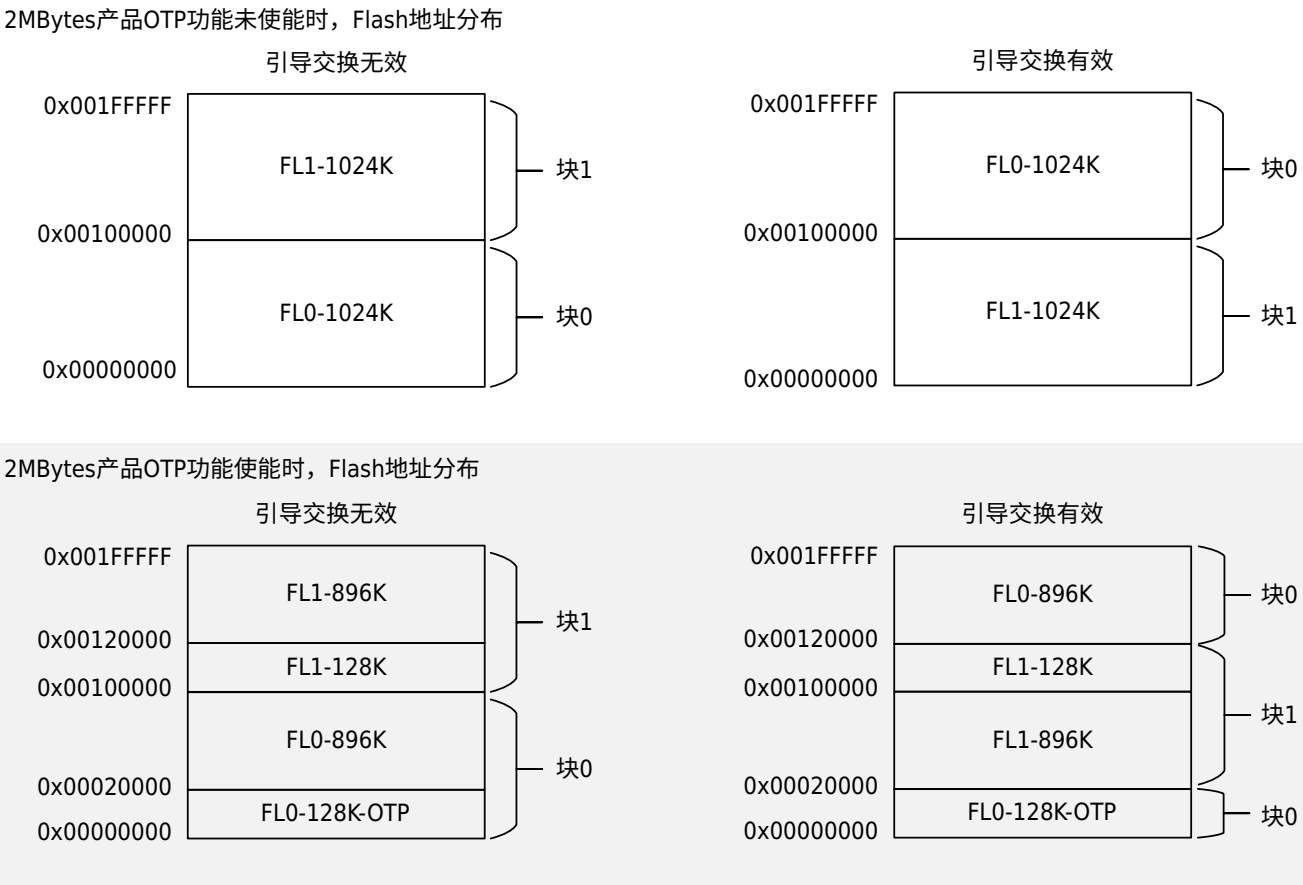


图 8-2 2MBytes 产品 Flash 地址分布

以 OTP 未启用的引导交换为例，介绍该功能的使用。

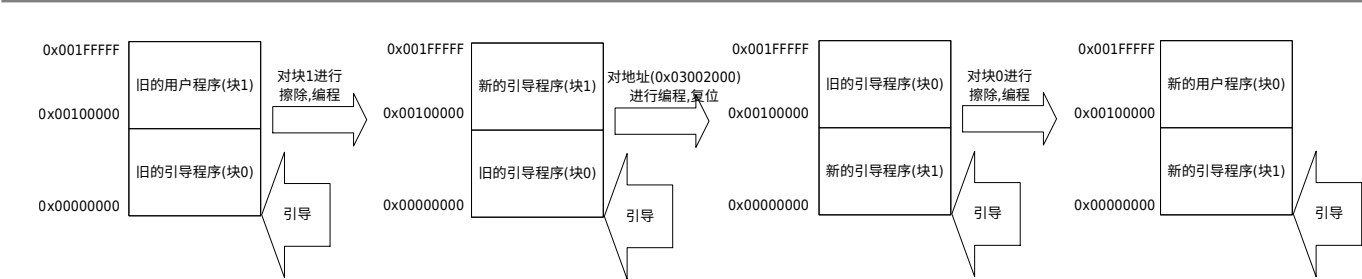


图 8-3 启动引导交换 1

当用户需要再次升级启动程序，由于地址 0x03002000 已经被编程过了，再次升级则需要对地址 0x03002000 进行擦除。操作流程如下所示：

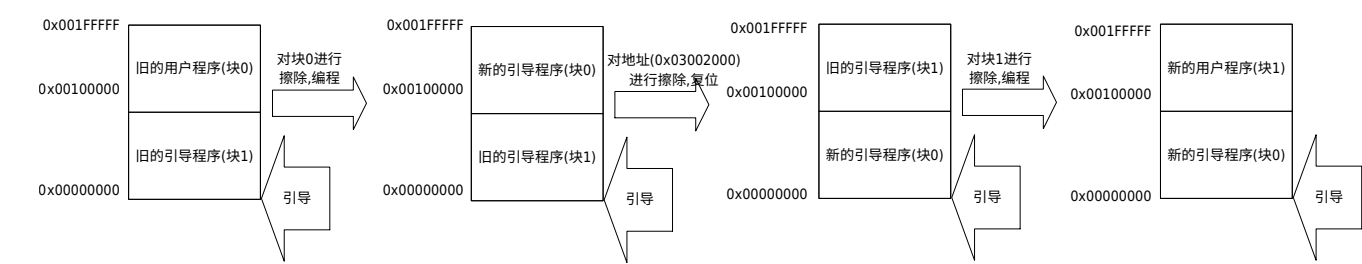


图 8-4 启动引导交换 2

8.3.7 数据可靠性和完整性

编程数据存储在带有纠错码 ECC 的 Flash 中，以防止数据损坏。

8.3.7.1 Flash 的 ECC

Flash 的 9 位 ECC 是对 128 位数据和 20 位地址相关信息计算出来的。

Flash 中的 ECC 具有以下功能：

- 通过 ECC 纠正 128 位数据和 9 位 ECC 的一位错误
- 通过 ECC 检测 20 位地址相关信息的一位错误
- 通过 ECC 检测 128 位数据和 9 位 ECC 以及 20 位地址相关信息的两位错误

Flash 中 ECC 的纠正和检测将会被验证和记录。

默认情况下，编程时会自动生成 ECC。另外，可以通过设置 EFM_F0ECCCR/F1ECCCR.GDIS=1，在编程时 ECC 使用 EFM_F0ECCDR/F1ECCDR.DWD 的数据。

默认情况下，在读取周期中，ECC 由硬件进行验证。另外，可以通过设置 EFM_F0ECCCR/F1ECCCR.VDIS=1，禁止 ECC 的纠正和检测，读出的 ECC 将发送到寄存器 EFM_F0ECCDR/F1ECCDR.DRD。
20 位地址相关信息：

- 位[19]：Reserved 位，固定为 0。
- 位[18]：0x0300 xxxx 空间为 1，块 0 和块 1 空间为 0。
- 位[17]：块 0 和块 1 的个别扇区使用救济扇区为 1，其他为 0。
- 位[16]：Reserved 位，固定为 0。
- 位[15:0]：地址[19:4]。

救济扇区：块 0 和块 1 分别有 2 个额外的扇区用来替换量产测试中发现的不良扇区。

8.3.7.2 ECC 错误

Flash 块 0 和块 1 分别有 2 个 ECC 错误记录寄存器，和 1 个 ECC 错误溢出标志位。

如果发生 ECC 错误，它将首先设置 ERR0。如果 ERR0 未被清除，并且出现新的 ECC 错误，它将设置 ERR1。如果 ERR0 和 ERR1 都有效，并且出现新的 ECC 错误，则将设置 EFM_FSR.ECEROF0/1。因此，客户需要及时清除错误记录寄存器。

8.3.7.3 错误注入读

客户可以通过设置错误注入读模式来故意设置数据错误，该模式可用于 ECC 逻辑自检。

错误注入读模式设定步骤如下：

1. 解除 Flash 的寄存器写保护。（EFM_FAPRT 先写 0x0123，再写 0x3210）
2. 设定 ECC 校验允许。（例：EFM_CKPR=0x77，EFM_CKCR=0x03000300）
3. 设定错误注入读模式。（EFM_EIEN.F0_EIEN/F1_EIEN=1）
4. 将数据写入 EFM_F0EIR/F1EIRn (n=0-4)。
 - 在上面的寄存器中，可以只设置 1 位“1”，用于检查 1 位 ECC 错误。
 - 可以设置 2 位“1”，用于检查 2 位 ECC 错误。
5. 读取已经编程的 Flash 地址。
6. 检查 EFM_CKSR，EFM_FSR，EFM_F0ERRn，EFM_F1ERRn (n=0/1) 寄存器。
7. 设定回 Normal 读模式。（EFM_EIEN.F0_EIEN/F1_EIEN=0）

上述程序应在和检查地址不同的 Flash 块上执行，或者在其他存储器（如 RAM）上执行。

8.3.8 中断

EFM 模块共有 3 个中断，分别是 PE（编程/擦除）错误中断，总线冲突中断和操作结束中断，可通过读取 EFM_FSR 寄存器查询状态标志位。

1. PE 错误中断 EFM_PEERR:

● 置位:

- ▶ 对已锁存的 OTP 地址发出编程, 擦除, 全擦除操作 (OTPWERR0=1)。
- ▶ OTP 功能启用后, 对 0x03000000~0x03001FFF 空间进行擦除, 全擦除操作 (OTPWERR0=1)。
- ▶ 对写保护扇区发出编程, 擦除, 全擦除操作 (PRTWERR0/1=1)。
- ▶ 编程地址非以 4 对齐或者数据大小非 32 位 (PGSZERR0/1=1)。
对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址如果没有连续依次写入 (PGSZERR0/1=1)。
- ▶ 单编程回读模式时, 编程地址硬件自动读取值与写入值不一致 (MISMTCH0/1=1)。

● 清零:

寄存器 EFM_FSCLR 对应标志清除位写 1, 状态位清零。

2. Flash 读写冲突中断 EFM_COLERR:

● 置位:

- ▶ Flash 忙期间 (RDY0=0 或 RDY1=0) 对该块 Flash 读写操作。(连续编程模式, 对 Flash 连续写除外)

● 清零:

寄存器 EFM_FSCLR 对应清除位置 1, 状态位清零。

3. 操作结束中断 EFM_OPTEND:

● 置位:

- ▶ 编程模式: 单个地址编程结束 (OPTEND0/1=1)。
- ▶ 擦除模式: 扇区擦除, 全擦除结束 (OPTEND0/1=1)。

● 清零:

寄存器 EFM_FSCLR 对应清除位置 1, 状态位清零。

8.4 寄存器

8.4.1 寄存器总表

EFM 基地址: 0x40010400

表 8-4 嵌入式 Flash (EFM) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	EFM_FAPRT	Flash 访问写保护寄存器	32
0x0004	EFM_KEY1	Flash 密钥 1 寄存器	32
0x0008	EFM_KEY2	Flash 密钥 2 寄存器	32
0x0014	EFM_FSTP	Flash 停止寄存器	32
0x0018	EFM_FRMC	Flash 读模式寄存器	32
0x001C	EFM_FWMC	Flash 擦写模式寄存器	32
0x0020	EFM_FSR	Flash 状态寄存器	32
0x0024	EFM_FSCLR	Flash 状态清除寄存器	32
0x0028	EFM_FITE	Flash 中断许可寄存器	32
0x002C	EFM_FSWP	Flash 引导交换状态寄存器	32
0x0030	EFM_FRMC2	Flash 读模式寄存器 2	32
0x0040	EFM_CHIPID	芯片专属标志寄存器	32
0x0050	FEM_UQID0	UNIQUE ID 寄存器 0	32
0x0054	EFM_UQID1	UNIQUE ID 寄存器 1	32
0x0058	EFM_UQID2	UNIQUE ID 寄存器 2	32
0x0180	EFM_WLOCK	Flash 写保护锁存寄存器	32
0x0190	EFM_FONWPRT0	Flash0 写保护寄存器 0	32
0x0194	EFM_FONWPRT1	Flash0 写保护寄存器 1	32
0x0198	EFM_FONWPRT2	Flash0 写保护寄存器 2	32
0x019C	EFM_FONWPRT3	Flash0 写保护寄存器 3	32
0x01A0	EFM_F1NWPRT0	Flash1 写保护寄存器 0	32
0x01A4	EFM_F1NWPRT1	Flash1 写保护寄存器 1	32
0x01A8	EFM_F1NWPRT2	Flash1 写保护寄存器 2	32
0x01AC	EFM_F1NWPRT3	Flash1 写保护寄存器 3	32
0x0350	EFM_CKCR	Flash 校验控制寄存器	32
0x0354	EFM_CKPR	Flash 校验保护寄存器	32

偏移地址	寄存器	描述	位宽
0x0358	EFM_CKSR	Flash 校验状态寄存器	32
0x035C	EFM_EIEN	Flash 错误注入使能寄存器	32
0x0370	EFM_F0ECCCR	Flash0 ECC 控制寄存器	32
0x0374	EFM_F0ECCDR	Flash0 ECC 数据寄存器	32
0x0378+0x04*n	EFM_F0ERRn	Flash0 错误记录寄存器 n (n=0~1)	32
0x0380+0x04*n	EFM_F0EIRn	Flash0 错误注入位使能寄存器 n (n=0~4)	32
0x03B0	EFM_F1ECCCR	Flash1 ECC 控制寄存器	32
0x03B4	EFM_F1ECCDR	Flash1 ECC 数据寄存器	32
0x03B8+0x04*n	EFM_F1ERRn	Flash1 错误记录寄存器 n (n=0~1)	32
0x03C0+0x04*n	EFM_F1EIRn	Flash1 错误注入位使能寄存器 n (n=0~4)	32

8.4.2 Flash 访问写保护寄存器 (EFM_FAPRT)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																FAPRT															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	FAPRT	EFM 寄存器写保护	EFM 寄存器访问写保护寄存器。 解除方法：对 FAPRT 先写“16 位数据 0x0123”再写“16 位数据 0x3210”。 在解除保护状态下，写入任意数据，EFM 寄存器再次进入保护状态。 EFM 寄存器访问保护有效时，该寄存器读出值为 0x00000000。 EFM 寄存器访问保护无效时，该寄存器读出值为 0x00000001。	RW

8.4.3 Flash 密钥 1 寄存器 (EFM_KEY1)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	KEY1																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	KEY1	密钥 1 寄存器	EFM_FWMC 写保护寄存器。 对 EFM_KEY1 先写 0x01234567，再写 0xFEDCBA98，解除对 EFM_FWMC 的写保护。 该寄存器读出值为 0x00000000。	RW

8.4.4 Flash 密钥 2 寄存器 (EFM_KEY2)

Offset	Bit Position																															
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	KEY2																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	KEY2	密钥 2 寄存器	对 OTP 锁存地址的写保护寄存器。 对 EFM_KEY2 先写 0x10325476，再写 0xEFCDA89，解除对 OTP 锁存地址的写保护。 该寄存器读出值为 0x00000000。	RW

8.4.5 Flash 停止寄存器 (EFM_FSTP)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														F1STP	F0STP
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	F1STP	Flash 块 1 停止模式控制	0b0: Flash 块 1 活动状态 0b1: Flash 块 1 处于停止模式  说明 当该位由 1 设为 0 后，请在确认 FSR.RDY1 位为 1 后，进行 Flash 块 1 访问。	RW
0	F0STP	Flash 块 0 停止模式控制	0b0: Flash 块 0 活动状态 0b1: Flash 块 0 处于停止模式  说明 当该位由 1 设为 0 后，请在确认 FSR.RDY0 位为 1 后，进行 Flash 块 0 访问。	RW

8.4.6 Flash 读模式寄存器 (EFM_FRMC)

Offset	Bit Position																																
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved												CRST	PREFE	DCACHE	ICACHE	Reserved								LVM	Reserved				FLWT			
Access													RW	RW	RW	RW									RW					RW			

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19	CRST	缓存复位位	0b0: 缓存数据解除复位 0b1: 缓存数据复位 (缓冲、缓存、预取指)	RW
18	PREFE	ICODE 预取指许可位	0b0: 关闭 ICODE 预取指功能 0b1: ICODE 预取指功能使能	RW
17	DCACHE	DCODE 缓存许可位	0b0: 关闭 DCODE 缓存功能 0b1: DCODE 缓存功能使能	RW
16	ICACHE	ICODE 缓存许可位	0b0: 关闭 ICODE 缓存功能 0b1: ICODE 缓存功能使能	RW
15:9	Reserved	保留	-	-
8	LVM	低电压读模式	0b0: 正常电压读模式 0b1: 低电压读模式  说明 超低速运行模式时需要置 LVM 为“1”。	RW
7:4	Reserved	保留	-	-
3:0	FLWT	Flash 读插入的等待周期	0b0000: 不插入读等待周期 0b0001: 插入 1 个读等待周期 0b0010: 插入 2 个读等待周期 0b1110: 插入 14 个读等待周期 0b1111: 插入 15 个读等待周期	RW

8.4.7 Flash 擦写模式寄存器 (EFM_FWMC)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Reset	0x00030000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
Name	Reserved														KEY2LOCK	KEY1LOCK	Reserved						BLKECCRSEL	BLKECCWSEL	BUSHLDCTL	Reserved						PEMOD																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Access															RW	RW																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	KEY2LOCK	Flash 密钥 2 保护状态位	写 1 锁定 EFM_KEY2 寄存器，写 0 无效 读取： 0b0: EFM_KEY2 序列解锁 0b1: EFM_KEY2 序列未解锁  说明 当 EFM_KEY2 写入错误序列后，该位保持为 1 直至复位。	RW
16	KEY1LOCK	Flash 密钥 1 保护状态位	写 1 锁定 EFM_KEY1 寄存器，写 0 无效 读取： 0b0: EFM_KEY1 序列解锁 0b1: EFM_KEY1 序列未解锁  说明 当 EFM_KEY1 写入错误序列后，该位保持为 1 直至复位。	RW
15:11	Reserved	保留	-	-
10	BLKECCRSEL	Flash 读 128 位数据及 9 位 ECC 全 1 的 ECC 选择位	0b0: 无 ECC 错误 0b1: 由算法判断 ECC 错误	RW
9	BLKECCWSEL	Flash 编程 128 位全 1 数据的 ECC 选择位	0b0: 编程时自动生成的数据或者 ECCDR 设定的数据 0b1: 0x1FF	RW
8	BUSHLDCTL	Flash 擦除/编程期间总线控制	0b0: Flash 编程擦除期间，总线被占用 0b1: Flash 编程擦除期间，总线释放	RW
7:3	Reserved	保留	-	-
2:0	PEMOD	Flash 编程/擦除/全擦除模式	0b000: 只读模式 0b001: 单编程模式 0b010: 单编程回读模式 0b011: 连续编程模式 0b100: 扇区擦除模式 0b101: 单块 Flash 全擦除模式 0b110: 两块 Flash 全擦除模式 0b111: 只读模式	RW

8.4.8 Flash 状态寄存器 (EFM_FSR)

Offset	Bit Position																																		
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x01000100																																		
Name	Reserved							RDY1	BRMER1	ECEROF1	COLERR1	OPTEND1	MISMTCH1	PGSZERR1	PRTWERR1	Reserved							RDY0	BRMER0	ECEROF0	COLERR0	OPTEND0	MISMTCH0	PGSZERR0	PRTWERR0	OTPWERR0				
Access								RO	RO	RO	RO	RO	RO	RO	RO	PRTWERR1								RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24	RDY1	Flash 块 1 忙/空闲状态	0b0: Flash 块 1 忙状态 0b1: Flash 块 1 空闲状态	RO
23	BRMER1	Flash 块 1 Blank 读错误标志位	0b0: Flash 块 1 Blank 读数据是全 1 0b1: Flash 块 1 Blank 读数据不是全 1	RO
22	ECEROF1	Flash 块 1 ECC 错误溢出标志位	0b0: Flash 块 1 ECC 错误没有溢出 0b1: Flash 块 1 ECC 错误溢出	RO
21	COLERR1	Flash 块 1 读写冲突错误标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - RDY1 未置位情况下, 对 Flash 块 1 读写操作 - 连续编程模式下, 对 Flash 块 1 写不置位 清零条件: - EFM_FSCLR 对应清零位写 1	RO
20	OPTEND1	Flash 块 1 空间编程/擦除/全擦除操作结束标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 编程/擦除/全擦除操作结束 清零条件: - EFM_FSCLR 对应清零位写 1	RO
19	MISMTCH1	Flash 块 1 空间单编程回读值不一致标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 单编程回读模式, Flash 块 1 编程后读数据和写入值不一致 清零条件: - EFM_FSCLR 对应清零位写 1	RO

位/位域	标记	位名	功能描述	读写
18	PGSZERR1	Flash 块 1 空间编程地址和大小不对齐标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 编程地址非以 4 对齐, 或者发生非 32 位写操作 - 对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址没有连续依次写入 (连续依次写入过程中终止写入场景不会触发置位) 清零条件: - EFM_FSCLR 对应清零位写 1	RO
17	PRTWERR1	Flash 块 1 写保护错误标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 对写保护扇区进行编程, 擦除, 全擦除操作 清零条件: - EFM_FSCLR 对应清零位写 1	RO
16:9	Reserved	保留	-	-
8	RDY0	Flash 块 0 忙/空闲状态	0b0: Flash 块 0 忙状态 0b1: Flash 块 0 空闲状态	RO
7	BRMER0	Flash 块 0 Blank 读错误标志位	0b0: Flash 块 0 Blank 读数据是全 1 0b1: Flash 块 0 Blank 读数据不是全 1	RO
6	ECEROF0	Flash 块 0 ECC 错误溢出标志位	0b0: Flash 块 0 ECC 错误没有溢出 0b1: Flash 块 0 ECC 错误溢出	RO
5	COLERR0	Flash 块 0 读写冲突错误标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - RDY0 未置位情况下, 对 Flash 块 0 读写操作 - 连续编程模式下, 对 Flash 块 0 写不置位 清零条件: - EFM_FSCLR 对应清零位写 1	RO
4	OPTEND0	Flash 块 0 空间编程/擦除/全擦除操作结束标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 编程/擦除/全擦除操作结束 清零条件: - EFM_FSCLR 对应清零位写 1	RO
3	MISMTCH0	Flash 块 0 空间单编程回读值不一致标志位	0b0: 未产生标志 0b1: 产生标志 置位条件: - 单编程回读模式, Flash 块 0 编程后读数据和写入值不一致 清零条件: - EFM_FSCLR 对应清零位写 1	RO

位/位域	标记	位名	功能描述	读写
2	PGSZERR0	Flash 块 0 空间编程地址和大小不对齐标志位	● 0b0: 未产生标志 ● 0b1: 产生标志 置位条件: ● 编程地址非以 4 对齐, 或者发生非 32 位写操作 ● 对末位地址为 0x0, 0x4, 0x8, 0xC 的编程地址没有连续依次写入 (连续依次写入过程中终止写入场景不会触发置位) 清零条件: ● EFM_FSCLR 对应清零位写 1	RO
1	PRTWERR0	Flash 块 0 写保护错误标志位	● 0b0: 未产生标志 ● 0b1: 产生标志 置位条件: ● 对写保护扇区进行编程, 擦除, 全擦除操作 清零条件: ● EFM_FSCLR 对应清零位写 1	RO
0	OTPWERR0	Flash 块 0 的 OTP 擦写错误标志位	● 0b0: 未产生标志 ● 0b1: 产生标志 置位条件: ● 对 OTP 锁存区域进行编程, 擦除, 全擦除操作 清零条件: ● EFM_FSCLR 对应清零位写 1	RO

8.4.9 Flash 状态清除寄存器 (EFM_FSCLR)

Offset	Bit Position																																													
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
Reset	0x00000000																																													
Name	Reserved								BRMERCLR1		ECEROFCLR1		COLERRCLR1		OPTENDCLR1		MISMTCHCLR1		PGSZERRCLR1		PRTWERRCLR1		Reserved								BRMERCLR0		ECEROFCLR0		COLERRCLR0		OPTENDCLR0		MISMTCHCLR0		PGSZERRCLR0		PRTWERRCLR0		OTPWERRCLR0	
Access									RW		RW		RW		RW		RW		RW		RW										RW		RW		RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	BRMERCLR1	Flash 块 1 Blank 读错误标志清除	写 1 清除 FSR.BRMER1 位, 写 0 无效; 读出恒为 0	RW
22	ECEROFCLR1	Flash 块 1 ECC 错误溢出标志清除	写 1 清除 FSR.ECEROF1 位, 写 0 无效; 读出恒为 0	RW
21	COLERRCLR1	Flash 块 1 读写冲突错误标志清除	写 1 清除 FSR.COLERR1 位, 写 0 无效; 读出恒为 0	RW
20	OPTENDCLR1	Flash 块 1 空间编程/擦除/全擦除操作结束标志清除	写 1 清除 FSR.OPTEND1 位, 写 0 无效; 读出恒为 0	RW
19	MISMTCHCLR1	Flash 块 1 空间单编程回读值不一致标志清除	写 1 清除 FSR.MISMTCH1 位, 写 0 无效; 读出恒为 0	RW
18	PGSZERRCLR1	Flash 块 1 空间编程地址和大小不对齐标志清除	写 1 清除 FSR.PGSZERR1 位, 写 0 无效; 读出恒为 0	RW
17	PRTWERRCLR1	Flash 块 1 写保护错误标志清除	写 1 清除 FSR.PRTWERR1 位, 写 0 无效; 读出恒为 0	RW
16:8	Reserved	保留	-	-
7	BRMERCLR0	Flash 块 0 Blank 读错误标志清除	写 1 清除 FSR.BRMER0 位, 写 0 无效; 读出恒为 0	RW
6	ECEROFCLR0	Flash 块 0 ECC 错误溢出标志清除	写 1 清除 FSR.ECEROF0 位, 写 0 无效; 读出恒为 0	RW
5	COLERRCLR0	Flash 块 0 读写冲突错误标志清除	写 1 清除 FSR.COLERR0 位, 写 0 无效; 读出恒为 0	RW
4	OPTENDCLR0	Flash 块 0 空间编程/擦除/全擦除操作结束标志清除	写 1 清除 FSR.OPTEND0 位, 写 0 无效; 读出恒为 0	RW

位/位域	标记	位名	功能描述	读写
3	MISMTCHCLR0	Flash 块 0 空间单编程回读值不一致标志清除	写 1 清除 FSR.MISMTCH0 位，写 0 无效；读出恒为 0	RW
2	PGSZERRCLR0	Flash 块 0 空间编程地址和大小不对齐标志清除	写 1 清除 FSR.PGSZERR0 位，写 0 无效；读出恒为 0	RW
1	PRTWERRCLR0	Flash 块 0 写保护错误标志清除	写 1 清除 FSR.PRTWERR0 位，写 0 无效；读出恒为 0	RW
0	OTPWERRCLR0	Flash 块 0 的 OTP 擦写错误标志清除	写 1 清除 FSR.OTPWERR0 位，写 0 无效；读出恒为 0	RW

8.4.10 Flash 中断许可寄存器 (EFM_FITE)

Offset	Bit Position																																			
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																														COLERRITE		OPTENDITE		PEERRITE	
Access																															RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	COLERRITE	读写冲突错误中断许可	● 0b0：中断不许可 ● 0b1：中断许可	RW
1	OPTENDITE	操作结束中断许可	● 0b0：中断不许可 ● 0b1：中断许可	RW
0	PEERRITE	编程/擦除错误中断许可	● 0b0：中断不许可 ● 0b1：中断许可	RW

8.4.11 Flash 引导交换状态寄存器 (EFM_FSWP)

Offset	Bit Position																															
0x002C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	Reserved																															FSWP
Access																																RO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	FSWP	块 0 和块 1 地址交换位	0b0: Flash 块 0 和块 1 地址不交换。复位后 CPU 从 Flash 块 0 启动 0b1: Flash 块 0 和块 1 地址交换。复位后 CPU 从 Flash 块 1 启动  说明 寄存器初始值由 Flash 地址 0x03002000~0x03002002 的值决定。其数据为 0x5A5A5A 时，复位启动后初值为 1；否则为 0。	RO

8.4.12 Flash 读模式寄存器 2 (EFM_FRMC2)

Offset	Bit Position																																	
0x0030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																BRM1	Reserved																BRM0
Access																	RW																	RW

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	BRM1	Flash 块 1 Blank 读模式	0b0: Normal 读模式 0b1: Blank 读模式	RW
15:1	Reserved	保留	-	-
0	BRM0	Flash 块 0 Blank 读模式	0b0: Normal 读模式 0b1: Blank 读模式	RW


说明
 Blank 读模式会禁止 ECC 验证，一旦 Blank 读确认结束，请设定回 Normal 读模式。

8.4.13 芯片专属标志寄存器 (EFM_CHIPID)

Offset	Bit Position																															
0x0040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x584804A8																															
Name	CHIPID																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	CHIPID	芯片专属标志	CHIPID[31:16]: XH 的 ASCII 码 0x5848 CHIPID[15:0]: 产品型名, 例如 0x04A8	RO

8.4.14 UNIQUE ID 寄存器 (EFM_UQID0)

Offset	Bit Position																															
0x0050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	LOT_ID[47:16]																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	LOT_ID[47:16]	LOT ID	芯片 LOT ID[47:16]	RO

8.4.15 UNIQUE ID 寄存器 (EFM_UQID1)

Offset	Bit Position																															
0x0054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved								WAFER_ID								LOT_ID[15:0]															
Access									RO								RO															

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	WAFER_ID	Wafer ID	Wafer ID	RO
15:0	LOT_ID[15:0]	LOT ID	芯片 LOT ID[15:0]	RO

8.4.16 UNIQUE ID 寄存器 (EFM_UQID2)

Offset	Bit Position																															
0x0058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																RO Y_LOCATION								RO X_LOCATION							
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	Y_LOCATION	Y 坐标	芯片在 Wafer 上的 Y 坐标	RO
7:0	X_LOCATION	X 坐标	芯片在 Wafer 上的 X 坐标	RO

8.4.17 Flash 写保护锁定寄存器 (EFM_WLOCK)

Offset	Bit Position																															
0x0180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								WLOCK[7:0]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	WLOCK[7]	F1NWPRT3 寄存器锁定位	0b0: F1NWPRT3 寄存器控制的扇区写禁止未 lockup 0b1: F1NWPRT3 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
6	WLOCK[6]	F1NWPRT2 寄存器锁定位	0b0: F1NWPRT2 寄存器控制的扇区写禁止未 lockup 0b1: F1NWPRT2 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
5	WLOCK[5]	F1NWPRT1 寄存器锁定位	0b0: F1NWPRT1 寄存器控制的扇区写禁止未 lockup 0b1: F1NWPRT1 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
4	WLOCK[4]	F1NWPRT0 寄存器锁定位	0b0: F1NWPRT0 寄存器控制的扇区写禁止未 lockup 0b1: F1NWPRT0 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
3	WLOCK[3]	F0NWPRT3 寄存器锁定位	0b0: F0NWPRT3 寄存器控制的扇区写禁止未 lockup 0b1: F0NWPRT3 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW

位/位域	标记	位名	功能描述	读写
2	WLOCK[2]	F0NWPRT2 寄存器锁定位	● 0b0: F0NWPRT2 寄存器控制的扇区写禁止未 lockup ● 0b1: F0NWPRT2 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
1	WLOCK[1]	F0NWPRT1 寄存器锁定位	● 0b0: F0NWPRT1 寄存器控制的扇区写禁止未 lockup ● 0b1: F0NWPRT1 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW
0	WLOCK[0]	F0NWPRT0 寄存器锁定位	● 0b0: F0NWPRT0 寄存器控制的扇区写禁止未 lockup ● 0b1: F0NWPRT0 寄存器控制的扇区写禁止 lockup  说明 该位一旦设定为 1，只能通过复位才能恢复成 0。	RW

8.4.18 Flash0 写保护寄存器 0 (EFM_FONWPRT0)

Offset	Bit Position																															
0x0190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	FONWPRT31	FONWPRT30	FONWPRT29	FONWPRT28	FONWPRT27	FONWPRT26	FONWPRT25	FONWPRT24	FONWPRT23	FONWPRT22	FONWPRT21	FONWPRT20	FONWPRT19	FONWPRT18	FONWPRT17	FONWPRT16	FONWPRT15	FONWPRT14	FONWPRT13	FONWPRT12	FONWPRT11	FONWPRT10	FONWPRT9	FONWPRT8	FONWPRT7	FONWPRT6	FONWPRT5	FONWPRT4	FONWPRT3	FONWPRT2	FONWPRT1	FONWPRT0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	FONWPRTx (x=31~0)	Flash 块 0 扇区 x 的写许可位	0b0: Flash 块 0 扇区 x 的写禁止 0b1: Flash 块 0 扇区 x 的写许可  说明 WLOCK[0]=0 时, 该位支持写入 0 或 1。 WLOCK[0]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.19 Flash0 写保护寄存器 1 (EFM_FONWPRT1)

Offset	Bit Position																															
0x0194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	FONWPRT63	FONWPRT62	FONWPRT61	FONWPRT60	FONWPRT59	FONWPRT58	FONWPRT57	FONWPRT56	FONWPRT55	FONWPRT54	FONWPRT53	FONWPRT52	FONWPRT51	FONWPRT50	FONWPRT49	FONWPRT48	FONWPRT47	FONWPRT46	FONWPRT45	FONWPRT44	FONWPRT43	FONWPRT42	FONWPRT41	FONWPRT40	FONWPRT39	FONWPRT38	FONWPRT37	FONWPRT36	FONWPRT35	FONWPRT34	FONWPRT33	FONWPRT32
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	FONWPRTx (x=63~32)	Flash 块 0 扇区 x 的写许可位	0b0: Flash 块 0 扇区 x 的写禁止 0b1: Flash 块 0 扇区 x 的写许可  说明 WLOCK[1]=0 时, 该位支持写入 0 或 1。 WLOCK[1]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.20 Flash0 写保护寄存器 2 (EFM_FONWPRT2)

Offset	Bit Position																															
0x0198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	FONWPRT95	FONWPRT94	FONWPRT93	FONWPRT92	FONWPRT91	FONWPRT90	FONWPRT89	FONWPRT88	FONWPRT87	FONWPRT86	FONWPRT85	FONWPRT84	FONWPRT83	FONWPRT82	FONWPRT81	FONWPRT80	FONWPRT79	FONWPRT78	FONWPRT77	FONWPRT76	FONWPRT75	FONWPRT74	FONWPRT73	FONWPRT72	FONWPRT71	FONWPRT70	FONWPRT69	FONWPRT68	FONWPRT67	FONWPRT66	FONWPRT65	FONWPRT64
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	FONWPRTx (x=95~64)	Flash 块 0 扇区 x 的写许可位	0b0: Flash 块 0 扇区 x 的写禁止 0b1: Flash 块 0 扇区 x 的写许可  说明 WLOCK[2]=0 时, 该位支持写入 0 或 1。 WLOCK[2]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.21 Flash0 写保护寄存器 3 (EFM_FONWPRT3)

Offset	Bit Position																															
0x019C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	FONWPRT127	FONWPRT126	FONWPRT125	FONWPRT124	FONWPRT123	FONWPRT122	FONWPRT121	FONWPRT120	FONWPRT119	FONWPRT118	FONWPRT117	FONWPRT116	FONWPRT115	FONWPRT114	FONWPRT113	FONWPRT112	FONWPRT111	FONWPRT110	FONWPRT109	FONWPRT108	FONWPRT107	FONWPRT106	FONWPRT105	FONWPRT104	FONWPRT103	FONWPRT102	FONWPRT101	FONWPRT100	FONWPRT99	FONWPRT98	FONWPRT97	FONWPRT96
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	FONWPRTx (x=127~96)	Flash 块 0 扇区 x 的写许可位	0b0: Flash 块 0 扇区 x 的写禁止 0b1: Flash 块 0 扇区 x 的写许可  说明 WLOCK[3]=0 时, 该位支持写入 0 或 1。 WLOCK[3]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.22 Flash1 写保护寄存器 0 (EFM_F1NWPRT0)

Offset	Bit Position																															
0x01A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	F1NWPRT31	F1NWPRT30	F1NWPRT29	F1NWPRT28	F1NWPRT27	F1NWPRT26	F1NWPRT25	F1NWPRT24	F1NWPRT23	F1NWPRT22	F1NWPRT21	F1NWPRT20	F1NWPRT19	F1NWPRT18	F1NWPRT17	F1NWPRT16	F1NWPRT15	F1NWPRT14	F1NWPRT13	F1NWPRT12	F1NWPRT11	F1NWPRT10	F1NWPRT9	F1NWPRT8	F1NWPRT7	F1NWPRT6	F1NWPRT5	F1NWPRT4	F1NWPRT3	F1NWPRT2	F1NWPRT1	F1NWPRT0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	F1NWPRTx (x=31~0)	Flash 块 1 扇区 x 的写许可位	0b0: Flash 块 1 扇区 x 的写禁止 0b1: Flash 块 1 扇区 x 的写许可  说明 WLOCK[4]=0 时, 该位支持写入 0 或 1。 WLOCK[4]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.23 Flash1 写保护寄存器 1 (EFM_F1NWPRT1)

Offset	Bit Position																															
0x01A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	F1NWPRT63	F1NWPRT62	F1NWPRT61	F1NWPRT60	F1NWPRT59	F1NWPRT58	F1NWPRT57	F1NWPRT56	F1NWPRT55	F1NWPRT54	F1NWPRT53	F1NWPRT52	F1NWPRT51	F1NWPRT50	F1NWPRT49	F1NWPRT48	F1NWPRT47	F1NWPRT46	F1NWPRT45	F1NWPRT44	F1NWPRT43	F1NWPRT42	F1NWPRT41	F1NWPRT40	F1NWPRT39	F1NWPRT38	F1NWPRT37	F1NWPRT36	F1NWPRT35	F1NWPRT34	F1NWPRT33	F1NWPRT32
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	F1NWPRTx (x=63~32)	Flash 块 1 扇区 x 的写许可位	0b0: Flash 块 1 扇区 x 的写禁止 0b1: Flash 块 1 扇区 x 的写许可  说明 WLOCK[5]=0 时, 该位支持写入 0 或 1。 WLOCK[5]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.24 Flash1 写保护寄存器 2 (EFM_F1NWPRT2)

Offset	Bit Position																															
0x01A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	F1NWPRT95	F1NWPRT94	F1NWPRT93	F1NWPRT92	F1NWPRT91	F1NWPRT90	F1NWPRT89	F1NWPRT88	F1NWPRT87	F1NWPRT86	F1NWPRT85	F1NWPRT84	F1NWPRT83	F1NWPRT82	F1NWPRT81	F1NWPRT80	F1NWPRT79	F1NWPRT78	F1NWPRT77	F1NWPRT76	F1NWPRT75	F1NWPRT74	F1NWPRT73	F1NWPRT72	F1NWPRT71	F1NWPRT70	F1NWPRT69	F1NWPRT68	F1NWPRT67	F1NWPRT66	F1NWPRT65	F1NWPRT64
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	F1NWPRTx (x=95~64)	Flash 块 1 扇区 x 的写许可位	0b0: Flash 块 1 扇区 x 的写禁止 0b1: Flash 块 1 扇区 x 的写许可  说明 WLOCK[6]=0 时, 该位支持写入 0 或 1。 WLOCK[6]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.25 Flash1 写保护寄存器 3 (EFM_F1NWPRT3)

Offset	Bit Position																															
0x01AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	F1NWPRT127	F1NWPRT126	F1NWPRT125	F1NWPRT124	F1NWPRT123	F1NWPRT122	F1NWPRT121	F1NWPRT120	F1NWPRT119	F1NWPRT118	F1NWPRT117	F1NWPRT116	F1NWPRT115	F1NWPRT114	F1NWPRT113	F1NWPRT112	F1NWPRT111	F1NWPRT110	F1NWPRT109	F1NWPRT108	F1NWPRT107	F1NWPRT106	F1NWPRT105	F1NWPRT104	F1NWPRT103	F1NWPRT102	F1NWPRT101	F1NWPRT100	F1NWPRT99	F1NWPRT98	F1NWPRT97	F1NWPRT96
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	F1NWPRTx (x=127~96)	Flash 块 1 扇区 x 的写许可位	0b0: Flash 块 1 扇区 x 的写禁止 0b1: Flash 块 1 扇区 x 的写许可  说明 WLOCK[7]=0 时, 该位支持写入 0 或 1。 WLOCK[7]=1 时, 该位只支持写 0, 写 1 无效。	RW

8.4.26 Flash 校验控制寄存器 (EFM_CKCR)

Offset	Bit Position																															
0x0350	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x02000200																															
Name	Reserved				F1ECCMOD				Reserved				F1ECCOAD				Reserved				F0ECCMOD				Reserved				F0ECCOAD			
Access					RW								RW								RW								RW			

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:24	F1ECCMOD	Flash 块 1 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
23:17	Reserved	保留	-	-
16	F1ECCOAD	Flash 块 1 的 ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
15:10	Reserved	保留	-	-
9:8	F0ECCMOD	Flash 块 0 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	F0ECCOAD	Flash 块 0 的 ECC 校验 出错后操作	0b0: Non-maskable interrupt 0b1: Reset	RW

 **说明**

20 位地址相关信息的一位错误归类于本寄存器说明里面的 2 位错误。

8.4.27 Flash 校验保护寄存器 (EFM_CKPR)

Offset	Bit Position																															
0x0354	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								CKPRKW				CKPRC			
Access																									RW				RW			

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:1	CKPRKW	写入关键码	当 CKPRC 需要置 1 或者清零时，必须同时向 CKPRKW 写入 0x3B	RW
0	CKPRC	Flash 校验控制 寄存器写使能	0b0: Flash 校验控制寄存器写入禁止 0b1: Flash 校验控制寄存器写入使能  说明 该位控制 EFM_CKCR 寄存器的写操作。当 CKPRC 为 1 时，允许对 EFM_CKCR 进行写操作；当 CKPRC 为 0 时，不允许对 EFM_CKCR 进行写操作。	RW

 **说明**

向本寄存器写入 0x00000077 可以将 CKPRC 置 1；写入 0x00000076 可以将 CKPRC 清零。

8.4.28 Flash 校验状态寄存器 (EFM_CKSR)

Offset	Bit Position																															
0x0358	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														F1_2ERR	F1_1ERR	Reserved														F0_2ERR	F0_1ERR
Access															RW	RW															RW	RW

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	F1_2ERR	Flash 块 1 发生 ECC 2-bit 错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未产生 2-bit ECC 错误 ● 0b1：产生 2-bit ECC 错误	RW
16	F1_1ERR	Flash 块 1 发生 ECC 1-bit 错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未产生 1-bit ECC 错误 ● 0b1：产生 1-bit ECC 错误	RW
15:2	Reserved	保留	-	-
1	F0_2ERR	Flash 块 0 发生 ECC 2-bit 错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未产生 2-bit ECC 错误 ● 0b1：产生 2-bit ECC 错误	RW
0	F0_1ERR	Flash 块 0 发生 ECC 1-bit 错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未产生 1-bit ECC 错误 ● 0b1：产生 1-bit ECC 错误	RW

 **说明**

EFM_CKSR 确认和清除时，推荐把 EFM_FSR.ECEROF0/1，EFM_F0ERRn (n=0~1)，EFM_F1ERRn (n=0~1) 一起确认和清除，以保持这些寄存器能同步更新反映最新结果。

20 位地址相关信息的一位错误归类于本寄存器说明里面的 2 位错误。

8.4.29 Flash 错误注入使能寄存器 (EFM_EIEN)

Offset	Bit Position																																	
0x035C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																F1_EIEN	Reserved																F0_EIEN
Access																	RW																	RW

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	F1_EIEN	Flash 块 1 错误注入使能 (错误注入读模式)	0b0: 错误注入禁止 (Normal 读模式) 0b1: 错误注入有效 (错误注入读模式)	RW
15:1	Reserved	保留	-	-
0	F0_EIEN	Flash 块 0 错误注入使能 (错误注入读模式)	0b0: 错误注入禁止 (Normal 读模式) 0b1: 错误注入有效 (错误注入读模式)	RW

8.4.30 Flash0 ECC 控制寄存器 (EFM_F0ECCCR)

Offset	Bit Position																																	
0x0370	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																														RW		VDIS	
Access																															RW		GDIS	

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	VDIS	解除自动验证 ECC	0b0: 自动验证 ECC 0b1: 将 ECC 发送到 ECCDR.DRD, 然后进行软件验证	RW
0	GDIS	解除自动生成 ECC	0b0: ECC 使用编程时自动生成的数据 0b1: ECC 使用 ECCDR.DWD 的数据	RW

8.4.31 Flash0 ECC 数据寄存器 (EFM_F0ECCDR)

Offset	Bit Position																															
0x0374	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								DRD								Reserved								DWD							
Access									RO																RW							

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24:16	DRD	读取到的 ECC	读取到的 ECC	RO
15:9	Reserved	保留	-	-
8:0	DWD	编程时的 ECC	编程时的 ECC	RW

8.4.32 Flash0 错误记录寄存器 n (EFM_F0ERRn) (n=0~1)

EFM_F0ERR0 寄存器指示 ECC 错误 0 的类型和此错误的地址索引（偏移地址）。

EFM_F0ERR1 寄存器指示 ECC 错误 1 的类型和此错误的地址索引（偏移地址）。

偏移地址（Offset）：0x0378+0x04*n

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								INDEX																Reserved		FATAL	VALID				
Access									RW																		RW	RW				

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:4	INDEX	偏移地址	INDEX[19]: 固定为 0。 INDEX[18]: 0x0300XXXX 空间为 1, 块 0 和块 1 空间为 0。 INDEX[17]: 块 0 和块 1 的个别扇区使用救济扇区为 1, 其他为 0。 INDEX[16]: 固定为 0。 INDEX[15:0]: 对应于所访问的单块 Flash 的地址右移 4 位的偏移地址。	RW
3:2	Reserved	保留	-	-
1	FATAL	记录的错误是致命错误	● 0b0: 可纠正 ● 0b1: 致命	RW
0	VALID	寄存器内容有效	● 0b0: 无效 ● 0b1: 有效	RW

- FATAL=0, VALID=1: 检测到 128 位数据和 9 位 ECC 的一位错误。
- FATAL=1, VALID=1: 检测到 20 位地址相关信息的一位错误, 检测到 128 位数据和 9 位 ECC 以及 20 位地址相关信息的两位错误。
- EFM_FSR.ECEROF0=1: 当 EFM_F0ERR0.VALID=1 和 EFM_F0ERR1.VALID=1 时, 又 1 个新的 ECC 错误发生。

 **说明**

要清除致命错误, 需要将位[1]设置为 1, 并将位[0]设置为 0。

要清除可纠正错误, 需要将位[0]设置为 0。

如果该寄存器保存有关可纠正错误的信息, 则在该寄存器中的位[0]和位[1]设置为 0 之后, INDEX[19:0]位字段将被清除。如果位[1]被设置为 1, 而位[0]被设置为 0, 则 INDEX[19:0]位字段总是被清除。

8.4.33 Flash0 错误注入位使能寄存器 n (EFM_F0EIRn) (n=0~4)

偏移地址 (Offset) : 0x0380+0x04*n

Offset	Bit Position																															
.	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name																																
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIR[i] (i=31~0)	错误注入位使能 (在错误注入读模式下, 将 ECC 错误注入 Flash 数据)	0b0: 不向 Flash 读数据中的位[n*32+i]注入错误 0b1: 向 Flash 读数据中的位[n*32+i]注入错误	RW

 **说明**

对于 EFM_F0EIR4 寄存器:

- 位[31:12]: 用于 Flash 的地址相关信息位。
- 位[11:9]: Reserved。
- 位[8:0]: 用于 Flash 的 ECC 位。

8.4.34 Flash1 ECC 控制寄存器 (EFM_F1ECCCR)

Offset	Bit Position																															
0x03B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														VDIS	GDIS
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	VDIS	解除自动验证 ECC	0b0: 自动验证 ECC 0b1: 将 ECC 发送到 ECCDR.DRD, 然后进行软件验证	RW
0	GDIS	解除自动生成 ECC	0b0: ECC 使用编程时自动生成的数据 0b1: ECC 使用 ECCDR.DWD 的数据	RW

8.4.35 Flash1 ECC 数据寄存器 (EFM_F1ECCDR)

Offset	Bit Position																															
0x03B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								DRD								Reserved								DWD							
Access									RO																RW							

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24:16	DRD	读取到的 ECC	读取到的 ECC	RO
15:9	Reserved	保留	-	-
8:0	DWD	编程时的 ECC	编程时的 ECC	RW

8.4.36 Flash1 错误记录寄存器 n (EFM_F1ERRn) (n=0~1)

EFM_F1ERR0 寄存器指示 ECC 错误 0 的类型和此错误的地址索引（偏移地址）。

EFM_F1ERR1 寄存器指示 ECC 错误 1 的类型和此错误的地址索引（偏移地址）。

偏移地址（Offset）：0x03B8+0x04*n

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								INDEX																Reserved		FATAL	VALID				
Access									RW																		RW	RW				

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:4	INDEX	偏移地址	INDEX[19]: 固定为 0。 INDEX[18]: 0x0300XXXX 空间为 1, 块 0 和块 1 空间为 0。 INDEX[17]: 块 0 和块 1 的个别扇区使用救济扇区为 1, 其他为 0。 INDEX[16]: 固定为 0。 INDEX[15:0]: 对应于所访问的单块 Flash 的地址右移 4 位的偏移地址。	RW
3:2	Reserved	保留	-	-
1	FATAL	记录的错误是致命错误	● 0b0: 可纠正 ● 0b1: 致命	RW
0	VALID	寄存器内容有效	● 0b0: 无效 ● 0b1: 有效	RW

- FATAL=0, VALID=1: 检测到 128 位数据和 9 位 ECC 的一位错误。
- FATAL=1, VALID=1: 检测到 20 位地址相关信息的一位错误, 检测到 128 位数据和 9 位 ECC 以及 20 位地址相关信息的两位错误。
- EFM_FSR.ECEROF1=1: 当 EFM_F1ERR0.VALID=1 和 EFM_F1ERR1.VALID=1 时, 又 1 个新的 ECC 错误发生。

 **说明**

要清除致命错误, 需要将位[1]设置为 1, 并将位[0]设置为 0。

要清除可纠正错误, 需要将位[0]设置为 0。

如果该寄存器保存有关可纠正错误的信息, 则在该寄存器中的位[0]和位[1]设置为 0 之后, INDEX[19:0]位字段将被清除。如果位[1]被设置为 1, 而位[0]被设置为 0, 则 INDEX[19:0]位字段总是被清除。

8.4.37 Flash1 错误注入位使能寄存器 n (EFM_F1EIRn) (n=0~4)

偏移地址 (Offset) : 0x03C0+0x04*n

Offset	Bit Position																															
.	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name																																
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIR[i] (i=31~0)	错误注入位使能 (在错误注入读模式下, 将 ECC 错误注入 Flash 数据)	0b0: 不向 Flash 读数据中的位[n*32+i]注入错误 0b1: 向 Flash 读数据中的位[n*32+i]注入错误	RW

 **说明**

对于 EFM_F1EIR4 寄存器:

- 位[31:12]: 用于 Flash 的地址相关信息位。
- 位[11:9]: Reserved。
- 位[8:0]: 用于 Flash 的 ECC 位。

8.5 注意事项

1. 在 Flash 进行擦写时，发生复位、擦写操作会强制停止，Flash 数据将得不到保证。用户需在复位解除后对地址擦除后再次进行操作。
2. 擦写操作结束后硬件电路会自动复位缓存回路。
3. 对同一地址重复编程，不能确保数据的正确性。
4. 连续编程模式时，Flash 模拟电路会有高电压状态，长期高压状态会影响 Flash 特性，一旦连续编程结束请立即退出连续编程模式。禁止在连续编程模式下，MCU 进入低功耗模式（睡眠模式、停止模式、掉电模式）。
5. 使用读加速缓存时，请确保缓存 RAM 不掉电。（PWC_PRAMLPC.PRAMPDC2=0）
6. 编程、擦除时设定总线释放（FWMC.BUSHLCTL=1）时，编程、擦除期间如需响应中断，请把向量和子程序设定到 RAM 或者编程、擦除以外的 Flash 块上。
7. ECC 校验许可时，未编程的地址的 128 位数据以及 9 位 ECC 全 1，如果发生错误，ECC 的纠正和检测不保证。

9 内置 SRAM (SRAM)

9.1 简介

本产品带有 512KB 系统 SRAM (SRAMH/SRAM1/SRAM2/SRAM3/SRAM4) 和 4KB 备份 SRAM (Ret SRAM)。

各 SRAM 可按照字节、半字 (16 位) 或全字 (32 位) 访问。高速 SRAM (SRAMH) 读写操作最快可以在 CPU 最高速度 (240MHz) 下 0 等待 (即 1 周期) 执行, 也可插入等待周期。各个 SRAM 的读写访问的等待周期由 SRAM 等待控制寄存器 (SRAM_WTCR) 设定。

Ret SRAM 可在 Power down 模式下提供 4KB 的数据保持空间。

SRAM1、SRAM2、SRAM3、SRAM4、SRAMH 和 Ret SRAM 带有 ECC 校验 (Error Checking and Correcting), ECC 校验为纠一检二码, 可以对读取数据和地址纠正一位错误, 或者检查两位错误。

9.2 功能说明

9.2.1 等待周期

各个 SRAM 的读写访问的等待周期和 CPU 时钟频率的关系如下表所示:

表 9-1 SRAM 读写访问的等待周期设定和 CPU 时钟频率的关系

等待周期 (CPU 访问周期)	访问高速 SRAM (SRAMH) 允许的 CPU 时钟频率范围	访问掉电模式备份 SRAM (Ret SRAM) 允许的 CPU 时钟频率范围	访问其它 SRAM (SRAM1,2,3,4) 允许的 CPU 时钟频率范围
0wait (1 个 CPU 周期访问)	0~240MHz	0~120MHz	0~200MHz
1wait (2 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
2wait (3 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
3wait (4 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
4wait (5 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
5wait (6 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
6wait (7 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz
7wait (8 个 CPU 周期访问)	0~240MHz	0~240MHz	0~240MHz

9.2.2 SRAM 空间分配

表 9-2 SRAM 空间分配

名称	容量	地址范围	校验方式
SRAM1	128KB	0x20000000~0x2001FFFF	ECC check
SRAM2	128KB	0x20020000~0x2003FFFF	ECC check
SRAM3	96KB	0x20040000~0x20057FFF	ECC check
SRAM4	32KB	0x20058000~0x2005FFFF	ECC check

名称	容量	地址范围	校验方式
Ret SRAM	4KB	0x200F0000~0x200F0FFF	ECC check
SRAMH	128KB	0x1FFE0000~0x1FFFFFFF	ECC check

9.2.3 ECC 校验错误注入

SRAM 的 ECC 校验带有错误注入功能，可以对 32-bit 数据及 7-bit 冗余码注入错误。根据 SRAM 错误注入使能寄存器和 SRAM 错误注入位使能寄存器的设定，硬件电路将 SRAM 读数据按位取反，以达到错误注入的目的。通过 SRAM 错误注入位使能寄存器的设定，可以制造 1-bit 错误和多 bit 错误，由于 ECC 电路只能检出不超过 2-bit 错误，因此 3-bit 及以上的误差有可能不会被检出。

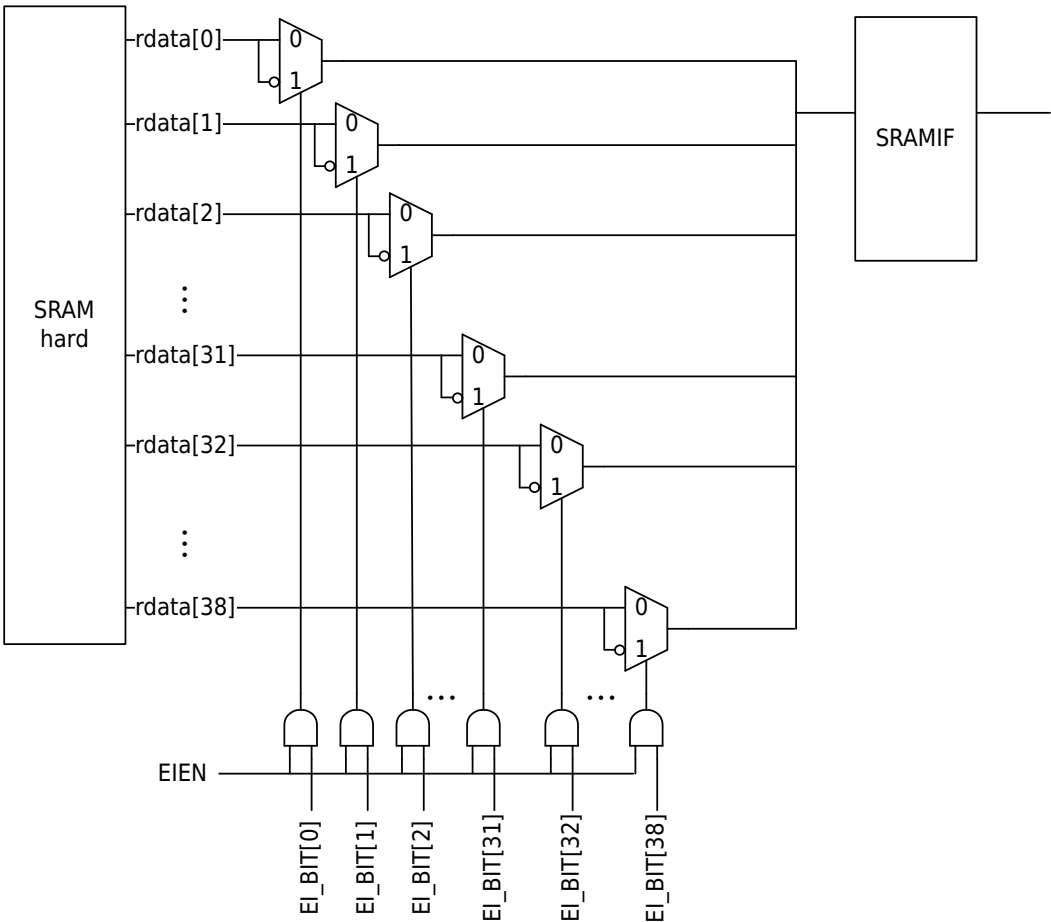


图 9-1 SRAM 39-bit 错误注入功能示意图

说明
0x20000000~0x20000002 不支持错误注入。

CACHE 内部的 RAM 的 ECC 校验带有错误注入功能，可以对 128-bit 数据及 9-bit 冗余码注入错误。根据 RAM 错误注入使能寄存器和 SRAM 错误注入位使能寄存器的设定，硬件电路将 RAM 读数据按位取反，以达到错误注入的目的。通过 RAM 错误注入位使能寄存器的设定，可以制造 1-bit 错误和多 bit 错误，由于 ECC 电路只能检出不超过 2-bit 错误，因此 3-bit 及以上的误差有可能不会被检出。

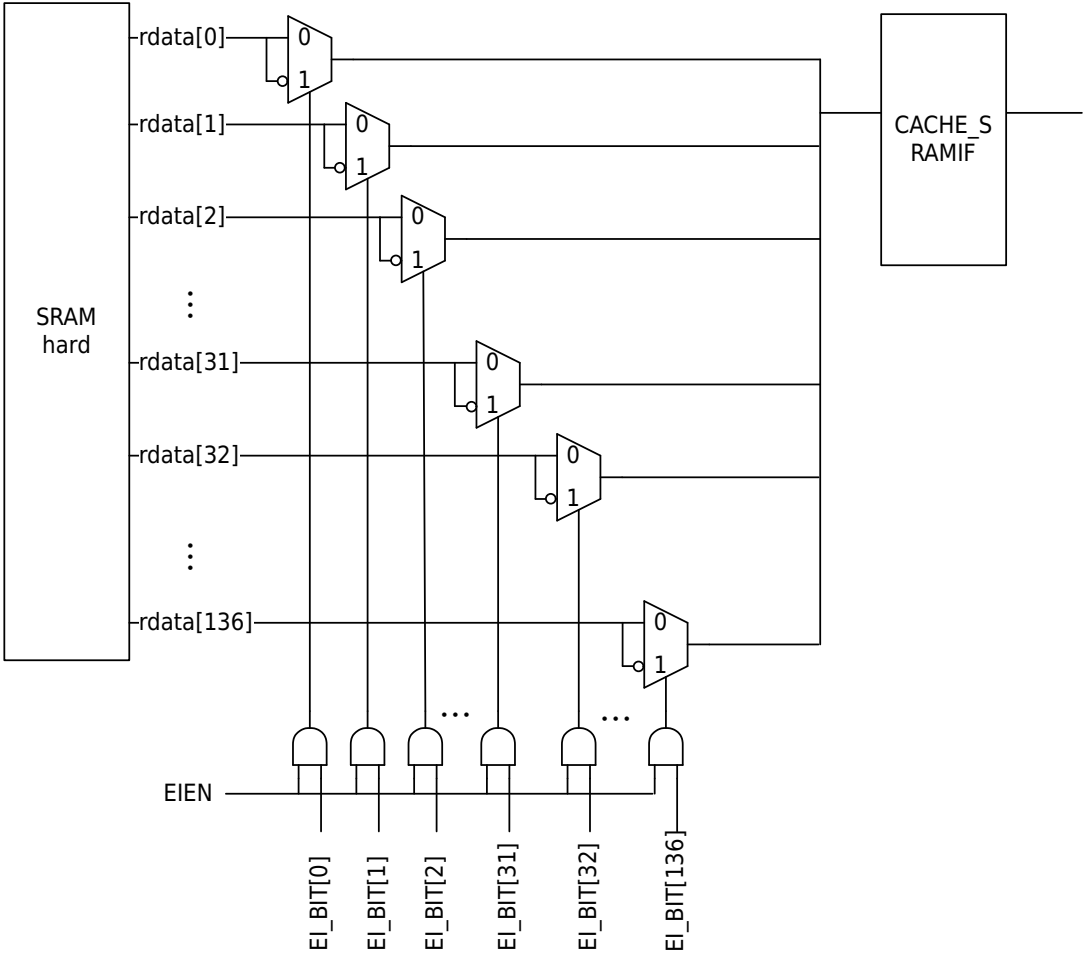


图 9-2 CACHE RAM 137-bit 错注入功能示意图

9.3 寄存器

9.3.1 寄存器总表

SRAMC 基地址：0x40050800

表 9-3 内置 SRAM (SRAM) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	SRAMC_WTCR	SRAM 等待控制寄存器	32
0x04	SRAMC_WTPR	SRAM 等待控制保护寄存器	32
0x08	SRAMC_CKCR	SRAM 校验控制寄存器	32
0x0C	SRAMC_CKPR	SRAM 校验控制保护寄存器	32
0x10	SRAMC_CKSR	SRAM 校验状态寄存器	32
0x14	SRAM1_EIEN	SRAM1 错误注入使能寄存器	32
0x18	SRAM1_EIBIT0	SRAM1 错误注入位使能寄存器 0	32
0x1C	SRAM1_EIBIT1	SRAM1 错误注入位使能寄存器 1	32
0x20	SRAM1_ECCERRADDR	SRAM1 错误地址寄存器	32
0x24	SRAM2_EIEN	SRAM2 错误注入使能寄存器	32
0x28	SRAM2_EIBIT0	SRAM2 错误注入位使能寄存器 0	32
0x2C	SRAM2_EIBIT1	SRAM2 错误注入位使能寄存器 1	32
0x30	SRAM2_ECCERRADDR	SRAM2 错误地址寄存器	32
0x34	SRAM3_EIEN	SRAM3 错误注入使能寄存器	32
0x38	SRAM3_EIBIT0	SRAM3 错误注入位使能寄存器 0	32
0x3C	SRAM3_EIBIT1	SRAM3 错误注入位使能寄存器 1	32
0x40	SRAM3_ECCERRADDR	SRAM3 错误地址寄存器	32
0x44	SRAM4_EIEN	SRAM4 错误注入使能寄存器	32
0x48	SRAM4_EIBIT0	SRAM4 错误注入位使能寄存器 0	32
0x4C	SRAM4_EIBIT1	SRAM4 错误注入位使能寄存器 1	32
0x50	SRAM4_ECCERRADDR	SRAM4 错误地址寄存器	32
0x54	SRAMH_EIEN	SRAMH 错误注入使能寄存器	32
0x58	SRAMH_EIBIT0	SRAMH 错误注入位使能寄存器 0	32
0x5C	SRAMH_EIBIT1	SRAMH 错误注入位使能寄存器 1	32
0x60	SRAMH_ECCERRADDR	SRAMH 错误地址寄存器	32
0x64	SRAMB_EIEN	Ret SRAM 错误注入使能寄存器	32

偏移地址	寄存器	描述	位宽
0x68	SRAMB_EIBIT0	Ret SRAM 错误注入位使能寄存器 0	32
0x6C	SRAMB_EIBIT1	Ret SRAM 错误注入位使能寄存器 1	32
0x70	SRAMB_ECCERRADDR	Ret SRAM 错误地址寄存器	32
0x74	CACHE_EIEN	CACHERAM 错误注入使能寄存器	32
0x78	CACHE_EIBIT0	CACHERAM 错误注入位使能寄存器 0	32
0x7C	CACHE_EIBIT1	CACHERAM 错误注入位使能寄存器 1	32
0x80	CACHE_EIBIT2	CACHERAM 错误注入位使能寄存器 2	32
0x84	CACHE_EIBIT3	CACHERAM 错误注入位使能寄存器 3	32
0x88	CACHE_EIBIT4	CACHERAM 错误注入位使能寄存器 4	32
0x8C	CACHE_ECCERRADDR	CACHERAM 错误地址寄存器	32

9.3.2 SRAM 等待控制寄存器 (SRAMC_WTCR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	SRAMBWWT			Reserved	SRAMBRWT			Reserved	SRAMHWWT			Reserved	SRAMHRWT			Reserved	SRAM4WWT			Reserved	SRAM4RWT			Reserved	SRAM123WWT			Reserved	SRAM123RWT		
Access		RW				RW				RW				RW				RW				RW				RW				RW		

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:28	SRAMBWWT	Ret SRAM 写入周期选择	● 0b000: 1 周期写 ● 0b001: 2 周期写 ● ● 0b111: 8 周期写	RW
27	Reserved	保留	-	-
26:24	SRAMBRWT	Ret SRAM 读取周期选择	● 0b000: 1 周期读 ● 0b001: 2 周期读 ● ● 0b111: 8 周期读	RW
23	Reserved	保留	-	-
22:20	SRAMHWWT	SRAMH 写入周期选择	● 0b000: 1 周期写 ● 0b001: 2 周期写 ● ● 0b111: 8 周期写	RW
19	Reserved	保留	-	-
18:16	SRAMHRWT	SRAMH 读取周期选择	● 0b000: 1 周期读 ● 0b001: 2 周期读 ● ● 0b111: 8 周期读	RW
15	Reserved	保留	-	-
14:12	SRAM4WWT	SRAM4 写入周期选择	● 0b000: 1 周期写 ● 0b001: 2 周期写 ● ● 0b111: 8 周期写	RW
11	Reserved	保留	-	-
10:8	SRAM4RWT	SRAM4 读取周期选择	● 0b000: 1 周期读 ● 0b001: 2 周期读 ● ● 0b111: 8 周期读	RW
7	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
6:4	SRAM123WWT	SRAM1, SRAM2 和 SRAM3 写入周期选择	● 0b000: 1 周期写 ● 0b001: 2 周期写 ● ● 0b111: 8 周期写	RW
3	Reserved	保留	-	-
2:0	SRAM123RWT	SRAM1, SRAM2 和 SRAM3 读取周期选择	● 0b000: 1 周期读 ● 0b001: 2 周期读 ● ● 0b111: 8 周期读	RW

9.3.3 SRAM 等待控制保护寄存器 (SRAMC_WTPR)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								WTPRKW				WTPRC			
Access																									RW				RW			

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:1	WTPRKW	写入关键码	当 WTPRC 需要置 1 或者清零时，必须同时向 WTPRKW 写入 0x3B	RW
0	WTPRC	SRAM 等待控制寄存器写控制	● 0b0: SRAM 等待控制寄存器写入禁止 ● 0b1: SRAM 等待控制寄存器写入使能  说明 该位控制 SRAM_WTCR 寄存器的写操作。当 WTPRC 为 1 时，允许对 SRAM_WTCR 进行写操作；当 WTPRC 为 0 时，不允许对 SRAM_WTCR 进行写操作。	RW

 **说明**

向本寄存器写入 0x00000077 可以将 WTPRC 置 1；写入 0x00000076 可以将 WTPRC 清零。

9.3.4 SRAM 校验控制寄存器 (SRAMC CKCR)



说明

在允许产生 RAM ECC 校验的情况下，当存取数据时，必须对所用 RAM 空间以字为单位进行初始化。

[illegible]

位/位域	标记	位名	功能描述	读写
31:30	CACHE_ECCMOD	CACHERAM 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
29:28	SRAMH_ECCMOD	SRAMH 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW

位/位域	标记	位名	功能描述	读写
27:26	SRAMB_ECCMOD	Ret SRAM 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
25:24	SRAM4_ECCMOD	SRAM4 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
23:22	SRAM3_ECCMOD	SRAM3 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW

位/位域	标记	位名	功能描述	读写
21:20	SRAM2_ECCMOD	SRAM2 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
19:18	SRAM1_ECCMOD	SRAM1 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断/复位 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断/复位	RW
17	SRAMB_ECCOAD	Ret SRAM ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
16	SRAM4_ECCOAD	SRAM4 ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
15	SRAM3_ECCOAD	SRAM3 ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
14	SRAM2_ECCOAD	SRAM2 ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
13	SRAM1_ECCOAD	SRAM1 ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
12	SRAMH_ECCOAD	SRAMH ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
11	CACHE_ECCOAD	CACHERAM ECC 校验出错后操作	● 0b0: Non-maskable interrupt ● 0b1: Reset	RW
10:0	Reserved	保留	-	-

9.3.5 SRAM 校验控制保护寄存器 (SRAMC_CKPR)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								CKPRKW				CKPRC			
Access																									RW				RW			

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:1	CKPRKW	写入关键码	当 CKPRC 需要置 1 或者清零时，必须同时向 CKPRKW 写入 0x3B	RW
0	CKPRC	SRAM 校验控制寄存器写使能	0b0：SRAM 校验控制寄存器写入禁止 0b1：SRAM 校验控制寄存器写入使能 说明 该位控制 SRAM_CKCR 寄存器的写操作。当 CKPRC 为 1 时，允许对 SRAM_CKCR 进行写操作；当 CKPRC 为 0 时，不允许对 SRAM_CKCR 进行写操作。	RW

说明

向本寄存器写入 0x00000077 可以将 CKPRC 置 1；写入 0x00000076 可以将 CKPRC 清零。

9.3.6 SRAM 校验状态寄存器 (SRAMC_CKSR)

Offset	Bit Position																																			
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved													CACHE_2ERR	CACHE_1ERR	SRAMH_2ERR	SRAMH_1ERR	SRAM3_2ERR	SRAM3_1ERR	SRAM2_2ERR	SRAM2_1ERR	SRAM1_2ERR	SRAM1_1ERR	Reserved	SRAMB_2ERR	SRAMB_1ERR	SRAM4_2ERR	SRAM4_1ERR	Reserved							
Access														RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	Reserved	RW	RW	RW	RW						

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18	CACHE_2ERR	CACHERAM 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
17	CACHE_1ERR	CACHERAM 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
16	SRAMH_2ERR	SRAMH 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
15	SRAMH_1ERR	SRAMH 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
14	SRAM3_2ERR	SRAM3 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
13	SRAM3_1ERR	SRAM3 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
12	SRAM2_2ERR	SRAM2 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
11	SRAM2_1ERR	SRAM2 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW

位/位域	标记	位名	功能描述	读写
10	SRAM1_2ERR	SRAM1 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
9	SRAM1_1ERR	SRAM1 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
8	<i>Reserved</i>	<i>保留</i>	-	-
7	SRAMB_2ERR	Ret SRAM 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
6	SRAMB_1ERR	Ret SRAM 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
5	SRAM4_2ERR	SRAM4 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
4	SRAM4_1ERR	SRAM4 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW
3:0	<i>Reserved</i>	<i>保留</i>	-	-

9.3.7 SRAM1 错误注入使能寄存器 (SRAM1_EIEN)

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.8 SRAM1 错误注入位使能寄存器 0 (SRAM1_EIBIT0)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 SRAM1 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误 说明 每个 Bit 对应一个输出数据位。	RW

9.3.9 SRAM1 错误注入位使能寄存器 1 (SRAM1_EIBIT1)

Offset	Bit Position																															
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 SRAM1 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

SRAM1 输出数据的 38~32 位为 ECC 位。

9.3.10 SRAM1 错误地址寄存器 (SRAM1_ECCERRADDR)

Offset	Bit Position																															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ECCERRADDR															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16:0	ECCERRADDR	校验错误地址	记录 SRAM1 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.11 SRAM2 错误注入使能寄存器 (SRAM2_EIEN)

Offset	Bit Position																																					
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																																					EIEN
Access																																						RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.12 SRAM2 错误注入位使能寄存器 0 (SRAM2_EIBIT0)

Offset	Bit Position																																					
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	EIBIT[31:0]																																					
Access	RW																																					

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 SRAM2 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.13 SRAM2 错误注入位使能寄存器 1 (SRAM2_EIBIT1)

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 SRAM2 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

SRAM2 输出数据的 38~32 位为 ECC 位。

9.3.14 SRAM2 错误地址寄存器 (SRAM2_ECCERRADDR)

Offset	Bit Position																															
0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ECCERRADDR															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16:0	ECCERRADDR	校验错误地址	记录 SRAM2 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.15 SRAM3 错误注入使能寄存器 (SRAM3_EIEN)

Offset	Bit Position																															
0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.16 SRAM3 错误注入位使能寄存器 0 (SRAM3_EIBIT0)

Offset	Bit Position																															
0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 SRAM3 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.17 SRAM3 错误注入位使能寄存器 1 (SRAM3_EIBIT1)

Offset	Bit Position																															
0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 SRAM3 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

 **说明**
SRAM3 输出数据的 38~32 位为 ECC 位。

9.3.18 SRAM3 错误地址寄存器 (SRAM3_ECCERRADDR)

Offset	Bit Position																															
0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ECCERRADDR															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16:0	ECCERRADDR	校验错误地址	记录 SRAM3 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.19 SRAM4 错误注入使能寄存器 (SRAM4_EIEN)

Offset	Bit Position																															
0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.20 SRAM4 错误注入位使能寄存器 0 (SRAM4_EIBIT0)

Offset	Bit Position																															
0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 SRAM4 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.21 SRAM4 错误注入位使能寄存器 1 (SRAM4_EIBIT1)

Offset	Bit Position																															
0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 SRAM4 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

SRAM4 输出数据的 38~32 位为 ECC 位。

9.3.22 SRAM4 错误地址寄存器 (SRAM4_ECCERRADDR)

Offset	Bit Position																															
0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ECCERRADDR															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14:0	ECCERRADDR	校验错误地址	记录 SRAM4 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.23 SRAMH 错误注入使能寄存器 (SRAMH_EIEN)

Offset	Bit Position																															
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.24 SRAMH 错误注入位使能寄存器 0 (SRAMH_EIBIT0)

Offset	Bit Position																															
0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 SRAMH 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.25 SRAMH 错误注入位使能寄存器 1 (SRAMH_EIBIT1)

Offset	Bit Position																															
0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 SRAMH 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

SRAMH 输出数据的 38~32 位为 ECC 位。

9.3.26 SRAMH 错误地址寄存器 (SRAMH_ECCERRADDR)

Offset	Bit Position																															
0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ECCERRADDR															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16:0	ECCERRADDR	校验错误地址	记录 SRAMH 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.27 Ret SRAM 错误注入使能寄存器 (SRAMB_EIEN)

Offset	Bit Position																															
0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.28 Ret SRAM 错误注入位使能寄存器 0 (SRAMB_EIBIT0)

Offset	Bit Position																															
0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 Ret SRAM 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误 说明 每个 Bit 对应一个输出数据位。	RW

9.3.29 Ret SRAM 错误注入位使能寄存器 1 (SRAMB_EIBIT1)

Offset	Bit Position																															
0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 Ret SRAM 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

Ret SRAM 输出数据的 38~32 位为 ECC 位。

9.3.30 Ret SRAM 错误地址寄存器 (SRAMB_ECCERRADDR)

Offset	Bit Position																															
0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ECCERRADDR											
Access																					RO											

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	ECCERRADDR	校验错误地址	记录 Ret SRAM 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.3.31 CACHERAM 错误注入使能寄存器 (CACHE_EIEN)

Offset	Bit Position																															
0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 错误注入禁止 0b1: 错误注入使能	RW

9.3.32 CACHERAM 错误注入位使能寄存器 0 (CACHE_EIBIT0)

Offset	Bit Position																															
0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 CACHERAM 输出数据的 31~0 位是否取反： 0b0: 不取反，该位不注入 ECC 错误 0b1: 取反，该位注入 ECC 错误 说明 每个 Bit 对应一个输出数据位。	RW

9.3.33 CACHERAM 错误注入位使能寄存器 1 (CACHE_EIBIT1)

Offset	Bit Position																															
0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[63:32]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[63:32]	错误注入位使能	控制 CACHERAM 输出数据的 63~32 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.34 CACHERAM 错误注入位使能寄存器 2 (CACHE_EIBIT2)

Offset	Bit Position																															
0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[95:64]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[95:64]	错误注入位使能	控制 CACHERAM 输出数据的 95~64 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.35 CACHERAM 错误注入位使能寄存器 3 (CACHE_EIBIT3)

Offset	Bit Position																															
0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[127:96]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[127:96]	错误注入位使能	控制 CACHERAM 输出数据的 127~96 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

9.3.36 CACHERAM 错误注入位使能寄存器 4 (CACHE_EIBIT4)

Offset	Bit Position																															
0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												EIBIT[136:128]																			
Access													RW																			

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	EIBIT[136:128]	错误注入位使能	控制 CACHERAM 输出数据的 136~128 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

 **说明**
CACHERAM 输出数据的 136~128 位为 ECC 位。

9.3.37 CACHERAM 错误地址寄存器 (CACHE_ECCERRADDR)

Offset	Bit Position																															
0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				RO ECCERRADDR											
Access																																

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:0	ECCERRADDR	校验错误地址	记录 CACHERAM 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

9.4 注意事项

- 1. 在允许产生 RAM ECC 校验错误的情况下，当存取数据时，必须对所用 RAM 空间以字为单位进行初始化。当从 SRAMH 空间执行指令时，必须对所用 RAM 空间+3 字的区域以字为单位进行初始化。
- 2. 备份 SRAM 在使用前需要将 VBAT 域初始化，初始化方法请参见《电源控制 (PWC) 》“电池备份电源域”章节相关描述。
- 3. SRAM1 地址 0x20000000，0x20000001，0x20000002 不支持 ECC 校验，如果在这三个地址跨区访问不会访问地址 0x20000003，在 ECC 模式时，如果在这三个地址进行非跨区访问，则会访问地址 0x20000003。
- 4. 当开启 ECC 模式后，ECC 计算以字为单位，所以当访问字节或半字时，SRAM 控制器都会自动访问对应的字，再经过内部处理来读写字节或者半字。
- 5. 因为 CPU 对 CPU-I/ CPU-D 总线的处理和 CPU-S 总线不同，所以当插入 NOP 指令时，CPU-I/ CPU-D 总线和 CPU-S 总线产生空拍的情况不同，而且也与时钟配置相关，通常情况下需要插入一条以上的 NOP 指令 CPU-S 总线才会产生空拍，而只需插入一条 NOP 指令，CPU-I/ CPU-D 总线即可产生空拍。当开启错误注入时，在背靠背的写-读同一个地址时，写和读操作需间隔至少 2 个 HCLK 周期，软件可以通过类似插入 NOP 指令等方法来插入间隔，若间隔时间不够，读操作时可能不会产生 ECC 错误。

10 通用 IO (GPIO)

10.1 简介

本章中使用的一些略称：

- Px (x=A~I) 表示一组端口，如 PA 表示 PA0~PA15 这一组的 16 个 I/O 端口。
- Pxy (x=A~H, y=0~15; x=I, y=0~13, 以下同) 表示单个 I/O 端口，如 PB10 端口表示 PB 组中第 11 个 I/O。
- GPIO (General Purpose Input Output) 通用输入输出。
- NOD (Nmos Open Drain) NMOS 开漏输出模式。

10.2 主要特性

- 每组 Port 配有 16 个 I/O Pin，根据实际配置可能不足 16 个
- 支持上拉
- 支持推挽，开漏输出模式
- 支持高、中、低型驱动模式
- 支持 CMOS/Schmitt 两种输入模式自由切换
- 支持外部中断的输入
- 支持 I/O pin 周边功能复用，一个 I/O pin 最多可具有 64 个可选的复用功能
- 各个 I/O pin 可独立编程
- 各个 I/O pin 可以选择 2 个功能同时有效（不支持 2 个输出功能同时有效）

10.3 功能说明

10.3.1 功能框图

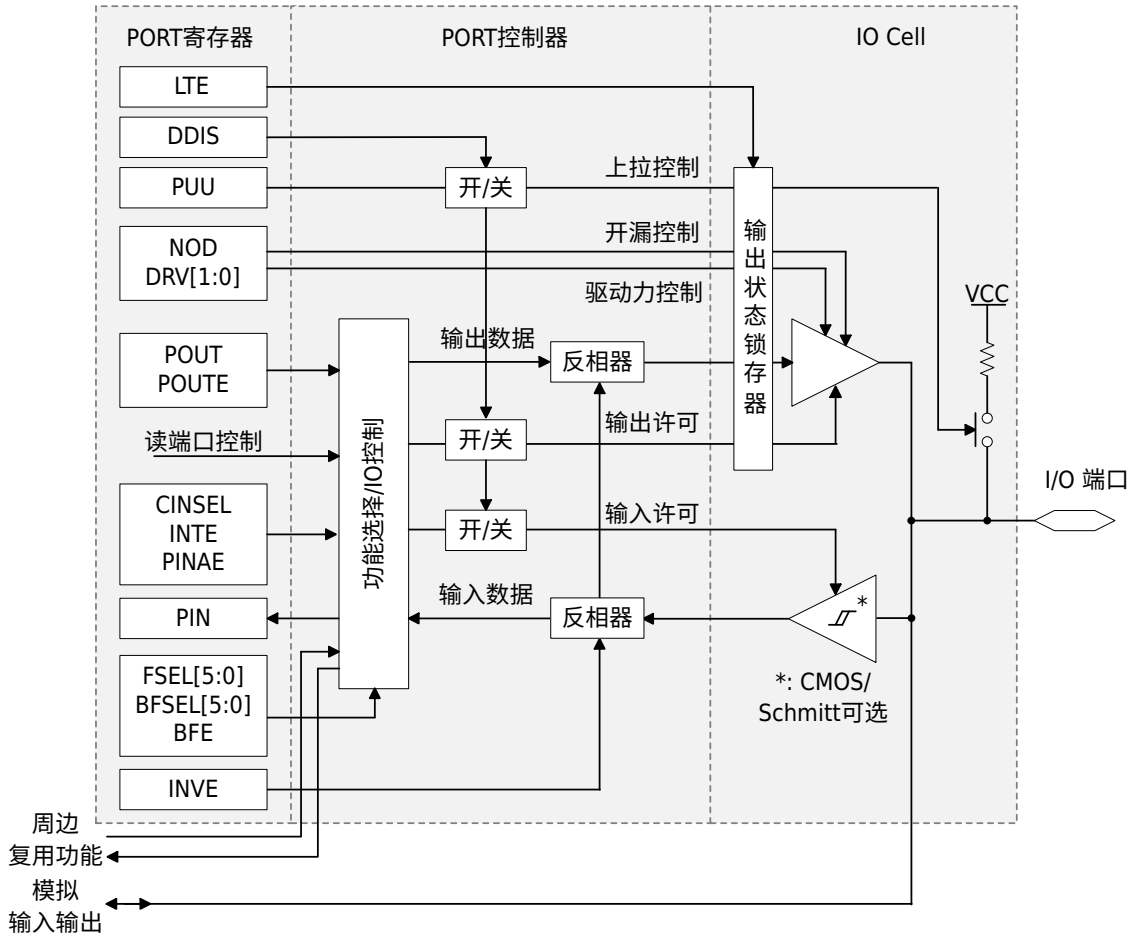


图 10-1 端口基本结构示意图

详细 GPIO 端口数目、5V 耐用、驱动能力配置请参见《数据手册》的“引脚配置及功能”章节。

10.3.2 通用输入输出 GPIO 功能

通用输入功能 GPI

各 I/O 都具有通用输入 GPI 功能，且在数字功能禁止位 PCRxy.DDIS 为 0 时，GPI 功能始终有效，与功能选择寄存器中 PFSRxy 的 FSEL[5:0] 设定值无关。通过访问端口输入数据寄存器 PIDRx 可以获取当前端口的状态。也可以通过端口控制寄存器 PCRxy 的 PIN 位查询相应的单 I/O 端口状态，PIDRx.PIN[y] 寄存器位与 PCRxy.PIN 位等价。

默认情况下，为了降低功耗，I/O 的输入 MOS 是被关闭的。只有在读取 PIDRx，PCRxy 寄存器时，才会被打开。根据需要，也可以通过设置寄存器 PINAER.PINAE[x] 为 1，让 I/O 的输入 MOS 一直处于打开状态。

当系统运行在高速时钟下，且 PINAER 设置为 0 时，由于 I/O 输入存在延迟，单周期可能无法正确读取输入状态值。此时需要设置寄存器 PCCR.RDWT[2:0]，插入若干等待周期。具体参考寄存器 PCCR 说明。

通用输出功能 GPO

除输入专用的端口，其他 I/O 端口都具有通用输出 GPO 功能。通过设置端口功能选择寄存器 PFSRxy.FSEL[5:0] 为 0x0 可以有效 GPO 功能。

GPO 功能有效时，可以通过设置通用输出许可寄存器 POERx 来允许或者禁止 I/O 的输出，通用输出数据寄存器 PODRx 来控制的输出值。使用下面的 3 个寄存器也可以控制 I/O 的输出值：输出数据清零寄存器

PORRx, 输出数据置位寄存器 POSRx, 输出数据翻转寄存器 POTRx。对上述寄存器中相应位写 1 可使对应 I/O 输出 0、1、翻转。写 0 时 I/O 输出状态不改变。

上面的寄存器都是 16 个 PORT 一组一起操作的。为了方便对单个 I/O 的控制, 也可通过设置 PCRxy.POUTE 来允许或者禁止 I/O 的输出, PCRxy.POUTE 寄存器位与 POERx.POUTEy 等价。可通过设置 PCRxy.POUT 来控制 I/O 的输出值, PCRxy.POUT 寄存器位与 PODRx.POUTy 等价。PCRxy 适合用于控制单个 PORT, POERx/PODRx 适合于控制 16 位整组 PORT。

系统复位后, 除了 JTAG 复用端口 PA13, PA14, PA15, PB3, PB4, 副振荡器复用端口 PC14, PC15 以外, 其他端口的初始功能均为 GPO (FSEL[5:0]=0x0), 且处于高阻态 (输出禁止 POUTE=0)。



说明

端口 PI13 与 MD 复用, 复位端口 NRST 解除时 PI13/MD 须固定为低电平, 以便系统能够进入用户模式。模式确立之后 PI13 可作为普通 IO 使用。

10.3.3 周边功能

通过功能选择寄存器 PFSRxy 的 FSEL[5:0], 每个端口可以配置最多 64 个功能。其中包括 FSEL[5:0]=0x0 对应的通用输出 GPO 功能。各端口具体配置的功能请参见《数据手册》的“引脚配置及功能-引脚功能说明”。

JTAG/SWD 调试功能, 使用寄存器 PSPCR 选择。PSPCR.SPFE[z] (z=0~4) 为 1 时, 对应端口的 PFSRxy.FSEL[5:0]寄存器位无效, 即 SPFE 优先级高于 FSEL。PSPCR 寄存器初始值为 0x1F, JTAG/SWD 功能有效。如果要将这些端口设置为 JTAG/SWD 以外的功能, 需要先对对应 SPFE[z]位写 0。

10.3.4 双周边功能

有些应用情况, 需要将一个端口同时设成两种功能。这种情况可以先由 PFSRxy.FSEL[5:0]选择好一个功能, 再通过设置公共控制寄存器 PCCR.BFSEL[5:0]选择第二个功能, 并设置 PFSRxy.BFE 为 1。例如: 设置 PFSRxy.FSEL[5:0]=0x2, PCCR.BFSEL[5:0]=0x5, PFSRxy.BFE=0x1, 则 Pxy 上的功能 2 与功能 5 就将同时有效。禁止在同一端口上同时有效 2 个输出功能。

10.3.5 Event Port 输入输出功能

支持 4 组 Event Port, 每组 16 个端口。Event Port1 包含 EVNTP100~EVNTP115, Event Port2 包含 EVNTP200~EVNTP215, 以此类推。EVNTPmn (m=1~4, n=0~15) 端口可以作为触发源, 根据端口输入生成事件来触发其它周边设备 (如 TIMER, ADC, DMA 等) 开始特定的动作。也可以作为被触发对象, 接收事件, 自动输入或输出。

作为触发源时, 设置 PEVNTRISRm、PEVNTFALRm、PEVNTNFCR 选择上升沿或下降沿检测, 以及数字滤波功能, 并设置功能选择寄存器 PFSRxy 选择 EVNTPmn 功能。当选择的边沿从端口输入时, 则生成事件 EVENT_PORTm, 输出到其它周边设备以触发其开始动作。

作为被触发对象时, 设置 AOS 模块 PEVNT_TRGSELM 选择触发事件源, 设置 PEVNTDIRRm 选择输出或输入功能。输出功能时, 选定事件发生时 EVNTPmn 根据 PEVNTODRm、PEVNTORRm、PEVNTOSRm 设定值输出指定电平或翻转。输入功能时, 选定事件发生时, EVNTPmn 输入状态保存进寄存器 PEVNTIDRm。

使用 Event Port 功能时, 需要先将功能时钟控制 0 寄存器 (PWC_FCG0) 的自动运行系统 AOS 功能使能位设置为有效。

10.3.6 外部中断 EIRQ 输入功能

除 PI13/MD 引脚外, 每个 I/O 端口都具有外部中断输入功能。当 PCRxy.INTE 位设为 1 时, 此 I/O 将作为外部中断源 EIRQy 被允许输入 (如: PA0 对应 EIRQ0, PA2 对应 EIRQ2)。每个 EIRQy 可配置的 I/O 不止一个, 使用时每个 EIRQy 不要同时允许多个 I/O 输入。EIRQy 输入功能与 PFSRxy.FSEL 选择的周边功能 (包括 GPIO) 可同时有效。

I/O 端口作为外部中断 EIRQ 使用时, 需要结合中断控制器 INTC, 设置滤波, 中断触发沿, 中断号等。详细请参见《中断控制器 (INTC)》章节相关描述。

10.3.7 模拟功能

部分 I/O 端口带有模拟输入输出功能（包括主副振荡器）。在用作模拟功能时，请将寄存器 PCRxy.DDIS 写 1，禁止当前端口的数字功能。

10.3.8 通用控制

上拉电阻

各 I/O 端口带有内部的上拉电阻。可以设置寄存器 PCRxy.PUU 位允许此功能，在 I/O 端口无输入时，内部处于弱上拉状态。在 I/O 端口处于输出状态时，上拉功能将自动无效。



说明

当 I/O 端口选择 I2Cx_SCL/I2Cx_SDA 功能时，将无视寄存器 PUU 的设定，内部上拉功能被强制无效。

PA11/USBFS_DM，PA12/USBFS_DP，PB14/USBHS_DM 及 PB15/USBHS_DP 引脚与 USB 端口复用，4 个引脚均内藏约 400kΩ 的下拉电阻，且一直有效。

驱动能力控制

各 I/O 端口都具有高、中、低 3 档驱动能力可调，可根据需要设置寄存器 PCRxy.DRV[1:0]。

本功能只在端口处于输出状态时才有效。

开漏输出模式

设置 PCRxy.NOD 位，可以将 I/O 端口设置成 NMOS 开漏输出模式。当 NOD 有效时，对应端口可正常输出 0，而输出 1 时端口将处于高阻态。

当 I/O 端口选择 I2Cx_SCL/I2Cx_SDA 功能时，将无视寄存器 NOD 的设定，开漏输出模式强制有效。

CMOS/Schmitt 输入模式

各 I/O 端口支持 CMOS 和 Schmitt 两种输入模式。可以设置寄存器 PCRxy.CINSEL 位选项，设置为 0 表示 Schmitt 输入，1 表示 CMOS 输入。默认为 Schmitt 输入。

在使用 I/O 输入功能时，请根据实际需求设置输入类型。

以上所述通用控制功能，如无特别说明，它们与端口具体选择的功能即 FSEL[5:0]的设置无关。

10.4 寄存器

除 Event Port 相关寄存只支持 32-bit 访问外，其他寄存器支持 16-bit 和 32-bit 访问，不支持 8-bit 访问。
32-bit 访问时这些寄存器结合方式如下：

表 10-1 32-bit 访问时 PORT 寄存器一览

地址	Bit31 ... Bit16	Bit15 ... Bit0
0x40053800+0x10*n	Reserved	PIDRx
0x40053804+0x10*n	POERx	PODRx
0x40053808+0x10*n	PORRx	POSRx
0x4005380C+0x10*n	Reserved	POTRx
0x40053BF4	Reserved	PSPCR
0x40053BF8	PINAER	PCCR
0x40053BFC	Reserved	PWPR
0x40053C00+0x40*n+0x04*y	PFSRxy	PCRxy



说明

除特殊说明，本节所有寄存器偏移地址计算公式中：

- x=A~H 对应 n=0~7, y=0~15。
- x=I 对应 n=8, y=0~13。

10.4.1 寄存器总表

通用 IO (GPIO) 基地址：0x40053800

表 10-2 通用 IO (GPIO) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00+0x10*n	PIDRx	通用输入数据寄存器 (x=A~I)	16
0x04+0x10*n	PODRx	通用输出数据寄存器 (x=A~I)	16
0x06+0x10*n	POERx	通用输出许可寄存器 (x=A~I)	16
0x08+0x10*n	POSRx	通用输出置位寄存器 (x=A~I)	16
0x0A+0x10*n	PORRx	通用输出复位寄存器 (x=A~I)	16
0x0C+0x10*n	POTRx	通用输出翻转寄存器 (x=A~I)	16
0x3F4	PSPCR	特殊控制寄存器	16
0x3F8	PCCR	公共控制寄存器	16
0x3FA	PINAER	输入控制寄存器	16
0x3FC	PWPR	写保护寄存器	16
0x400+0x40*n+0x4*y	PCRxy	通用控制寄存器 (x=A~I)	16

偏移地址	寄存器	描述	位宽
0x402+0x40*n+0x4*y	PFSRxy	功能选择寄存器 (x=A~I)	16

Event Port 基地址：0x40010800

表 10-3 Event Port 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x100+0x1C*(m-1)	PEVNTDIRRm	Event Port 方向选择寄存器	32
0x104+0x1C*(m-1)	PEVNTIDRm	Event Port 输入数据寄存器	32
0x108+0x1C*(m-1)	PEVNTODRm	Event Port 输出数据寄存器	32
0x10C+0x1C*(m-1)	PEVNTORRm	Event Port 输出数据复位寄存器	32
0x110+0x1C*(m-1)	PEVNTOSRm	Event Port 输出数据置位寄存器	32
0x114+0x1C*(m-1)	PEVNTRISRm	Event Port 上升沿输入许可寄存器	32
0x118+0x1C*(m-1)	PEVNTFALRm	Event Port 下降沿输入许可寄存器	32
0x170	PEVNTNFCR	Event Port 输入滤波控制寄存器	32

 说明

Event Port 寄存器偏移地址中 m=1~4。

10.4.2 通用输入寄存器（PIDRx）

偏移地址（Offset）：0x00+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xXXXX															
Name	PIN15	PIN14	PIN13	PIN12	PIN11	PIN10	PIN9	PIN8	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
Access	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
15:0	PINy (y=15~0)	输入状态	分别对应 Px15~Px0 端口： ● 0b0：I/O 端口输入状态为低电平 ● 0b1：I/O 端口输入状态为高电平	RO

 **说明**

端口数字功能未被禁止（PCRxy.DDIS=0）时，通过读取此寄存器可以获取端口的输入状态，与功能选择寄存器的 PFSRxy.FSEL[5:0]设定值无关。

不存在端口对应位的读出值不定。

端口数字功能被禁止（PCRxy.DDIS=1）时，由于 I/O 输入 MOS 处于关闭状态，对应 PIN 位读出值为固定值 0x1。

10.4.3 通用输出数据寄存器（PODRx）

偏移地址（Offset）：0x04+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	POUT15	POUT14	POUT13	POUT12	POUT11	POUT10	POUT9	POUT8	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:0	POUTy (y=15~0)	输出数据	分别对应 Px15~Px0 端口： ● 0b0：输出低电平 ● 0b1：输出高电平	RW

 **说明**

当 I/O 端口被设置成 GPO 功能时，改写此寄存器可以更改对应端口的输出状态。

10.4.4 通用输出许可寄存器 (POERx)

偏移地址 (Offset) : 0x06+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	POUTE15	POUTE14	POUTE13	POUTE12	POUTE11	POUTE10	POUTE9	POUTE8	POUTE7	POUTE6	POUTE5	POUTE4	POUTE3	POUTE2	POUTE1	POUTE0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:0	POUTE _y (y=15~0)	输出许可	分别对应 Px15~Px0 端口： ● 0b0：输出禁止 ● 0b1：输出许可	RW



说明

当 I/O 端口被设置成 GPO 功能时，且此寄存器设为 1 时，PODRx 设定值将输出到对应 I/O 端口。此寄存器设为 0 时，输出关闭，端口为高阻态。
禁止将不存在的端口对应位设置为 1。

10.4.5 通用输出置位寄存器 (POSRx)

偏移地址 (Offset) : 0x08+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	POS15	POS14	POS13	POS12	POS11	POS10	POS9	POS8	POS7	POS6	POS5	POS4	POS3	POS2	POS1	POS0
Access	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
15:0	POS _y (y=15~0)	输出高	分别对应 Px15~Px0 端口： ● 0b0：对应的 PODRx.POUT _y 无变化 ● 0b1：对应的 PODRx.POUT _y 置 1	WO



说明

此寄存器的读出值始终为 0x0000。32-bit 访问时，同一 I/O 的 POR_y 与 POS_y 同时写 1 时，POR_y 优先级更高，即对应 POUT_y 清零。

10.4.6 通用输出复位寄存器 (PORRx)

偏移地址 (Offset) : 0x0A+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	POR15	POR14	POR13	POR12	POR11	POR10	POR9	POR8	POR7	POR6	POR5	POR4	POR3	POR2	POR1	POR0
Access	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
15:0	PORy (y=15~0)	输出低	分别对应 Px15~Px0 端口： ● 0b0：对应的 PODRx.POUTy 无变化 ● 0b1：对应的 PODRx.POUTy 清零	WO

 **说明**

此寄存器的读出值始终为 0x0000。

10.4.7 通用输出翻转寄存器 (POTRx)

偏移地址 (Offset) : 0x0C+0x10*n

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	POT15	POT14	POT13	POT12	POT11	POT10	POT9	POT8	POT7	POT6	POT5	POT4	POT3	POT2	POT1	POT0
Access	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
15:0	POTy (y=15~0)	输出翻转	分别对应 Px15~Px0 端口： ● 0b0：对应的 PODRx.POUTy 无变化 ● 0b1：对应的 PODRx.POUTy 取反	WO

 **说明**

此寄存器的读出值始终为 0x0000。

10.4.8 特殊控制寄存器 (PSPCR)

Offset	Bit Position															
0x3F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x001F															
Name	Reserved											SPFE[4:0]				
Access												RW				

位/位域	标记	位名	功能描述	读写
15:5	Reserved	保留	-	-
4	SPFE[4]	特殊功能选择	● 0b0: NJTRST 功能无效 ● 0b1: NJTRST 功能有效	RW
3	SPFE[3]	特殊功能选择	● 0b0: JTDI 功能无效 ● 0b1: JTDI 功能有效	RW
2	SPFE[2]	特殊功能选择	● 0b0: JTDO_TRACESWO 功能无效 ● 0b1: JTDO_TRACESWO 功能有效	RW
1	SPFE[1]	特殊功能选择	● 0b0: JTMS_SWDIO 功能无效 ● 0b1: JTMS_SWDIO 功能有效	RW
0	SPFE[0]	特殊功能选择	● 0b0: JTCK_SWCLK 功能无效 ● 0b1: JTCK_SWCLK 功能有效	RW

 **说明**

PSPCR.SPFE[4:0]功能选择位优先级高于 PFSRxy.FSEL[5:0]功能选择位。

10.4.9 公共控制寄存器 (PCCR)

Offset	Bit Position															
0x3F8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x1000															
Name	Reserved	RDWT			Reserved						BFSEL					
Access		RW									RW					

位/位域	标记	位名	功能描述	读写																		
15	Reserved	保留	-	-																		
14:12	RDWT	读端口等待	设置读寄存器 PIDRx, PCRxy 时插入的等待周期数 <table border="1"><thead><tr><th>工作频率</th><th>最小等待周期</th><th>设定范围</th></tr></thead><tbody><tr><td>0<HCLK≤50MHz</td><td>0 周期</td><td>0~7</td></tr><tr><td>50MHz <HCLK≤100MHz</td><td>1 周期</td><td>1~7</td></tr><tr><td>100MHz<HCLK≤150MHz</td><td>2 周期</td><td>2~7</td></tr><tr><td>150MHz<HCLK≤200MHz</td><td>3 周期</td><td>3~7</td></tr><tr><td>200MHz<HCLK≤240MHz</td><td>4 周期</td><td>4~7</td></tr></tbody></table>	工作频率	最小等待周期	设定范围	0<HCLK≤50MHz	0 周期	0~7	50MHz <HCLK≤100MHz	1 周期	1~7	100MHz<HCLK≤150MHz	2 周期	2~7	150MHz<HCLK≤200MHz	3 周期	3~7	200MHz<HCLK≤240MHz	4 周期	4~7	RW
工作频率	最小等待周期	设定范围																				
0<HCLK≤50MHz	0 周期	0~7																				
50MHz <HCLK≤100MHz	1 周期	1~7																				
100MHz<HCLK≤150MHz	2 周期	2~7																				
150MHz<HCLK≤200MHz	3 周期	3~7																				
200MHz<HCLK≤240MHz	4 周期	4~7																				
11:6	Reserved	保留	-	-																		
5:0	BFSEL	副功能选择	各端口的功能配置请参见《数据手册》的“引脚配置及功能-引脚功能说明”。	RW																		

10.4.10 输入控制寄存器 (PINAER)

Offset	Bit Position															
0x3FA	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved								PINAER							
Access									RW							

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8:0	PINAER	输入常开	0b0: 输入 MOS 常开无效 0b1: 输入 MOS 常开有效 PINAER 的每一位与 I/O 端口的对应关系: - PINAER[0]控制 PA0~PA15 - PINAER[1]控制 PB0~PB15 - PINAER[2]控制 PC0~PC15 - PINAER[3]控制 PD0~PD15 - PINAER[4]控制 PE0~PE15 - PINAER[5]控制 PF0~PF15 - PINAER[6]控制 PG0~PG15 - PINAER[7]控制 PH0~PH15 - PINAER[8]控制 PI0~PI13	RW

10.4.11 写保护寄存器 (PWPR)

Offset	Bit Position																
0x3FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x0000																
Name	WO WP								Reserved								WE
Access																	

位/位域	标记	位名	功能描述	读写
15:8	WP	写保护码	当 WE 需要置 1 时，必须同时向 WP 写入 0xA5；读出恒为 0	WO
7:1	Reserved	保留	-	-
0	WE	写许可	● 0b0：PSPCR、PCCR、PCRxy、PFSRxy 寄存器写禁止 ● 0b1：PSPCR、PCCR、PCRxy、PFSRxy 寄存器写许可	RW

说明

向本寄存器写入 0xA501 可以将 WE 置 1；写入其他值会将 WE 清零。

10.4.12 通用控制寄存器 (PCRxy)

偏移地址 (Offset) : $0x400+0x40*n+0x4*y$

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0X00 ⁽¹⁾															
Name	DDIS	LTE	Reserved	INTE	Reserved	CINSEL	INVE	PIN	Reserved	PUU	DRV	Reserved	NOD	POUTE	POUT	
Access	RW	RW		RW		RW	RW	RO		RW	RW		RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	DDIS	数字功能禁止	0b0: 数字功能有效 0b1: 数字功能禁止	RW
14	LTE	输出状态锁存	0b0: 输出锁存无效 0b1: 输出锁存有效	RW
13	Reserved	保留	-	-
12	INTE	外部中断许可	0b0: 外部中断输入禁止 0b1: 外部中断输入许可	RW
11	Reserved	保留	-	-
10	CINSEL	输入模式选择	0b0: Schmitt 输入 0b1: CMOS 输入	RW
9	INVE	反相许可	0b0: 输入输出数据不反相 0b1: 输入输出数据反相	RW
8	PIN	输入状态	0b0: I/O 端口输入状态为低电平 0b1: I/O 端口输入状态为高电平 与寄存器 PIDRx.PINy 功能一致	RO
7	Reserved	保留	-	-
6	PUU	上拉许可	0b0: 内部上拉 (pullup) 电阻无效 0b1: 内部上拉 (pullup) 电阻有效	RW
5:4	DRV	驱动模式选择	0b00: 低驱动力模式 0b01: 中驱动力模式 0b10/0b11: 高驱动力模式	RW
3	Reserved	保留	-	-
2	NOD	NMOS 开漏	0b0: 正常 CMOS 输出模式 0b1: NMOS 开漏输出	RW
1	POUTE	输出许可	0b0: 输出禁止 0b1: 输出许可 与寄存器 POERx.POUTEy 功能一致	RW
0	POUT	输出数据	0b0: 输出低电平 0b1: 输出高电平 与寄存器 PODRx.POUTy 功能一致	RW



说明

- 1. 外部低速振荡器 XTAL32（XTAL32_IN，XTAL32_OUT）复用端口 PCRC14 和 PCRC15 的复位值为 0x8100。
- 2. DDIS 设为 1 时，对应端口的所有数字功能全部强制无效，包括通用输入输出，周边的数字输入输出，上拉功能，以及外部中断输入功能。当端口作为模拟输入时，请将 DDIS 位设为 1。
- 3. LTE 设为 1 输出锁存有效时，端口当前的输出状态保持，直至 LTE 写成 0。本功能主要在端口功能切换时使用。为避免功能切换时端口输出意想不到的毛刺而导致系统误动作，在功能切换前，先将 LTE 写 1 锁住端口的输出状态，再改写寄存器选择寄存器切换功能，最后再将 LTE 写 0 解除锁存，端口状态更新为新功能。
- 4. INVE 设为 1 时，端口的输入输出数据都会进行反相，包含 GPIO 功能，和其他周边的输入输出功能。
- 5. CINSEL 用于设置 I/O 输入模式，作用于 GPI，EIRQ 及周边输入等所有数字输入功能。

10.4.13 功能选择寄存器（PFSRxy）

偏移地址（Offset）：0x402+0x40*n+0x4*y

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							BFE	Reserved	FSEL						
Access								RW		RW						

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	BFE	副功能许可	控制 PCCR.BFSEL 选择的副功能是否有效 0b0：副功能禁止 0b1：副功能有效	RW
7:6	Reserved	保留	-	-
5:0	FSEL	功能选择	各端口的功能配置请参见《数据手册》的“引脚配置及功能-引脚功能说明”。	RW



说明

- I/O 端口可以通过 FSEL[5:0]选择配置在该端口上多个功能中的一个。FSEL[5:0]设为 0 表示选择 Func0，设为 1 表示选择 Func1，以此类推，设 15 表示选择 Func15。其中 Func0 对应的通用输出功能 GPO。
- PA13、PA14、PA15、PB3、PB4 端口复位后初始状态为 JTAG/ SWD 功能有效，在配置 FSEL[5:0]选择功能时需要先将寄存器 PSPCR 相应位写 0 无效 JTAG/ SWD 功能。PC14、PC15 端口复位后初始状态为数字功能禁止状态，在选择数字功能时需要先将相应寄存器 PCRxy 的 DDIS 位写 0。

10.4.14 Event Port 方向选择寄存器（PEVNTDIRm）

偏移地址（Offset）：0x100+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PDIR[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	PDIR[15:0]	方向选择	● 0b0：Event Port 为输入功能 ● 0b1：Event Port 为输出功能	RW

10.4.15 Event Port 输入数据寄存器（PEVNTIDRm）

偏移地址（Offset）：0x104+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PIN[15:0]															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	PIN[15:0]	端口输入状态	● 0b0：事件触发时 Event Port 输入状态为低电平 ● 0b1：事件触发时 Event Port 输入状态为高电平	RO

说明

当 Event Port 的方向设置为输入状态时，在设置的事件触发时，相应 I/O 端口的输入状态被保存到本寄存器中。

10.4.16 Event Port 输出数据寄存器（PEVNTODRm）

偏移地址（Offset）：0x108+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																POUT[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	POUT[15:0]	端口输出值	0b0：Event Port 输出低电平 0b1：Event Port 输出高电平	RW

说明

当 Event Port 的方向设置为输出时，在设置的事件触发前，Event Port 输出值为对应 POUT 位的写入值。当设置的事件触发后，根据 PEVNTORRm，PEVNTOSRm 设定值，PEVNTODRm.POUT 相应位清 0、置 1 或者翻转，同时输出到 EVNTPmn 端口上。

10.4.17 Event Port 输出数据复位寄存器（PEVNTORRm）

偏移地址（Offset）：0x10C+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																POR[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	POR[15:0]	输出值复位	0b0：事件触发时对应 PEVNTODRm.POUT 无变化 0b1：事件触发时对应 PEVNTODRm.POUT 复位	RW

说明

当 PEVNTORRm.POR 与 PEVNTOSRm.POS 都设为 1 时，事件触发时对应 PEVNTODRm.POUT 翻转。

10.4.18 Event Port 输出数据置位寄存器（PEVNTOSRm）

偏移地址（Offset）：0x110+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																POS[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	POS[15:0]	输出值置位	0b0：事件触发时对应 PEVNTODRm.POUT 无变化 0b1：事件触发时对应 PEVNTODRm.POUT 置位	RW

说明

当 PEVNTORRm.POR 与 PEVNTOSRm.POS 都设为 1 时，事件触发时对应 PEVNTODRm.POUT 翻转。

10.4.19 Event Port 上升沿输入许可寄存器（PEVNTRISRm）

偏移地址（Offset）：0x114+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RIS[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RIS[15:0]	上升沿检测许可	0b0：EVNTPmn 上升沿事件检出无效 0b1：EVNTPmn 上升沿事件检出有效 PEVNTRISRm.RIS[n]对应 EVNTPmn	RW

说明

Event Port 作为事件源，当 RIS 位设为 1 时，对应 EVNTPmn 的输入上升沿时，输出事件，用以触发其他外设。EVNTPm0~15 的边沿事件是合并成一个事件 EVENT_PORTm 输出的，其中任何一个端口检测出边沿后都会输出事件 EVENT_PORTm。

10.4.20 Event Port 下降沿输入许可寄存器 (PEVNTFALRm)

偏移地址 (Offset) : 0x118+0x1C*(m-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																FAL[15:0]															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	FAL[15:0]	下降沿检测许可	● 0b0: EVNTPmn 下降沿事件检出无效 ● 0b1: EVNTPmn 下降沿事件检出有效 PEVNTISRm.FAL[n]对应 EVNTPmn	RW

 **说明**

Event Port 作为事件源，当 FAL 位设为 1 时，对应 EVNTP 的输入下降沿时，输出事件，用以触发其他外设。
EVNTPm0~15 的边沿事件是合并成一个事件 EVENT_PORTm 输出的，其中任何一个端口检测出边沿后都会输出事件 EVENT_PORTm。

10.4.21 Event Port 输入滤波控制寄存器 (PEVNTNFCR)

Offset	Bit Position																															
0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				DIVS4		NFEN4	Reserved				DIVS3		NFEN3	Reserved				DIVS2		NFEN2	Reserved				DIVS1		NFEN1				
Access					RW		RW					RW		RW					RW		RW					RW		RW				

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26:25	DIVS4	Event Port4 数字滤波采样时钟选择	0b00: PCLK1 0b01: PCLK1/8 0b10: PCLK1/32 0b11: PCLK1/64	RW
24	NFEN4	Event Port4 数字滤波许可	0b0: 数字滤波无效 0b1: 数字滤波有效	RW
23:19	Reserved	保留	-	-
18:17	DIVS3	Event Port3 数字滤波采样时钟选择	0b00: PCLK1 0b01: PCLK1/8 0b10: PCLK1/32 0b11: PCLK1/64	RW
16	NFEN3	Event Port3 数字滤波许可	0b0: 数字滤波无效 0b1: 数字滤波有效	RW
15:11	Reserved	保留	-	-
10:9	DIVS2	Event Port2 数字滤波采样时钟选择	0b00: PCLK1 0b01: PCLK1/8 0b10: PCLK1/32 0b11: PCLK1/64	RW
8	NFEN2	Event Port2 数字滤波许可	0b0: 数字滤波无效 0b1: 数字滤波有效	RW
7:3	Reserved	保留	-	-
2:1	DIVS1	Event Port1 数字滤波采样时钟选择	0b00: PCLK1 0b01: PCLK1/8 0b10: PCLK1/32 0b11: PCLK1/64	RW
0	NFEN1	Event Port1 数字滤波许可	0b0: 数字滤波无效 0b1: 数字滤波有效	RW

10.5 注意事项

1. 请不要将同一功能设置到多个端口上。
2. 使用模拟功能时，请将相应端口的数字功能关闭(PCRxy.DDIS=0b1)。
3. 请在输出锁存有效时（PCRxy.LTE=0b1）进行端口功能切换，以避免切换期间端口上输出期待之外毛刺。

11 中断控制器 (INTC)

11.1 简介

中断控制器 (INTC) 选择中断事件作为中断请求送到 NVIC 唤醒 WFI；选择中断事件作为事件输入 (RXEV) 唤醒 WFE；选择中断事件唤醒系统低功耗模式（休眠模式和停止模式）；控制外部中断和软件中断。

INTC 的主要规格如下：

- 1. NVIC 中断请求：INTC 配备了 457 个中断事件，处理后作为中断请求 (IRQ) 发送给 NVIC，支持 144 个 IRQ，每个 IRQ 可以根据中断选择寄存器选择对应的中断事件。
- 2. 可编程优先级：16 个可编程优先级（使用了 4 位中断优先级寄存器）。
- 3. 不可屏蔽中断：可以独立选择多种系统中断事件作为不可屏蔽中断，且各中断事件配备独立的使能选择、标志及标志清除寄存器。
- 4. 配备 16 个外部中断事件。
- 5. 配备多个中断事件，具体数目请参见[中断事件表](#)。
- 6. 配备 32 个软件中断事件。
- 7. 中断可唤醒系统休眠模式和停止模式。

表 11-1 外部中断输入管脚

管脚名	I/O	说明
EIRQ0	输入	外部中断事件 0
EIRQ1	输入	外部中断事件 1
EIRQ2	输入	外部中断事件 2
EIRQ3	输入	外部中断事件 3
EIRQ4	输入	外部中断事件 4
EIRQ5	输入	外部中断事件 5
EIRQ6	输入	外部中断事件 6
EIRQ7	输入	外部中断事件 7
EIRQ8	输入	外部中断事件 8
EIRQ9	输入	外部中断事件 9
EIRQ10	输入	外部中断事件 10
EIRQ11	输入	外部中断事件 11
EIRQ12	输入	外部中断事件 12
EIRQ13	输入	外部中断事件 13
EIRQ14	输入	外部中断事件 14
EIRQ15	输入	外部中断事件 15

11.2 向量表

11.2.1 中断向量表

表 11-2 中断向量表

向量地址	向量序号	IRQ 序号	中断源	说明
ARM 内核中断				
0x00000000	0	-	ARM core	Initial Stack Pointer
0x00000004	1	-	ARM core	Initial Program Counter
0x00000008	2	-	ARM core	Non-maskable Interrupt(NMI)
0x0000000C	3	-	ARM core	Hard Fault
0x00000010	4	-	ARM core	Mem Manage Fault
0x00000014	5	-	ARM core	Bus Fault
0x00000018	6	-	ARM core	Usage Fault
0x0000001C	7	-	ARM core	Reserved
0x00000020	8	-	ARM core	Reserved
0x00000024	9	-	ARM core	Reserved
0x00000028	10	-	ARM core	Reserved
0x0000002C	11	-	ARM core	Supervisor Call(SVCall)
0x00000030	12	-	ARM core	Debug Monitor
0x00000034	13	-	ARM core	Reserved
0x00000038	14	-	ARM core	Pendable request for system service(PendSV)
0x0000003C	15	-	ARM core	System tick timer(SysTick)
非 ARM 内核中断				
0x00000040	16	0	INTC_INTSEL0	寄存器 INTC_INTSEL0 选择的中断事件/软件中断 0
0x00000044	17	1	INTC_INTSEL1	寄存器 INTC_INTSEL1 选择的中断事件/软件中断 1
0x00000048	18	2	INTC_INTSEL2	寄存器 INTC_INTSEL2 选择的中断事件/软件中断 2
0x0000004C	19	3	INTC_INTSEL3	寄存器 INTC_INTSEL3 选择的中断事件/软件中断 3
0x00000050	20	4	INTC_INTSEL4	寄存器 INTC_INTSEL4 选择的中断事件/软件中断 4
0x00000054	21	5	INTC_INTSEL5	寄存器 INTC_INTSEL5 选择的中断事件/软件中断 5
0x00000058	22	6	INTC_INTSEL6	寄存器 INTC_INTSEL6 选择的中断事件/软件中断 6
0x0000005C	23	7	INTC_INTSEL7	寄存器 INTC_INTSEL7 选择的中断事件/软件中断 7
0x00000060	24	8	INTC_INTSEL8	寄存器 INTC_INTSEL8 选择的中断事件/软件中断 8
0x00000064	25	9	INTC_INTSEL9	寄存器 INTC_INTSEL9 选择的中断事件/软件中断 9
0x00000068	26	10	INTC_INTSEL10	寄存器 INTC_INTSEL10 选择的中断事件/软件中断 10
0x0000006C	27	11	INTC_INTSEL11	寄存器 INTC_INTSEL11 选择的中断事件/软件中断 11
0x00000070	28	12	INTC_INTSEL12	寄存器 INTC_INTSEL12 选择的中断事件/软件中断 12
0x00000074	29	13	INTC_INTSEL13	寄存器 INTC_INTSEL13 选择的中断事件/软件中断 13
0x00000078	30	14	INTC_INTSEL14	寄存器 INTC_INTSEL14 选择的中断事件/软件中断 14
0x0000007C	31	15	INTC_INTSEL15	寄存器 INTC_INTSEL15 选择的中断事件/软件中断 15
0x00000080	32	16	INTC_INTSEL16	寄存器 INTC_INTSEL16 选择的中断事件请求/软件中断
0x00000084	33	17	INTC_INTSEL17	寄存器 INTC_INTSEL17 选择的中断事件请求/软件中断
0x00000088	34	18	INTC_INTSEL18	寄存器 INTC_INTSEL18 选择的中断事件请求/软件中断
0x0000008C	35	19	INTC_INTSEL19	寄存器 INTC_INTSEL19 选择的中断事件请求/软件中断
0x00000090	36	20	INTC_INTSEL20	寄存器 INTC_INTSEL20 选择的中断事件请求/软件中断
0x00000094	37	21	INTC_INTSEL21	寄存器 INTC_INTSEL21 选择的中断事件请求/软件中断
0x00000098	38	22	INTC_INTSEL22	寄存器 INTC_INTSEL22 选择的中断事件请求/软件中断

向量地址	向量序号	IRQ 序号	中断源	说明
0x0000009C	39	23	INTC_INTSEL23	寄存器 INTC_INTSEL23 选择的 interrupt 事件请求/软件中断
0x000000A0	40	24	INTC_INTSEL24	寄存器 INTC_INTSEL24 选择的 interrupt 事件请求/软件中断
0x000000A4	41	25	INTC_INTSEL25	寄存器 INTC_INTSEL25 选择的 interrupt 事件请求/软件中断
0x000000A8	42	26	INTC_INTSEL26	寄存器 INTC_INTSEL26 选择的 interrupt 事件请求/软件中断
0x000000AC	43	27	INTC_INTSEL27	寄存器 INTC_INTSEL27 选择的 interrupt 事件请求/软件中断
0x000000B0	44	28	INTC_INTSEL28	寄存器 INTC_INTSEL28 选择的 interrupt 事件请求/软件中断
0x000000B4	45	29	INTC_INTSEL29	寄存器 INTC_INTSEL29 选择的 interrupt 事件请求/软件中断
0x000000B8	46	30	INTC_INTSEL30	寄存器 INTC_INTSEL30 选择的 interrupt 事件请求/软件中断
0x000000BC	47	31	INTC_INTSEL31	寄存器 INTC_INTSEL31 选择的 interrupt 事件请求/软件中断
0x000000C0	48	32	INTC_INTSEL32	寄存器 INTC_INTSEL32 选择的 interrupt 事件请求
0x000000C4	49	33	INTC_INTSEL33	寄存器 INTC_INTSEL33 选择的 interrupt 事件请求
0x000000C8	50	34	INTC_INTSEL34	寄存器 INTC_INTSEL34 选择的 interrupt 事件请求
0x000000CC	51	35	INTC_INTSEL35	寄存器 INTC_INTSEL35 选择的 interrupt 事件请求
0x000000D0	52	36	INTC_INTSEL36	寄存器 INTC_INTSEL36 选择的 interrupt 事件请求
0x000000D4	53	37	INTC_INTSEL37	寄存器 INTC_INTSEL37 选择的 interrupt 事件请求
0x000000D8	54	38	INTC_INTSEL38	寄存器 INTC_INTSEL38 选择的 interrupt 事件请求
0x000000DC	55	39	INTC_INTSEL39	寄存器 INTC_INTSEL39 选择的 interrupt 事件请求
0x000000E0	56	40	INTC_INTSEL40	寄存器 INTC_INTSEL40 选择的 interrupt 事件请求
0x000000E4	57	41	INTC_INTSEL41	寄存器 INTC_INTSEL41 选择的 interrupt 事件请求
0x000000E8	58	42	INTC_INTSEL42	寄存器 INTC_INTSEL42 选择的 interrupt 事件请求
0x000000EC	59	43	INTC_INTSEL43	寄存器 INTC_INTSEL43 选择的 interrupt 事件请求
0x000000F0	60	44	INTC_INTSEL44	寄存器 INTC_INTSEL44 选择的 interrupt 事件请求
0x000000F4	61	45	INTC_INTSEL45	寄存器 INTC_INTSEL45 选择的 interrupt 事件请求
0x000000F8	62	46	INTC_INTSEL46	寄存器 INTC_INTSEL46 选择的 interrupt 事件请求
0x000000FC	63	47	INTC_INTSEL47	寄存器 INTC_INTSEL47 选择的 interrupt 事件请求
0x00000100	64	48	INTC_INTSEL48	寄存器 INTC_INTSEL48 选择的 interrupt 事件请求
0x00000104	65	49	INTC_INTSEL49	寄存器 INTC_INTSEL49 选择的 interrupt 事件请求
0x00000108	66	50	INTC_INTSEL50	寄存器 INTC_INTSEL50 选择的 interrupt 事件请求
0x0000010C	67	51	INTC_INTSEL51	寄存器 INTC_INTSEL51 选择的 interrupt 事件请求
0x00000110	68	52	INTC_INTSEL52	寄存器 INTC_INTSEL52 选择的 interrupt 事件请求
0x00000114	69	53	INTC_INTSEL53	寄存器 INTC_INTSEL53 选择的 interrupt 事件请求
0x00000118	70	54	INTC_INTSEL54	寄存器 INTC_INTSEL54 选择的 interrupt 事件请求
0x0000011C	71	55	INTC_INTSEL55	寄存器 INTC_INTSEL55 选择的 interrupt 事件请求
0x00000120	72	56	INTC_INTSEL56	寄存器 INTC_INTSEL56 选择的 interrupt 事件请求
0x00000124	73	57	INTC_INTSEL57	寄存器 INTC_INTSEL57 选择的 interrupt 事件请求
0x00000128	74	58	INTC_INTSEL58	寄存器 INTC_INTSEL58 选择的 interrupt 事件请求
0x0000012C	75	59	INTC_INTSEL59	寄存器 INTC_INTSEL59 选择的 interrupt 事件请求
0x00000130	76	60	INTC_INTSEL60	寄存器 INTC_INTSEL60 选择的 interrupt 事件请求
0x00000134	77	61	INTC_INTSEL61	寄存器 INTC_INTSEL61 选择的 interrupt 事件请求
0x00000138	78	62	INTC_INTSEL62	寄存器 INTC_INTSEL62 选择的 interrupt 事件请求
0x0000013C	79	63	INTC_INTSEL63	寄存器 INTC_INTSEL63 选择的 interrupt 事件请求
0x00000140	80	64	INTC_INTSEL64	寄存器 INTC_INTSEL64 选择的 interrupt 事件请求
0x00000144	81	65	INTC_INTSEL65	寄存器 INTC_INTSEL65 选择的 interrupt 事件请求
0x00000148	82	66	INTC_INTSEL66	寄存器 INTC_INTSEL66 选择的 interrupt 事件请求
0x0000014C	83	67	INTC_INTSEL67	寄存器 INTC_INTSEL67 选择的 interrupt 事件请求
0x00000150	84	68	INTC_INTSEL68	寄存器 INTC_INTSEL68 选择的 interrupt 事件请求

向量地址	向量序号	IRQ 序号	中断源	说明
0x00000154	85	69	INTC_INTSEL69	寄存器 INTC_INTSEL69 选择的中断事件请求
0x00000158	86	70	INTC_INTSEL70	寄存器 INTC_INTSEL70 选择的中断事件请求
0x0000015C	87	71	INTC_INTSEL71	寄存器 INTC_INTSEL71 选择的中断事件请求
0x00000160	88	72	INTC_INTSEL72	寄存器 INTC_INTSEL72 选择的中断事件请求
0x00000164	89	73	INTC_INTSEL73	寄存器 INTC_INTSEL73 选择的中断事件请求
0x00000168	90	74	INTC_INTSEL74	寄存器 INTC_INTSEL74 选择的中断事件请求
0x0000016C	91	75	INTC_INTSEL75	寄存器 INTC_INTSEL75 选择的中断事件请求
0x00000170	92	76	INTC_INTSEL76	寄存器 INTC_INTSEL76 选择的中断事件请求
0x00000174	93	77	INTC_INTSEL77	寄存器 INTC_INTSEL77 选择的中断事件请求
0x00000178	94	78	INTC_INTSEL78	寄存器 INTC_INTSEL78 选择的中断事件请求
0x0000017C	95	79	INTC_INTSEL79	寄存器 INTC_INTSEL79 选择的中断事件请求
0x00000180	96	80	INTC_INTSEL80	寄存器 INTC_INTSEL80 选择的中断事件请求
0x00000184	97	81	INTC_INTSEL81	寄存器 INTC_INTSEL81 选择的中断事件请求
0x00000188	98	82	INTC_INTSEL82	寄存器 INTC_INTSEL82 选择的中断事件请求
0x0000018C	99	83	INTC_INTSEL83	寄存器 INTC_INTSEL83 选择的中断事件请求
0x00000190	100	84	INTC_INTSEL84	寄存器 INTC_INTSEL84 选择的中断事件请求
0x00000194	101	85	INTC_INTSEL85	寄存器 INTC_INTSEL85 选择的中断事件请求
0x00000198	102	86	INTC_INTSEL86	寄存器 INTC_INTSEL86 选择的中断事件请求
0x0000019C	103	87	INTC_INTSEL87	寄存器 INTC_INTSEL87 选择的中断事件请求
0x000001A0	104	88	INTC_INTSEL88	寄存器 INTC_INTSEL88 选择的中断事件请求
0x000001A4	105	89	INTC_INTSEL89	寄存器 INTC_INTSEL89 选择的中断事件请求
0x000001A8	106	90	INTC_INTSEL90	寄存器 INTC_INTSEL90 选择的中断事件请求
0x000001AC	107	91	INTC_INTSEL91	寄存器 INTC_INTSEL91 选择的中断事件请求
0x000001B0	108	92	INTC_INTSEL92	寄存器 INTC_INTSEL92 选择的中断事件请求
0x000001B4	109	93	INTC_INTSEL93	寄存器 INTC_INTSEL93 选择的中断事件请求
0x000001B8	110	94	INTC_INTSEL94	寄存器 INTC_INTSEL94 选择的中断事件请求
0x000001BC	111	95	INTC_INTSEL95	寄存器 INTC_INTSEL95 选择的中断事件请求
0x000001C0	112	96	INTC_INTSEL96	寄存器 INTC_INTSEL96 选择的中断事件请求
0x000001C4	113	97	INTC_INTSEL97	寄存器 INTC_INTSEL97 选择的中断事件请求
0x000001C8	114	98	INTC_INTSEL98	寄存器 INTC_INTSEL98 选择的中断事件请求
0x000001CC	115	99	INTC_INTSEL99	寄存器 INTC_INTSEL99 选择的中断事件请求
0x000001D0	116	100	INTC_INTSEL100	寄存器 INTC_INTSEL100 选择的中断事件请求
0x000001D4	117	101	INTC_INTSEL101	寄存器 INTC_INTSEL101 选择的中断事件请求
0x000001D8	118	102	INTC_INTSEL102	寄存器 INTC_INTSEL102 选择的中断事件请求
0x000001DC	119	103	INTC_INTSEL103	寄存器 INTC_INTSEL103 选择的中断事件请求
0x000001E0	120	104	INTC_INTSEL104	寄存器 INTC_INTSEL104 选择的中断事件请求
0x000001E4	121	105	INTC_INTSEL105	寄存器 INTC_INTSEL105 选择的中断事件请求
0x000001E8	122	106	INTC_INTSEL106	寄存器 INTC_INTSEL106 选择的中断事件请求
0x000001EC	123	107	INTC_INTSEL107	寄存器 INTC_INTSEL107 选择的中断事件请求
0x000001F0	124	108	INTC_INTSEL108	寄存器 INTC_INTSEL108 选择的中断事件请求
0x000001F4	125	109	INTC_INTSEL109	寄存器 INTC_INTSEL109 选择的中断事件请求
0x000001F8	126	110	INTC_INTSEL110	寄存器 INTC_INTSEL110 选择的中断事件请求
0x000001FC	127	111	INTC_INTSEL111	寄存器 INTC_INTSEL111 选择的中断事件请求
0x00000200	128	112	INTC_INTSEL112	寄存器 INTC_INTSEL112 选择的中断事件请求
0x00000204	129	113	INTC_INTSEL113	寄存器 INTC_INTSEL113 选择的中断事件请求
0x00000208	130	114	INTC_INTSEL114	寄存器 INTC_INTSEL114 选择的中断事件请求

向量地址	向量序号	IRQ 序号	中断源	说明
0x0000020C	131	115	INTC_INTSEL115	寄存器 INTC_INTSEL115 选择的中断事件请求
0x00000210	132	116	INTC_INTSEL116	寄存器 INTC_INTSEL116 选择的中断事件请求
0x00000214	133	117	INTC_INTSEL117	寄存器 INTC_INTSEL117 选择的中断事件请求
0x00000218	134	118	INTC_INTSEL118	寄存器 INTC_INTSEL118 选择的中断事件请求
0x0000021C	135	119	INTC_INTSEL119	寄存器 INTC_INTSEL119 选择的中断事件请求
0x00000220	136	120	INTC_INTSEL120	寄存器 INTC_INTSEL120 选择的中断事件请求
0x00000224	137	121	INTC_INTSEL121	寄存器 INTC_INTSEL121 选择的中断事件请求
0x00000228	138	122	INTC_INTSEL122	寄存器 INTC_INTSEL122 选择的中断事件请求
0x00000234	141	125	INTC_INTSEL123	寄存器 INTC_INTSEL123 选择的中断事件请求
0x00000238	142	126	INTC_INTSEL124	寄存器 INTC_INTSEL124 选择的中断事件请求
0x0000023C	143	127	INTC_INTSEL125	寄存器 INTC_INTSEL125 选择的中断事件请求
0x00000240	144	128	INTC_VSSEL128	寄存器 INTC_VSSEL128 通过使能位选择的中断事件请求共用该向量
0x00000244	145	129	INTC_VSSEL129	寄存器 INTC_VSSEL129 通过使能位选择的中断事件请求共用该向量
0x00000248	146	130	INTC_VSSEL130	寄存器 INTC_VSSEL130 通过使能位选择的中断事件请求共用该向量
0x0000024C	147	131	INTC_VSSEL131	寄存器 INTC_VSSEL131 通过使能位选择的中断事件请求共用该向量
0x00000250	148	132	INTC_VSSEL132	寄存器 INTC_VSSEL132 通过使能位选择的中断事件请求共用该向量
0x00000254	149	133	INTC_VSSEL133	寄存器 INTC_VSSEL133 通过使能位选择的中断事件请求共用该向量
0x00000258	150	134	INTC_VSSEL134	寄存器 INTC_VSSEL134 通过使能位选择的中断事件请求共用该向量
0x0000025C	151	135	INTC_VSSEL135	寄存器 INTC_VSSEL135 通过使能位选择的中断事件请求共用该向量
0x00000260	152	136	INTC_VSSEL136	寄存器 INTC_VSSEL136 通过使能位选择的中断事件请求共用该向量
0x00000264	153	137	INTC_VSSEL137	寄存器 INTC_VSSEL137 通过使能位选择的中断事件请求共用该向量
0x00000268	154	138	INTC_VSSEL138	寄存器 INTC_VSSEL138 通过使能位选择的中断事件请求共用该向量
0x0000026C	155	139	INTC_VSSEL139	寄存器 INTC_VSSEL139 通过使能位选择的中断事件请求共用该向量
0x00000270	156	140	INTC_VSSEL140	寄存器 INTC_VSSEL140 通过使能位选择的中断事件请求共用该向量
0x00000274	157	141	INTC_VSSEL141	寄存器 INTC_VSSEL141 通过使能位选择的中断事件请求共用该向量
0x00000278	158	142	INTC_VSSEL142	寄存器 INTC_VSSEL142 通过使能位选择的中断事件请求共用该向量
0x0000027C	159	143	INTC_VSSEL143	寄存器 INTC_VSSEL143 通过使能位选择的中断事件请求共用该向量



说明

INTC_INTSEL 寄存器选择的中断事件编号请参见[中断事件表](#)。

11.2.2 中断事件表

中断事件由外设产生，作为 IRQ 输入时称为中断源；作为 RXEV 输入时称为事件源；作为外设触发事件时称为触发源；中断事件也可以作为从低功耗模式返回的条件。

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
0	0x000	PORT	EIRQ0	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[0]
1	0x001		EIRQ1	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[1]
2	0x002		EIRQ2	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[2]
3	0x003		EIRQ3	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[3]
4	0x004		EIRQ4	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[4]
5	0x005		EIRQ5	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[5]
6	0x006		EIRQ6	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[6]
7	0x007		EIRQ7	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[7]
8	0x008		EIRQ8	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[8]
9	0x009		EIRQ9	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[9]
10	0x00A		EIRQ10	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[10]
11	0x00B		EIRQ11	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[11]
12	0x00C		EIRQ12	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[12]
13	0x00D		EIRQ13	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[13]
14	0x00E		EIRQ14	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[14]
15	0x00F		EIRQ15	P/L	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[15]
16	0x010	Timer0_3	TMR0_3_CMPA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[16]
17	0x011		TMR0_3_CMPB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[17]
18	0x012	Timer0_4	TMR0_4_CMPA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[18]
19	0x013		TMR0_4_CMPB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[19]
20	0x014	Timer0_5	TMR0_5_CMPA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[20]
21	0x015		TMR0_5_CMPB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[21]
22	0x016	Timer0_1	TMR0_1_OVFA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[22]
23	0x017		TMR0_1_OVFB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[23]
24	0x018	Timer0_2	TMR0_2_OVFA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[24]
25	0x019		TMR0_2_OVFB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[25]
26	0x01A	Timer0_3	TMR0_3_OVFA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[26]
27	0x01B		TMR0_3_OVFB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[27]
28	0x01C	Timer0_4	TMR0_4_OVFA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[28]
29	0x01D		TMR0_4_OVFB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[29]
30	0x01E	Timer0_5	TMR0_5_OVFA	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[30]
31	0x01F		TMR0_5_OVFB	P	✓	✓	INTSEL0~31	INTSEL32~37	VSSEL128[31]
32	0x020	DMA_1	DMA_1_TC0	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[0]
33	0x021		DMA_1_TC1	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[1]
34	0x022		DMA_1_TC2	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[2]
35	0x023		DMA_1_TC3	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[3]
36	0x024		DMA_1_TC4	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[4]
37	0x025		DMA_1_TC5	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[5]
38	0x026		DMA_1_TC6	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[6]
39	0x027		DMA_1_TC7	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[7]
40	0x028		DMA_1 BTC0	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[8]
41	0x029		DMA_1 BTC1	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[9]
42	0x02A		DMA_1 BTC2	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[10]
43	0x02B		DMA_1 BTC3	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[11]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
44	0x02C	DMA_1	DMA_1_BTC4	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[12]
45	0x02D		DMA_1_BTC5	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[13]
46	0x02E		DMA_1_BTC6	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[14]
47	0x02F		DMA_1_BTC7	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[15]
48	0x030		DMA_1_ERR	P	✓	-	INTSEL0~31	INTSEL38~43	VSSEL129[16]
49	0x031	EFM	EFM_PEERR	P	✓	-	INTSEL0~31	INTSEL38~43	VSSEL129[17]
50	0x032		EFM_RDCOL	P	✓	-	INTSEL0~31	INTSEL38~43	VSSEL129[18]
51	0x033		EFM_OPTEND	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[19]
52	0x034	USBFS	USBFS_SOF	-	-	✓	INTSEL0~31	INTSEL38~43	VSSEL129[20]
53	0x035	USBHS	USBHS_SOF	-	-	✓	INTSEL0~31	INTSEL38~43	VSSEL129[21]
54	0x036	QSPI	QSPI_INTR	L	✓	-	INTSEL0~31	INTSEL38~43	VSSEL129[22]
55	0x037	DCU	DCU1	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[23]
56	0x038		DCU2	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[24]
57	0x039		DCU3	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[25]
58	0x03A		DCU4	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[26]
59	0x03B		DCU5	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[27]
60	0x03C		DCU6	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[28]
61	0x03D		DCU7	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[29]
62	0x03E		DCU8	P	✓	✓	INTSEL0~31	INTSEL38~43	VSSEL129[30]
63	0x03F	FPU	FPU_ERR	L	✓	-	INTSEL0~31	INTSEL38~43	VSSEL129[31]
64	0x040	DMA_2	DMA_2_TC0	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[0]
65	0x041		DMA_2_TC1	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[1]
66	0x042		DMA_2_TC2	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[2]
67	0x043		DMA_2_TC3	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[3]
68	0x044		DMA_2_TC4	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[4]
69	0x045		DMA_2_TC5	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[5]
70	0x046		DMA_2_TC6	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[6]
71	0x047		DMA_2_TC7	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[7]
72	0x048		DMA_2_BTC0	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[8]
73	0x049		DMA_2_BTC1	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[9]
74	0x04A		DMA_2_BTC2	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[10]
75	0x04B		DMA_2_BTC3	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[11]
76	0x04C		DMA_2_BTC4	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[12]
77	0x04D		DMA_2_BTC5	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[13]
78	0x04E		DMA_2_BTC6	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[14]
79	0x04F		DMA_2_BTC7	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[15]
80	0x050		DMA_2_ERR	P	✓	-	INTSEL0~31	INTSEL44~49	VSSEL130[16]
81	0x051	-	-	-	-	-	INTSEL0~31	INTSEL44~49	VSSEL130[17]
82	0x052	-	-	-	-	-	INTSEL0~31	INTSEL44~49	VSSEL130[18]
83	0x053	MAU	MAU_SQRT	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[19]
84	0x054	DVP	DVP_FRAMSTA	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[20]
85	0x055		DVP_LINESTA	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[21]
86	0x056		DVP_LINEEND	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[22]
87	0x057		DVP_FRAMEND	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[23]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
88	0x058	DVP	DVP_SQUERR	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[24]
89	0x059		DVP_FIFOERR	L	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[25]
90	0x05A		DVP_DMAREQ	-	-	✓	INTSEL0~31	INTSEL44~49	VSSEL130[26]
91	0x05B	FMAC	FMAC_1_FIR	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[27]
92	0x05C		FMAC_2_FIR	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[28]
93	0x05D		FMAC_3_FIR	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[29]
94	0x05E		FMAC_4_FIR	P	✓	✓	INTSEL0~31	INTSEL44~49	VSSEL130[30]
95	0x05F	-	-	-	-	-	INTSEL0~31	INTSEL44~49	VSSEL130[31]
96	0x060	Timer0_1	TMR0_1_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[0]
97	0x061		TMR0_1_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[1]
98	0x062	Timer0_2	TMR0_2_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[2]
99	0x063		TMR0_2_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[3]
100	0x064	Timer2_1	TMR2_1_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[4]
101	0x065		TMR2_1_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[5]
102	0x066		TMR2_1_OVFA	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[6]
103	0x067		TMR2_1_OVFB	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[7]
104	0x068	Timer2_2	TMR2_2_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[8]
105	0x069		TMR2_2_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[9]
106	0x06A		TMR2_2_OVFA	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[10]
107	0x06B		TMR2_2_OVFB	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[11]
108	0x06C	Timer2_3	TMR2_3_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[12]
109	0x06D		TMR2_3_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[13]
110	0x06E		TMR2_3_OVFA	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[14]
111	0x06F		TMR2_3_OVFB	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[15]
112	0x070	Timer2_4	TMR2_4_CMPA	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[16]
113	0x071		TMR2_4_CMPB	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[17]
114	0x072		TMR2_4_OVFA	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[18]
115	0x073		TMR2_4_OVFB	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[19]
116	0x074	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[20]
117	0x075	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[21]
118	0x076	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[22]
119	0x077	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[23]
120	0x078	RTC	RTC_TP	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[24]
121	0x079		RTC_ALM	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[25]
122	0x07A		RTC_PRD	P	✓	✓	INTSEL0~31	INTSEL50~55	VSSEL131[26]
123	0x07B	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[27]
124	0x07C	-	-	-	-	-	INTSEL0~31	INTSEL50~55	VSSEL131[28]
125	0x07D	XTAL	XTAL_STOP	L	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[29]
126	0x07E	WKTM	WKTM_PRD	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[30]
127	0x07F	SWDT	SWDT_REFUDF	P	✓	-	INTSEL0~31	INTSEL50~55	VSSEL131[31]
128	0x080	Timer6_1	TMR6_1_GCMA	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[0]
129	0x081		TMR6_1_GCMB	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[1]
130	0x082		TMR6_1_GCMC	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[2]
131	0x083		TMR6_1_GCMD	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[3]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
132	0x084	Timer6_1	TMR6_1_GCME	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[4]
133	0x085		TMR6_1_GCMF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[5]
134	0x086		TMR6_1_GOVF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[6]
135	0x087		TMR6_1_GUDF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[7]
136	0x088	Timer4_1	TMR4_1_GCMUH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[8]
			TMR4_1_SCM0	-	-	✓			
137	0x089		TMR4_1_GCMUL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[9]
			TMR4_1_SCM1	-	-	✓			
138	0x08A		TMR4_1_GCMVH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[10]
			TMR4_1_SCM2	-	-	✓			
139	0x08B		TMR4_1_GCMVL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[11]
			TMR4_1_SCM3	-	-	✓			
140	0x08C		TMR4_1_GCMWH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[12]
			TMR4_1_SCM4	-	-	✓			
141	0x08D		TMR4_1_GCMWL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[13]
			TMR4_1_SCM5	-	-	✓			
142	0x08E		TMR4_1_GOVF	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[14]
143	0x08F		TMR4_1_GUDF	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[15]
144	0x090	Timer6_2	TMR6_2_GCMA	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[16]
145	0x091		TMR6_2_GCMB	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[17]
146	0x092		TMR6_2_GCMC	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[18]
147	0x093		TMR6_2_GCMD	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[19]
148	0x094		TMR6_2_GCME	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[20]
149	0x095		TMR6_2_GCMF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[21]
150	0x096		TMR6_2_GOVF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[22]
151	0x097		TMR6_2_GUDF	P	✓	✓	INTSEL0~31	INTSEL56~61	VSSEL132[23]
152	0x098	Timer4_2	TMR4_2_GCMUH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[24]
			TMR4_2_SCM0	-	-	✓			
153	0x099		TMR4_2_GCMUL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[25]
			TMR4_2_SCM1	-	-	✓			
154	0x09A		TMR4_2_GCMVH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[26]
			TMR4_2_SCM2	-	-	✓			
155	0x09B		TMR4_2_GCMVL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[27]
			TMR4_2_SCM3	-	-	✓			
156	0x09C		TMR4_2_GCMWH	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[28]
			TMR4_2_SCM4	-	-	✓			
157	0x09D		TMR4_2_GCMWL	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[29]
			TMR4_2_SCM5	-	-	✓			
158	0x09E		TMR4_2_GOVF	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[30]
159	0x09F		TMR4_2_GUDF	L	✓	-	INTSEL0~31	INTSEL56~61	VSSEL132[31]
160	0x0A0	Timer6_3	TMR6_3_GCMA	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[0]
161	0x0A1		TMR6_3_GCMB	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[1]
162	0x0A2		TMR6_3_GCMC	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[2]
163	0x0A3		TMR6_3_GCMD	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[3]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
164	0x0A4	Timer6_3	TMR6_3_GCME	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[4]
165	0x0A5		TMR6_3_GCMF	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[5]
166	0x0A6		TMR6_3_GOVF	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[6]
167	0x0A7		TMR6_3_GUDF	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[7]
168	0x0A8	Timer4_3	TMR4_3_GCMUH	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[8]
			TMR4_3_SCM0	-	-	✓			
169	0x0A9		TMR4_3_GCMUL	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[9]
			TMR4_3_SCM1	-	-	✓			
170	0x0AA		TMR4_3_GCMVH	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[10]
			TMR4_3_SCM2	-	-	✓			
171	0x0AB		TMR4_3_GCMVL	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[11]
			TMR4_3_SCM3	-	-	✓			
172	0x0AC		TMR4_3_GCMWH	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[12]
			TMR4_3_SCM4	-	-	✓			
173	0x0AD		TMR4_3_GCMWL	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[13]
			TMR4_3_SCM5	-	-	✓			
174	0x0AE		TMR4_3_GOVF	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[14]
175	0x0AF		TMR4_3_GUDF	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[15]
176	0x0B0	Timer6_1	TMR6_1_GDTE	P	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[16]
177	0x0B1	-	-	-	-	-	INTSEL0~31	INTSEL62~67	VSSEL133[17]
178	0x0B2	-	-	-	-	-	INTSEL0~31	INTSEL62~67	VSSEL133[18]
179	0x0B3	Timer6_1	TMR6_1_SCMA	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[19]
180	0x0B4		TMR6_1_SCMB	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[20]
181	0x0B5	Timer4_1	TMR4_1_RLOU	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[21]
182	0x0B6		TMR4_1_RLOV	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[22]
183	0x0B7		TMR4_1_RLOW	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[23]
184	0x0B8	Timer6_2	TMR6_2_GDTE	P	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[24]
185	0x0B9	-	-	-	-	-	INTSEL0~31	INTSEL62~67	VSSEL133[25]
186	0x0BA	-	-	-	-	-	INTSEL0~31	INTSEL62~67	VSSEL133[26]
187	0x0BB	Timer6_2	TMR6_2_SCMA	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[27]
188	0x0BC		TMR6_2_SCMB	P	✓	✓	INTSEL0~31	INTSEL62~67	VSSEL133[28]
189	0x0BD	Timer4_2	TMR4_2_RLOU	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[29]
190	0x0BE		TMR4_2_RLOV	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[30]
191	0x0BF		TMR4_2_RLOW	L	✓	-	INTSEL0~31	INTSEL62~67	VSSEL133[31]
192	0x0C0	Timer6_3	TMR6_3_GDTE	P	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[0]
193	0x0C1	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[1]
194	0x0C2	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[2]
195	0x0C3	Timer6_3	TMR6_3_SCMA	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[3]
196	0x0C4		TMR6_3_SCMB	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[4]
197	0x0C5	Timer4_3	TMR4_3_RLOU	L	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[5]
198	0x0C6		TMR4_3_RLOV	L	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[6]
199	0x0C7		TMR4_3_RLOW	L	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[7]
200	0x0C8	USART_3	USART_3_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[8]
201	0x0C9	USART_4	USART_4_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[9]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
202	0x0CA	USART_5	USART_5_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[10]
203	0x0CB	USART_8	USART_8_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[11]
204	0x0CC	USART_9	USART_9_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[12]
205	0x0CD	USART_10	USART_10_RTO	L	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[13]
206	0x0CE	PKE	PKE	L	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[14]
207	0x0CF	PKERAM	PKERAM_ECCER	L	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[15]
208	0x0D0	Timer6_4	TMR6_4_GCMA	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[16]
209	0x0D1		TMR6_4_GCMB	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[17]
210	0x0D2		TMR6_4_GCMC	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[18]
211	0x0D3		TMR6_4_GCMD	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[19]
212	0x0D4		TMR6_4_GCME	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[20]
213	0x0D5		TMR6_4_GCMF	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[21]
214	0x0D6		TMR6_4_GOVF	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[22]
215	0x0D7		TMR6_4_GUDF	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[23]
216	0x0D8		TMR6_4_GDTE	P	✓	-	INTSEL0~31	INTSEL68~73	VSSEL134[24]
217	0x0D9	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[25]
218	0x0DA	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[26]
219	0x0DB	Timer6_4	TMR6_4_SCMA	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[27]
220	0x0DC		TMR6_4_SCMB	P	✓	✓	INTSEL0~31	INTSEL68~73	VSSEL134[28]
221	0x0DD	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[29]
222	0x0DE	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[30]
223	0x0DF	-	-	-	-	-	INTSEL0~31	INTSEL68~73	VSSEL134[31]
224	0x0E0	Timer6_5	TMR6_5_GCMA	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[0]
225	0x0E1		TMR6_5_GCMB	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[1]
226	0x0E2		TMR6_5_GCMC	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[2]
227	0x0E3		TMR6_5_GCMD	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[3]
228	0x0E4		TMR6_5_GCME	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[4]
229	0x0E5		TMR6_5_GCMF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[5]
230	0x0E6		TMR6_5_GOVF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[6]
231	0x0E7		TMR6_5_GUDF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[7]
232	0x0E8		TMR6_5_GDTE	P	✓	-	INTSEL0~31	INTSEL74~79	VSSEL135[8]
233	0x0E9	-	-	-	-	-	INTSEL0~31	INTSEL74~79	VSSEL135[9]
234	0x0EA	-	-	-	-	-	INTSEL0~31	INTSEL74~79	VSSEL135[10]
235	0x0EB	Timer6_5	TMR6_5_SCMA	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[11]
236	0x0EC		TMR6_5_SCMB	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[12]
237	0x0ED	TimerA_1	TMRA_1_OVF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[13]
238	0x0EE		TMRA_1_UDF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[14]
239	0x0EF		TMRA_1_CMP	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[15]
240	0x0F0	Timer6_6	TMR6_6_GCMA	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[16]
241	0x0F1		TMR6_6_GCMB	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[17]
242	0x0F2		TMR6_6_GCMC	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[18]
243	0x0F3		TMR6_6_GCMD	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[19]
244	0x0F4		TMR6_6_GCME	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[20]
245	0x0F5		TMR6_6_GCMF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[21]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
246	0x0F6	Timer6_6	TMR6_6_GOVF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[22]
247	0x0F7		TMR6_6_GUDF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[23]
248	0x0F8		TMR6_6_GDTE	P	✓	-	INTSEL0~31	INTSEL74~79	VSSEL135[24]
249	0x0F9	-	-	-	-	-	INTSEL0~31	INTSEL74~79	VSSEL135[25]
250	0x0FA	-	-	-	-	-	INTSEL0~31	INTSEL74~79	VSSEL135[26]
251	0x0FB	Timer6_6	TMR6_6_SCMA	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[27]
252	0x0FC		TMR6_6_SCMB	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[28]
253	0x0FD	TimerA_2	TMRA_2_OVF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[29]
254	0x0FE		TMRA_2_UDF	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[30]
255	0x0FF		TMRA_2_CMP	P	✓	✓	INTSEL0~31	INTSEL74~79	VSSEL135[31]
256	0x100	Timer6_7	TMR6_7_GCMA	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[0]
257	0x101		TMR6_7_GCMB	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[1]
258	0x102		TMR6_7_GCMC	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[2]
259	0x103		TMR6_7_GCMD	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[3]
260	0x104		TMR6_7_GCME	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[4]
261	0x105		TMR6_7_GCMF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[5]
262	0x106		TMR6_7_GOVF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[6]
263	0x107		TMR6_7_GUDF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[7]
264	0x108		TMR6_7_GDTE	P	✓	-	INTSEL0~31	INTSEL80~85	VSSEL136[8]
265	0x109	-	-	-	-	-	INTSEL0~31	INTSEL80~85	VSSEL136[9]
266	0x10A	-	-	-	-	-	INTSEL0~31	INTSEL80~85	VSSEL136[10]
267	0x10B	Timer6_7	TMR6_7_SCMA	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[11]
268	0x10C		TMR6_7_SCMB	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[12]
269	0x10D	TimerA_3	TMRA_3_OVF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[13]
270	0x10E		TMRA_3_UDF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[14]
271	0x10F		TMRA_3_CMP	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[15]
272	0x110	Timer6_8	TMR6_8_GCMA	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[16]
273	0x111		TMR6_8_GCMB	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[17]
274	0x112		TMR6_8_GCMC	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[18]
275	0x113		TMR6_8_GCMD	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[19]
276	0x114		TMR6_8_GCME	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[20]
277	0x115		TMR6_8_GCMF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[21]
278	0x116		TMR6_8_GOVF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[22]
279	0x117		TMR6_8_GUDF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[23]
280	0x118		TMR6_8_GDTE	P	✓	-	INTSEL0~31	INTSEL80~85	VSSEL136[24]
281	0x119	-	-	-	-	-	INTSEL0~31	INTSEL80~85	VSSEL136[25]
282	0x11A	-	-	-	-	-	INTSEL0~31	INTSEL80~85	VSSEL136[26]
283	0x11B	Timer6_8	TMR6_8_SCMA	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[27]
284	0x11C		TMR6_8_SCMB	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[28]
285	0x11D	TimerA_4	TMRA_4_OVF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[29]
286	0x11E		TMRA_4_UDF	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[30]
287	0x11F		TMRA_4_CMP	P	✓	✓	INTSEL0~31	INTSEL80~85	VSSEL136[31]
288	0x120	EMB	EMB_GR0	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[0]
289	0x121		EMB_GR1	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[1]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
290	0x122	EMB	EMB_GR2	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[2]
291	0x123		EMB_GR3	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[3]
292	0x124		EMB_GR4	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[4]
293	0x125		EMB_GR5	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[5]
294	0x126		EMB_GR6	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[6]
295	0x127	USBHS	USBHS_EP1_OUT	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[7]
296	0x128		USBHS_EP1_IN	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[8]
297	0x129		USBHS_GLB	L	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[9]
298	0x12A		USBHS_WKUP	P	✓	-	INTSEL0~31	INTSEL86~91	VSSEL137[10]
299	0x12B	AOS_STRG ⁽³⁾	AOS_STRG	-	-	✓	INTSEL0~31	INTSEL86~91	VSSEL137[11]
300	0x12C	USART_1	USART_1_EI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[12]
301	0x12D		USART_1_RI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[13]
302	0x12E		USART_1_TI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[14]
303	0x12F		USART_1_TCI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[15]
304	0x130		USART_1_RTO	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[16]
305	0x131	USART_2	USART_2_EI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[17]
306	0x132		USART_2_RI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[18]
307	0x133		USART_2_TI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[19]
308	0x134		USART_2_TCI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[20]
309	0x135		USART_2_RTO	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[21]
310	0x136	SPI_1	SPI_1_SPRI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[22]
311	0x137		SPI_1_SPTI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[23]
312	0x138		SPI_1_SPII	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[24]
313	0x139		SPI_1_SPEI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[25]
314	0x13A		SPI_1_SPEND	-	-	✓	INTSEL0~31	INTSEL86~91	VSSEL137[26]
315	0x13B	SPI_2	SPI_2_SPRI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[27]
316	0x13C		SPI_2_SPTI	P	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[28]
317	0x13D		SPI_2_SPII	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[29]
318	0x13E		SPI_2_SPEI	L	✓	✓	INTSEL0~31	INTSEL86~91	VSSEL137[30]
319	0x13F		SPI_2_SPEND	-	-	✓	INTSEL0~31	INTSEL86~91	VSSEL137[31]
320	0x140	TimerA_5	TMRA_5_OVF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[0]
321	0x141		TMRA_5_UDF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[1]
322	0x142		TMRA_5_CMP	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[2]
323	0x143	TimerA_6	TMRA_6_OVF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[3]
324	0x144		TMRA_6_UDF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[4]
325	0x145		TMRA_6_CMP	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[5]
326	0x146	TimerA_7	TMRA_7_OVF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[6]
327	0x147		TMRA_7_UDF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[7]
328	0x148		TMRA_7_CMP	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[8]
329	0x149	TimerA_8	TMRA_8_OVF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[9]
330	0x14A		TMRA_8_UDF	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[10]
331	0x14B		TMRA_8_CMP	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[11]
332	0x14C	USART_3	USART_3_EI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[12]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
333	0x14D	USART_3	USART_3_RI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[13]
334	0x14E		USART_3_TI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[14]
335	0x14F		USART_3_TCI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[15]
336	0x150	USART_4	USART_4_EI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[16]
337	0x151		USART_4_RI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[17]
338	0x152		USART_4_TI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[18]
339	0x153		USART_4_TCI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[19]
340	0x154	CAN_1	CAN_1_HOST	L	✓	-	INTSEL0~31	INTSEL92~97	VSSEL138[20]
341	0x155	CAN_2	CAN_2_HOST	L	✓	-	INTSEL0~31	INTSEL92~97	VSSEL138[21]
342	0x156	SPI_3	SPI_3_SPRI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[22]
343	0x157		SPI_3_SPTI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[23]
344	0x158		SPI_3_SPII	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[24]
345	0x159		SPI_3_SPEI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[25]
346	0x15A		SPI_3_SPEND	-	-	✓	INTSEL0~31	INTSEL92~97	VSSEL138[26]
347	0x15B	SPI_4	SPI_4_SPRI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[27]
348	0x15C		SPI_4_SPTI	P	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[28]
349	0x15D		SPI_4_SPII	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[29]
350	0x15E		SPI_4_SPEI	L	✓	✓	INTSEL0~31	INTSEL92~97	VSSEL138[30]
351	0x15F		SPI_4_SPEND	-	-	✓	INTSEL0~31	INTSEL92~97	VSSEL138[31]
352	0x160	TimerA_9	TMRA_9_OVF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[0]
353	0x161		TMRA_9_UDF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[1]
354	0x162		TMRA_9_CMP	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[2]
355	0x163	TimerA_10	TMRA_10_OVF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[3]
356	0x164		TMRA_10_UDF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[4]
357	0x165		TMRA_10_CMP	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[5]
358	0x166	TimerA_11	TMRA_11_OVF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[6]
359	0x167		TMRA_11_UDF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[7]
360	0x168		TMRA_11_CMP	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[8]
361	0x169	TimerA_12	TMRA_12_OVF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[9]
362	0x16A		TMRA_12_UDF	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[10]
363	0x16B		TMRA_12_CMP	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[11]
364	0x16C	USART_5	USART_5_BRKWKPI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[12]
365	0x16D		USART_5_EI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[13]
366	0x16E		USART_5_RI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[14]
367	0x16F		USART_5_TI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[15]
368	0x170		USART_5_TCI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[16]
369	0x171	USART_6	USART_6_EI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[17]
370	0x172		USART_6_RI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[18]
371	0x173		USART_6_TI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[19]
372	0x174		USART_6_TCI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[20]
373	0x175		USART_6_RTO	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[21]
374	0x176	SPI_5	SPI_5_SPRI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[22]
375	0x177		SPI_5_SPTI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[23]
376	0x178		SPI_5_SPII	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[24]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
377	0x179	SPI_5	SPI_5_SPEI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[25]
378	0x17A		SPI_5_SPEND	-	-	✓	INTSEL0~31	INTSEL98~103	VSSEL139[26]
379	0x17B	SPI_6	SPI_6_SPRI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[27]
380	0x17C		SPI_6_SPTI	P	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[28]
381	0x17D		SPI_6_SPII	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[29]
382	0x17E		SPI_6_SPEI	L	✓	✓	INTSEL0~31	INTSEL98~103	VSSEL139[30]
383	0x17F		SPI_6_SPEND	-	-	✓	INTSEL0~31	INTSEL98~103	VSSEL139[31]
384	0x180	I2S_1	I2S_1_TXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[0]
385	0x181		I2S_1_RXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[1]
386	0x182		I2S_1_ERRIRQOUT	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[2]
387	0x183	I2S_2	I2S_2_TXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[3]
388	0x184		I2S_2_RXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[4]
389	0x185		I2S_2_ERRIRQOUT	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[5]
390	0x186	USART_7	USART_7_EI	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[6]
391	0x187		USART_7_RI	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[7]
392	0x188		USART_7_TI	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[8]
393	0x189		USART_7_TCI	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[9]
394	0x18A		USART_7_RTO	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[10]
395	0x18B	USART_8	USART_8_EI	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[11]
396	0x18C		USART_8_RI	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[12]
397	0x18D		USART_8_TI	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[13]
398	0x18E		USART_8_TCI	L	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[14]
399	0x18F	USBFS	USBFS_GLB	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[15]
400	0x190		USBFS_WKUP	P	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[16]
401	0x191	HASH	HASH_INT	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[17]
402	0x192	SDIOC_1	SDIOC_1_DMAR	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[18]
403	0x193		SDIOC_1_DMAW	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[19]
404	0x194		SDIOC_1_SD	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[20]
405	0x195	SDIOC_2	SDIOC_2_DMAR	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[21]
406	0x196		SDIOC_2_DMAW	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[22]
407	0x197		SDIOC_2_SD	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[23]
408	0x198	EVENT port	EVENT_PORT1	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[24]
409	0x199		EVENT_PORT2	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[25]
410	0x19A		EVENT_PORT3	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[26]
411	0x19B		EVENT_PORT4	P	✓	✓	INTSEL0~31	INTSEL104~109	VSSEL140[27]
412	0x19C	ETHMAC	ETH_GLB_INT	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[28]
413	0x19D		ETH_WKP_INT	L	✓	-	INTSEL0~31	INTSEL104~109	VSSEL140[29]
414	0x19E		ETH_PPS_OUT_0	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[30]
415	0x19F		ETH_PPS_OUT_1	-	-	✓	INTSEL0~31	INTSEL104~109	VSSEL140[31]
416	0x1A0	I2S_3	I2S_3_TXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[0]
417	0x1A1		I2S_3_RXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[1]
418	0x1A2		I2S_3_ERRIRQOUT	L	✓	-	INTSEL0~31	INTSEL110~115	VSSEL141[2]
419	0x1A3	I2S_4	I2S_4_TXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[3]
420	0x1A4		I2S_4_RXIRQOUT	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[4]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
421	0x1A5	I2S_4	I2S_4_ERRIRQOUT	L	✓	-	INTSEL0~31	INTSEL110~115	VSSEL141[5]
422	0x1A6	USART_9	USART_9_EI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[6]
423	0x1A7		USART_9_RI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[7]
424	0x1A8		USART_9_TI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[8]
425	0x1A9		USART_9_TCI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[9]
426	0x1AA	USART_10	USART_10_BRKWKPI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[10]
427	0x1AB		USART_10_EI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[11]
428	0x1AC		USART_10_RI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[12]
429	0x1AD		USART_10_TI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[13]
430	0x1AE		USART_10_TCI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[14]
431	0x1AF	SKE	SKE	L	✓	-	INTSEL0~31	INTSEL110~115	VSSEL141[15]
432	0x1B0	I2C_1	I2C_1_RXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[16]
433	0x1B1		I2C_1_TXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[17]
434	0x1B2		I2C_1_TEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[18]
435	0x1B3		I2C_1_EEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[19]
436	0x1B4	I2C_2	I2C_2_RXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[20]
437	0x1B5		I2C_2_TXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[21]
438	0x1B6		I2C_2_TEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[22]
439	0x1B7		I2C_2_EEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[23]
440	0x1B8	I2C_3	I2C_3_RXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[24]
441	0x1B9		I2C_3_TXI	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[25]
442	0x1BA		I2C_3_TEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[26]
443	0x1BB		I2C_3_EEI	L	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[27]
444	0x1BC	CMP	CMP1	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[28]
445	0x1BD		CMP2	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[29]
446	0x1BE		CMP3	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[30]
447	0x1BF		CMP4	P	✓	✓	INTSEL0~31	INTSEL110~115	VSSEL141[31]
448	0x1C0	I2C_4	I2C_4_RXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[0]
449	0x1C1		I2C_4_TXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[1]
450	0x1C2		I2C_4_TEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[2]
451	0x1C3		I2C_4_EEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[3]
452	0x1C4	I2C_5	I2C_5_RXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[4]
453	0x1C5		I2C_5_TXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[5]
454	0x1C6		I2C_5_TEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[6]
455	0x1C7		I2C_5_EEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[7]
456	0x1C8	I2C_6	I2C_6_RXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[8]
457	0x1C9		I2C_6_TXI	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[9]
458	0x1CA		I2C_6_TEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[10]
459	0x1CB		I2C_6_EEI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[11]
460	0x1CC	USART_1	USART_1_WUPI	边沿	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[12]
461	0x1CD	PVD	PVD_PVD1	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[13]
462	0x1CE		PVD_PVD2	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[14]
463	0x1CF	OTS	OTS	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[15]
464	0x1D0	FCM	FCMFERRI	L	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[16]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
465	0x1D1	FCM	FCMMENDI	L	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[17]
466	0x1D2		FCMCOVFI	L	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[18]
467	0x1D3	WDT	WDT_REFUDF	P	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[19]
468	0x1D4	CTC	CTC_ERR	P	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[20]
469	0x1D5	ERMU	ERMU_LPI	P	✓	-	INTSEL0~31	INTSEL116~121	VSSEL142[21]
470	0x1D6	USART_1	USART_1_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[22]
471	0x1D7	USART_2	USART_2_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[23]
472	0x1D8	USART_3	USART_3_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[24]
473	0x1D9	USART_4	USART_4_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[25]
474	0x1DA	USART_5	USART_5_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[26]
475	0x1DB	USART_6	USART_6_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[27]
476	0x1DC	USART_7	USART_7_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[28]
477	0x1DD	USART_8	USART_8_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[29]
478	0x1DE	USART_9	USART_9_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[30]
479	0x1DF	USART_10	USART_10_TENDI	L	✓	✓	INTSEL0~31	INTSEL116~121	VSSEL142[31]
480	0x1E0	ADC_1	ADC_1_EOCA	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[0]
481	0x1E1		ADC_1_EOCB	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[1]
482	0x1E2		ADC_1_CMP0	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[2]
483	0x1E3		ADC_1_CMP1	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[3]
484	0x1E4	ADC_2	ADC_2_EOCA	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[4]
485	0x1E5		ADC_2_EOCB	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[5]
486	0x1E6		ADC_2_CMP0	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[6]
487	0x1E7		ADC_2_CMP1	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[7]
488	0x1E8	ADC_3	ADC_3_EOCA	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[8]
489	0x1E9		ADC_3_EOCB	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[9]
490	0x1EA		ADC_3_CMP0	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[10]
491	0x1EB		ADC_3_CMP1	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[11]
492	0x1EC	TRNG	TRNG_END	P	✓	✓	INTSEL0~31	INTSEL122~127	VSSEL143[12]
493	0x1ED	CANRAM	CAN1RAM_ECCER	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[13]
494	0x1EE		CAN2RAM_ECCER	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[14]
495	0x1EF	MCANRAM	MCANRAM_ECCER	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[15]
496	0x1F0	NFC	NFC_INT	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[16]
497	0x1F1	MCAN_1	MCAN_1_INT0	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[17]
498	0x1F2		MCAN_1_INT1	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[18]
499	0x1F3	MCAN_2	MCAN_2_INT0	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[19]
500	0x1F4		MCAN_2_INT1	L	✓	-	INTSEL0~31	INTSEL122~127	VSSEL143[20]
501	0x1F5	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[21]
502	0x1F6	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[22]
503	0x1F7	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[23]
504	0x1F8	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[24]
505	0x1F9	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[25]
506	0x1FA	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[26]
507	0x1FB	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[27]
508	0x1FC	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[28]

序号	中断事件编号	模块	中断名称	中断类型 ⁽¹⁾	中断源	触发源	NVIC 中断请求 (IRQ) 编号映射 ⁽²⁾		
							NVIC 向量 0~31	NVIC 向量 32~127	NVIC 向量 128~143
509	0x1FD	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[29]
510	0x1FE	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[30]
511	0x1FF	-	-	-	-	-	INTSEL0~31	INTSEL122~127	VSSEL143[31]

 **说明**

1. “P”表示脉冲中断，“L”表示电平中断。电平中断必须在中断处理程序中清除中断标志位。

2. INTSEL 表示 INTC_INTSEL 寄存器，VSSEL 表示 INTC_VSSEL 寄存器。中断选择寄存器选择的编号如果未配置中断事件，则该寄存器设定无效。

3. AOS_STRG 由软件设定寄存器 INTSFTTRG 的 STRG 位产生。

11.3 功能说明

11.3.1 功能框图

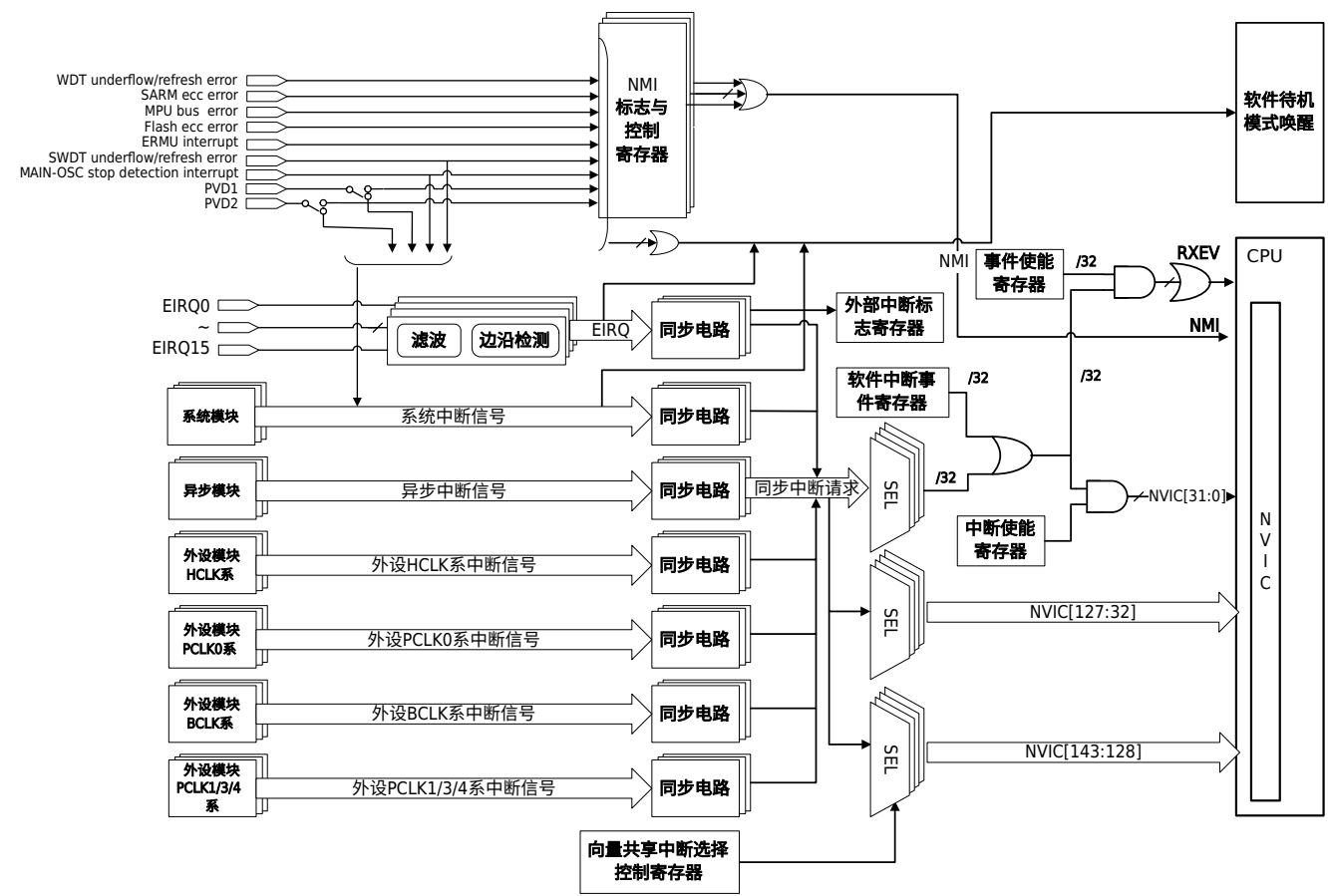


图 11-1 中断系统功能框图

11.3.2 不可屏蔽中断

不可屏蔽中断事件如下：

- ERMU 中断
- Flash ECC 校验错误中断
- XTAL 振荡故障中断
- WDT 下溢/刷新中断

- SWDT 下溢/刷新中断
- 低电压检测 PVD1 中断
- 低电压检测 PVD2 中断
- SRAM ECC 校验错误中断
- MPU 总线错误中断

不可屏蔽中断具有最高的优先级并且可以选择多个中断事件。各中断事件的状态可以通过中断标志寄存器 (INTC_NMIFR) 查询。请在不可屏蔽中断处理退出前确认所有的标志位都为“0”。

需要使用不可屏蔽中断时，请按照如下流程设定：

1. 为不可屏蔽中断事件配置相应的功能。
2. 对 INTC_NMIFCR 各位写 1，清除中断标志位，防止误动作。
3. 设定中断使能寄存器 INTC_NMIER，使能相应的中断事件。



注意

一旦 INTC_NMIER 的相应位被设定为 1，将不可更改，除非复位。

11.3.3 外部中断

配备了 16 个外部中断事件，中断状态可以通过中断标志寄存器 (INTC_EIFR) 查询。需要使用外部中断时，请按照如下流程设定：

1. 为相应的外部管脚设定外部中断功能。
2. 对 INTC_EIFCR 各位写 1，清除中断标志位，防止误动作。
3. 清除 INTC_EIRQCRx.EFEN 位 (x=0~15)，禁止数字滤波器。
4. 设定 INTC_EIRQCRx.EIRQTRG[1:0]位，选择触发边沿或电平；设定 EISMPCLK[1:0]位，选择数字滤波器采样时钟；设定 EFEN 位，使能数字滤波器。

11.3.4 中断事件多路器

中断控制器共使用了 144 个中断向量，提供了 3 种中断事件请求选择方式，通过灵活的组合来满足各种中断配置需求。具体的中断事件编号请参见“[中断事件表](#)”。

● 第一种方式：

共 32 个中断向量，对应 NVIC 的中断向量为 0~31。通过中断/事件选择寄存器 INTC_INTSEL0~31 进行选择，可以在所有中断事件请求中任选 1 个作为中断源。0~31 中断向量需要通过 INTC_IER 寄存器使能。

● 第二种方式：

共 96 个中断向量，对应的中断向量为 32~127，通过中断选择寄存器 INTC_INTSEL32~127 进行选择。96 个中断向量以 6 个向量为 1 组分成 16 组，同时所有中断事件请求以 32 个请求为 1 组也分成 16 组。每组中的 6 个中断向量通过中断选择寄存器，选择对应的中断事件请求组中 32 个请求里的任一请求。

● 第三种方式：

共 16 个中断向量，对应 NVIC 的中断向量为 128~143。通过中断使能寄存器 INTC_VSSEL128~143 进行中断事件请求使能，每 32 个外设中断事件请求可以共享 1 个中断向量。通过选择有外设标志位的中断事件请求，各外设都可以申请中断。具体参考[中断事件表](#)中 NVIC 向量 128~143 列。

11.3.5 软件中断

软件中断功能可以通过直接写软件中断寄存器 INTC_SWIER 发生一次中断请求。一共配备了 32 个软件中断，对应的 NVIC 中断请求编号为 0~31，由中断使能寄存器 INTC_IER 控制许可。

11.3.6 事件输入

编号为 0~31 的 NVIC 中断请求也可被用作事件输入 (RXEV) 来唤醒内核 (WFE)，通过事件使能寄存器 INTC_EVTER 控制许可。

中断请求和事件输入的功能框图如下：

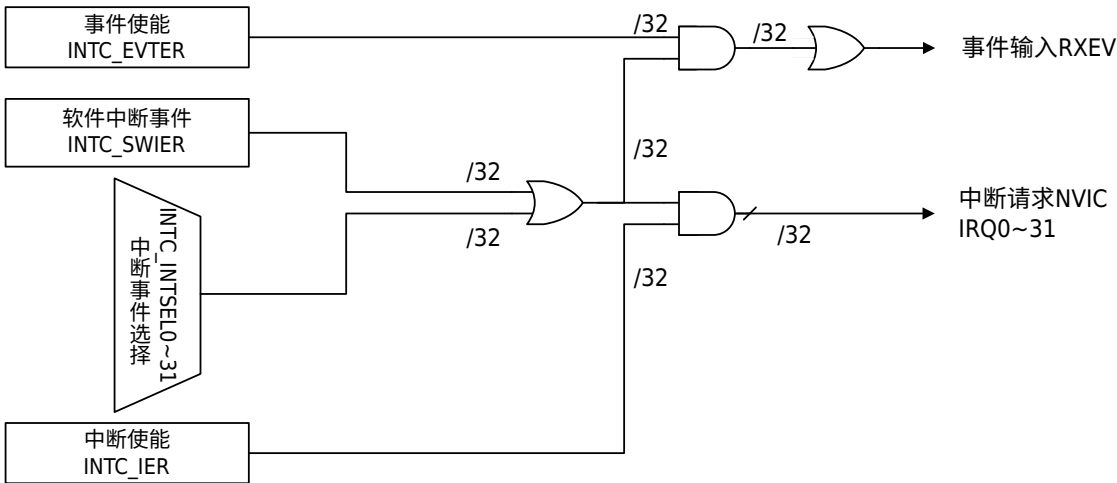


图 11-2 中断请求/事件输入选择

11.3.7 WFE 事件管理

中断控制器可以通过中断请求、事件输入两种方式唤醒内核（WFE）。

- 中断请求唤醒内核
选择一个中断事件，根据选择的中断请求设定 INTC_INTSEL 并使能 INTC_IER 寄存器（但不在 NVIC 中使能该中断请求）。INTC_EVTER 寄存器设定为非使能，同时使能 Cortex-M4 内核控制寄存器 SCR 中的 SEVONPEND 位。执行 WFE 使内核进入休眠模式。当选择的中断事件发生时，内核由休眠模式被唤醒，但不进入中断处理程序。
- 事件输入唤醒内核
选择一个中断事件，根据选择的中断请求设定 INTC_INTSEL 并使能 INTC_EVTER 寄存器。INTC_IER 寄存器设定为非使能。执行 WFE 使内核进入休眠模式。当选择的中断事件发生时，内核由休眠模式被唤醒。

11.3.8 噪声滤波器

为了消除输入噪声，外部中断 EIRQx（x=0~15）配置了数字和模拟两种滤波器。

数字滤波器通过 INTC_EIRQCRx 寄存器的 EFEN 位使能。使用时以 PCLK3 作为采样时钟对 EIRQ 输入信号进行采样，滤波宽度由 EIRQCR 寄存器的 EISMPCLK[1:0]位和 NFSEL[7:0]位的设定值共同决定。当输入电平小于采样周期×（NFSEL[7:0]设定值+1）时，输入电平被滤除。

例：当 NFSEL[7:0]=0x02 时，宽度小于 3 个采样周期的信号将被滤除。采样周期通过 INTC_EIRQCRx 寄存器的 EISMPCLK[1:0]位设定。

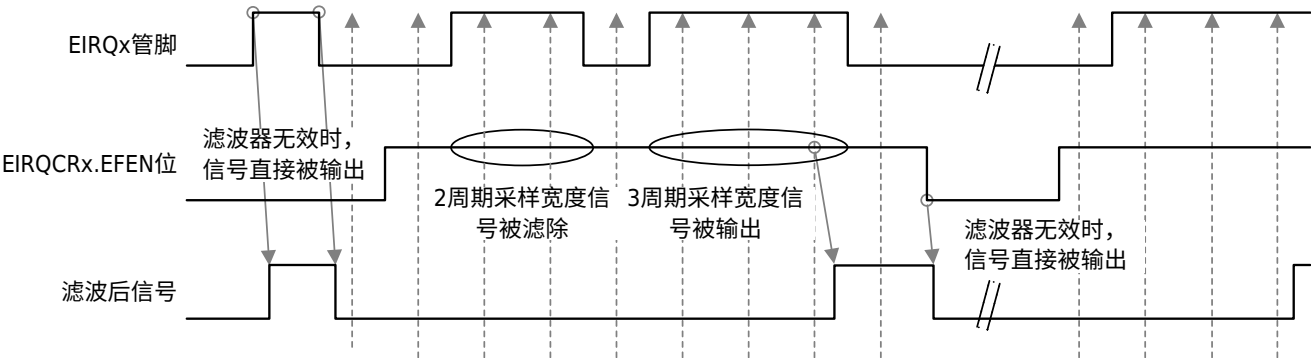


图 11-3 数字滤波器工作示意图（以 NFSEL[7:0]=0x02 为例）

模拟滤波器通过 INTC_EIRQCRx 寄存器的 NOCEN 位使能。使能后以 INTC_NOCCR 寄存器的 NOCSEL 位选择的滤波宽度对信号进行滤波。

两个滤波器彼此独立，可单独使能或者关闭。同时打开时，信号是先经过模拟滤波再经过数字滤波后进入到内部电路的。

在进入停止模式前，必须关闭数字滤波器。从停止模式返回后可再使能数字滤波器。模拟滤波器可以在停止模式时工作。

11.3.9 低功耗模式返回

休眠模式返回

通过中断请求唤醒睡眠模式时，需要进行如下设定：

- 选择一个中断事件作为中断请求
如果使用 0~31 号中断请求需要设定 INTC_IER 寄存器。
- 在 NVIC 中使能该中断请求
- 使用不可屏蔽中断时需要设定 INTC_NMIER 寄存器

停止模式返回

通过中断请求唤醒停止模式时，需要进行如下设定：

- 选择一个中断事件作为中断请求
如果使用 0~31 号中断请求需要设定 INTC_IER 寄存器。
- 通过 INTC_WKEN 寄存器使能该中断事件
如果使用不可屏蔽中断则通过 INTC_NMIER 寄存器使能中断事件，可以唤醒停止模式的不可屏蔽中断有：XTAL 振荡故障中断、SWDT 下溢/刷新中断。
- 在 NVIC 中使能该中断请求

掉电模式返回

掉电模式的返回可由电源控制 (PWC) 章节所表示的条件、NRST 管脚复位、上电复位以及欠压复位条件返回。返回后 CPU 进入复位中断处理。详细说明请参见《电源控制 (PWC)》章节相关描述。

不可屏蔽中断和 WFI 指令

执行 WFI 指令前，请确认不可屏蔽中断标志寄存器 INTC_NMIFR 的所有状态位为 0。

11.4 寄存器

11.4.1 寄存器总表

基地址：0x40051000

表 11-4 中断控制器 (INTC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	INTC_NOCCR	外部管脚中断滤波控制寄存器	32
0x0004	INTC_NMIER	不可屏蔽中断使能寄存器	32
0x0008	INTC_NMIFR	不可屏蔽中断标志寄存器	32
0x000C	INTC_NMIFCR	不可屏蔽中断标志清除寄存器	32
0x0010+0x4*x	INTC_EIRQCRx	外部中断控制寄存器 x (x=0~15)	32
0x0050	INTC_WKEN	停止模式唤醒事件使能寄存器	32
0x0054	INTC_EIFR	外部中断标志寄存器	32
0x0058	INTC_EIFCR	外部中断标志清除寄存器	32
0x005C+0x4*x	INTC_INTSELx	中断事件选择寄存器 x (x=0~31)	32
0x00DC+0x4*(y-32)	INTC_INTSELy	中断选择寄存器 y (y=32~127)	32
0x025C+0x4*(z-128)	INTC_VSSELz	向量共享中断选择寄存器 z (z=128~143)	32
0x029C	INTC_SWIER	软件中断事件寄存器	32
0x02A0	INTC_EVTER	事件使能寄存器	32
0x02A4	INTC_IER	中断使能寄存器	32
0x02A8	INTC_FPUIER	FPU 中断使能寄存器	32

11.4.2 外部管脚中断滤波控制寄存器 (INTC_NOCCR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																			NOCSEL												
Access																				RW												

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13:12	NOCSEL	模拟滤波滤波宽度选择	● 0b00: 滤波宽度档位 1 ● 0b01: 滤波宽度档位 2 ● 0b10: 滤波宽度档位 3 ● 0b11: 滤波宽度档位 4 各档位具体值请参见《数据手册》的“电气特性-EIRQ 滤波特性”章节相关描述。	RW
11:0	Reserved	保留	-	-

11.4.3 不可屏蔽中断使能寄存器 (INTC_NMIER)

Offset	Bit Position																																											
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
Reset	0x00000000																																											
Name	Reserved																			ERMUEN		FECERREN		WDTEN		BUSERREN		RECCERREN		Reserved		XATLSTPEN		Reserved		PVD2EN		PVD1EN		SWDTEN		Reserved		
Access																				RW	RW	RW	RW	RW			RW		RW				RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	ERMUEN	ERMU 中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
12	FECERREN	Flash ECC 校验错误中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
11	WDTEN	WDT 下溢/刷新中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
10	BUSERREN	MPU 主总线错误中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
9	RECCERREN	SRAM ECC 校验错误中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
8:6	Reserved	保留	-	-
5	XATLSTPEN	XTAL 振荡故障中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
4	Reserved	保留	-	-
3	PVD2EN	低电压检测 PVD2 中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
2	PVD1EN	低电压检测 PVD1 中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
1	SWDTEN	SWDT 下溢/刷新中断选择	0b0: 禁止中断作为不可屏蔽中断源 0b1: 选择中断作为不可屏蔽中断源	RW
0	Reserved	保留	-	-

11.4.4 不可屏蔽中断标志寄存器 (INTC_NMIFR)

Offset	Bit Position																																				
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved																			ERMUF	FECCERRF	WDTF	BUSERRF	RECCERRF	Reserved				XTALSTPF	Reserved	PVD2F	PVD1F	SWDTF	Reserved			
Access																				RO	RO	RO	RO	RO					RO		RO	RO	RO				

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	ERMUF	ERUM 中断标志	0b0: 没有发生 ERUM 中断申请 0b1: 发生 ERUM 中断申请	RO
12	FECCERRF	Flash ECC 校验错误中断标志	0b0: 没有发生 Flash ECC 校验错误申请 0b1: 发生 Flash ECC 校验错误申请	RO
11	WDTF	WDT 下溢/刷新中断标志	0b0: 没有发生 WDT 下溢/刷新申请 0b1: 发生 WDT 下溢/刷新申请	RO
10	BUSERRF	MPU 主总线错误中断标志	0b0: 没有发生 MPU 主总线错误申请 0b1: 发生 MPU 主总线错误申请	RO
9	RECCERRF	SRAM ECC 校验错误中断标志	0b0: 没有发生 SRAM ECC 校验错误申请 0b1: 发生 SRAM ECC 校验错误申请	RO
8:6	Reserved	保留	-	-
5	XTALSTPF	XTAL 振荡故障中断标志	0b0: 没有发生 XTAL 振荡故障申请 0b1: 发生 XTAL 振荡故障申请	RO
4	Reserved	保留	-	-
3	PVD2F	低电压检测 PVD2 中断标志	0b0: 没有发生低电压检测 PVD2 申请 0b1: 发生低电压检测 PVD2 申请	RO
2	PVD1F	低电压检测 PVD1 中断标志	0b0: 没有发生低电压检测 PVD1 申请 0b1: 发生低电压检测 PVD1 申请	RO
1	SWDTF	SWDT 下溢/刷新中断标志	0b0: 没有发生 SWDT 下溢/刷新申请 0b1: 发生 SWDT 下溢/刷新申请	RO
0	Reserved	保留	-	-

11.4.5 不可屏蔽中断标志清除寄存器 (INTC_NMIFCR)

Offset	Bit Position																																
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																		ERMUFCLR	FECCERRFCLR	WDTFCLR	BUSERRFCLR	RECCERRFCLR	Reserved			XATLSTPFCLR	Reserved		PVD2FCLR	PVD1FCLR	SWDTFCLR	Reserved
Access																			WO	WO	WO	WO	WO				WO			WO	WO		

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	ERMUFCLR	ERMU 中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
12	FECCERRFCLR	Flash ECC 校验错误中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
11	WDTFCLR	WDT 下溢/刷新中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
10	BUSERRFCLR	MPU 主总线错误中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
9	RECCERRFCLR	SRAM ECC 校验错误中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
8:6	Reserved	保留	-	-
5	XATLSTPFCLR	XTAL 振荡故障中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
4	Reserved	保留	-	-
3	PVD2FCLR	低电压检测 PVD2 中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
2	PVD1FCLR	低电压检测 PVD1 中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
1	SWDTFCLR	SWDT 下溢/刷新中断标志清除	写 0 无任何效果, 写 1 清除标志; 读出恒为 0	WO
0	Reserved	保留	-	-

11.4.6 外部中断控制寄存器 (INTC_EIRQCRx) (x=0~15)

偏移地址 (Offset) : 0x0010+0x4*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00020000																															
Name	Reserved								NFSEL					NOCEN	Reserved								EFEN	Reserved	EISMPCLK			Reserved	EIRQTRG			
Access									RW					RW									RW		RW				RW			

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	NFSEL	EIRQ 数字滤波宽度选择	设定范围为 0x00~0xFF 滤波宽度=采样时钟周期 × (NFSEL 设定值+1)	RW
15	NOCEN	EIRQ 模拟滤波器使能	0b0: 禁止模拟滤波器功能 0b1: 允许模拟滤波器功能 滤波宽度由 INTC_NOCCR.NOCSEL 决定。	RW
14:8	Reserved	保留	-	-
7	EFEN	EIRQ 数字滤波器使能	0b0: 禁止数字滤波器功能 0b1: 允许数字滤波器功能	RW
6	Reserved	保留	-	-
5:4	EISMPCLK	数字滤波采样周期选择	0b00: 1×PCLK3 周期 0b01: 8×PCLK3 周期 0b10: 32×PCLK3 周期 0b11: 64×PCLK3 周期	RW
3:2	Reserved	保留	-	-
1:0	EIRQTRG	触发选择	0b00: 下降沿 0b01: 上升沿 0b10: 双边沿 0b11: 低电平	RW

11.4.7 外部中断标志寄存器 (INTC_EIFR)

Offset	Bit Position																															
0x0054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RO EIF															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	EIF	EIRQ 标志位	对应 EIF15~EIF0： ● 0b0：EIRQ 事件未发生，或写 EIFCR 位清除 ● 0b1：选择的 EIRQ 事件发生	RO

11.4.8 外部中断标志清除寄存器 (INTC{EIFCR)

Offset	Bit Position																															
0x0058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																EIFCLR															
Access																	WO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	EIFCLR	EIRQ 标志清除位	对应 EIFCLR15~EIFCLR0，写 0 无任何效果，写 1 清除 EIFR 寄存器对应位；读出恒为 0	WO

11.4.9 中断事件选择寄存器 (INTC_INTSELx) (x=0~31)

偏移地址 (Offset) : 0x005C+0x4*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																								INTSEL							
Access																																

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	INTSEL	中断事件请求选择	选择 0x000~0x1FE 对应的中断事件请求，详见 中断事件表	RW

11.4.10 中断选择寄存器 (INTC_INTSELy) (y=32~127)

偏移地址 (Offset) : 0x00DC+0x4*(y-32)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																								INTSEL							
Access																																

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	INTSEL	中断事件请求选择	分别选择中断事件表中序号对应的事件，具体对应关系如下： ● INTC_INTSEL32~37：选择 0x000~0x01F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL38~43：选择 0x020~0x03F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL44~49：选择 0x040~0x05F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL50~55：选择 0x060~0x07F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL56~61：选择 0x080~0x09F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL62~67：选择 0x0A0~0x0BF 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL68~73：选择 0x0C0~0x0DF 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL74~79：选择 0x0E0~0x0FF 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL80~85：选择 0x100~0x11F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL86~91：选择 0x120~0x13F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL92~97：选择 0x140~0x15F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL98~103：选择 0x160~0x17F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL104~109：选择 0x180~0x19F 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL110~115：选择 0x1A0~0x1BF 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL116~121：选择 0x1C0~0x1DF 对应的中断事件请求，其他选择无效。 ● INTC_INTSEL122~127：选择 0x1E0~0x1FF 对应的中断事件请求，其他选择无效。	RW

11.4.11 向量共享中断选择寄存器 (INTC_VSSELz) (z=128~143)

偏移地址 (Offset) : 0x025C+0x4*(z-128)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	VSEL31	VSEL30	VSEL29	VSEL28	VSEL27	VSEL26	VSEL25	VSEL24	VSEL23	VSEL22	VSEL21	VSEL20	VSEL19	VSEL18	VSEL17	VSEL16	VSEL15	VSEL14	VSEL13	VSEL12	VSEL11	VSEL10	VSEL9	VSEL8	VSEL7	VSEL6	VSEL5	VSEL4	VSEL3	VSEL2	VSEL1	VSEL0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	VSELx (x=31~0)	中断使能	分别选择中断事件表中序号对应的事件，具体对应关系如下： ● INTC_VSSEL128：每一位 VSELx 分别使能 0x000~0x01F 对应的中断事件请求。 ● INTC_VSSEL129：每一位 VSELx 分别使能 0x020~0x03F 对应的中断事件请求。 ● INTC_VSSEL130：每一位 VSELx 分别使能 0x040~0x05F 对应的中断事件请求。 ● INTC_VSSEL131：每一位 VSELx 分别使能 0x060~0x07F 对应的中断事件请求。 ● INTC_VSSEL132：每一位 VSELx 分别使能 0x080~0x09F 对应的中断事件请求。 ● INTC_VSSEL133：每一位 VSELx 分别使能 0x0A0~0x0BF 对应的中断事件请求。 ● INTC_VSSEL134：每一位 VSELx 分别使能 0x0C0~0x0DF 对应的中断事件请求。 ● INTC_VSSEL135：每一位 VSELx 分别使能 0x0E0~0x0FF 对应的中断事件请求。 ● INTC_VSSEL136：每一位 VSELx 分别使能 0x100~0x11F 对应的中断事件请求。 ● INTC_VSSEL137：每一位 VSELx 分别使能 0x120~0x13F 对应的中断事件请求。 ● INTC_VSSEL138：每一位 VSELx 分别使能 0x140~0x15F 对应的中断事件请求。 ● INTC_VSSEL139：每一位 VSELx 分别使能 0x160~0x17F 对应的中断事件请求。 ● INTC_VSSEL140：每一位 VSELx 分别使能 0x180~0x19F 对应的中断事件请求。 ● INTC_VSSEL141：每一位 VSELx 分别使能 0x1A0~0x1BF 对应的中断事件请求。 ● INTC_VSSEL142：每一位 VSELx 分别使能 0x1C0~0x1DF 对应的中断事件请求。 ● INTC_VSSEL143：每一位 VSELx 分别使能 0x1E0~0x1FF 对应的中断事件请求。	RW

11.4.12 停止模式唤醒事件使能寄存器 (INTC_WKEN)

Offset	Bit Position																																
0x0050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																																

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	ETHWKEN	ETH_PMT 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
28	USFWKEN	USBFS_WKUP 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
27	USHWKEN	USBHS_WKUP 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
26	RXWKEN	USART_1 RX 线 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
25	TMR2OVFWKEN	TMR21_OVFA 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
24	TMR2GCMPPWKEN	TMR21_GCMA 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
23	TMR0CMPWKEN	TMR0_1_CMPA 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
22	RTCPRDWKEN	RTC_PRD 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
21	RTCALMWKEN	RTC_ALM 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
20	WKTMWKEN	WKTM_CMP 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
19	CMP1WKEN	CMP1 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
18	PVD2WKEN	PVD_PVD2 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
17	PVD1WKEN	PVD_PVD1 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW

位/位域	标记	位名	功能描述	读写
16	SWDTWKEN	SWDT_REFUDF 停止模式唤醒使能	0b0: 唤醒禁止 0b1: 唤醒许可	RW
15:0	EIRQWKEN	EIRQ 停止模式 唤醒使能	对应 EIRQWKEN15~EIRQWKEN0: 0b0: 唤醒禁止 0b1: 唤醒许可	RW

11.4.13 软件中断事件寄存器 (INTC_SWIER)

Offset	Bit Position																															
0x029C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SWIE																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	SWIE	软件中断事件	对应 SWIE31~SWIE0: 0b0: 不发生软件中断/事件 0b1: 发生软件中断/事件	RW

11.4.14 事件使能寄存器 (INTC_EVTER)

Offset	Bit Position																															
0x02A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EVTE																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EVTE	事件使能	对应 EVTE31~EVTE0: 0b0: 事件选择禁止 0b1: 事件选择许可	RW

11.4.15 中断使能寄存器 (INTC_IER)

Offset	Bit Position																															
0x02A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	IEN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	IEN	中断使能	IEN31~IEN0 分别对应 NVIC 中断请求 31~0: ● 0b0: 中断禁止 ● 0b1: 中断许可 禁止时, 各中断请求对应的中断事件将不能被 NVIC 接收。	RW

11.4.16 FPU 中断使能寄存器 (INTC_FPUIER)

Offset	Bit Position																															
0x02A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										IDCIEN	IXCIEN	UFCIEN	OFCIEN	DZCIEN	IOCIEN
Access																											RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	IDCIEN	FPU IDCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW
4	IXCIEN	FPU IXCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW
3	UFCIEN	FPU UFCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW
2	OFCIEN	FPU OFCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW
1	DZCIEN	FPU DZCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW
0	IOCIEN	FPU IOCI 中断使能位	● 0b0: 中断禁止 ● 0b1: 中断允许	RW

11.5 注意事项

关于 ARM 核中断的描述, 请参考 ARM 手册《ARM Processor Cortex®-M4 Technical Reference Manual (ARM DDI 0439D)》。

12 自动运行系统 (AOS)

12.1 简介

自动运行系统 (Automatic Operation System) 用于在不借助 CPU 的情况下实现外设之间的联动。利用外设产生的事件作为 AOS 源 (AOS Source)，如定时器的比较匹配、定时器的计数溢出、RTC 的周期信号、通信模块的收发数据的各种状态 (空闲、接收数据满、发送数据结束、发送数据空)、ADC 的转换结束等事件，来触发其他外设动作。被触发的外设动作称为 AOS 目标 (AOS Target)。

AOS 中还配备了 4 个可编程逻辑运算单元 (PLU)，用于 PORT 和 AOS 源的逻辑运算。用户可以根据需求选择一个或多个 AOS 源来触发同一个 AOS 目标。

12.2 主要特性

AOS 有多个触发源，详见 [AOS 源事件](#)。除特殊限制外，每个 AOS 目标可选择其中一个或多个作为触发源。需要选择多个触发源时可以使用 AOS 配备的逻辑运算单元 (PLU)，最多可以同时选择 4 个触发源。另外还可以通过公共触发源选择寄存器 1 和公共触发源选择寄存器 2 选择两个公共触发源，所有 AOS 目标共享这两个公共触发源。这样，对于一个 AOS 目标来说，最多可以选择 6 个触发源，其中任意一个发生触发事件时均可触发该 AOS 目标动作。

可以由外设电路硬件触发，也可通过写寄存器由软件触发。

能够作为 AOS 目标的外设及数量如下：

- 4 个 DCU 触发目标
- 17 个 DMA 触发目标
- 4 个高级控制定时器 (Timer6) 触发目标
- 1 个通用定时器 0 (Timer0) 触发目标
- 1 个通用定时器 2 (Timer2) 触发目标
- 4 个通用定时器 A (TimerA) 触发目标
- 2 个 Event Port 触发目标
- 2 个 HASH 触发目标
- 1 个温度传感器 (OTS) 触发目标
- 3 组每组 2 个 ADC 触发目标，用于 ADC1~ADC3 序列触发

12.3 功能说明

12.3.1 功能框图

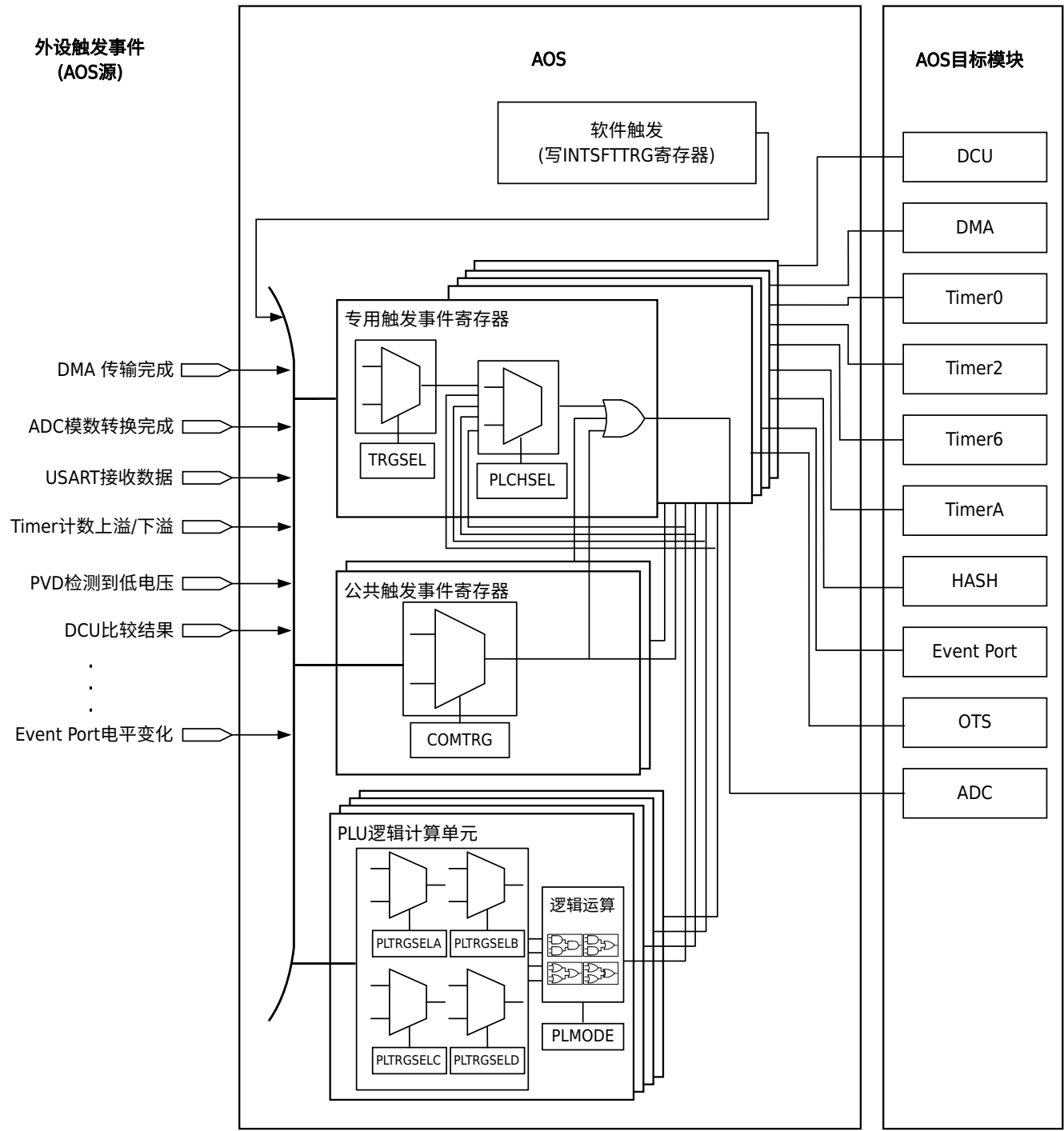


图 12-1 AOS 模块框图

12.3.2 AOS 源事件和目标

AOS 源事件编号见《中断控制器（INTC）》章节的中断事件表，表中“触发源”列中打“√”的事件可以用作 AOS 源。AOS 目标如下表所示。

表 12-1 AOS 目标列表

模块	序号	动作
DCU_1	0	触发加/减法运算，三角波/锯齿波模式下触发数据递增或递减
DCU_2	1	触发加/减法运算，三角波/锯齿波模式下触发数据递增或递减
DCU_3	2	触发加/减法运算，三角波/锯齿波模式下触发数据递增或递减
DCU_4	3	触发加/减法运算，三角波/锯齿波模式下触发数据递增或递减
DMA_1	4	触发通道 0 开始传输
	5	触发通道 1 开始传输
	6	触发通道 2 开始传输
	7	触发通道 3 开始传输
	8	触发通道 4 开始传输
	9	触发通道 5 开始传输
	10	触发通道 6 开始传输
	11	触发通道 7 开始传输
DMA_2	12	触发通道 0 开始传输
	13	触发通道 1 开始传输
	14	触发通道 2 开始传输
	15	触发通道 3 开始传输
	16	触发通道 4 开始传输
	17	触发通道 5 开始传输
	18	触发通道 6 开始传输
	19	触发通道 7 开始传输
DMA_1/2	20	触发事件触发通道重置
Timer6	21	触发计数器启动/停止/清零/刷新/捕获/递增/递减
	22	触发计数器启动/停止/清零/刷新/捕获/递增/递减
	23	触发计数器启动/停止/清零/刷新/捕获/递增/递减
	24	触发计数器启动/停止/清零/刷新/捕获/递增/递减
Event Port	25	触发 Event Port1/2 动作
	26	触发 Event Port3/4 动作
Timer0	27	触发计数器启动/停止/清零/捕获
Timer2	28	触发计数器启动/停止/清零/捕获

模块	序号	动作
HASH	29	触发 HASH 运算启动
	30	触发 HASH 最后一次运算启动
TimerA	31	触发单元 1/5/9 计数器启动/停止/清零/递增/递减
		触发单元 2/6/10 计数器输入捕获
	32	触发单元 2/6/10 计数器启动/停止/清零/递增/递减
		触发单元 1/5/9 计数器输入捕获
	33	触发单元 3/7/11 计数器启动/停止/清零/递增/递减
		触发单元 4/8/12 计数器输入捕获
	34	触发单元 4/8/12 计数器启动/停止/清零/递增/递减
		触发单元 3/7/11 计数器输入捕获
OTS	35	触发 OTS 开始测温
ADC_1	36、37	触发序列 A/B 开始模数转换
ADC_2	38、39	触发序列 A/B 开始模数转换
ADC_3	40、41	触发序列 A/B 开始模数转换

12.3.3 专用触发源

具有 AOS 目标的外设电路模块为每个 AOS 目标配有一个专用的外设触发源选择寄存器，当这个寄存器写入 AOS 源对应的事件编号时，该 AOS 目标即选择这个 AOS 源作为触发源。当 AOS 源的事件发生时，这个事件将通过 AOS 传递给 AOS 目标，作为 AOS 目标的外设电路根据自身设定开始动作。

12.3.4 公共触发源

除了各个 AOS 目标的专用外设触发源选择寄存器外，AOS 还配置了两个公共触发源选择寄存器 (AOS_COMTRGSEL1, AOS_COMTRGSEL2)。用于实现同一触发源触发一个或多个 AOS 目标的功能，也可以实现多个 AOS 触发源触发同一个 AOS 目标的功能。使用时首先在 AOS 目标专用外设触发源选择寄存器中将公共触发源使能位置为有效，再在公共触发源选择寄存器中写入 AOS 源对应的事件编号。当 AOS 源的事件发生时，这个事件将通过 AOS 的公共触发源传递给 AOS 目标，作为 AOS 目标的外设电路根据自身设定开始动作。当同时设置专用触发源和公共触发源后，可以最多同时有 3 个 AOS 源触发同一 AOS 目标，3 个 AOS 源中的任意一个发生触发事件时，都将触发该 AOS 目标。

所有 AOS 目标共享这两个公共触发源。因此使用时当其他 AOS 目标不使用公共触发源选择寄存器选定的事件时，需要在它的专用外设触发源选择寄存器中将公共触发源使能位置为无效，以防止错误的触发动作。

12.3.5 可编程逻辑运算单元 (PLU)

AOS 中一共配备了 4 个可编程逻辑运算单元 (PLU)，每个单元有 4 个输入 (PLINA、PLINB、PLINC、PLIND) 和 1 个输出 (PLOUT)。

四个输入 PLINA~D 的源不仅可以是 AOS 触发源，也可以是 PORT 输入，或者是固定值。逻辑运算后的结果不仅可以用于 AOS 触发目标模块，也可以通过 PORT 输出。

设置 AOS_PLUx_CR (x=0~3) 寄存器的 PLINySEL[1:0] (y=A、B、C、D) 位可以选择 PLINA~D 的输入源。设置 AOS_PLUx_CR 寄存器的 PLMODE[1:0]位可以选择需要的逻辑运算关系。

PLUx_INA、PLUx_INB、PLUx_INC、PLUx_IND 和 PLUx_OUT 的配置请参见《数据手册》中“引脚功能说明”章节。

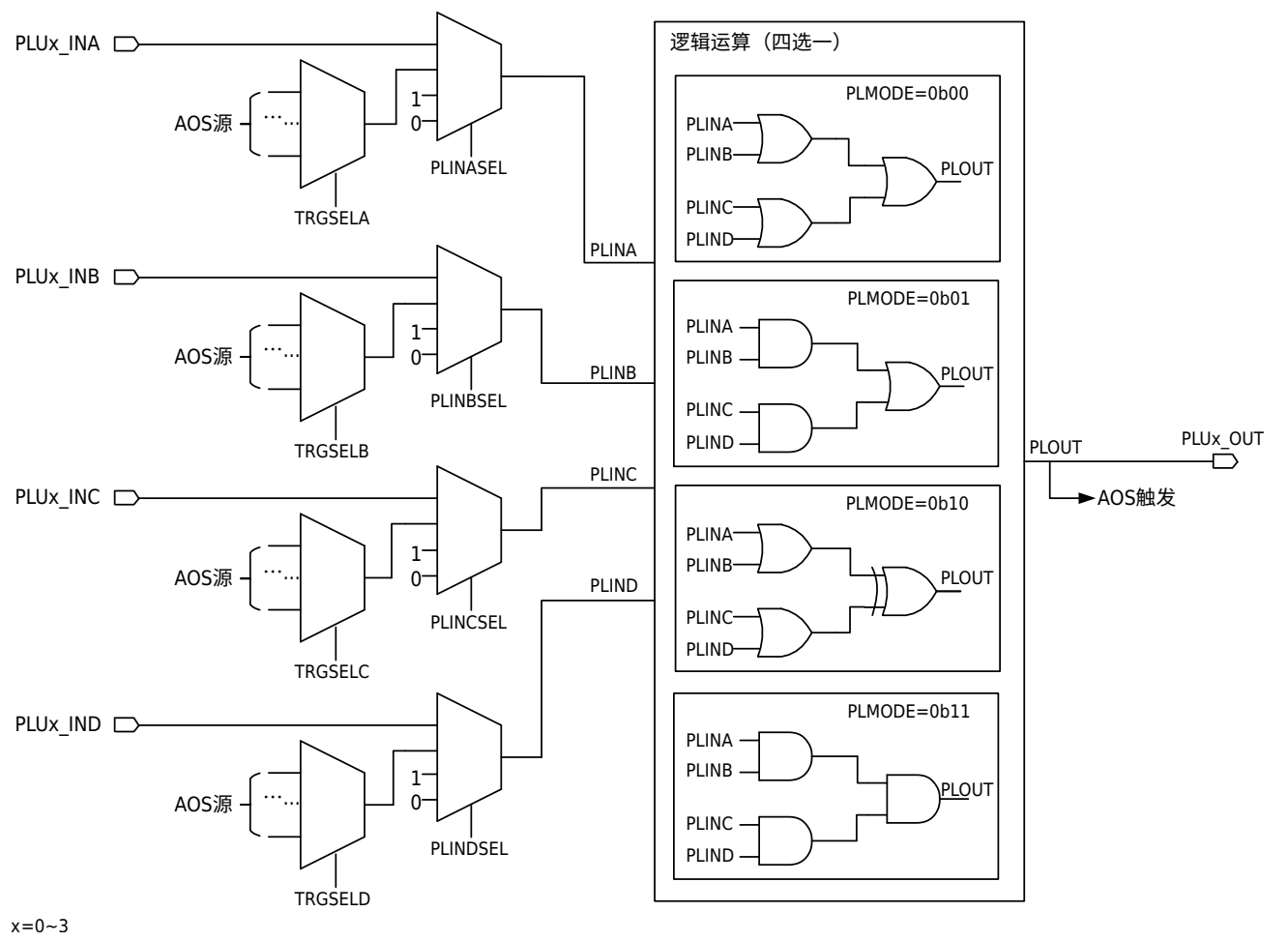


图 12-2 PLU 框图

12.4 寄存器

12.4.1 寄存器总表

基地址：0x40010800

表 12-2 自动运行系统 (AOS) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	INTSFTRG	软件触发事件寄存器	32
0x04*x	DCU_TRGSELx	DCU 触发源选择寄存器 (x=1~4)	32
0x14+0x04*x	DMA1_TRGSELx	DMA1 传输启动触发源选择寄存器 (x=0~7)	32
0x34+0x04*x	DMA2_TRGSELx	DMA2 传输启动触发源选择寄存器 (x=0~7)	32
0x54	DMA_RC_TRGSEL	DMA 通道重置触发源选择寄存器	32
0x58+0x04*x	TMR6_TRGSELx	Timer6 硬件触发事件选择寄存器 (x=0~3)	32
0x68	PEVNT_TRGSEL12	Event Port1/2 触发源选择寄存器	32
0x6C	PEVNT_TRGSEL34	Event Port3/4 触发源选择寄存器	32
0x70	TMR0_TRGSEL	Timer0 硬件触发事件选择寄存器	32
0x74	TMR2_TRGSEL	Timer2 硬件触发事件选择寄存器	32
0x78	HASH_TRGSELA	HASH 硬件触发事件选择寄存器 A	32
0x7C	HASH_TRGSELB	HASH 硬件触发事件选择寄存器 B	32
0x80+0x04*x	TMRA_TRGSELx	TimerA 硬件触发事件选择寄存器 (x=0~3)	32
0x90	OTS_TRGSEL	OTS 硬件触发事件选择寄存器	32
0x94+0x04*x	ADC1_TRGSELx	ADC1 转换开始片内触发源选择寄存器 (x=0~1)	32
0x9C+0x04*x	ADC2_TRGSELx	ADC2 转换开始片内触发源选择寄存器 (x=0~1)	32
0xA4+0x04*x	ADC3_TRGSELx	ADC3 转换开始片内触发源选择寄存器 (x=0~1)	32
0xAC	AOS_COMTRGSEL1	公共触发源选择寄存器 1	32
0xB0	AOS_COMTRGSEL2	公共触发源选择寄存器 2	32
0x200+0x004*x	AOS_PLUX_CR	PLU 控制寄存器 (x=0~3)	32
0x210+0x010*x	AOS_PLUX_TRGSELA	PLU 触发源选择寄存器 A (x=0~3)	32
0x214+0x010*x	AOS_PLUX_TRGSELB	PLU 触发源选择寄存器 B (x=0~3)	32
0x218+0x010*x	AOS_PLUX_TRGSELC	PLU 触发源选择寄存器 C (x=0~3)	32
0x21C+0x010*x	AOS_PLUX_TRGSELD	PLU 触发源选择寄存器 D (x=0~3)	32

12.4.2 软件触发事件寄存器 (INTSFTTRG)

该寄存器每次写 1 将产生一次触发事件。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															STRG
Access																																WO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	STRG	软件触发	写 1 产生一次软件触发事件，写 0 无效	WO

12.4.3 DCU 触发源选择寄存器 (DCU_TRGSELx) (x=1~4)

在 DCU 选择硬件触发启动模式后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生后，DCU 将由该事件触发启动并进行运算。

每个 DCU 单元对应一个寄存器。

偏移地址 (Offset) : 0x04*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x000001FF																																
Name	COMEN	Reserved												PLCHSEL				Reserved								TRGSEL							
Access	RW													RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.4 DMA1 传输启动触发源选择寄存器 (DMA1_TRGSELx) (x=0~7)

在 DMA1 使能后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，DMA1 将由该事件触发启动并进行传输。

每个数据传输通道对应一个寄存器。

偏移地址 (Offset) : 0x14+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved								TRGSEL									
Access	RW													RW									RW									

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.5 DMA2 传输启动触发源选择寄存器 (DMA2_TRGSELx) (x=0~7)

在 DMA2 使能后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，DMA2 将由该事件触发启动并进行传输。

每个数据传输通道对应一个寄存器。

偏移地址 (Offset) : 0x34+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved								TRGSEL									
Access	RW													RW									RW									

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.6 DMA 通道重置触发源选择寄存器 (DMA_RC_TRGSEL)

在 DMA 使能后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，DMA 将由该事件触发进行通道重置。

DMA1 和 DMA2 共用这个寄存器。

Offset	Bit Position																															
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN		Reserved										PLCHSEL		Reserved										TRGSEL							
Access	RW												RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.7 Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3)

在 Timer6 选择内部触发硬件同步后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设电路事件发生时，Timer6 将由该事件触发硬件同步（包括启动/停止/清零/刷新/递增/递减/输入捕获）。

所有 Timer6 单元共用这 4 个寄存器。

偏移地址 (Offset) : 0x58+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved								TRGSEL									
Access	RW													RW									RW									

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.8 Event Port 触发源选择寄存器 (PEVNT_TRGSEL12/PEVNT_TRGSEL34)

将触发事件的编号写入该寄存器中，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。Event Port 被触发输出指定电平或者锁存端口状态。

Event Port1/2 共用 PEVNT_TRGSEL12 寄存器，Event Port3/4 共用 PEVNT_TRGSEL34 寄存器。

偏移地址 (Offset)：0x68 (PEVNT_TRGSEL12)，0x6C (PEVNT_TRGSEL34)

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x000001FF																																
Name	COMEN	Reserved												PLCHSEL				Reserved								TRGSEL							
Access	RW													RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.9 Timer0 硬件触发事件选择寄存器 (TMR0_TRGSEL)

在 Timer0 选择硬件触发后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b0100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，Timer0 将由该事件触发动作（包括启动/停止/清零/输入捕获）。

所有 Timer0 单元及其通道共用这个寄存器。

Offset	Bit Position																															
0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved										TRGSEL							
Access		RW													RW																	

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.10 Timer2 硬件触发事件选择寄存器（TMR2_TRGSEL）

在 Timer2 选择硬件触发后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，Timer2 将由该事件触发动作（包括启动/停止/清零/输入捕获）。

所有 Timer2 单元及其通道共用这个寄存器。

Offset	Bit Position																																
0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x000001FF																																
Name	COMEN	Reserved												PLCHSEL				Reserved								TRGSEL							
Access	RW													RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.11 HASH 硬件触发事件选择寄存器 (HASH_TRGSELx) (x=A/B)

当数据写入 HASH_DR 后，请用 HASH_TRGSELA 寄存器来选择触发事件源，以启动 HASH 运算。当最后一组数据写入 HASH_DR 后，请用 HASH_TRGSELB 寄存器来选择触发事件源，以通知 HASH 进行最后一次运算。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，HASH 将由该事件触发启动运算。

 **注意**

请勿选择 DMA_BTcx (x=0~7) 以外的其他触发信号。

偏移地址 (Offset) : 0x78 (HASH_TRGSELA) , 0x7C (HASH_TRGSELB)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved								TRGSEL									
Access		RW													RW																	

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.12 TimerA 硬件触发事件选择寄存器（TMRA_TRGSELx）（x=0~3）

在 TimerA 选择内部硬件触发后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，TimerA 将由该事件触发动作（包括启动/停止/清零/输入捕获）。

所有 TimerA 单元共用这 4 个寄存器。

偏移地址（Offset）：0x80+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved										TRGSEL							
Access	RW													RW											RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.13 OTS 硬件触发事件选择寄存器 (OTS_TRGSEL)

在 OTS 选择硬件触发启动模式后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，OTS 将由该事件触发启动。

Offset	Bit Position																															
0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved												PLCHSEL	Reserved								TRGSEL									
Access	RW													RW									RW									

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0：禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0：禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1：允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000：使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001：使用 PLU0 运算后的事件触发本 AOS 目标 0b010：使用 PLU1 运算后的事件触发本 AOS 目标 0b011：使用 PLU2 运算后的事件触发本 AOS 目标 0b100：使用 PLU3 运算后的事件触发本 AOS 目标 - 其他：禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.14 ADC1 转换开始片内触发源选择寄存器 (ADC1_TRGSELx) (x=0~1)

在 ADC1 选择片内触发模式后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，ADC1 将由该事件触发模数转换。

ADC1 的序列 A 和序列 B 共用这 2 个寄存器。

偏移地址 (Offset) : 0x94+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved											PLCHSEL	Reserved											TRGSEL							
Access	RW												RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.15 ADC2 转换开始片内触发源选择寄存器 (ADC2_TRGSELx) (x=0~1)

在 ADC2 选择片内触发模式后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，ADC2 将由该事件触发模数转换。

ADC2 的序列 A 和序列 B 共用这 2 个寄存器。

偏移地址 (Offset) : 0x9C+0x04*x

Offset	Bit Position																																			
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x000001FF																																			
Name	COMEN	Reserved												PLCHSEL	Reserved												TRGSEL									
Access	RW													RW													RW									

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.16 ADC3 转换开始片内触发源选择寄存器 (ADC3_TRGSELx) (x=0~1)

在 ADC3 选择片内触发模式后，请用该寄存器来选择触发事件源。当 PLCHSEL[2:0]设为 0b000 时，请将触发事件的编号写入该寄存器的 TRGSEL[8:0]位中。当 PLCHSEL[2:0]设为 0b001、0b010、0b011、0b100 时，请将触发事件的编号写入对应的 PLU 触发源选择寄存器 AOS_PLU_TRGSELA/B/C/D 中，并设置 PLU 控制寄存器 AOS_PLU_CR。当编号对应的外设事件发生时，ADC3 将由该事件触发模数转换。

ADC3 的序列 A 和序列 B 共用这 2 个寄存器。

偏移地址 (Offset) : 0xA4+0x04*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	COMEN	Reserved											PLCHSEL	Reserved											TRGSEL							
Access	RW												RW												RW							

位/位域	标记	位名	功能描述	读写
31	COMEN[1]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL2 的公共触发事件触发本 AOS 目标	RW
30	COMEN[0]	公共触发使能	0b0: 禁止 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标 0b1: 允许 AOS_COMTRGSEL1 的公共触发事件触发本 AOS 目标	RW
29:19	Reserved	保留	-	-
18:16	PLCHSEL	PLU 选择位	0b000: 使用 TRGSEL[8:0]位选择的事件触发本 AOS 目标 0b001: 使用 PLU0 运算后的事件触发本 AOS 目标 0b010: 使用 PLU1 运算后的事件触发本 AOS 目标 0b011: 使用 PLU2 运算后的事件触发本 AOS 目标 0b100: 使用 PLU3 运算后的事件触发本 AOS 目标 - 其他: 禁止设置	RW
15:9	Reserved	保留	-	-
8:0	TRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.17 公共触发源选择寄存器（AOS_COMTRGSELx）（x=1~2）

在 AOS_COMTRGSEL1/2 写入触发事件的编号，当编号对应的外设电路事件发生时，如果一个或多个 AOS 目标的专用触发源选择寄存器的 COMEN 位值为 1，则编号对应的外设事件将触发这一个或多个 AOS 目标启动。

偏移地址（Offset）：0xAC（AOS_COMTRGSEL1），0xB0（AOS_COMTRGSEL2）

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																								COMTRG							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	COMTRG	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.18 PLU 控制寄存器 (AOS_PLUX_CR) (x=0~3)

AOS 中一共配备了 4 个可编程逻辑运算单元 (PLU)，每个单元有 4 个输入 (PLINA、PLINB、PLINC、PLIND) 和 1 个输出 (PLOUT)。

逻辑运算的输入不仅可以是 AOS 触发源，也可以是 PORT 输入。通过 PLINySEL[1:0] (y=A、B、C、D) 位可以选择输入源或者固定值。通过 PLMODE[1:0]位可以选择需要的逻辑运算关系。

偏移地址 (Offset) : 0x200+0x004*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PLINDSEL	PLINCSEL	PLINBSEL	PLINASEL	Reserved								PLMODE			
Access																	RW	RW	RW	RW									RW			

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:14	PLINDSEL	PLIND 输入源选择	0b00: PLIND 固定为 0 0b01: PLIND 固定为 1 0b10: PLIND 为 AOS_PLU_TRGSELD 选择的触发源 0b11: PLIND 为 PLUX_IND 端口输入值	RW
13:12	PLINCSEL	PLINC 输入源选择	0b00: PLINC 固定为 0 0b01: PLINC 固定为 1 0b10: PLINC 为 AOS_PLU_TRGSELC 选择的触发源 0b11: PLINC 为 PLUX_INC 端口输入值	RW
11:10	PLINBSEL	PLINB 输入源选择	0b00: PLINB 固定为 0 0b01: PLINB 固定为 1 0b10: PLINB 为 AOS_PLU_TRGSELB 选择的触发源 0b11: PLINB 为 PLUX_INB 端口输入值	RW
9:8	PLINASEL	PLINA 输入源选择	0b00: PLINA 固定为 0 0b01: PLINA 固定为 1 0b10: PLINA 为 AOS_PLU_TRGSELA 选择的触发源 0b11: PLINA 为 PLUX_INA 端口输入值	RW
7:2	Reserved	保留	-	-
1:0	PLMODE	逻辑运算选择	0b00: OR-OR, 即 (PLINA PLINB) (PLINC PLIND) 0b01: AND-OR, 即 (PLINA & PLINB) (PLINC & PLIND) 0b10: OR-XOR, 即 (PLINA PLINB) ^ (PLINC PLIND) 0b11: AND-AND, 即 (PLINA & PLINB) & (PLINC & PLIND)	RW



注意

当使用 PLU 运算结果作为 AOS 触发源时，请将 PLINySEL (y=A,B,C,D) 设为 0b00 或 0b10，且 PLMODE 设为 0b00。

12.4.19 PLU 触发源选择寄存器 A (AOS_PLUX_TRGSELA) (x=0~3)

当 PLU 的 PLINA 选择触发源输入时，请将对应触发事件的编号写入 AOS_PLUX_TRGSELA。

偏移地址 (Offset) : 0x210+0x010*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																								PLTRGSEL							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	PLTRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.20 PLU 触发源选择寄存器 B (AOS_PLUX_TRGSELB) (x=0~3)

当 PLU 的 PLINB 选择触发源输入时，请将对应触发事件的编号写入 AOS_PLUX_TRGSELB。

偏移地址 (Offset) : 0x214+0x010*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																								PLTRGSEL							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	PLTRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.21 PLU 触发源选择寄存器 C (AOS_PLUX_TRGSELC) (x=0~3)

当 PLU 的 PLINC 选择触发源输入时，请将对应触发事件的编号写入 AOS_PLUX_TRGSELC。

偏移地址 (Offset) : 0x218+0x010*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																				PLTRGSEL											
Access																					RW											

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	PLTRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

12.4.22 PLU 触发源选择寄存器 D (AOS_PLUX_TRGSELD) (x=0~3)

当 PLU 的 PLIND 选择触发源输入时，请将对应触发事件的编号写入 AOS_PLUX_TRGSELD。

偏移地址 (Offset) : 0x21C+0x010*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000001FF																															
Name	Reserved																				PLTRGSEL											
Access																					RW											

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:0	PLTRGSEL	触发源选择	写入要选择的外设电路事件的编号	RW

13 存储保护单元（MPU）

13.1 简介

MPU 可以提供对存储器的保护，通过阻止非授权的访问，可以提高系统的安全性。
本芯片内置了 1 个针对 CPU 的 MPU 单元，5 个针对主机的 MPU 单元和 1 个针对 IP 的 MPU 单元。

表 13-1 MPU 模块简介

模块		内容
ARM MPU		CPU 的存储保护单元 8 区域，详见 ARM MPU 说明
DMPU	系统 DMA_1 MPU: SMPU1	系统 DMA_1 的存储保护单元 16 区域，8 区域系统 DMA 专用，8 区域所有 DMA 共用
	系统 DMA_2 MPU: SMPU2	系统 DMA_2 的存储保护单元 16 区域，8 区域系统 DMA 专用，8 区域所有 DMA 共用
	USBFS-DMA MPU: FMPU	USBFS-DMA 的存储保护单元 8 区域，所有 DMA 共用
	USBHS-DMA MPU: HMPU	USBHS-DMA 的存储保护单元 8 区域，所有 DMA 共用
	ETH-DMA MPU: EMPU	ETH-DMA 的存储保护单元 8 区域，所有 DMA 共用
IPMPU		针对系统 IP 和安全相关 IP 的访问保护单元

其中 ARM MPU 提供 CPU 对全部 4G 地址空间的访问权限控制。
DMPU 包含 SMPU1/SMPU2/FMPU/HMPU/EMPU，分别提供系统 DMA_1/系统 DMA_2/USBFS-DMA/USBHS-DMA/ETH-DMA 对全部 4G 地址空间的读写访问权限控制。对禁止空间发生访问时，可以设置 MPU 动作为无视/总线错误/不可屏蔽中断/复位。
IPMPU 提供非特权模式时对系统 IP 和安全相关 IP 的访问权限控制。

13.2 功能说明

13.2.1 区域范围设置

MPU 以区域为单位对存储空间进行权限管理。每个区域可以独立设定基地址和区域大小，可设的范围为 32Bytes~4GBytes，大小必须为 2^N Bytes ($N=5\sim32$)，对应的基地址低 N 位为 0。

没有被任何区域覆盖的地址空间称为背景区域。

13.2.2 权限设置

每个区域包括背景区域可以针对各个 DMA 独立设置允许读/禁止读和允许写/禁止写。如果不同区域之间发生地址重叠，则设定的禁止优先。

13.2.3 MPU 动作选择

发生被禁止的访问时，该访问被无视（读访问读到 0，写访问忽略）的同时，可以设置对应的动作，可以设置为：

- 无视
- 总线错误
- 不可屏蔽中断
- 复位

13.2.4 启动 MPU

SMPU1/SMPU2/FMPU/HMPU/EMPU 可以独立使能。

建议在设置好区域范围/权限设置/动作选择后再使能 MPU。

13.3 操作示例

13.3.1 只允许部分空间访问

例：将背景区域权限设置为禁止读/禁止写，区域 0 设置为允许读/允许写，区域 1 设置为允许读禁止写，区域 0 范围覆盖区域 1。

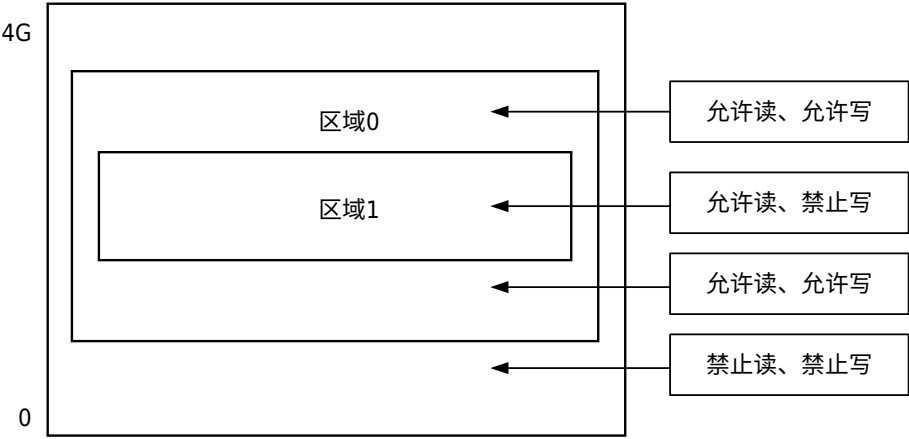


图 13-1 只允许部分空间访问示意图

13.3.2 只禁止部分空间访问

例：将背景区域权限设置为允许读/禁止写，区域 0 设置为允许读/允许写，区域 1 设置为允许读/禁止写，区域 0 覆盖区域 1，区域 2 设置为禁止读/禁止写。

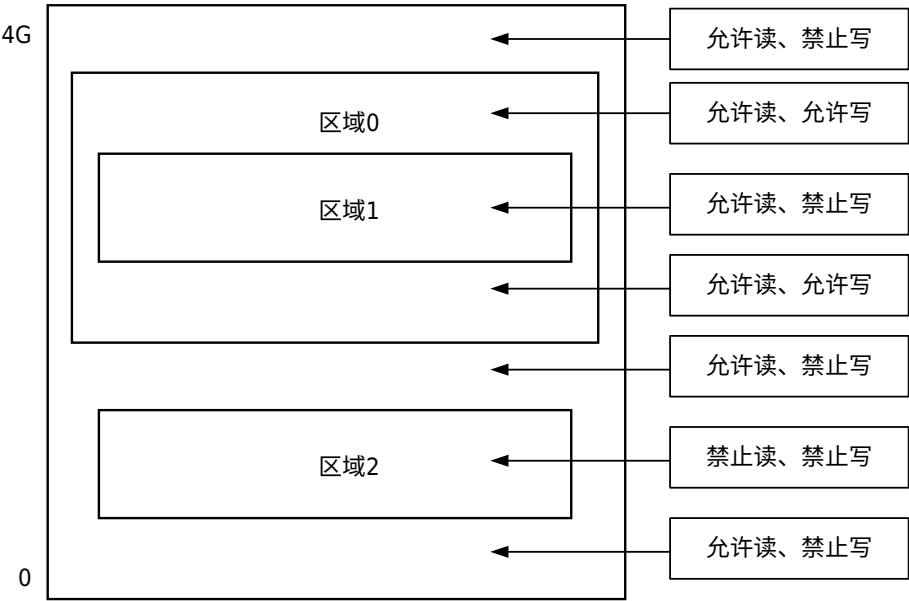


图 13-2 只禁止部分空间访问示意图

13.4 寄存器

13.4.1 寄存器总表

本模块的寄存器只能通过 CPU 来设置。

基地址：0x40050000

表 13-2 存储保护单元 (MPU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00+0x04*n	MPU_RGDn	区域范围描述寄存器(n=0~15) ⁽¹⁾	32
0x40	MPU_SR	状态标志寄存器	32
0x44	MPU_ECLR	标志清除寄存器	32
0x48	MPU_WP	写保护寄存器	32
0x4C	MPU_IPPR	IP 访问保护寄存器 ⁽¹⁾	32
0x50	MPU_S1RGE	SMPU1 区域使能寄存器 ⁽¹⁾	32
0x54	MPU_S1RGWP	SMPU1 区域写权限寄存器 ⁽¹⁾	32
0x58	MPU_S1RGRP	SMPU1 区域读权限寄存器 ⁽¹⁾	32
0x5C	MPU_S1CR	SMPU1 控制寄存器 ⁽¹⁾	32
0x60	MPU_S2RGE	SMPU2 区域使能寄存器 ⁽¹⁾	32
0x64	MPU_S2RGWP	SMPU2 区域写权限寄存器 ⁽¹⁾	32
0x68	MPU_S2RGRP	SMPU2 区域读权限寄存器 ⁽¹⁾	32
0x6C	MPU_S2CR	SMPU2 控制寄存器 ⁽¹⁾	32
0x70	MPU_FRGE	FMPU 区域使能寄存器 ⁽¹⁾	32
0x74	MPU_FRGWP	FMPU 区域写权限寄存器 ⁽¹⁾	32
0x78	MPU_FRGRP	FMPU 区域读权限寄存器 ⁽¹⁾	32
0x7C	MPU_FCR	FMPU 控制寄存器 ⁽¹⁾	32
0x80	MPU_HRGE	HMPU 区域使能寄存器 ⁽¹⁾	32
0x84	MPU_HRGWP	HMPU 区域写权限寄存器 ⁽¹⁾	32
0x88	MPU_HRGRP	HMPU 区域读权限寄存器 ⁽¹⁾	32
0x8C	MPU_HCR	HMPU 控制寄存器 ⁽¹⁾	32
0x90	MPU_ERGE	EMPU 区域使能寄存器 ⁽¹⁾	32
0x94	MPU_ERGWP	EMPU 区域写权限寄存器 ⁽¹⁾	32
0x98	MPU_ERGRP	EMPU 区域读权限寄存器 ⁽¹⁾	32
0x9C	MPU_ECR	EMPU 控制寄存器 ⁽¹⁾	32

**说明**

1. 将 MPU_WP.MPUWE 置 1 时，允许写入。

13.4.2 区域范围描述寄存器 (MPU_RGDn) (n=0~15)

偏移地址 (Offset) : 0x00+0x04*n

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	MPURGnADDR[31:5]																								MPURGnSIZE							
Access	RW																								RW							

位/位域	标记	位名	功能描述	读写
31:5	MPURGnADDR[31:5]	区域基地址	设定区域 n 的基地址，有效位数与区域大小有关。 区域基地址总长 32bits，其中 MPURGnADDR[MPURGnSIZE:0]固定为 0。 例如： 区域大小为 32Bytes 时，区域基地址为 MPURGnADDR[31:5]+MPURGnADDR[4:0]（低 5 位 固定为 0）。 区域大小为 64Bytes 时，区域基地址为 MPURGnADDR[31:6]+MPURGnADDR[5:0]（低 6 位 固定为 0）。	RW
4:0	MPURGnSIZE	区域大小	设定区域 n 的大小： ● 0~3：禁止设置 ● 4：32Bytes ● 5：64Bytes ● ● 30：2GBytes ● 31：4GBytes	RW

13.4.3 状态标志寄存器 (MPU_SR)

Offset	Bit Position																																
0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																												EMPUEAF	HMPUEAF	FMPUEAF	SMPU2EAF	SMPUIEAF
Access																													RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4	EMPUEAF	EMPU 错误标志	● 0b0: ETH-DMA 未发生错误访问 ● 0b1: ETH-DMA 发生了错误访问	RO
3	HMPUEAF	HMPU 错误标志	● 0b0: USBHS-DMA 未发生错误访问 ● 0b1: USBHS-DMA 发生了错误访问	RO
2	FMPUEAF	FMPU 错误标志	● 0b0: USBFS-DMA 未发生错误访问 ● 0b1: USBFS-DMA 发生了错误访问	RO
1	SMPU2EAF	SMPU2 错误标志	● 0b0: 系统 DMA_2 未发生错误访问 ● 0b1: 系统 DMA_2 发生了错误访问	RO
0	SMPU1EAF	SMPU1 错误标志	● 0b0: 系统 DMA_1 未发生错误访问 ● 0b1: 系统 DMA_1 发生了错误访问	RO

 注意

对本寄存器的写操作会被忽略，清除错误标志请使用 MPU_ECLR。

13.4.4 标志清除寄存器 (MPU_ECLR)

Offset	Bit Position																															
0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										EMPUECLR	HMPUECLR	FMPUECLR	SMPU2ECLR	SMPU1ECLR	
Access																											RW	RW	RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4	EMPUECLR	EMPU 错误标志清除	写 1 将错误标志清 0，写 0 无效；读出恒为 0	RW
3	HMPUECLR	HMPU 错误标志清除	写 1 将错误标志清 0，写 0 无效；读出恒为 0	RW
2	FMPUECLR	FMPU 错误标志清除	写 1 将错误标志清 0，写 0 无效；读出恒为 0	RW
1	SMPU2ECLR	SMPU2 错误标志清除	写 1 将错误标志清 0，写 0 无效；读出恒为 0	RW
0	SMPU1ECLR	SMPU1 错误标志清除	写 1 将错误标志清 0，写 0 无效；读出恒为 0	RW

13.4.5 写保护寄存器 (MPU_WP)

Offset	Bit Position																																
0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																WKEY																MPUWE
Access																	WO																RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:1	WKEY	写入码	在写入 MPUWE 时，必须同时向 WKEY 写入 0x4B52	WO
0	MPUWE	MPU 寄存器写允许	0b0: MPU 地址寄存器/控制寄存器不允许写入 0b1: MPU 地址寄存器/控制寄存器允许写入	RW

 说明

向本寄存器写入 0x96A5 可以将 MPUWE 置 1；写入 0x96A4 可以将 MPUWE 清 0；禁止写入其他值修改 MPUWE。

13.4.6 IP 访问保护寄存器 (MPU_IPPR)



说明

- 特权模式下不受本寄存器影响可以读写访问对象 IP。
- DMPU 读/写保护有效时，除 MPU_IPPR 以外的寄存器不可读/写访问。

Offset	Bit Position																															
0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	BUSERRE	Reserved	MSTPWRP	MSTPRDP	RW SYSCWRP	RW SYSCRDP	RW INTCWRP	RW INTCRDP	RW SRAMCWRP	RW SRAMCRDP	RW DMPUWRP	RW DMPURDP	RW RTCWRP	RW RTCRDP	RW BKSAMWRP	RW BKSAMRDP	RW SWDTPWRP	RW SWDTRDP	RW WDTWRP	RW WDTRDP	RW ERMUWRP	RW ERMURDP	RW EFMWRP	RW EFMURDP	RW CRCWRP	RW CRCRDP	RW TRNGWRP	RW TRNGRDP	RW HASHWRP	RW HASHRDP	RW ENCRYPTWRP	RW ENCRYPTRDP
Access	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	BUSERRE	总线错误允许	● 0b0: 忽略对保护 IP 的访问 ● 0b1: 发生对保护 IP 的访问时，返回总线错误	RW
30	Reserved	保留	-	-
29	MSTPWRP	MSTP 写保护	● 0b0: 允许对 PWC_FCG0/1/2/3、PWC_FCG0PC 的写操作 ● 0b1: 禁止对 PWC_FCG0/1/2/3、PWC_FCG0PC 的写操作	RW
28	MSTPRDP	MSTP 读保护	● 0b0: 允许对 PWC_FCG0/1/2/3、PWC_FCG0PC 的读操作 ● 0b1: 禁止对 PWC_FCG0/1/2/3、PWC_FCG0PC 的读操作	RW
27	SYSCWRP	SYSC 写保护	● 0b0: 允许对 RMU/CMU/PWC 的写操作 ● 0b1: 禁止对 RMU/CMU/PWC 的写操作 说明 BUSERRE 设置为 1 时，对 RMU/PWC 的写操作不会发生总线错误。	RW
26	SYSCRD	SYSC 读保护	● 0b0: 允许对 RMU/CMU/PWC 的读操作 ● 0b1: 禁止对 RMU/CMU/PWC 的读操作 说明 BUSERRE 设置为 1 时，对 RMU/PWC 的写操作不会发生总线错误。	RW
25	INTCWRP	INTC 写保护	● 0b0: 允许对 INTC 的写操作 ● 0b1: 禁止对 INTC 的写操作	RW
24	INTCRDP	INTC 读保护	● 0b0: 允许对 INTC 的读操作 ● 0b1: 禁止对 INTC 的读操作	RW
23	SRAMCWRP	SRAMC 写保护	● 0b0: 允许对 SRAMC 的写操作 ● 0b1: 禁止对 SRAMC 的写操作	RW

位/位域	标记	位名	功能描述	读写
22	SRAMCRDP	SRAMC 读保护	● 0b0: 允许对 SRAMC 的读操作 ● 0b1: 禁止对 SRAMC 的读操作	RW
21	DMPUWRP	DMPU 写保护*2	● 0b0: 允许对 DMPU 的写操作 ● 0b1: 禁止对 DMPU 的写操作	RW
20	DMPURDP	DMPU 读保护*2	● 0b0: 允许对 DMPU 的读操作 ● 0b1: 禁止对 DMPU 的读操作	RW
19	RTCWRP	RTC 写保护	● 0b0: 允许对 RTC 的写操作 ● 0b1: 禁止对 RTC 的写操作	RW
18	RTCRDP	RTC 读保护	● 0b0: 允许对 RTC 的读操作 ● 0b1: 禁止对 RTC 的读操作	RW
17	BKSRAMWRP	BKSRAM 写保护	● 0b0: 允许对 Ret SRAM 的写操作 ● 0b1: 禁止对 Ret SRAM 的写操作	RW
16	BKSRAMRDP	BKSRAM 读保护	● 0b0: 允许对 Ret SRAM 的读操作 ● 0b1: 禁止对 Ret SRAM 的读操作	RW
15	SWDTWRP	SWDT 写保护	● 0b0: 允许对 SWDT 的写操作 ● 0b1: 禁止对 SWDT 的写操作	RW
14	SWDTRDP	SWDT 读保护	● 0b0: 允许对 SWDT 的读操作 ● 0b1: 禁止对 SWDT 的读操作	RW
13	WDTWRP	WDT 写保护	● 0b0: 允许对 WDT 的写操作 ● 0b1: 禁止对 WDT 的写操作	RW
12	WDTRDP	WDT 读保护	● 0b0: 允许对 WDT 的读操作 ● 0b1: 禁止对 WDT 的读操作	RW
11	ERMUWRP	ERMU 写保护	● 0b0: 允许对 ERMU 的写操作 ● 0b1: 禁止对 ERMU 的写操作	RW
10	ERMURDP	ERMU 读保护	● 0b0: 允许对 ERMU 的读操作 ● 0b1: 禁止对 ERMU 的读操作	RW
9	EFMWRP	EFM 写保护	● 0b0: 允许对 EFM 的写操作 ● 0b1: 禁止对 EFM 的写操作	RW
8	EFMRDP	EFM 读保护	● 0b0: 允许对 EFM 的读操作 ● 0b1: 禁止对 EFM 的读操作	RW
7	CRCWRP	CRC 写保护	● 0b0: 允许对 CRC 的写操作 ● 0b1: 禁止对 CRC 的写操作	RW
6	CRCRDP	CRC 读保护	● 0b0: 允许对 CRC 的读操作 ● 0b1: 禁止对 CRC 的读操作	RW
5	TRNGWRP	TRNG 写保护	● 0b0: 允许对 TRNG 的写操作 ● 0b1: 禁止对 TRNG 的写操作	RW
4	TRNGRDP	TRNG 读保护	● 0b0: 允许对 TRNG 的读操作 ● 0b1: 禁止对 TRNG 的读操作	RW
3	HASHWRP	HASH 写保护	● 0b0: 允许对 HASH 的写操作 ● 0b1: 禁止对 HASH 的写操作	RW

位/位域	标记	位名	功能描述	读写
2	HASHRDP	HASH 读保护	● 0b0: 允许对 HASH 的读操作 ● 0b1: 禁止对 HASH 的读操作	RW
1	ENCRYPTWRP	SKE/PKE 写保护	● 0b0: 允许对 SKE/PKE 的写操作 ● 0b1: 禁止对 SKE/PKE 的写操作	RW
0	ENCRYPTRDP	SKE/PKE 读保护	● 0b0: 允许对 SKE/PKE 的读操作 ● 0b1: 禁止对 SKE/PKE 的读操作	RW

13.4.7 SMPU1 区域使能寄存器 (MPU_S1RGE)

Offset	Bit Position																															
0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																S1RG15E	S1RG14E	S1RG13E	S1RG12E	S1RG11E	S1RG10E	S1RG9E	S1RG8E	S1RG7E	S1RG6E	S1RG5E	S1RG4E	S1RG3E	S1RG2E	S1RG1E	S1RG0E
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S1RG15E	SMPU1 区域 15 使能	0b0: SMPU1 的区域 15 无效 0b1: SMPU1 的区域 15 有效	RW
14	S1RG14E	SMPU1 区域 14 使能	0b0: SMPU1 的区域 14 无效 0b1: SMPU1 的区域 14 有效	RW
13	S1RG13E	SMPU1 区域 13 使能	0b0: SMPU1 的区域 13 无效 0b1: SMPU1 的区域 13 有效	RW
12	S1RG12E	SMPU1 区域 12 使能	0b0: SMPU1 的区域 12 无效 0b1: SMPU1 的区域 12 有效	RW
11	S1RG11E	SMPU1 区域 11 使能	0b0: SMPU1 的区域 11 无效 0b1: SMPU1 的区域 11 有效	RW
10	S1RG10E	SMPU1 区域 10 使能	0b0: SMPU1 的区域 10 无效 0b1: SMPU1 的区域 10 有效	RW
9	S1RG9E	SMPU1 区域 9 使能	0b0: SMPU1 的区域 9 无效 0b1: SMPU1 的区域 9 有效	RW
8	S1RG8E	SMPU1 区域 8 使能	0b0: SMPU1 的区域 8 无效 0b1: SMPU1 的区域 8 有效	RW
7	S1RG7E	SMPU1 区域 7 使能	0b0: SMPU1 的区域 7 无效 0b1: SMPU1 的区域 7 有效	RW
6	S1RG6E	SMPU1 区域 6 使能	0b0: SMPU1 的区域 6 无效 0b1: SMPU1 的区域 6 有效	RW
5	S1RG5E	SMPU1 区域 5 使能	0b0: SMPU1 的区域 5 无效 0b1: SMPU1 的区域 5 有效	RW
4	S1RG4E	SMPU1 区域 4 使能	0b0: SMPU1 的区域 4 无效 0b1: SMPU1 的区域 4 有效	RW
3	S1RG3E	SMPU1 区域 3 使能	0b0: SMPU1 的区域 3 无效 0b1: SMPU1 的区域 3 有效	RW
2	S1RG2E	SMPU1 区域 2 使能	0b0: SMPU1 的区域 2 无效 0b1: SMPU1 的区域 2 有效	RW
1	S1RG1E	SMPU1 区域 1 使能	0b0: SMPU1 的区域 1 无效 0b1: SMPU1 的区域 1 有效	RW

位/位域	标记	位名	功能描述	读写
0	S1RG0E	SMPU1 区域 0 使能	0b0: SMPU1 的区域 0 无效 0b1: SMPU1 的区域 0 有效	RW

13.4.8 SMPU1 区域写权限寄存器 (MPU_S1RGWP)

Offset	Bit Position																															
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																S1RG15WP	S1RG14WP	S1RG13WP	S1RG12WP	S1RG11WP	S1RG10WP	S1RG9WP	S1RG8WP	S1RG7WP	S1RG6WP	S1RG5WP	S1RG4WP	S1RG3WP	S1RG2WP	S1RG1WP	S1RG0WP
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S1RG15WP	SMPU1 区域 15 写权限	0b0: SMPU1 的区域 15 允许系统 DMA_1 写 0b1: SMPU1 的区域 15 禁止系统 DMA_1 写	RW
14	S1RG14WP	SMPU1 区域 14 写权限	0b0: SMPU1 的区域 14 允许系统 DMA_1 写 0b1: SMPU1 的区域 14 禁止系统 DMA_1 写	RW
13	S1RG13WP	SMPU1 区域 13 写权限	0b0: SMPU1 的区域 13 允许系统 DMA_1 写 0b1: SMPU1 的区域 13 禁止系统 DMA_1 写	RW
12	S1RG12WP	SMPU1 区域 12 写权限	0b0: SMPU1 的区域 12 允许系统 DMA_1 写 0b1: SMPU1 的区域 12 禁止系统 DMA_1 写	RW
11	S1RG11WP	SMPU1 区域 11 写权限	0b0: SMPU1 的区域 11 允许系统 DMA_1 写 0b1: SMPU1 的区域 11 禁止系统 DMA_1 写	RW
10	S1RG10WP	SMPU1 区域 10 写权限	0b0: SMPU1 的区域 10 允许系统 DMA_1 写 0b1: SMPU1 的区域 10 禁止系统 DMA_1 写	RW
9	S1RG9WP	SMPU1 区域 9 写权限	0b0: SMPU1 的区域 9 允许系统 DMA_1 写 0b1: SMPU1 的区域 9 禁止系统 DMA_1 写	RW
8	S1RG8WP	SMPU1 区域 8 写权限	0b0: SMPU1 的区域 8 允许系统 DMA_1 写 0b1: SMPU1 的区域 8 禁止系统 DMA_1 写	RW
7	S1RG7WP	SMPU1 区域 7 写权限	0b0: SMPU1 的区域 7 允许系统 DMA_1 写 0b1: SMPU1 的区域 7 禁止系统 DMA_1 写	RW
6	S1RG6WP	SMPU1 区域 6 写权限	0b0: SMPU1 的区域 6 允许系统 DMA_1 写 0b1: SMPU1 的区域 6 禁止系统 DMA_1 写	RW
5	S1RG5WP	SMPU1 区域 5 写权限	0b0: SMPU1 的区域 5 允许系统 DMA_1 写 0b1: SMPU1 的区域 5 禁止系统 DMA_1 写	RW
4	S1RG4WP	SMPU1 区域 4 写权限	0b0: SMPU1 的区域 4 允许系统 DMA_1 写 0b1: SMPU1 的区域 4 禁止系统 DMA_1 写	RW
3	S1RG3WP	SMPU1 区域 3 写权限	0b0: SMPU1 的区域 3 允许系统 DMA_1 写 0b1: SMPU1 的区域 3 禁止系统 DMA_1 写	RW

位/位域	标记	位名	功能描述	读写
2	S1RG2WP	SMPU1 区域 2 写权限	0b0: SMPU1 的区域 2 允许系统 DMA_1 写 0b1: SMPU1 的区域 2 禁止系统 DMA_1 写	RW
1	S1RG1WP	SMPU1 区域 1 写权限	0b0: SMPU1 的区域 1 允许系统 DMA_1 写 0b1: SMPU1 的区域 1 禁止系统 DMA_1 写	RW
0	S1RG0WP	SMPU1 区域 0 写权限	0b0: SMPU1 的区域 0 允许系统 DMA_1 写 0b1: SMPU1 的区域 0 禁止系统 DMA_1 写	RW

13.4.9 SMPU1 区域读权限寄存器 (MPU_S1RGRP)

Offset	Bit Position																															
0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																S1RG15RP	S1RG14RP	S1RG13RP	S1RG12RP	S1RG11RP	S1RG10RP	S1RG9RP	S1RG8RP	S1RG7RP	S1RG6RP	S1RG5RP	S1RG4RP	S1RG3RP	S1RG2RP	S1RG1RP	S1RG0RP
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S1RG15RP	SMPU1 区域 15 读权限	0b0: SMPU1 的区域 15 允许系统 DMA_1 读 0b1: SMPU1 的区域 15 禁止系统 DMA_1 读	RW
14	S1RG14RP	SMPU1 区域 14 读权限	0b0: SMPU1 的区域 14 允许系统 DMA_1 读 0b1: SMPU1 的区域 14 禁止系统 DMA_1 读	RW
13	S1RG13RP	SMPU1 区域 13 读权限	0b0: SMPU1 的区域 13 允许系统 DMA_1 读 0b1: SMPU1 的区域 13 禁止系统 DMA_1 读	RW
12	S1RG12RP	SMPU1 区域 12 读权限	0b0: SMPU1 的区域 12 允许系统 DMA_1 读 0b1: SMPU1 的区域 12 禁止系统 DMA_1 读	RW
11	S1RG11RP	SMPU1 区域 11 读权限	0b0: SMPU1 的区域 11 允许系统 DMA_1 读 0b1: SMPU1 的区域 11 禁止系统 DMA_1 读	RW
10	S1RG10RP	SMPU1 区域 10 读权限	0b0: SMPU1 的区域 10 允许系统 DMA_1 读 0b1: SMPU1 的区域 10 禁止系统 DMA_1 读	RW
9	S1RG9RP	SMPU1 区域 9 读权限	0b0: SMPU1 的区域 9 允许系统 DMA_1 读 0b1: SMPU1 的区域 9 禁止系统 DMA_1 读	RW
8	S1RG8RP	SMPU1 区域 8 读权限	0b0: SMPU1 的区域 8 允许系统 DMA_1 读 0b1: SMPU1 的区域 8 禁止系统 DMA_1 读	RW
7	S1RG7RP	SMPU1 区域 7 读权限	0b0: SMPU1 的区域 7 允许系统 DMA_1 读 0b1: SMPU1 的区域 7 禁止系统 DMA_1 读	RW
6	S1RG6RP	SMPU1 区域 6 读权限	0b0: SMPU1 的区域 6 允许系统 DMA_1 读 0b1: SMPU1 的区域 6 禁止系统 DMA_1 读	RW
5	S1RG5RP	SMPU1 区域 5 读权限	0b0: SMPU1 的区域 5 允许系统 DMA_1 读 0b1: SMPU1 的区域 5 禁止系统 DMA_1 读	RW

位/位域	标记	位名	功能描述	读写
4	S1RG4RP	SMPU1 区域 4 读权限	0b0: SMPU1 的区域 4 允许系统 DMA_1 读 0b1: SMPU1 的区域 4 禁止系统 DMA_1 读	RW
3	S1RG3RP	SMPU1 区域 3 读权限	0b0: SMPU1 的区域 3 允许系统 DMA_1 读 0b1: SMPU1 的区域 3 禁止系统 DMA_1 读	RW
2	S1RG2RP	SMPU1 区域 2 读权限	0b0: SMPU1 的区域 2 允许系统 DMA_1 读 0b1: SMPU1 的区域 2 禁止系统 DMA_1 读	RW
1	S1RG1RP	SMPU1 区域 1 读权限	0b0: SMPU1 的区域 1 允许系统 DMA_1 读 0b1: SMPU1 的区域 1 禁止系统 DMA_1 读	RW
0	S1RG0RP	SMPU1 区域 0 读权限	0b0: SMPU1 的区域 0 允许系统 DMA_1 读 0b1: SMPU1 的区域 0 禁止系统 DMA_1 读	RW

13.4.10 SMPU1 控制寄存器 (MPU_S1CR)

Offset	Bit Position																															
0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								SMPU1E	Reserved			SMPU1ACT	SMPU1BWP	SMPU1BRP	
Access																									RW				RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SMPU1E	SMPU1 使能	0b0: SMPU1 无效 0b1: SMPU1 有效	RW
6:4	Reserved	保留	-	-
3:2	SMPU1ACT	SMPU1 动作选择	设定系统 DMA_1 发生被禁止的访问时的动作 0b00: 读访问读到 0, 写访问忽略 0b01: 读访问读到 0, 写访问忽略; 产生总线错误 0b10: 读访问读到 0, 写访问忽略; 产生不可屏蔽中断 0b11: 复位	RW
1	SMPU1BWP	SMPU1 背景写权限设置	0b0: SMPU1 背景空间允许系统 DMA_1 写 0b1: SMPU1 背景空间禁止系统 DMA_1 写	RW
0	SMPU1BRP	SMPU1 背景读权限设置	0b0: SMPU1 背景空间允许系统 DMA_1 读 0b1: SMPU1 背景空间禁止系统 DMA_1 读	RW



说明

多个区域设定重叠时, 优先级为: 设定的禁止>设定的允许。

13.4.11 SMPU2 区域使能寄存器 (MPU_S2RGE)

Offset	Bit Position																																					
0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																S2RG15E	S2RG14E	S2RG13E	S2RG12E	S2RG11E	S2RG10E	S2RG9E	S2RG8E	S2RG7E	S2RG6E	S2RG5E	S2RG4E	S2RG3E	S2RG2E	S2RG1E	S2RG0E						
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S2RG15E	SMPU2 区域 15 使能	0b0: SMPU2 的区域 15 无效 0b1: SMPU2 的区域 15 有效	RW
14	S2RG14E	SMPU2 区域 14 使能	0b0: SMPU2 的区域 14 无效 0b1: SMPU2 的区域 14 有效	RW
13	S2RG13E	SMPU2 区域 13 使能	0b0: SMPU2 的区域 13 无效 0b1: SMPU2 的区域 13 有效	RW
12	S2RG12E	SMPU2 区域 12 使能	0b0: SMPU2 的区域 12 无效 0b1: SMPU2 的区域 12 有效	RW
11	S2RG11E	SMPU2 区域 11 使能	0b0: SMPU2 的区域 11 无效 0b1: SMPU2 的区域 11 有效	RW
10	S2RG10E	SMPU2 区域 10 使能	0b0: SMPU2 的区域 10 无效 0b1: SMPU2 的区域 10 有效	RW
9	S2RG9E	SMPU2 区域 9 使能	0b0: SMPU2 的区域 9 无效 0b1: SMPU2 的区域 9 有效	RW
8	S2RG8E	SMPU2 区域 8 使能	0b0: SMPU2 的区域 8 无效 0b1: SMPU2 的区域 8 有效	RW
7	S2RG7E	SMPU2 区域 7 使能	0b0: SMPU2 的区域 7 无效 0b1: SMPU2 的区域 7 有效	RW
6	S2RG6E	SMPU2 区域 6 使能	0b0: SMPU2 的区域 6 无效 0b1: SMPU2 的区域 6 有效	RW
5	S2RG5E	SMPU2 区域 5 使能	0b0: SMPU2 的区域 5 无效 0b1: SMPU2 的区域 5 有效	RW
4	S2RG4E	SMPU2 区域 4 使能	0b0: SMPU2 的区域 4 无效 0b1: SMPU2 的区域 4 有效	RW
3	S2RG3E	SMPU2 区域 3 使能	0b0: SMPU2 的区域 3 无效 0b1: SMPU2 的区域 3 有效	RW
2	S2RG2E	SMPU2 区域 2 使能	0b0: SMPU2 的区域 2 无效 0b1: SMPU2 的区域 2 有效	RW
1	S2RG1E	SMPU2 区域 1 使能	0b0: SMPU2 的区域 1 无效 0b1: SMPU2 的区域 1 有效	RW

位/位域	标记	位名	功能描述	读写
0	S2RG0E	SMPU2 区域 0 使能	0b0: SMPU2 的区域 0 无效 0b1: SMPU2 的区域 0 有效	RW

13.4.12 SMPU2 区域写权限寄存器 (MPU_S2RGWP)

Offset	Bit Position																															
0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																S2RG15WP	S2RG14WP	S2RG13WP	S2RG12WP	S2RG11WP	S2RG10WP	S2RG9WP	S2RG8WP	S2RG7WP	S2RG6WP	S2RG5WP	S2RG4WP	S2RG3WP	S2RG2WP	S2RG1WP	S2RG0WP
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S2RG15WP	SMPU2 区域 15 写权限	0b0: SMPU2 的区域 15 允许系统 DMA_2 写 0b1: SMPU2 的区域 15 禁止系统 DMA_2 写	RW
14	S2RG14WP	SMPU2 区域 14 写权限	0b0: SMPU2 的区域 14 允许系统 DMA_2 写 0b1: SMPU2 的区域 14 禁止系统 DMA_2 写	RW
13	S2RG13WP	SMPU2 区域 13 写权限	0b0: SMPU2 的区域 13 允许系统 DMA_2 写 0b1: SMPU2 的区域 13 禁止系统 DMA_2 写	RW
12	S2RG12WP	SMPU2 区域 12 写权限	0b0: SMPU2 的区域 12 允许系统 DMA_2 写 0b1: SMPU2 的区域 12 禁止系统 DMA_2 写	RW
11	S2RG11WP	SMPU2 区域 11 写权限	0b0: SMPU2 的区域 11 允许系统 DMA_2 写 0b1: SMPU2 的区域 11 禁止系统 DMA_2 写	RW
10	S2RG10WP	SMPU2 区域 10 写权限	0b0: SMPU2 的区域 10 允许系统 DMA_2 写 0b1: SMPU2 的区域 10 禁止系统 DMA_2 写	RW
9	S2RG9WP	SMPU2 区域 9 写权限	0b0: SMPU2 的区域 9 允许系统 DMA_2 写 0b1: SMPU2 的区域 9 禁止系统 DMA_2 写	RW
8	S2RG8WP	SMPU2 区域 8 写权限	0b0: SMPU2 的区域 8 允许系统 DMA_2 写 0b1: SMPU2 的区域 8 禁止系统 DMA_2 写	RW
7	S2RG7WP	SMPU2 区域 7 写权限	0b0: SMPU2 的区域 7 允许系统 DMA_2 写 0b1: SMPU2 的区域 7 禁止系统 DMA_2 写	RW
6	S2RG6WP	SMPU2 区域 6 写权限	0b0: SMPU2 的区域 6 允许系统 DMA_2 写 0b1: SMPU2 的区域 6 禁止系统 DMA_2 写	RW
5	S2RG5WP	SMPU2 区域 5 写权限	0b0: SMPU2 的区域 5 允许系统 DMA_2 写 0b1: SMPU2 的区域 5 禁止系统 DMA_2 写	RW
4	S2RG4WP	SMPU2 区域 4 写权限	0b0: SMPU2 的区域 4 允许系统 DMA_2 写 0b1: SMPU2 的区域 4 禁止系统 DMA_2 写	RW
3	S2RG3WP	SMPU2 区域 3 写权限	0b0: SMPU2 的区域 3 允许系统 DMA_2 写 0b1: SMPU2 的区域 3 禁止系统 DMA_2 写	RW

位/位域	标记	位名	功能描述	读写
2	S2RG2WP	SMPU2 区域 2 写权限	0b0: SMPU2 的区域 2 允许系统 DMA_2 写 0b1: SMPU2 的区域 2 禁止系统 DMA_2 写	RW
1	S2RG1WP	SMPU2 区域 1 写权限	0b0: SMPU2 的区域 1 允许系统 DMA_2 写 0b1: SMPU2 的区域 1 禁止系统 DMA_2 写	RW
0	S2RG0WP	SMPU2 区域 0 写权限	0b0: SMPU2 的区域 0 允许系统 DMA_2 写 0b1: SMPU2 的区域 0 禁止系统 DMA_2 写	RW

13.4.13 SMPU2 区域读权限寄存器 (MPU_S2RGRP)

Offset	Bit Position																															
0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																S2RG15RP	S2RG14RP	S2RG13RP	S2RG12RP	S2RG11RP	S2RG10RP	S2RG9RP	S2RG8RP	S2RG7RP	S2RG6RP	S2RG5RP	S2RG4RP	S2RG3RP	S2RG2RP	S2RG1RP	S2RG0RP
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	S2RG15RP	SMPU2 区域 15 读权限	0b0: SMPU2 的区域 15 允许系统 DMA_2 读 0b1: SMPU2 的区域 15 禁止系统 DMA_2 读	RW
14	S2RG14RP	SMPU2 区域 14 读权限	0b0: SMPU2 的区域 14 允许系统 DMA_2 读 0b1: SMPU2 的区域 14 禁止系统 DMA_2 读	RW
13	S2RG13RP	SMPU2 区域 13 读权限	0b0: SMPU2 的区域 13 允许系统 DMA_2 读 0b1: SMPU2 的区域 13 禁止系统 DMA_2 读	RW
12	S2RG12RP	SMPU2 区域 12 读权限	0b0: SMPU2 的区域 12 允许系统 DMA_2 读 0b1: SMPU2 的区域 12 禁止系统 DMA_2 读	RW
11	S2RG11RP	SMPU2 区域 11 读权限	0b0: SMPU2 的区域 11 允许系统 DMA_2 读 0b1: SMPU2 的区域 11 禁止系统 DMA_2 读	RW
10	S2RG10RP	SMPU2 区域 10 读权限	0b0: SMPU2 的区域 10 允许系统 DMA_2 读 0b1: SMPU2 的区域 10 禁止系统 DMA_2 读	RW
9	S2RG9RP	SMPU2 区域 9 读权限	0b0: SMPU2 的区域 9 允许系统 DMA_2 读 0b1: SMPU2 的区域 9 禁止系统 DMA_2 读	RW
8	S2RG8RP	SMPU2 区域 8 读权限	0b0: SMPU2 的区域 8 允许系统 DMA_2 读 0b1: SMPU2 的区域 8 禁止系统 DMA_2 读	RW
7	S2RG7RP	SMPU2 区域 7 读权限	0b0: SMPU2 的区域 7 允许系统 DMA_2 读 0b1: SMPU2 的区域 7 禁止系统 DMA_2 读	RW
6	S2RG6RP	SMPU2 区域 6 读权限	0b0: SMPU2 的区域 6 允许系统 DMA_2 读 0b1: SMPU2 的区域 6 禁止系统 DMA_2 读	RW
5	S2RG5RP	SMPU2 区域 5 读权限	0b0: SMPU2 的区域 5 允许系统 DMA_2 读 0b1: SMPU2 的区域 5 禁止系统 DMA_2 读	RW

位/位域	标记	位名	功能描述	读写
4	S2RG4RP	SMPU2 区域 4 读权限	0b0: SMPU2 的区域 4 允许系统 DMA_2 读 0b1: SMPU2 的区域 4 禁止系统 DMA_2 读	RW
3	S2RG3RP	SMPU2 区域 3 读权限	0b0: SMPU2 的区域 3 允许系统 DMA_2 读 0b1: SMPU2 的区域 3 禁止系统 DMA_2 读	RW
2	S2RG2RP	SMPU2 区域 2 读权限	0b0: SMPU2 的区域 2 允许系统 DMA_2 读 0b1: SMPU2 的区域 2 禁止系统 DMA_2 读	RW
1	S2RG1RP	SMPU2 区域 1 读权限	0b0: SMPU2 的区域 1 允许系统 DMA_2 读 0b1: SMPU2 的区域 1 禁止系统 DMA_2 读	RW
0	S2RG0RP	SMPU2 区域 0 读权限	0b0: SMPU2 的区域 0 允许系统 DMA_2 读 0b1: SMPU2 的区域 0 禁止系统 DMA_2 读	RW

13.4.14 SMPU2 控制寄存器 (MPU_S2CR)

Offset	Bit Position																															
0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								SMPU2E	Reserved			SMPU2ACT	SMPU2BWP	SMPU2BRP	
Access																									RW				RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SMPU2E	SMPU2 使能	0b0: SMPU2 无效 0b1: SMPU2 有效	RW
6:4	Reserved	保留	-	-
3:2	SMPU2ACT	SMPU2 动作选择	设定系统 DMA_2 发生被禁止的访问时的动作 0b00: 读访问读到 0, 写访问忽略 0b01: 读访问读到 0, 写访问忽略; 产生总线错误 0b10: 读访问读到 0, 写访问忽略; 产生不可屏蔽中断 0b11: 复位	RW
1	SMPU2BWP	SMPU2 背景写权限设置	0b0: SMPU2 背景空间允许系统 DMA_2 写 0b1: SMPU2 背景空间禁止系统 DMA_2 写	RW
0	SMPU2BRP	SMPU2 背景读权限设置	0b0: SMPU2 背景空间允许系统 DMA_2 读 0b1: SMPU2 背景空间禁止系统 DMA_2 读	RW



说明

多个区域设定重叠时, 优先级为: 设定的禁止>设定的允许。

13.4.15 FMPU 区域使能寄存器 (MPU_FRGE)

Offset	Bit Position																																	
0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																										FRG7E	FRG6E	FRG5E	FRG4E	FRG3E	FRG2E	FRG1E	FRG0E
Access																											RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	FRG7E	FMPU 区域 7 使能	● 0b0: FMPU 的区域 7 无效 ● 0b1: FMPU 的区域 7 有效	RW
6	FRG6E	FMPU 区域 6 使能	● 0b0: FMPU 的区域 6 无效 ● 0b1: FMPU 的区域 6 有效	RW
5	FRG5E	FMPU 区域 5 使能	● 0b0: FMPU 的区域 5 无效 ● 0b1: FMPU 的区域 5 有效	RW
4	FRG4E	FMPU 区域 4 使能	● 0b0: FMPU 的区域 4 无效 ● 0b1: FMPU 的区域 4 有效	RW
3	FRG3E	FMPU 区域 3 使能	● 0b0: FMPU 的区域 3 无效 ● 0b1: FMPU 的区域 3 有效	RW
2	FRG2E	FMPU 区域 2 使能	● 0b0: FMPU 的区域 2 无效 ● 0b1: FMPU 的区域 2 有效	RW
1	FRG1E	FMPU 区域 1 使能	● 0b0: FMPU 的区域 1 无效 ● 0b1: FMPU 的区域 1 有效	RW
0	FRG0E	FMPU 区域 0 使能	● 0b0: FMPU 的区域 0 无效 ● 0b1: FMPU 的区域 0 有效	RW

13.4.16 FMPU 区域写权限寄存器 (MPU_FRGWP)

Offset	Bit Position																																
0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																								FRG7WP	FRG6WP	FRG5WP	FRG4WP	FRG3WP	FRG2WP	FRG1WP	FRG0WP	
Access																									RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	FRG7WP	FMPU 区域 7 写权限	● 0b0: FMPU 的区域 7 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 7 禁止 USBFS-DMA 写	RW
6	FRG6WP	FMPU 区域 6 写权限	● 0b0: FMPU 的区域 6 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 6 禁止 USBFS-DMA 写	RW
5	FRG5WP	FMPU 区域 5 写权限	● 0b0: FMPU 的区域 5 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 5 禁止 USBFS-DMA 写	RW
4	FRG4WP	FMPU 区域 4 写权限	● 0b0: FMPU 的区域 4 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 4 禁止 USBFS-DMA 写	RW
3	FRG3WP	FMPU 区域 3 写权限	● 0b0: FMPU 的区域 3 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 3 禁止 USBFS-DMA 写	RW
2	FRG2WP	FMPU 区域 2 写权限	● 0b0: FMPU 的区域 2 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 2 禁止 USBFS-DMA 写	RW
1	FRG1WP	FMPU 区域 1 写权限	● 0b0: FMPU 的区域 1 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 1 禁止 USBFS-DMA 写	RW
0	FRG0WP	FMPU 区域 0 写权限	● 0b0: FMPU 的区域 0 允许 USBFS-DMA 写 ● 0b1: FMPU 的区域 0 禁止 USBFS-DMA 写	RW

13.4.17 FMPU 区域读权限寄存器 (MPU_FRGRP)

Offset	Bit Position																															
0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								FRG7RP	FRG6RP	FRG5RP	FRG4RP	FRG3RP	FRG2RP	FRG1RP	FRG0RP
Access																									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	FRG7RP	FMPU 区域 7 读权限	● 0b0: FMPU 的区域 7 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 7 禁止 USBFS-DMA 读	RW
6	FRG6RP	FMPU 区域 6 读权限	● 0b0: FMPU 的区域 6 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 6 禁止 USBFS-DMA 读	RW
5	FRG5RP	FMPU 区域 5 读权限	● 0b0: FMPU 的区域 5 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 5 禁止 USBFS-DMA 读	RW
4	FRG4RP	FMPU 区域 4 读权限	● 0b0: FMPU 的区域 4 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 4 禁止 USBFS-DMA 读	RW
3	FRG3RP	FMPU 区域 3 读权限	● 0b0: FMPU 的区域 3 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 3 禁止 USBFS-DMA 读	RW
2	FRG2RP	FMPU 区域 2 读权限	● 0b0: FMPU 的区域 2 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 2 禁止 USBFS-DMA 读	RW
1	FRG1RP	FMPU 区域 1 读权限	● 0b0: FMPU 的区域 1 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 1 禁止 USBFS-DMA 读	RW
0	FRG0RP	FMPU 区域 0 读权限	● 0b0: FMPU 的区域 0 允许 USBFS-DMA 读 ● 0b1: FMPU 的区域 0 禁止 USBFS-DMA 读	RW

13.4.18 FMPU 控制寄存器 (MPU_FCR)

Offset	Bit Position																															
0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										FMPUE	Reserved		FMPUACT	FMPUBWP	FMPUBRP
Access																											RW			RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	FMPUE	FMPU 使能	0b0: FMPU 无效 0b1: FMPU 有效	RW
6:4	Reserved	保留	-	-
3:2	FMPUACT	FMPU 动作选择	设定 USBFS-DMA 发生被禁止的访问时的动作 0b00: 读访问读到 0, 写访问忽略 0b01: 读访问读到 0, 写访问忽略; 产生总线错误 0b10: 读访问读到 0, 写访问忽略; 产生不可屏蔽中断 0b11: 复位	RW
1	FMPUBWP	FMPU 背景写权限设置	0b0: FMPU 背景空间允许 USBFS-DMA 写 0b1: FMPU 背景空间禁止 USBFS-DMA 写	RW
0	FMPUBRP	FMPU 背景读权限设置	0b0: FMPU 背景空间允许 USBFS-DMA 读 0b1: FMPU 背景空间禁止 USBFS-DMA 读	RW



说明

多个区域设定重叠时, 优先级为: 设定的禁止>设定的允许。

13.4.19 HMPU 区域使能寄存器 (MPU_HRGE)

Offset	Bit Position																															
0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								HRG7E	HRG6E	HRG5E	HRG4E	HRG3E	HRG2E	HRG1E	HRG0E
Access																									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	HRG7E	HMPU 区域 7 使能	0b0: HMPU 的区域 7 无效 0b1: HMPU 的区域 7 有效	RW
6	HRG6E	HMPU 区域 6 使能	0b0: HMPU 的区域 6 无效 0b1: HMPU 的区域 6 有效	RW
5	HRG5E	HMPU 区域 5 使能	0b0: HMPU 的区域 5 无效 0b1: HMPU 的区域 5 有效	RW
4	HRG4E	HMPU 区域 4 使能	0b0: HMPU 的区域 4 无效 0b1: HMPU 的区域 4 有效	RW
3	HRG3E	HMPU 区域 3 使能	0b0: HMPU 的区域 3 无效 0b1: HMPU 的区域 3 有效	RW
2	HRG2E	HMPU 区域 2 使能	0b0: HMPU 的区域 2 无效 0b1: HMPU 的区域 2 有效	RW
1	HRG1E	HMPU 区域 1 使能	0b0: HMPU 的区域 1 无效 0b1: HMPU 的区域 1 有效	RW
0	HRG0E	HMPU 区域 0 使能	0b0: HMPU 的区域 0 无效 0b1: HMPU 的区域 0 有效	RW

13.4.20 HMPU 区域写权限寄存器 (MPU_HRGWP)

Offset	Bit Position																															
0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								HRG7WP	HRG6WP	HRG5WP	HRG4WP	HRG3WP	HRG2WP	HRG1WP	HRG0WP
Access																									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	HRG7WP	HMPU 区域 7 写权限	● 0b0: HMPU 的区域 7 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 7 禁止 USBHS-DMA 写	RW
6	HRG6WP	HMPU 区域 6 写权限	● 0b0: HMPU 的区域 6 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 6 禁止 USBHS-DMA 写	RW
5	HRG5WP	HMPU 区域 5 写权限	● 0b0: HMPU 的区域 5 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 5 禁止 USBHS-DMA 写	RW
4	HRG4WP	HMPU 区域 4 写权限	● 0b0: HMPU 的区域 4 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 4 禁止 USBHS-DMA 写	RW
3	HRG3WP	HMPU 区域 3 写权限	● 0b0: HMPU 的区域 3 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 3 禁止 USBHS-DMA 写	RW
2	HRG2WP	HMPU 区域 2 写权限	● 0b0: HMPU 的区域 2 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 2 禁止 USBHS-DMA 写	RW
1	HRG1WP	HMPU 区域 1 写权限	● 0b0: HMPU 的区域 1 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 1 禁止 USBHS-DMA 写	RW
0	HRG0WP	HMPU 区域 0 写权限	● 0b0: HMPU 的区域 0 允许 USBHS-DMA 写 ● 0b1: HMPU 的区域 0 禁止 USBHS-DMA 写	RW

13.4.21 HMPU 区域读权限寄存器 (MPU_HRGRP)

Offset	Bit Position																															
0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								HRG7RP	HRG6RP	HRG5RP	HRG4RP	HRG3RP	HRG2RP	HRG1RP	HRG0RP
Access																									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	HRG7RP	HMPU 区域 7 读权限	● 0b0: HMPU 的区域 7 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 7 禁止 USBHS-DMA 读	RW
6	HRG6RP	HMPU 区域 6 读权限	● 0b0: HMPU 的区域 6 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 6 禁止 USBHS-DMA 读	RW
5	HRG5RP	HMPU 区域 5 读权限	● 0b0: HMPU 的区域 5 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 5 禁止 USBHS-DMA 读	RW
4	HRG4RP	HMPU 区域 4 读权限	● 0b0: HMPU 的区域 4 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 4 禁止 USBHS-DMA 读	RW
3	HRG3RP	HMPU 区域 3 读权限	● 0b0: HMPU 的区域 3 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 3 禁止 USBHS-DMA 读	RW
2	HRG2RP	HMPU 区域 2 读权限	● 0b0: HMPU 的区域 2 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 2 禁止 USBHS-DMA 读	RW
1	HRG1RP	HMPU 区域 1 读权限	● 0b0: HMPU 的区域 1 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 1 禁止 USBHS-DMA 读	RW
0	HRG0RP	HMPU 区域 0 读权限	● 0b0: HMPU 的区域 0 允许 USBHS-DMA 读 ● 0b1: HMPU 的区域 0 禁止 USBHS-DMA 读	RW

13.4.22 HMPU 控制寄存器 (MPU_HCR)

Offset	Bit Position																															
0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								HMPUE	Reserved		HMPUACT	HMPUBWP	HMPUBRP		
Access																									RW			RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	HMPUE	HMPU 使能	0b0: HMPU 无效 0b1: HMPU 有效	RW
6:4	Reserved	保留	-	-
3:2	HMPUACT	HMPU 动作选择	设定 USBHS-DMA 发生被禁止的访问时的动作 0b00: 读访问读到 0, 写访问忽略 0b01: 读访问读到 0, 写访问忽略; 产生总线错误 0b10: 读访问读到 0, 写访问忽略; 产生不可屏蔽中断 0b11: 复位	RW
1	HMPUBWP	HMPU 背景写权限设置	0b0: HMPU 背景空间允许 USBHS-DMA 写 0b1: HMPU 背景空间禁止 USBHS-DMA 写	RW
0	HMPUBRP	HMPU 背景读权限设置	0b0: HMPU 背景空间允许 USBHS-DMA 读 0b1: HMPU 背景空间禁止 USBHS-DMA 读	RW

 **说明**

多个区域设定重叠时, 优先级为: 设定的禁止>设定的允许。

13.4.23 EMPU 区域使能寄存器 (MPU_ERGE)

Offset	Bit Position																																
0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																								ERG7E	ERG6E	ERG5E	ERG4E	ERG3E	ERG2E	ERG1E	ERG0E	
Access																									RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	ERG7E	EMPU 区域 7 使能	● 0b0: EMPU 的区域 7 无效 ● 0b1: EMPU 的区域 7 有效	RW
6	ERG6E	EMPU 区域 6 使能	● 0b0: EMPU 的区域 6 无效 ● 0b1: EMPU 的区域 6 有效	RW
5	ERG5E	EMPU 区域 5 使能	● 0b0: EMPU 的区域 5 无效 ● 0b1: EMPU 的区域 5 有效	RW
4	ERG4E	EMPU 区域 4 使能	● 0b0: EMPU 的区域 4 无效 ● 0b1: EMPU 的区域 4 有效	RW
3	ERG3E	EMPU 区域 3 使能	● 0b0: EMPU 的区域 3 无效 ● 0b1: EMPU 的区域 3 有效	RW
2	ERG2E	EMPU 区域 2 使能	● 0b0: EMPU 的区域 2 无效 ● 0b1: EMPU 的区域 2 有效	RW
1	ERG1E	EMPU 区域 1 使能	● 0b0: EMPU 的区域 1 无效 ● 0b1: EMPU 的区域 1 有效	RW
0	ERG0E	EMPU 区域 0 使能	● 0b0: EMPU 的区域 0 无效 ● 0b1: EMPU 的区域 0 有效	RW

13.4.24 EMPU 区域写权限寄存器 (MPU_ERGWP)

Offset	Bit Position																															
0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								ERG7WP	ERG6WP	ERG5WP	ERG4WP	ERG3WP	ERG2WP	ERG1WP	ERG0WP
Access																									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	ERG7WP	EMPU 区域 7 写权限	● 0b0: EMPU 的区域 7 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 7 禁止 ETH-DMA 写	RW
6	ERG6WP	EMPU 区域 6 写权限	● 0b0: EMPU 的区域 6 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 6 禁止 ETH-DMA 写	RW
5	ERG5WP	EMPU 区域 5 写权限	● 0b0: EMPU 的区域 5 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 5 禁止 ETH-DMA 写	RW
4	ERG4WP	EMPU 区域 4 写权限	● 0b0: EMPU 的区域 4 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 4 禁止 ETH-DMA 写	RW
3	ERG3WP	EMPU 区域 3 写权限	● 0b0: EMPU 的区域 3 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 3 禁止 ETH-DMA 写	RW
2	ERG2WP	EMPU 区域 2 写权限	● 0b0: EMPU 的区域 2 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 2 禁止 ETH-DMA 写	RW
1	ERG1WP	EMPU 区域 1 写权限	● 0b0: EMPU 的区域 1 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 1 禁止 ETH-DMA 写	RW
0	ERG0WP	EMPU 区域 0 写权限	● 0b0: EMPU 的区域 0 允许 ETH-DMA 写 ● 0b1: EMPU 的区域 0 禁止 ETH-DMA 写	RW

13.4.25 EMPU 区域读权限寄存器 (MPU_ERGRP)

Offset	Bit Position																																	
0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																										ERG7RP	ERG6RP	ERG5RP	ERG4RP	ERG3RP	ERG2RP	ERG1RP	ERG0RP
Access																											RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	ERG7RP	EMPU 区域 7 读权限	● 0b0: EMPU 的区域 7 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 7 禁止 ETH-DMA 读	RW
6	ERG6RP	EMPU 区域 6 读权限	● 0b0: EMPU 的区域 6 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 6 禁止 ETH-DMA 读	RW
5	ERG5RP	EMPU 区域 5 读权限	● 0b0: EMPU 的区域 5 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 5 禁止 ETH-DMA 读	RW
4	ERG4RP	EMPU 区域 4 读权限	● 0b0: EMPU 的区域 4 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 4 禁止 ETH-DMA 读	RW
3	ERG3RP	EMPU 区域 3 读权限	● 0b0: EMPU 的区域 3 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 3 禁止 ETH-DMA 读	RW
2	ERG2RP	EMPU 区域 2 读权限	● 0b0: EMPU 的区域 2 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 2 禁止 ETH-DMA 读	RW
1	ERG1RP	EMPU 区域 1 读权限	● 0b0: EMPU 的区域 1 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 1 禁止 ETH-DMA 读	RW
0	ERG0RP	EMPU 区域 0 读权限	● 0b0: EMPU 的区域 0 允许 ETH-DMA 读 ● 0b1: EMPU 的区域 0 禁止 ETH-DMA 读	RW

13.4.26 EMPU 控制寄存器 (MPU_ECR)

Offset	Bit Position																															
0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EMPUE	Reserved				EMPUACT	EMPUBWP	EMPUBRP
Access																									RW					RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	EMPUE	EMPU 使能	0b0: EMPU 无效 0b1: EMPU 有效	RW
6:4	Reserved	保留	-	-
3:2	EMPUACT	EMPU 动作选择	设定 ETH-DMA 发生被禁止的访问时的动作 0b00: 读访问读到 0, 写访问忽略 0b01: 读访问读到 0, 写访问忽略; 产生总线错误 0b10: 读访问读到 0, 写访问忽略; 产生不可屏蔽中断 0b11: 复位	RW
1	EMPUBWP	EMPU 背景写权限设置	0b0: EMPU 背景空间允许 ETH-DMA 写 0b1: EMPU 背景空间禁止 ETH-DMA 写	RW
0	EMPUBRP	EMPU 背景读权限设置	0b0: EMPU 背景空间允许 ETH-DMA 读 0b1: EMPU 背景空间禁止 ETH-DMA 读	RW

说明

多个区域设定重叠时, 优先级为: 设定的禁止>设定的允许。

14 键盘扫描控制模块 (KEYSCAN)

14.1 简介

本产品搭载键盘扫描控制模块 (KEYSCAN) 1 个单元。KEYSCAN 模块支持键盘阵列 (行和列) 扫描, 列是由独立的扫描输出 KEYOUT_m ($m=0\sim7$) 驱动, 而行 KEYIN_n ($n=0\sim15$) 则作为 EIRQ_n ($n=0\sim15$) 输入被检测。本模块通过行扫描查询法实现按键识别功能。

14.2 主要特性

- EIRQ₀~EIRQ₁₅ 可以独立选择作为键盘阵列的行输入
- KEYOUT 可以通过寄存器选择
- 间隔一定时间依次输出低电平来扫描键盘阵列
- 扫描时间可以设置
- EIRQ 中断检出时停止扫描, 根据 SSR.INDEX 值和 EIRQ 中断标志 (INTC_EIFR.EIF) 来定位被按下的键

14.3 功能说明

14.3.1 功能框图

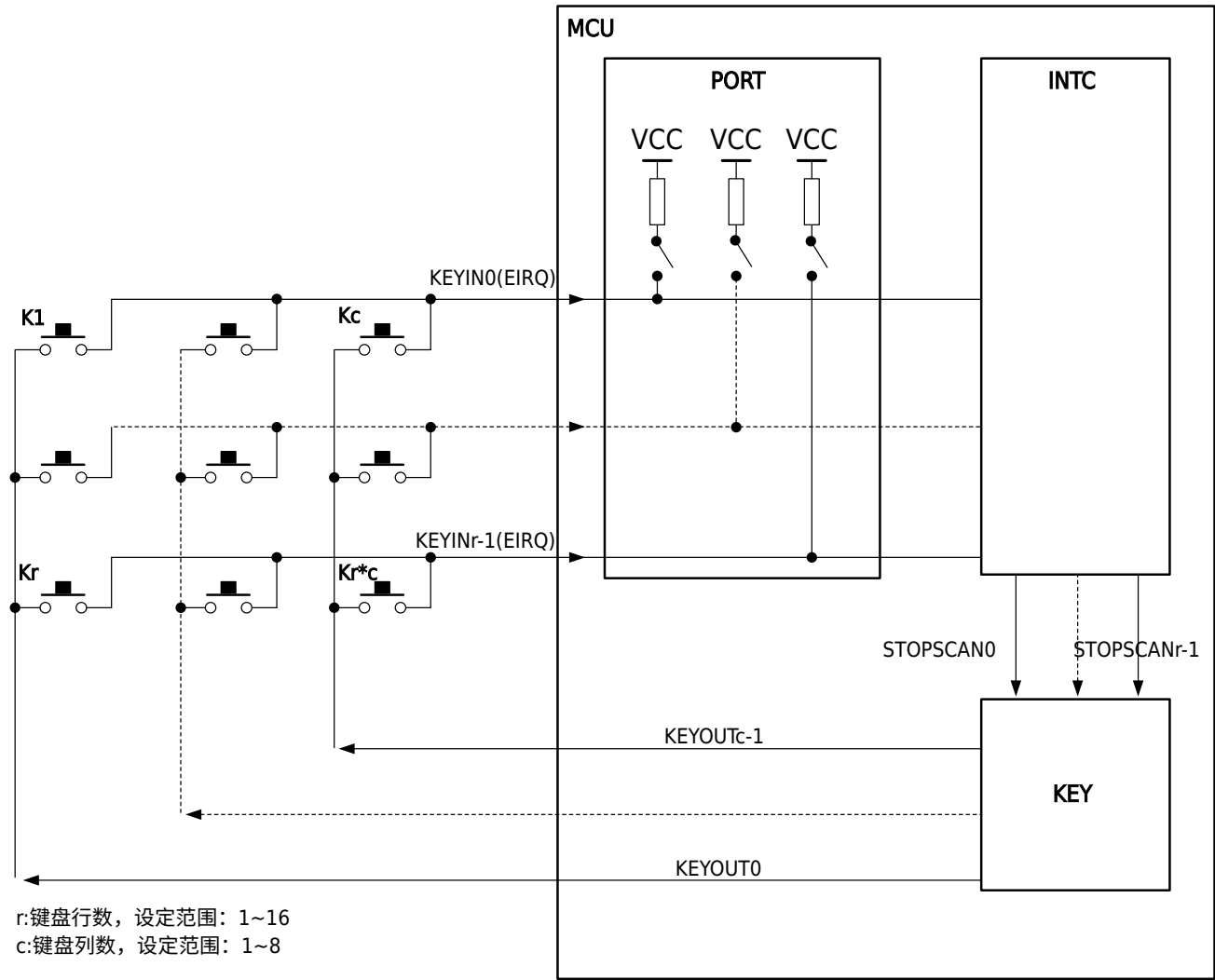


图 14-1 KEYSCAN 系统框图

表 14-1 KEYSCAN 管脚说明

管脚名	方向	功能描述
KEYINn (n=0~15)	输入	键盘行输入信号
KEYOUTm (m=0~7)	输出	键盘列输出信号

14.3.2 按键识别功能

当有键被按下时，键盘的行和列短接，行产生下降沿，从而产生 EIRQ 中断标志，通过比较中断标志位 (INTC_EIFR.EIF) 和 SSR.INDEX[2:0]的值来定位当前被按下的键。

通过寄存器 SCR.KEYINSEL[15:0]，KEYIN 可以从 EIRQ0~EIRQ15 中独立选择，通过寄存器 SCR.KEYOUTSEL[2:0]可以选择使用的 KEYOUT 管脚，从而可以灵活的选择键盘的行列数目，最大可以支持 16 行*8 列的键盘阵列。

14.3.3 键盘扫描功能

键盘扫描功能为：不断循环的给键盘阵列的列输出低电平，从而当有键被按下时，会产生对应的 EIRQ 中断标志。

当 KEYSCAN_SER.SEN 设置为 1 后，KEYOUT0 输出低电平，KEYOUT1~KEYOUTn（n 通过 KEYSCAN_SCR.KEYOUTSEL[2:0]设定）为 Hi-Z，经过 KEYSCAN_SCR.T_LLEVEL[4:0]设定的时间后，KEYOUT0~KEYOUTn 管脚全部为 Hi-Z，经过 KEYSCAN_SCR.T_HIZ[2:0]设定的时间后，KEYOUT1 输出低电平，其余 KEYOUT 管脚为 Hi-Z，依次类推循环。有键被按下并产生 EIRQ 中断标志时，键盘扫描功能停止，对应的中断标志清零后，扫描自动重启。

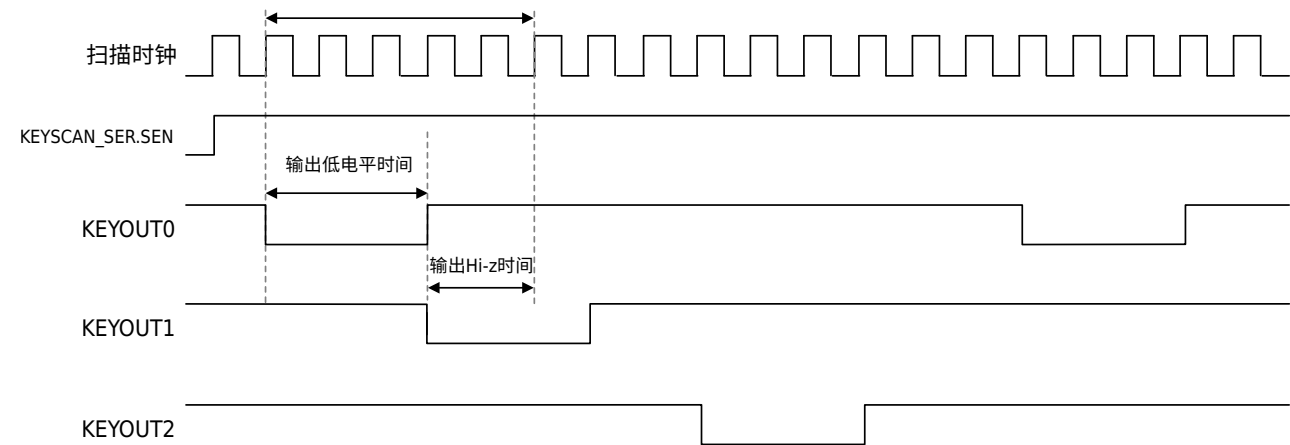


图 14-2 键盘扫描功能示意图

14.4 寄存器

14.4.1 寄存器总表

基地址：0x40050C00

表 14-2 键盘扫描控制模块 (KEYSCAN) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	KEYSCAN_SCR	扫描控制寄存器	32
0x04	KEYSCAN_SER	扫描使能寄存器	32
0x08	KEYSCAN_SSR	扫描状态寄存器	32

14.4.2 扫描控制寄存器 (KEYSCAN_SCR)



说明

该寄存器只在 KEYCAN_SER.SEN=0 时设定有效。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	T_HIZ		T_LLEVEL				Reserved		CKSEL		Reserved		KEYOUTSEL		KEYINSEL																	
Access	RW		RW						RW				RW		RW																	

位/位域	标记	位名	功能描述	读写
31:29	T_HIZ	输出 Hi-Z 时间	KEYOUT 输出 Hi-Z 时间（扫描时钟数） 0b000: 4 个扫描时钟数 0b001: 8 个扫描时钟数 0b010: 16 个扫描时钟数 0b011: 32 个扫描时钟数 0b100: 64 个扫描时钟数 0b101: 256 个扫描时钟数 0b110: 512 个扫描时钟数 0b111: 1024 个扫描时钟数 扫描周期 = 输出低电平时间 + 输出 Hi-Z 时间	RW
28:24	T_LLEVEL	输出低电平时间	KEYOUT 输出低电平时间（扫描时钟数） 输出低电平时间 = 2^{T_LLEVEL} 个扫描时钟数 扫描周期 = 输出低电平时间 + 输出 Hi-Z 时间 说明 0b00000 和 0b00001 禁止设定，最大可以设定值为 0b11000。	RW
23:22	Reserved	保留	-	-
21:20	CKSEL	扫描时钟源选择位	0b00: 系统时钟 HCLK 0b01: 内部低速振荡器 LRC 0b10: 外部低速振荡器 XTAL32 0b11: 禁止设置	RW
19	Reserved	保留	-	-
18:16	KEYOUTSEL	KEYOUT 输出选择位	0b000: 禁止设置 0b001: KEYOUT0~KEYOUT1 0b010: KEYOUT0~KEYOUT2 0b011: KEYOUT0~KEYOUT3 0b100: KEYOUT0~KEYOUT4 0b101: KEYOUT0~KEYOUT5 0b110: KEYOUT0~KEYOUT6 0b111: KEYOUT0~KEYOUT7	RW

位/位域	标记	位名	功能描述	读写
15:0	KEYINSEL	行输入选择位	对应 KEYINSEL15~KEYINSEL0，选择的行作为键盘阵列的行使用，并作为 EIRQn（n=15~0）被检测： ● 0b0：不用作键盘阵列的行使用 ● 0b1：用作键盘阵列的行使用	RW

14.4.3 扫描使能寄存器 (KEYSCAN_SER)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															SEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	SEN	扫描使能位	● 0b0：扫描禁止 ● 0b1：扫描使能	RW

14.4.4 扫描状态寄存器 (KEYSCAN_SSR)

 说明

该寄存器只在 KEYCAN_SER.SEN=1 时读取出来的数据才有意义。

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														INDEX	
Access																															RO	

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	INDEX	当前工作的 SCAN 管脚索引位	● 0b000：当前工作的 SCAN 管脚为 KEYOUT0 ● 0b001：当前工作的 SCAN 管脚为 KEYOUT1 ● 0b010：当前工作的 SCAN 管脚为 KEYOUT2 ● 0b011：当前工作的 SCAN 管脚为 KEYOUT3 ● 0b100：当前工作的 SCAN 管脚为 KEYOUT4 ● 0b101：当前工作的 SCAN 管脚为 KEYOUT5 ● 0b110：当前工作的 SCAN 管脚为 KEYOUT6 ● 0b111：当前工作的 SCAN 管脚为 KEYOUT7	RO

14.5 注意事项

本模块驱动键盘列，而键盘行检测则由中断控制模块 (INTC) 的外部 EIRQ 功能实现，EIRQ 需要选择下降沿检测，并开启数字滤波功能，设定合适的滤波时间。

STOP 模式下如果使用此功能，则需设定好扫描相关参数后，选择内部低速振荡器 LRC 或者外部低速振荡器 XTAL32 时钟作为扫描时钟。

如果使用芯片内部上拉电阻，则请参见《数据手册》中“电气特性-I/O 端口特性”章节相关描述，选择合适的扫描时间以及滤波时间。

15 内部时钟校准器 (CTC)

15.1 简介

内部时钟校准器 (Clock Trimming Controller, 以下称 CTC) 可以自动校准内部高速振荡器 (HRC)。由于工作环境的影响 HRC 的频率可能会产生偏差, 用 CTC 基于外部高精度参考时钟, 采用硬件方式自动调整 HRC 的频率可以得到一个精准的 HRC 时钟。

15.2 主要特性

- 三个外部参考时钟源: XTAL、XTAL32、CTCREF
- 用于频率测量并具有重载功能的 16 位校准计数器
- 用于频率校准的 8 位校准偏差值和 6 位校准值
- 用于提示校准失败的错误中断

15.3 功能说明

15.3.1 功能框图

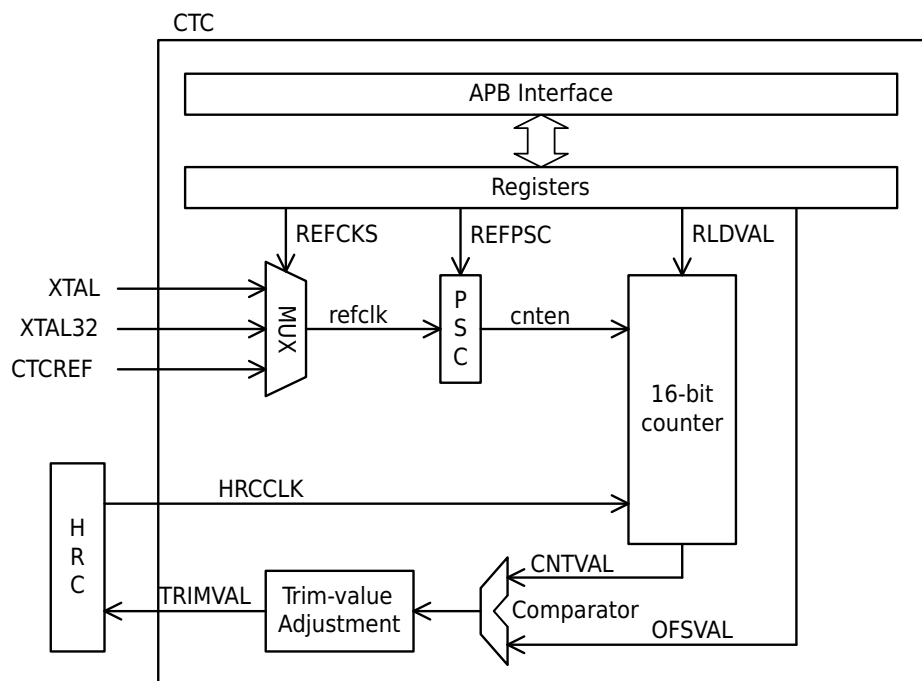


图 15-1 CTC 功能框图

15.3.2 参考时钟

CTC 有以下三个参考时钟, 通过 CTC_CR1 寄存器的 REFCKS[1:0]位选择。

- 外部高速时钟 (XTAL)
- 外部低速时钟 (XTAL32)
- 外部基准时钟 (CTCREF)

HRC 的抖动和校准过程中的计数误差都会影响校准精度, 可以通过设置 CTC_CR1 寄存器中的 REFPSC[2:0]位为参考时钟选择合适的分频来减小误差。误差的计算公式如下:

$$TE = \frac{0.099\%}{\sqrt{(F_{hrc} \div F_{ref}) \times PSC_{ref} \div 10^5}} + \frac{4}{(F_{hrc} \div F_{ref}) \times PSC_{ref}}$$

其中：

- F_{hrc} 表示 HRC 的目标频率，单位 Hz；
- F_{ref} 表示参考时钟的频率，单位 Hz；
- PSC_{ref} 表示参考时钟分频数。

下面两个表分别列举了 HRC 目标频率为 16MHz 和 20MHz 时使用不同频率的参考时钟并在不同分频条件下校准时的误差值，供快速查阅。



说明

- “不可设”表示该设定超出了校准计数器的范围，无法完成校准。
- “不推荐”表示该设定误差超过 2%，故不推荐。

表 15-1 HRC 目标频率为 20MHz 时的测量误差

PSC _{ref}	F _{ref}				
	32.768kHz	4MHz	8MHz	12MHz	24MHz
8	0.530%	不推荐	不推荐	不推荐	不推荐
32	0.244%	不推荐	不推荐	不推荐	不推荐
128	不可设	1.863%	3.000%	不推荐	不推荐
256	不可设	1.188%	1.863%	2.453%	不推荐
512	不可设	0.775%	1.188%	1.540%	2.453%
1024	不可设	0.516%	0.775%	0.992%	1.540%
2048	不可设	0.348%	0.516%	0.653%	0.992%
4096	不可设	0.238%	0.348%	0.437%	0.653%

表 15-2 HRC 目标频率为 16MHz 时的测量误差

PSC _{ref}	F _{ref}				
	32.768kHz	4MHz	8MHz	12MHz	24MHz
8	0.603%	不推荐	不推荐	不推荐	不推荐
32	0.276%	不推荐	不推荐	不推荐	不推荐
128	不可设	2.165%	3.519%	不推荐	不推荐
256	不可设	1.369%	2.165%	2.866%	不推荐
512	不可设	0.887%	1.369%	1.784%	2.866%
1024	不可设	0.587%	0.887%	1.140%	1.784%
2048	不可设	0.395%	0.587%	0.746%	1.140%
4096	不可设	0.269%	0.395%	0.497%	0.746%

15.3.3 频率校准

在 CTC_CR1 寄存器的 CTCEN 位写为 1 后，16 位校准计数器在参考时钟的触发下从 CTC_CR2 寄存器中重载 RLDVAL 并开始向下计数，计数时钟由 HRC 提供。计数开始后，如果参考时钟触发计数停止，那么 16 位校准计数器将停止计数。否则 16 位校准计数器将持续计数并在计数值下溢为 0xFFFF 时停止。

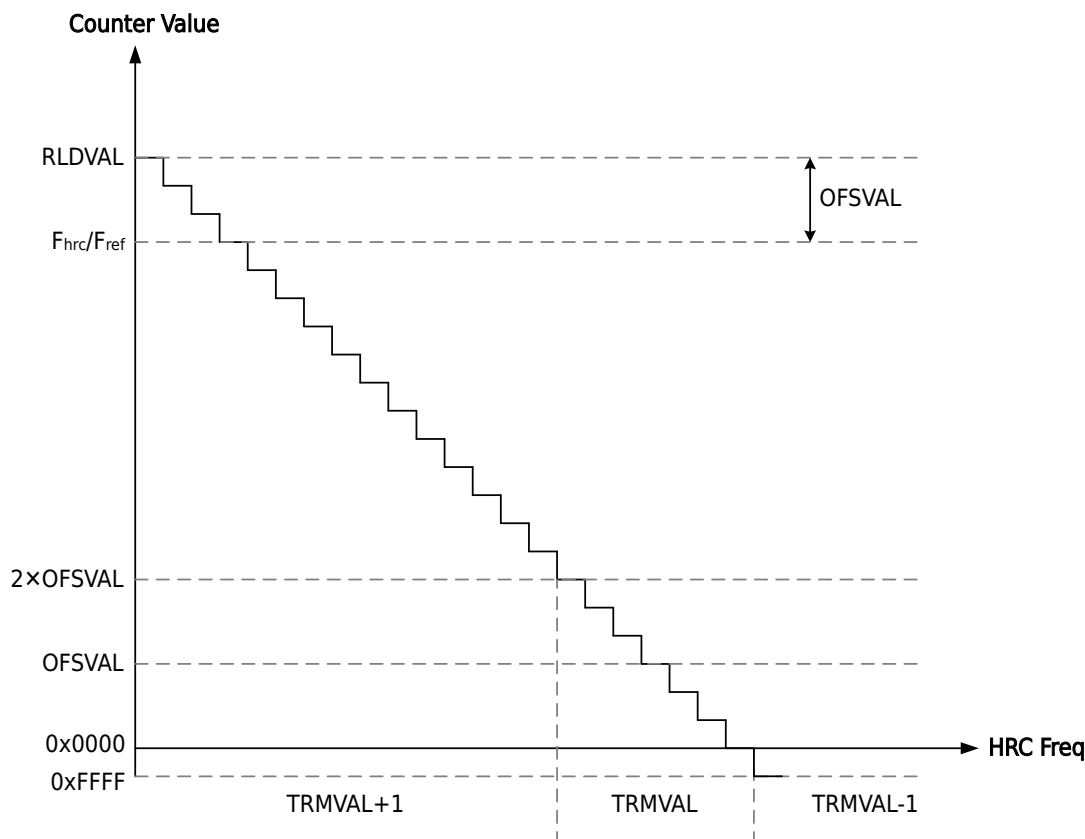


图 15-2 CTC 校准示意图

16 位校准计数器停止计数后，HRC 频率校准开始。由于在设定校准计数重载值 RLDVAL 时加上了校准目标偏差值 OFSVAL，因此如果 16 位校准计数器的值大于 $2 \times \text{OFSVAL}$ 且没有发生下溢，说明当前频率比期望频率慢，需要增大 CTC_CR1 寄存器中 TRMVAL 的值来提高频率。反之，如果 16 位校准计数器的值发生下溢，则说明当前频率比期望频率快，需要减小 TRMVAL 的值以降低频率。而如果 16 位校准计数器的值小于 $2 \times \text{OFSVAL}$ ，说明当前频率处于期望频率范围内，就不需要调整 TRMVAL 的值。

- Counter > ($\text{OFSVAL} \times 2$) 时停止计数，TRMVAL 的值自动加 1。
- $0 \leq \text{Counter} \leq (\text{OFSVAL} \times 2)$ 时停止计数，TRMVAL 的值不变，CTC_STR 寄存器中的 TRIMOK 位自动变为 1。
- Counter = 0xFFFF，CTC_CR1 寄存器中的 TRMVAL 的值自动减 1。

在每次 TRMVAL 调整后会自动开始下一次校准，在软件将 CTCEN 清零或者发生校准错误时停止校准。

15.3.4 中断

TRMVAL 的中间值是 0x00。如果自动加 1 的结果大于 0x1F，TRMVAL 的值将保持 0x1F 并发生校准上溢错误，这时 CTC_STR 寄存器的 TRMOVF 位自动变为 1。反之，如果自动减 1 的结果小于 0x20，TRMVAL 的值将保持 0x20 并发生校准下溢错误，这时 CTC_STR 寄存器中的 TRMUDF 位自动变为 1。发生校准上溢错误或下溢错误时校准会自动停止 (CTCEN 自动清零)，如果 CTC_CR1 寄存器的 ERRIE 位为 1，就会发生一次校准错误中断请求。具体设定方法请参见《中断控制器 (INTC)》章节。

15.4 操作示例

15.4.1 使用 XTAL32 作为参考时钟将 HRC 校准到 20MHz 且校准目标偏差小于 0.5%

XTAL32 设为 32 分频，经查表 15-1 可知校准误差为 0.244%，小于 0.5%的校准目标偏差，方案可行。

1. 设定寄存器 CTC_CR2 = 0x4CAD0062，其中：
 - $OFSVAL = (20 \div 0.032768) \times 32 \times 0.5\% \approx 98$ (0x62)
 - $RLDVAL = (20 \div 0.032768) \times 32 + 98 \approx 19629$ (0x4CAD)
2. 设定寄存器 CTC_CR1 = 0x000000E1，其中：
 - TRMVAL=0x00 (从中间值开始校准)
 - CTCEN=1 (启动 CTC)
 - ERRIE=1 (允许错误中断)
 - REFCKS=0b10 (选择 XTAL32 作为参考时钟)
 - REFPSC=0b001 (选择参考时钟的 32 分频)
3. 一段时间后，确认 CTC_STR.TRIMOK 位的状态：
 - 如果 TRIMOK=1，CTCEN 写 0 停止 CTC；
 - 如果 TRIMOK=0，继续等待一段时间后继续本步骤。
4. 确认 CTC_STR.CTCBSY=0，CTC 停止，校准完成。

途中如果发生错误中断，则表示由于超出可校准范围发生了校准上溢或者校准下溢，CTCEN 位会自动清零，校准停止。这时请修正 OFSVAL，RLDVAL 以及参考时钟设定后再次尝试校准。



说明

频率校准是通过 CTC_CR1 寄存器中的 CTCEN 位来启动或停止的。因此，启动频率校准后，请勿改变任何其他寄存器的设定值。如需修改设置，请先将 CTCEN 位写 0，并确认 CTC_STR 寄存器中的 CTCBSY 位为 0 后再进行修改。

15.5 寄存器

15.5.1 寄存器总表

基地址：0x40049C00

表 15-3 内部时钟校准器 (CTC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	CTC_CR1	时钟校准控制寄存器 1	32
0x04	CTC_CR2	时钟校准控制寄存器 2	32
0x08	CTC_STR	时钟校准状态寄存器	32

15.5.2 时钟校准控制寄存器 1 (CTC_CR1)

Offset	Bit Position																																				
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved										TRMVAL						Reserved										CTCEN		ERRIE		REFCKS		Reserved		REFPSC		
Access											RW																RW		RW		RW						RW

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21:16	TRMVAL	HRC 校准值	TRMVAL 为 6 位有符号校准值，中间值是 0。 ● 0x20: -32 ● 0x21: -31 ● ● 0x3F: -1 ● 0x00: 0 (中间值) ● 0x01: +1 ● ● 0x1E: +30 ● 0x1F: +31 CTCEN 位写 1 时 TRMVAL 为只读，由硬件自动修改。如需修改请将 CTCEN 位写 0 并确认 CTC_STR.CTCBSY 位为 0 后再修改。 在任何情况下 HRC 最终的频率校准值均为出厂校准值加上 TRMVAL 的结果。因此如果出厂校准值下的频率已满足要求无需校准，请将 TRMVAL 设为 0x00。	RW
15:8	Reserved	保留	-	-
7	CTCEN	校准使能	用于启动频率校准。 ● 0b0: 停止频率校准 ● 0b1: 启动频率校准  说明 ● 置位条件：软件置位 ● 清零条件： ▶ 软件清零 ▶ 校准发生上溢或者下溢时自动清零	RW
6	ERRIE	错误中断允许	用于选择发生校准上溢或者下溢时是否产生错误中断。 ● 0b0: 禁止发生错误中断 ● 0b1: 允许发生错误中断	RW

位/位域	标记	位名	功能描述	读写
5:4	REFCKS	参考时钟选择	用于为频率校准选择参考时钟源。 ● 0b00/0b01: 选择 CTCREF 作为参考时钟 ● 0b10: 选择 XTAL32 作为参考时钟 ● 0b11: 选择 XTAL 作为参考时钟 CTCEN 位为 1 时禁止修改 REFCKS 的值。如需修改请将 CTCEN 位写 0 并确认 CTC_STR.CTCBSY 位为 0 后再修改。	RW
3	Reserved	保留	-	-
2:0	REFPSC	参考时钟分频数选择	用于选择参考时钟的分频数。 ● 0b000: 8 分频 ● 0b001: 32 分频 ● 0b010: 128 分频 ● 0b011: 256 分频 ● 0b100: 512 分频 ● 0b101: 1024 分频 ● 0b110: 2048 分频 ● 0b111: 4096 分频 CTCEN 位为 1 时禁止修改 REFPSC 的值。如需修改请将 CTCEN 位写 0 并确认 CTC_STR.CTCBSY 位为 0 后再修改。	RW

15.5.3 时钟校准控制寄存器 2 (CTC_CR2)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RLDVAL																Reserved								OFSVAL							
Access	RW																								RW							

位/位域	标记	位名	功能描述	读写
31:16	RLDVAL	校准计数重载值	RLDVAL 设定的是 16 位校准计数器的重载值 $RLDVAL = (F_{hrc} \div (F_{ref} \div PSC_{ref})) + OFSVAL$ ● F_{hrc}: HRC 目标频率; ● F_{ref}: 参考时钟频率; ● OFSVAL: 校准计数偏差值; ● PSC_{ref}: 参考时钟分频数。  说明 CTCEN 位为 1 时禁止修改 RLDVAL 的值。如需修改请将 CTCEN 位写 0 并确认 CTC_STR.CTCBSY 位为 0 后再修改。	RW
15:8	Reserved	保留	-	-
7:0	OFSVAL	校准计数偏差值	OFSVAL 设定的是 HRC 在校准时允许的偏差值。 $OFSVAL = (F_{hrc} \div (F_{ref} \div PSC_{ref})) \times TA$ ● F_{hrc}: HRC 目标频率; ● F_{ref}: 参考时钟频率; ● TA: 校准精度, 例如 0.5%; ● PSC_{ref}: 参考时钟分频数。  说明 CTCEN 位为 1 时禁止修改 OFSVAL 的值。如需修改请将 CTCEN 位写 0 并确认 CTC_STR.CTCBSY 位为 0 后再修改。	RW

15.5.4 时钟校准状态寄存器 (CTC_STR)

Offset	Bit Position																																			
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																																CTCBSY	TRMUDF	TRMOVF	TRIMOK
Access																																	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3	CTCBSY	CTC 状态标志	0b0: CTC 处于停止状态 0b1: CTC 处于工作状态 CTC_CR1.CTCEN=1 时置位，在软件将 CTCEN 清零或者发生校准错误时清零。	RO
2	TRMUDF	校准下溢标志	校准下溢表示 TRMVAL 已减至 0x20，但仍未校准到目标范围。校准下溢时自动停止校准，ERRIE 为 1 时将发生错误中断。 0b0: 未发生校准下溢 0b1: 发生了校准下溢 在 CTC_CR1.CTCEN 写 1 时清零。	RO
1	TRMOVF	校准上溢标志	校准上溢表示 TRMVAL 已增至 0x1F，但仍未校准到目标范围。校准上溢时自动停止校准，ERRIE 为 1 时将发生错误中断。 0b0: 未发生校准上溢 0b1: 发生了校准上溢 在 CTC_CR1.CTCEN 写 1 时清零。	RO
0	TRIMOK	校准成功标志	0b0: HRC 频率未校准到目标范围 0b1: HRC 频率已校准到目标范围 在 CTC_CR1.CTCEN 写 1 时清零。	RO

16 DMA 控制器 (DMA)

16.1 简介

DMA 用于在存储器和外围功能模块之间传送数据，能够在 CPU 不参与的情况下，实现存储器之间、存储器和外围功能模块之间以及外围功能模块之间的数据交换。

16.2 主要特性

- DMA 总线独立于 CPU 总线，按照 AMBA AHB-Lite 总线协议传输
- 拥有 2 个 DMA 控制单元，共 16 个独立通道，可以独立操作不同的 DMA 传输功能
- 每个通道的启动源通过独立的触发源选择寄存器配置
- 每次请求传输一个数据块
- 数据块最小为 1 个数据，最多可以是 1024 个数据
- 每个数据的宽度可配置为 8-bit、16-bit 或 32-bit
- 可以配置 1~65535 次传输或无限次传输
- 源地址和目标地址可以独立配置为固定、自增、自减、重复跳转或指定偏移量的跳转
- 可产生 3 种中断：块传输完成中断、传输完成中断和传输错误中断。每种中断都可以配置是否屏蔽。其中块传输完成、传输完成可作为事件输出，可作为其他外围模块的触发源
- 支持连锁传输功能，可实现一次请求传输多个数据块
- 支持外部事件触发通道重置
- 支持软件触发启动传输和软件触发通道重置
- 不使用时可设置进入模块停止状态以降低功耗
- DMA 访问时 AHB 总线中的 HPROT 值可通过寄存器设置

16.3 功能说明

16.3.1 DMA 框图

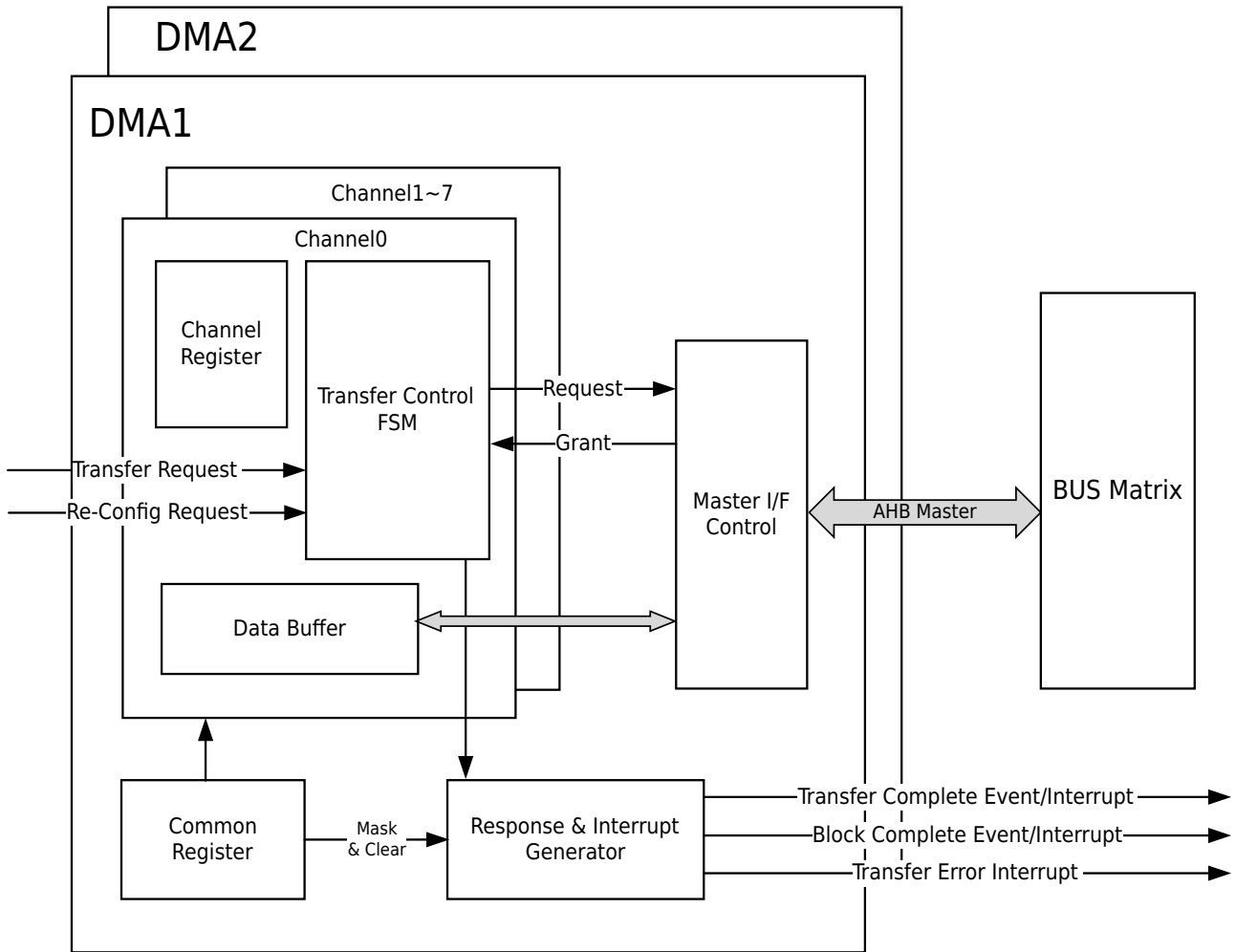


图 16-1 DMA 结构图

16.3.2 使能 DMA 控制器

使用 DMA 时需要先将功能时钟控制 0 寄存器 FCG0.DMAx 位写 0 允许 DMA 的时钟供给，再将 DMA 使能寄存器 DMA_EN.EN 位写 1。同时，DMA 需要外围电路来启动传输，所以 FCG0.AOS 外设电路触发功能位也需写 0。

当不使用 DMA，或者芯片需要进入 STOP 模式时，请将 DMA_EN.EN 设定为 0，并写 FCG0 关闭 DMA 时钟。EN 写 0 前请先确认寄存器 DMA_CHSTAT.DMAACT 为 0，确保 DMA 已完成所有传输。

16.3.3 通道选择和通道优先级

每个 DMA 控制单元中包含 8 个通道，每个通道可以独立配置传输功能。

通道优先级顺序为：通道 0 > 通道 1 > ... > 通道 7。

当一个 DMA 单元有多个通道请求传输时将按照优先级顺序执行。但已处于传输中的通道不会被打断，高优先通道需等当前通道传输完成后才会启动。

16.3.4 启动 DMA

DMA 通过外围电路产生的请求启动，这些请求通过触发源选择寄存器 DMA_TRGSELx (x=0~7) 进行配置，各通道独立配置启动请求源。DMA 可以通过软件写软件触发寄存器 DMA_SWREQ.SWREQ 启动 DMA 并开始

传输。当外围电路产生启动请求或软件写寄存器产生启动请求时，并且 DMA 传输使能有效 DMA_EN.EN=1，传输通道处于许可状态 DMA_CHEN.CHEN[x]=1，则启动通道 x 传输。

使用前，需要先将功能时钟控制 0 寄存器 (FCG0) 的外设电路触发功能及 DMA 功能使能位置为有效。

16.3.5 数据块

DMA 每次启动传输的数据量用块 (block) 表示，块的大小由数据控制寄存器 DMA_DTCTLx.BLKSIZE 设定，最大可以设置 1024 个数据。每个的数据宽度由 DMA_CHCTLx.HSIZE 决定，可以是 8-bit、16-bit 或 32-bit。

16.3.6 传输地址控制

传输的源地址和目标地址可以由寄存器设定为固定、递增、递减、重复跳转或者不连续跳转。

- **固定：**源地址、目标地址将在传输过程中固定不变。
- **递增及递减：**源地址、目标地址将在每传输完成 1 个数据后根据 HSIZE 的值进行增减。例如当 HSIZE 为 8-bit 时，地址将每次增加/减少 1，为 16-bit 时每次增加/减少 2，为 32-bit 时每次增加/减少 4。
- **重复跳转：**传输指定数量的数据后，源、目标地址将重新返回至最初的地址设定值。地址重复跳转前需要传输的数据量，即重复区域的大小由寄存器 DMA_RPT 设定。
- **不连续跳转：**传输指定数量的数据后，源、目标地址将跳过指定偏移量。地址跳转的偏移量，以及跳转前需要传输的数据量，即不连续区域的大小，由寄存器 DMA_SNSEQCTL/DMA_DNSEQCTL 设定。当地址重复跳转与不连续跳转的条件同时满足时，执行地址重复跳转。

16.3.7 传输次数

DMA 传输的总数据块的个数由数据控制寄存器 DMA_DTCTLx.CNT 位设定。传输次数最多可设置 65535 次。每传输一个数据块寄存器值减 1，当寄存器值减为 0 时代表本通道全部数据传输完成，自动清除通道传输许可位 DMA_CHEN.CHEN[x]，并产生传输完成中断。如果传输开始时 DMA_DTCTLx.CNT 被设为 0，表示无限次传输，每次启动请求传输一个数据块，但不清除通道传输许可位，也不会产生传输完成中断。



说明

CHEN[x]自动清 0 后，需要再次启动此通道时，需要重新设置通道配置寄存器后再将 CHEN[x]写 1 使能本通道。否则将按照上次传输结束时的状态开始后续的传输，即源/目标地址为上次传输对应的下个地址，CNT 为 0 的无限次传输。

16.3.8 连锁传输

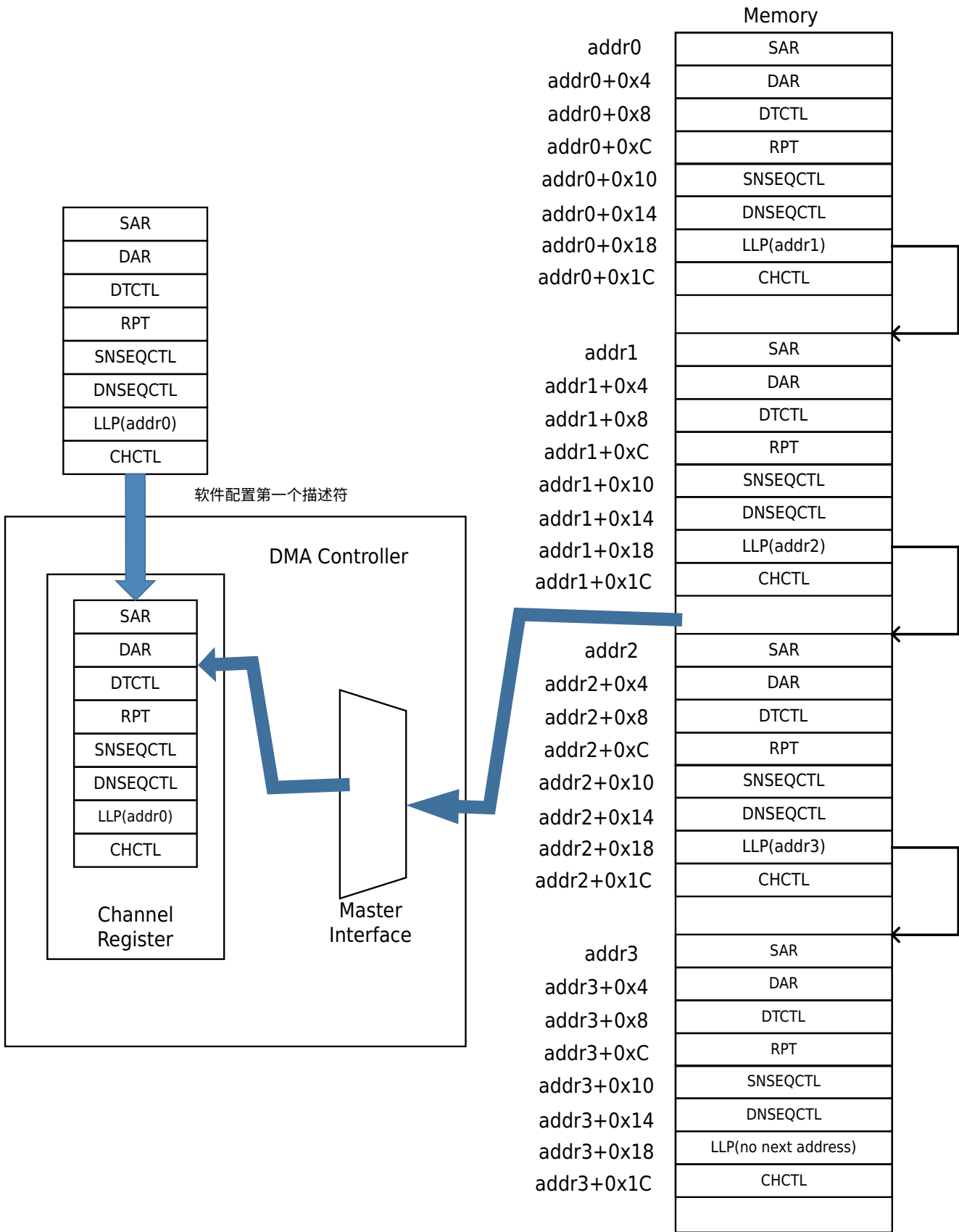
DMA 控制器具有连锁传输功能。连锁传输需要配置以下 8 个寄存器共 8 个 word，称为一个描述符 (descriptor)，包含了连锁传输的源地址、目标地址、数据控制信息、地址控制信息、链指针以及传输控制信息。

- DMA_SARx
- DMA_DARx
- DMA_DTCTLx
- DMA_RPTx
- DMA_SNSEQCTLx
- DMA_DNSEQCTLx
- DMA_LL Px
- DMA_CHCTLx

其中 LLP 称为链指针 (Linked-List Pointer)，其中的值代表下一个描述符在存储器中的首地址。使用连锁传输时首先写通道控制寄存器 DMA_CHCTLx.LLPEN 将连锁传输使能，并将第一个传输的描述符信息写入相应寄存器中。再将后续传输的描述符按顺序初始化在存储器中。需要结束连锁传输时，将最后一个描述符中 DMA_CHCTLx.LLPEN 置为无效，DMA 控制器将在传输完成后结束连锁传输。

当一个描述符的最后一次传输结束时，LLP 指定的下一个描述符将被从存储器中载入通道配置寄存器。等待下一次传输请求输入，开始新描述符的第一次传输。或者根据寄存器 DMA_CHCTLx.LLPRUN 的设置，在载入新描述符后直接开始第一次传输。

- 当 DMA_CHCTLx.LLPRUN = 0，在当前描述符的最后一次块传输之后，新的信道描述符从存储器加载到信道寄存器中，BTC 和 TC 中断照常生成，然后等待下一次传输请求以开始传输这个新加载的描述符。
- 当 DMA_CHCTLx.LLPRUN = 1，新描述符的传输在加载后直接开始，并且对于旧描述符的最后一个块传输既不生成 BTC 也不生成 TC 中断。



16.3.9 不连续地址传输

使用不连续地址传输能够实现源地址和目标地址在传输完一定量的数据后按照一定的偏移量跳转。以源地址不连续传输为例：首先根据需要将通道控制寄存器 DMA_CHCTLx.SNSEQEN 置 1 使不连续地址传输有效，跳转的方向按照 DMA_CHCTLx.SINC 的设定向前或向后跳转，然后配置寄存器 DMA_SNSEQCTLx 设置具体的跳

转方式。目标地址的控制与源地址相似，对应控制寄存器分别为 DMA_CHCTLx.DNSEQEN、DMA_CHCTLx.DINC、DMA_DNSEQCTLx。传输过程按照下图方式进行。

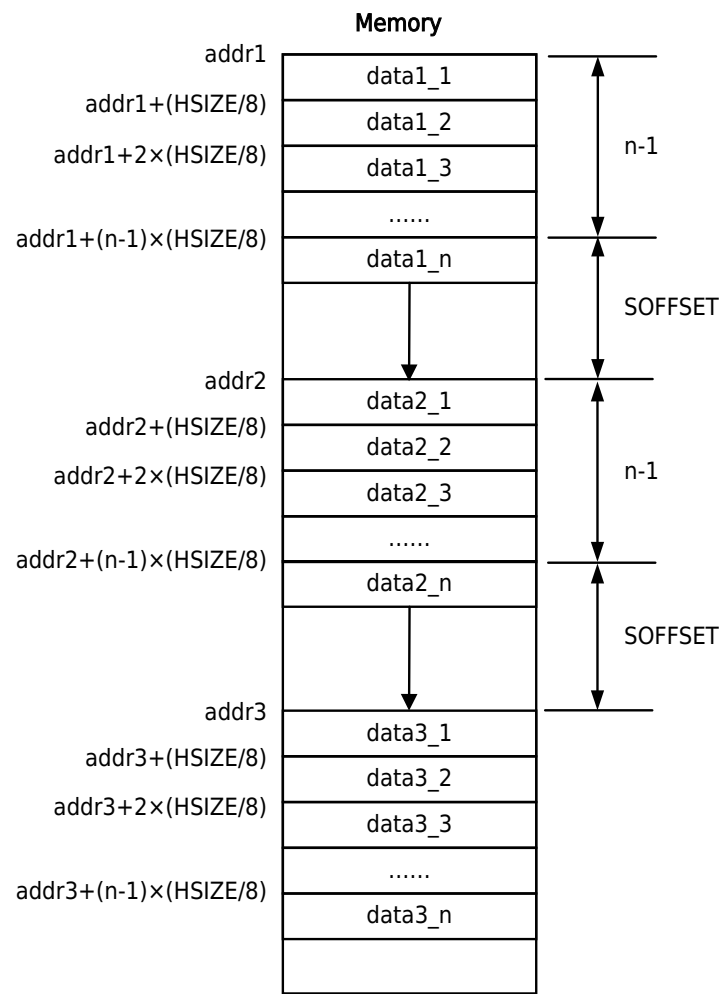


图 16-3 不连续地址传输示意图（以源地址为例）

$n = \text{SNSCNT}$
 $\text{addr2} = \text{addr1} + (n-1) \times (\text{HSIZE}/8) + \text{SOFFSET} \times (\text{HSIZE}/8)$
 $\text{addr3} = \text{addr2} + (n-1) \times (\text{HSIZE}/8) + \text{SOFFSET} \times (\text{HSIZE}/8)$
 其中 `HSIZE` 为数据宽度 (bit)，可以是字节 (8bits)，半字 (16bits) 和字 (32bits)。

16.3.10 通道重置

通道重置功能是指通过外围电路的事件请求来修改通道内部状态寄存器，重新配置下一次数据的传输方式。设置寄存器 `DMA_RCFGCTL.RCFGGEN` 为 1，允许通道重置。通过触发源选择寄存器 `DMA_RC_TRGSEL` 选择重置请求源，或通过软件写软件触发寄存器 `DMA_SWREQ.SWRCFGREQ` 触发通道重置功能。当选择的重置请求源输入或软件写寄存器产生重置请求时，寄存器 `DMA_RCFGCTL.RCFGCHS` 选择的通道会按指定的方式进行更新。重置请求只更新内部状态，不会启动实际的数据传输。

通道的重置方式有如下三种：链指针式、不连续式、重复式。

- 选择链指针式重置时，通道的描述符和内部状态全部更新为链指针 `LLP` 指向的新描述符。之后的传输请求，按新描述符进行传输。
- 选择不连续式、重复式重置时，通道内部状态按下表所述进行更新。

表 16-1 通道重置说明

重置方式 (寄存器 DMA_RCFGCTL)			通道内部状态		
CNTMD	SARMD	DARMD	剩余传输次数计数器	下次传输的源地址	下次传输的目标地址
0x0	任意	任意	保持不变	根据 SARMD 更新	根据 DARMD 更新
0x1	0x0	任意	保持不变	保持不变	根据 DARMD 更新
	0x1		更新为正常状态下，下一次源地址不连续跳转发生后的数值	更新为下一个源地址不连续传输区域的首地址	
	0x2、0x3		更新为正常状态下，下一次源地址重复跳转发生后的数值	更新为寄存器 DMA_SARx 的初始设定值	
0x2、0x3	任意	0x0	保持不变	根据 SARMD 更新	保持不变
		0x1	更新为正常状态下，下一次目标地址不连续跳转发生后的数值		更新为下一个目标地址不连续传输区域的首地址
		0x2、0x3	更新为正常状态下，下一次目标地址重复跳转发生后的数值		更新为寄存器 DMA_DARx 的初始设定值



说明

当重置功能有效时，该通道使用寄存器 DMA_RPTBx 和 DMA_SNSEQCTLBx，DMA_DNSEQCTLBx 控制传输地址的重复跳转和不连续跳转。寄存器 DMA_RPTx 和 DMA_SNSEQCTLx，DMA_DNSEQCTLx 无效。

下图所示 DMA 动作中，每次传输请求启动一个数据块的传输。第 1 次重置请求发生后，控制器跳过了数据块 blk1_3、blk1_4，传输地址更新为下个不连续区域的首地址即 addr2。第 2 次重置请求发生后，剩余传输次数更新为 0，即所有数据传输完成，通道许可位自动清 0，产生传输完成中断和事件。

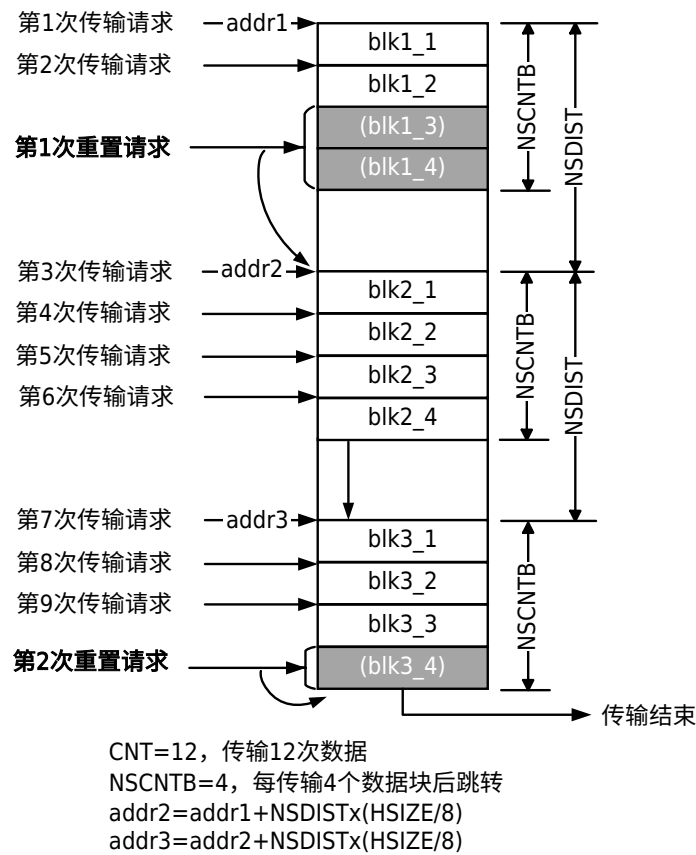


图 16-4 不连续式重置示意图

16.3.11 传输提前终止

传输过程中通道使能寄存器 DMA_CHEN.CHENx 保持有效, 非连锁传输时, 数据控制寄存器 DMA_DTCTLx 设定的传输次数完成后自动置为无效, 连锁传输时, 最后一次连锁传输的传输次数完成后自动置为无效。传输过程中如果软件对 DMA_CHENCLR.CHENCLR x 写 1, 则 DMA 将在完成当次数据读写后终止传输, 且 DMA_CHEN.CHENx 位清 0。

**注意**

软件对 CHENCLR x 位写 1 提前终止传输时, DMA 内部不会保存被终止时的传输状态。在没有重新设置通道配置寄存器 (描述符) 状态, 将 CHENx 写 1 再次允许此通道, 传输请求输入后, DMA 将重新传输被终止的数据块, 而不是断点续传。

16.3.12 中断

DMA 控制器可以产生以下 3 种中断:

- **数据块完成中断 DMA_BTCx:** 完成一个数据块传输后产生。
- **传输完成中断 DMA_TCx:** 完成寄存器 DMA_DTCTLx.CNT 设置的传输次数后产生。
- **传输错误中断 DMA_ERR:** 当启动请求溢出 (即通道的上一次请求还未响应时此通道再次触发启动请求) 时, 或者, 传输过程中发生总线错误 (如访问了非法地址或受保护的地址) 时产生中断, 其中, 总线错误时会立即终止本次传输。

以上中断除了启动请求溢出错误外, 其他中断都可以通过寄存器 DMA_CHCTLx.IE 设置中断的有效或无效。另外所有中断还配有独立的 MASK 寄存器, 对中断进行屏蔽。

上述的 DMA_BTCx、DMA_TCx 中断同时也可以作为事件信号输出, 可用作其他外围电路的触发源, 事件输出受 MASK 寄存器控制, 但不受中断许可位 DMA_CHCTLx.IE 控制。DMA_BTCx、DMA_TCx、DMA_ERR 事件发生会将对应的状态位寄存器置位, 不受 DMA_CHCTLx.IE 或 MASK 寄存器影响。

16.4 操作示例

16.4.1 存储器到存储器的传输

目标

将 12 个数据从 RAM 地址 0x20000000 传送到 0x20001000，数据宽度为 32-bit。

寄存器设置

1. DMA_EN.EN 写 1 使能 DMA 控制器。
2. 选择一个通道，例如通道 0，配置通道寄存器以实现：
 - a. 写 DMA_SAR0 配置源地址为 SRAM 区域 0x20000000。
 - b. 写 DMA_DAR0 配置目标地址为 SRAM 区域 0x20001000。
 - c. 写 DMA_DTCTL0 配置数据块的大小为 4，传输次数为 3 次，每次传输完 1 个数据块后产生块传输完成中断，3 次传输完成后产生一个传输完成中断。
 - d. 写 DMA_RPT 寄存器配置源地址重复区域大小为 6，即传输完成 6 个地址后重新载入初始源地址。
 - e. 配置通道控制寄存器 DMA_CHCTL0 以实现：
 - 1) 源地址和目标地址连锁传输无效。
 - 2) 源地址重复传输有效，目标地址更新方式为自增。
 - 3) 数据宽度为 32-bit。
 - 4) 中断使能有效。
 - f. 配置触发源控制器 DMA_TRGSEL0，选择软件触发作为 DMA 通道 0 的启动请求。
 - g. 通道使能位 DMA_CHEN.CHEN0 写 1，使能通道 0。
3. 写外设事件软件触发寄存器 INTSFTRG.STRG 为 1，发送第一个软件启动请求，DMA 开始传输数据。

传输过程

由于传输的数据块大小为 4，当软件写 INTSFTRG.STRG 为 1 后开始第一次传输，当一个数据块传输完成后，传输次数 DMA_MONDTCTL0.CNT 减 1，并产生一个块传输完成中断，软件可以在中断子程序中继续写 INTSFTRG.STRG 开始第二次传输。在第二次传输中，由于设置了源地址重复区域大小为 6，因此在传输完 2 个地址后源地址将重新载入初始地址 0x20000000 并继续传输剩余的 2 个地址。第二次传输完成后，传输次数 DMA_MONDTCTL0.CNT 减 1，并产生一个块传输完成中断，软件可以在中断子程序中继续写 INTSFTRG.STRG 开始第三次传输。第三次传输完成后，传输次数 DMA_MONDTCTL0.CNT 减为 0，即本次传输全部完成，DMA 产生一个块传输完成中断和一个传输完成中断，同时通道使能位 DMA_CHEN.CHEN0 将被自动清零。

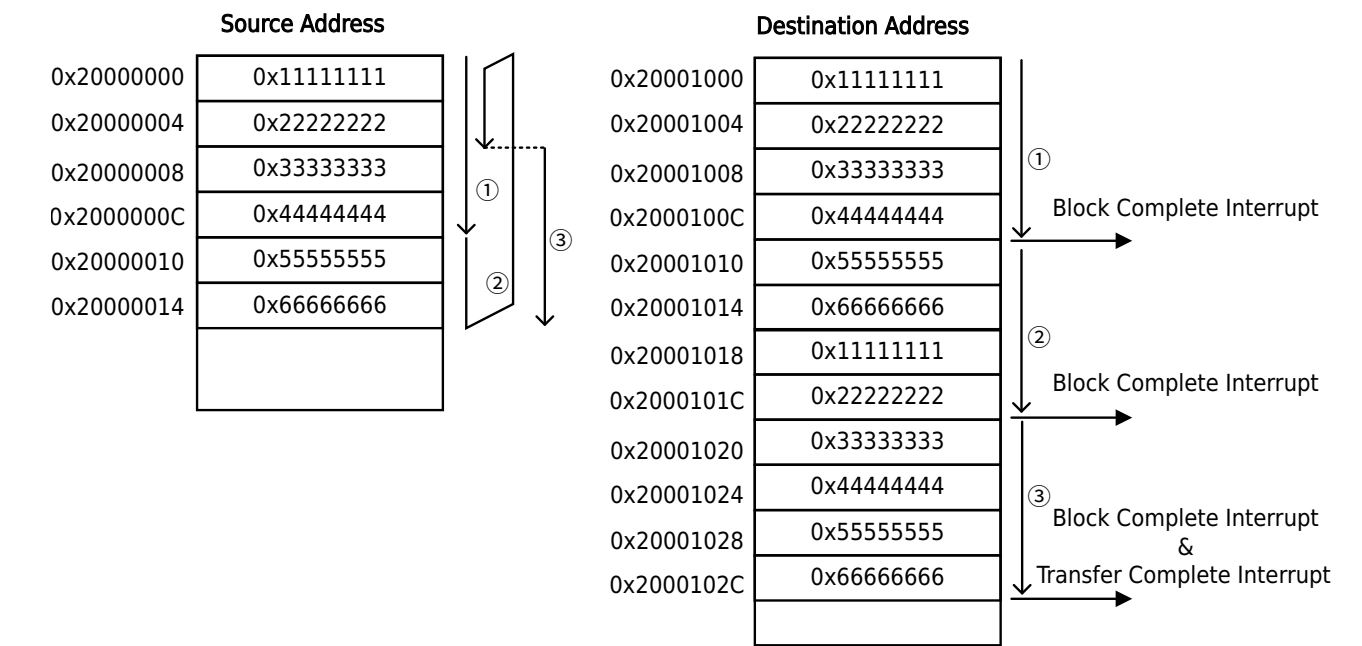


图 16-5 应用例 1：存储器到存储器传输

16.4.2 存储器到外围电路的传输

目标

将 10 个宽度为 16-bit 的数据从 RAM 地址 0x20000000 传送到通信模块的发送缓冲寄存器中，通信模块每发送完一个数据产生一次传送请求。当最后一个数据发送完成后 DMA 产生一个传输完成中断。

寄存器设定

1. DMA_EN.EN 写 1 使能 DMA 控制器。
2. 配置 DMA_INTMSK1 寄存器，屏蔽块传输完成中断，使能传输完成中断。
3. 选择一个通道，例如通道 0，配置通道寄存器以实现：
 - a. 写 DMA_SAR0 配置源地址为 SRAM 区域 0x20000000。
 - b. 写 DMA_DAR0 配置目标地址为外围电路的寄存器地址 0x40000000。
 - c. 写 DMA_DTCTL0 配置数据块的大小为 1，传输次数为 10 次，每个传输请求传送一次，每次 1 个数据。
 - d. 配置通道控制寄存器 DMA_CHCTL0 以实现：
 - 1) 源地址和目标地址连锁传输无效。
 - 2) 源地址更新方式为自增，目标地址为固定。
 - 3) 源地址和目标地址数据宽度为 16-bit。
 - 4) 中断使能有效。
 - e. 配置触发源控制器 DMA_TRGSEL0，选择通信模块的发送寄存器空作为 DMA 通道 0 的启动请求。
 - f. 通道使能位 DMA_CHEN.CHEN0 写 1，使能通道 0。

传输过程

通道使能后，DMA 等待来自通信模块的传输请求。当传输请求产生后，DMA 把 RAM 中的数据传送至通信模块的发送缓冲寄存器中，并等待第二次来自通信模块的传输请求，由于屏蔽了块传输完成中断，DMA 此时不产生中断请求。当 10 个数据全部传送完成后，DMA 产生一个传输完成中断，并且通道使能位 DMA_CHEN.CHEN0 将被自动清零。

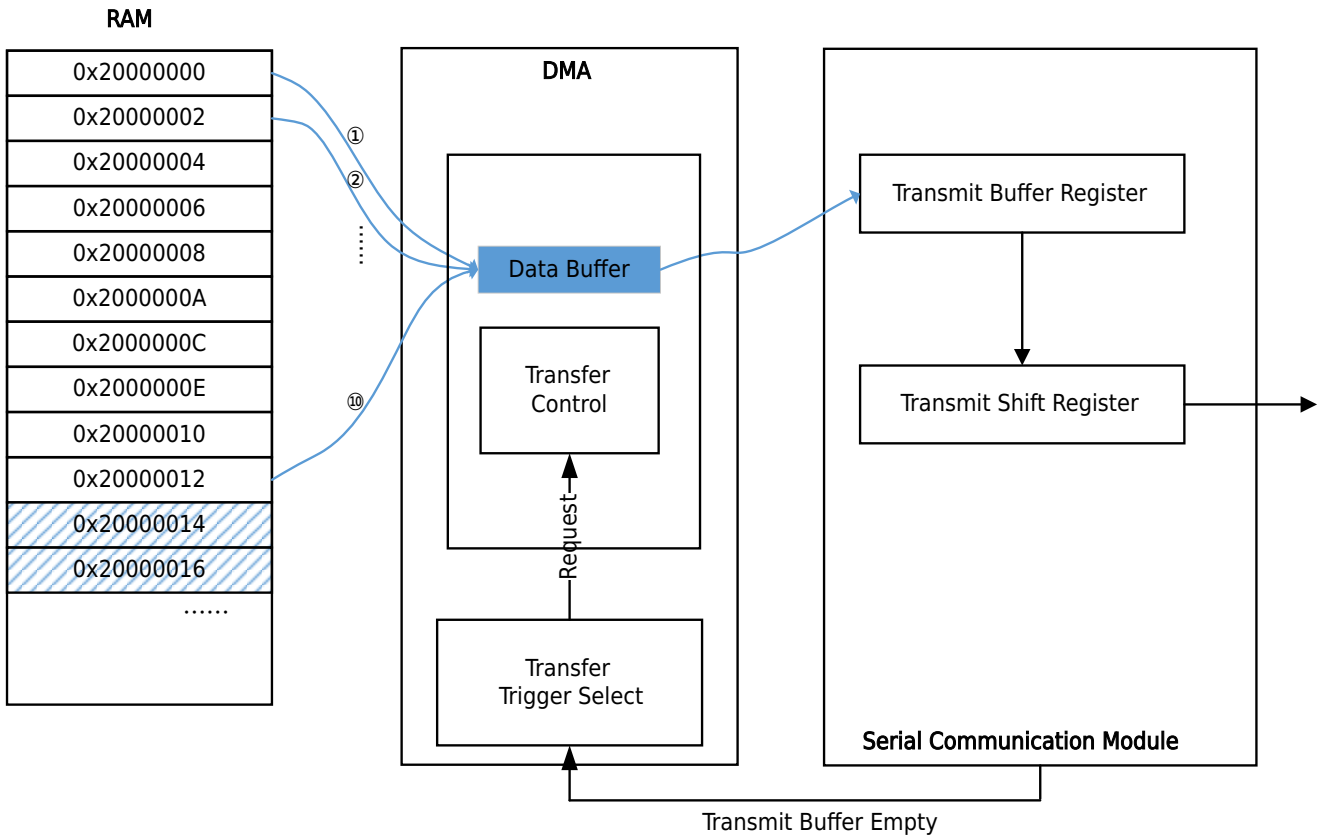


图 16-6 应用例 2：存储器到外围电路的传输

16.4.3 存储器到存储器的连锁传输

目标

通过连锁传输将 RAM 中 3 块数据传输到 3 个不同地址。其中，第一块数据为 10 个 32 位数据，从 0x20000000 传到 0x20001000。第二块数据为 20 个 16 位数据，从 0x20000100 传到 0x20001100。第三块数据为 40 个 8 位数据，从 0x20000200 传到 0x20001200。

寄存器设定

1. DMA_EN.EN 写 1 使能 DMA 控制器。
2. 选择一个通道，配置通道寄存器，例如选择通道 0，配置第一次传输的描述符（descriptor0）：
 - a. 写 DMA_SAR0 配置源地址为 SRAM 区域 0x20000000。
 - b. 写 DMA_DAR0 配置目标地址为 SRAM 区域 0x20001000。
 - c. 写 DMA_DTCTL0 配置数据块的大小为 10。
 - d. 在链指针寄存器 DMA_LLPO 中写入第二个描述符（descriptor1）的地址 0x20002000。
 - e. 配置通道控制寄存器 DMA_CHCTL0，配置第一个数据块的传输参数以实现：
 - 1) 连锁传输有效。
 - 2) 连锁传输模式为直接启动下一次传输。
 - 3) 源地址和目标地址更新方式为自增。
 - 4) 数据宽度为 32-bit。
 - 5) 中断使能无效。
3. 在 RAM 空间的 0x20002000 地址中配置第二次传输的描述符（descriptor1），包括：
 - a. 0x20002000 中写入 32 位数据 0x20000100，此为第二次传输的源地址。
 - b. 0x20002004 中写入 32 位数据 0x20001100，此为第二次传输的目标地址。
 - c. 0x20002008 中配置数据块的大小为 20。

- d. 0x20002018 中写入 32 位数据 0x20002020，此为第三次传输的描述符 (descriptor2) 所在地址。
- e. 0x2000201C 中写入第二次传输的控制数据，该控制数据实现：
 - 1) 连锁传输有效。
 - 2) 连锁传输模式为直接启动下一次传输。
 - 3) 源地址和目标地址更新方式为自增。
 - 4) 数据宽度为 16-bit。
 - 5) 中断使能无效。
- 4. 在 RAM 空间的 0x20002020 地址中配置第三次传输的描述符 (descriptor2)，包括：
 - a. 0x20002020 中写入 32 位数据 0x20000200，此为第三次传输的源地址。
 - b. 0x20002024 中写入 32 位数据 0x20001200，此为第三次传输的目标地址。
 - c. 0x20002028 配置数据块的大小为 40。
 - d. 0x20002038 中写入 32 位数据 0x0，代表该次传输为连锁传输的最后一次传输。
 - e. 0x2000203C 中写入第三次传输的控制数据实现：
 - 1) 连锁传输无效。
 - 2) 源地址和目标地址更新方式为自增。
 - 3) 数据宽度为 8-bit。
 - 4) 中断使能有效。
- 5. 配置传输启动触发源选择寄存器 DMA_TRGSEL0，选择软件触发作为 DMA 通道 0 的启动请求。
- 6. 通道使能位 DMA_CHEN.CHEN0 写 1，使能通道 0。
- 7. 写软件触发寄存器 INTSFTRG.STRG 为 1，发送一个启动请求，DMA 开始传输数据。

传输过程

软件启动 DMA 开始传输。第一次传输完成后，由于设置了连锁传输模式为直接启动下一次传输并且中断无效，DMA 将第二次传输的描述符 (descriptor1) 读入到通道寄存器中。根据描述符配置参数直接开始第二次传输。第二次传输完成后，将第三次传输的描述符 (descriptor2) 读入到通道寄存器中。根据描述符配置参数开始第三次传输。第三次传输完成后，根据配置信息该次为连锁传输的最后一次，且由于中断使能有效，因此 DMA 将产生一个传输完成中断，并清除通道使能位 DMA_CHEN.CHEN0。

16.5 寄存器

16.5.1 寄存器总表

- DMA1 基地址: 0x40053000
- DMA2 基地址: 0x40053400

表 16-2 DMA 控制器 (DMA) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	DMA_EN	DMA 使能寄存器	32
0x04	DMA_INTSTAT0	中断状态寄存器 0	32
0x08	DMA_INTSTAT1	中断状态寄存器 1	32
0x0C	DMA_INTMASK0	中断屏蔽寄存器 0	32
0x10	DMA_INTMASK1	中断屏蔽寄存器 1	32
0x14	DMA_INTCLR0	中断复位寄存器 0	32
0x18	DMA_INTCLR1	中断复位寄存器 1	32
0x1C	DMA_CHEN	通道使能寄存器	32
0x20	DMA_REQSTAT	传输请求状态寄存器	32
0x24	DMA_CHSTAT	通道状态观测寄存器	32
0x2C	DMA_RCFGCTL	通道重置控制寄存器	32
0x30	DMA_SWREQ	软件触发寄存器	32
0x34	DMA_CHENCLR	通道使能复位寄存器	32
0x40+0x40*x	DMA_SARx	传输源地址寄存器	32
0x44+0x40*x	DMA_DARx	传输目标地址寄存器	32
0x48+0x40*x	DMA_DTCTLx	数据控制寄存器	32
0x4C+0x40*x	DMA_RPTx	重复区域大小寄存器	32
0x4C+0x40*x	DMA_RPTBx	重复区域大小寄存器 B	32
0x50+0x40*x	DMA_SNSEQCTLx	源地址不连续传输控制寄存器	32
0x50+0x40*x	DMA_SNSEQCTLBx	源地址不连续传输控制寄存器 B	32
0x54+0x40*x	DMA_DNSEQCTLx	目标地址不连续传输控制寄存器	32
0x54+0x40*x	DMA_DNSEQCTBx	目标地址不连续传输控制寄存器 B	32
0x58+0x40*x	DMA_LL Px	链指针寄存器	32
0x5C+0x40*x	DMA_CHCTLx	通道控制寄存器	32
0x60+0x40*x	DMA_MONSARx	传输源地址监视寄存器	32

偏移地址	寄存器	描述	位宽
0x64+0x40*x	DMA_MONDARx	传输目标地址监视寄存器	32
0x68+0x40*x	DMA_MONDTCTLx	数据控制监视寄存器	32
0x6C+0x40*x	DMA_MONRPTx	重复区域计数器监视寄存器	32
0x70+0x40*x	DMA_MONSNSEQCTLx	源地址不连续传输计数器监视寄存器	32
0x74+0x40*x	DMA_MONDNSEQCTLx	目标地址不连续传输计数器监视寄存器	32

 **说明**

表中的 x 代表存在多个寄存器，取值为 0~7。

16.5.2 DMA 使能寄存器 (DMA_EN)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EN	DMA 使能位	0b0: 禁止 0b1: 使能	RW

16.5.3 中断状态寄存器 0 (DMA_INTSTAT0)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								REQERR								Reserved								TRNERR							
Access									RO																RO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	REQERR	传输请求溢出错误中断位	对应 REQERR7~REQERR0: 0b0: 该通道没有发生传输请求溢出错误 0b1: 该通道发生了传输请求溢出错误, 即上次请求还处在等待状态时又来了传输请求  说明 每个 Bit 对应一个传输通道。	RO
15:8	Reserved	保留	-	-
7:0	TRNERR	传输错误中断位	对应 TRNERR7~TRNERR0: 0b0: 该通道没有发生传输错误 0b1: 该通道发生了传输错误  说明 每个 Bit 对应一个传输通道。	RO

16.5.4 中断状态寄存器 1 (DMA_INTSTAT1)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								BTC								Reserved								TC							
Access									RO																RO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	BTC	块传输完成中断位	对应 BTC7~BTC0，该中断在完成一个数据块的传输后发生： ● 0b0：该通道没有块传输中断发生 ● 0b1：该通道发生块传输中断  说明 每个 Bit 对应一个传输通道。	RO
15:8	Reserved	保留	-	-
7:0	TC	传输完成中断位	对应 TC7~TC0，该中断在完成传输 DMA_DTCTLx.CNT 设置的次数后发生： ● 0b0：该通道没有传输完成中断发生 ● 0b1：该通道发生传输完成中断  说明 每个 Bit 对应一个传输通道。	RO

16.5.5 中断屏蔽寄存器 0 (DMA_INTMASK0)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								MSKREQERR								Reserved								MSKTRNERR							
Access									RW																RW							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	MSKREQERR	传输请求溢出中断屏蔽	对应 MSKREQERR7~MSKREQERR0: ● 0b0: 不屏蔽传输请求溢出中断 ● 0b1: 屏蔽传输请求溢出中断  说明 每个 Bit 对应一个传输通道。	RW
15:8	Reserved	保留	-	-
7:0	MSKTRNERR	传输错误中断屏蔽	对应 MSKTRNERR7~MSKTRNERR0: ● 0b0: 不屏蔽传输错误中断 ● 0b1: 屏蔽传输错误中断  说明 每个 Bit 对应一个传输通道。	RW

16.5.6 中断屏蔽寄存器 1 (DMA_INTMASK1)

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								MSKBTC								Reserved								MSKTC							
Access									RW																RW							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	MSKBTC	块传输完成中断屏蔽	对应 MSKBTC7~MSKBTC0： ● 0b0：不屏蔽块传输完成中断 ● 0b1：屏蔽块传输完成中断  说明 每个 Bit 对应一个传输通道。	RW
15:8	Reserved	保留	-	-
7:0	MSKTC	传输完成中断屏蔽	对应 MSKTC7~MSKTC0： ● 0b0：不屏蔽传输完成中断 ● 0b1：屏蔽传输完成中断  说明 每个 Bit 对应一个传输通道。	RW

16.5.7 中断复位寄存器 0 (DMA_INTCLR0)

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								CLRREQERR								Reserved								CLRTRNERR							
Access									WO																WO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	CLRREQERR	传输请求溢出错误中断复位	对应 CLRREQERR7~CLRREQERR0: 写 0 无任何效果, 写 1 复位传输请求溢出错误中断状态位; 读出恒为 0  说明 每个 Bit 对应一个传输通道。	WO
15:8	Reserved	保留	-	-
7:0	CLRTRNERR	传输错误中断复位	对应 CLRTRNERR7~CLRTRNERR0: 写 0 无任何效果, 写 1 复位传输错误中断状态位; 读出恒为 0  说明 每个 Bit 对应一个传输通道。	WO

16.5.8 中断复位寄存器 1 (DMA_INTCLR1)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								CLR BTC								Reserved								CLR RTC							
Access									WO																WO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	CLRBTC	块传输完成中断复位	对应 CLRBTC7~CLRBTC0： 写 0 无任何效果，写 1 复位块传输完成中断状态位；读出恒为 0  说明 每个 Bit 对应一个传输通道。	WO
15:8	Reserved	保留	-	-
7:0	CLRRTC	传输完成中断复位	对应 CLRRTC7~CLRRTC0： 写 0 无任何效果，写 1 复位传输完成中断状态位；读出恒为 0  说明 每个 Bit 对应一个传输通道。	WO

16.5.9 通道使能寄存器 (DMA_CHEN)

Offset	Bit Position																															
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								CHEN							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	CHEN	通道使能位	对应 CHEN7~CHEN0: ● 0b0: 该通道无效 ● 0b1: 该通道有效 对 CHEN[x]写 1, CHEN[x]置 1。对 CHEN[x]写 0 无效。 对 DMA_CHENCLR.CHENCLR[x]位写 1 时, CHEN[x]清 0, 并强制终止通道 x 正在进行的传输。传输过程中使能位保持为 1, 当传输完传输次数寄存器 DMA_DTCTLx.CNT 设置的传输次数后将会自动清 0。如果 DMA_DTCTLx.CNT 设置为 0, 则传输完成后不会自动清零, 即无限次传输。  说明 ● 每个 Bit 对应一个传输通道。 ● 请在通道 x 的 8 个描述符寄存器全部设置结束后将 CHEN[x]写 1, 使能本通道。CHEN[x]为 1 时, 对通道 x 的描述符寄存器的写操作无效。	RW

16.5.10 传输请求状态寄存器 (DMA_REQSTAT)

Offset	Bit Position																																
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																RCFGREQ	Reserved								CHREQ							
Access																	RO									RO							

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	RCFGREQ	通道重置请求标志	● 0b0: 无通道重置请求 ● 0b1: 有通道重置请求  说明 当外部重置请求输入后置 1；当通道重置启动后或者通道重置禁止时清 0。	RO
15:8	Reserved	保留	-	-
7:0	CHREQ	通道传输请求标志位	对应 CHREQ7~CHREQ0: ● 0b0: 该通道无传输请求 ● 0b1: 该通道有传输请求  说明 每个 Bit 对应一个传输通道。 当外部传输请求输入后对应位置 1；当该通道传输启动后，发生传输出错，或传输许可位（DMAEN 或 CHEN[x]）写 0 时，该位清 0。 当该位为 1 状态，该通道传输请求再次输入，则发生传输请求溢出错误，第二次的请求被忽略，但本标志位保持 1，继续等待传输。	RO

16.5.11 通道状态观测寄存器 (DMA_CHSTAT)

Offset	Bit Position																															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								CHACT								Reserved								RCFGACT		DMAACT					
Access									RO																RO		RO					

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	CHACT	传输动作中通道监视位	对应 CHACT7~CHACT0： ● 0b0：该通道空闲中 ● 0b1：该通道处于动作中  说明 每个 Bit 对应一个传输通道。	RO
15:2	Reserved	保留	-	-
1	RCFGACT	DMA 通道重置动作中监视位	● 0b0：DMA 未处于通道重置动作中 ● 0b1：DMA 处于通道重置动作中	RO
0	DMAACT	DMA 动作中监视位	● 0b0：DMA 未处于传输动作中 ● 0b1：DMA 处于传输动作中	RO

16.5.12 通道重置控制寄存器 (DMA_RCFGCTL)

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved										CNTMD		DARMD		SARMD		Reserved		RCFGCHS				Reserved					RCFGLLP		RCFGEN		
Access											RW		RW		RW				RW									RW		RW		

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21:20	CNTMD	剩余传输次数计数器重置方式	0b00: 保持不变 0b01: 按源地址方式 0b10/0b11: 按目标地址方式 当选择源地址方式时, 且源地址选择不连续式重置, 则剩余传输次数计数器更新为 DMA_SNSEQCTLBx.SNSCNTB 指定传输次数后的状态; 源地址选择重复式时, 剩余传输次数计数器更新为 DMA_RPTBx.SRPTB 指定传输次数后的状态。源地址选择保持时, 剩余传输次数计数器也保持不变。当选择目标地址方式时, 与源地址方式类似。	RW
19:18	DARMD	目标地址重置方式	0b00: 保持不变 0b01: 不连续式重置, 下次传输的目标地址更新为 $\text{addr\_base} + (\text{DMA\_DNSEQCTLBx.DNSDIST} \times \text{DMA\_CHCTLx.HSIZE}(\text{bit}) / 8)$, 其中: addr_base 表示当前不连续传输区域的首地址 0b10/0b11: 重复式重置, 下次传输的目标地址更新为 DMA_DARx 寄存器初始设置值  注意 只允许在本通道目标地址不连续传输有效 (DMA_CHCTLx.DNSEQEN=1) 状态下设置 DARMD[1:0] 为 0b01。 只允许在本通道目标地址重复传输有效 (DMA_CHCTLx.DRPTEN=1) 状态下设置 DARMD[1] 为 1。	RW
17:16	SARMD	源地址重置方式	0b00: 保持不变 0b01: 不连续式重置, 下次传输的源地址更新为 $\text{addr\_base} + (\text{DMA\_SNSEQCTLBx.SNSDIST} \times \text{DMA\_CHCTLx.HSIZE}(\text{bit}) / 8)$, 其中: addr_base 表示当前不连续传输区域的首地址 0b10/0b11: 重复式重置, 下次传输的源地址更新为 DMA_SARx 寄存器初始设置值  注意 只允许在本通道源地址不连续传输有效 (DMA_CHCTLx.SNSEQEN=1) 状态下设置 SARMD[1:0] 为 0b01。 只允许在本通道源地址重复传输有效 (DMA_CHCTLx.SRPTEN=1) 状态下设置 SARMD[1] 为 1。	RW

位/位域	标记	位名	功能描述	读写
15:12	Reserved	保留	-	-
11:8	RCFGCHS	重置通道选择	● 0x0：通道 0 ● 0x1：通道 1 ● ... ● 0x7：通道 7 ● 其他：禁止设置	RW
7:2	Reserved	保留	-	-
1	RCFGLLP	链指针式通道重置	● 0b0：链指针式重置无效 ● 0b1：链指针式重置有效  注意 RCFGPLL 设为 1 时，通道将重新载入存储器中的新描述符，故本寄存器的 Bit21-Bit16 全部无效。	RW
0	RCFGEN	通道重置许可	● 0b0：禁止事件触发通道配置寄存器强制更新 ● 0b1：允许事件触发通道配置寄存器强制更新	RW

 **注意**
请在 DMA_EN.EN 为 0 时设置本寄存器，本寄存器必须在重置通道的第一次传输前设置。

16.5.13 软件触发寄存器 (DMA_SWREQ)

Offset	Bit Position																																
0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	SWRCFGWP								SWREQWP								SWRCFGREQ	Reserved								SWREQ							
Access	WO								WO								WO									WO							

位/位域	标记	位名	功能描述	读写
31:24	SWRCFGWP	软件通道重置请求写保护	软件重置请求写保护，SWRCFGWP 写入值为 0xA2 解除写保护	WO
23:16	SWREQWP	软件通道触发请求写保护	软件触发请求写保护，SWREQWP 写入值为 0xA1 解除写保护	WO
15	SWRCFGREQ	软件通道重置请求	写 1（SWRCFGWP 同时写 0xA2）产生一次软件通道重置请求，写 0 无效  说明 软件触发通道重置功能，与 AOS 触发效果相同。	WO
14:8	Reserved	保留	-	-
7:0	SWREQ	软件通道触发请求	对应 SWREQ7~SWREQ0： 写 1（SWREQWP 同时写 0xA1）产生一次软件通道重置请求，写 0 无效  说明 每个 Bit 对应一个传输通道。 软件触发启动并进行传输，与 AOS 触发效果相同。	WO

16.5.14 通道使能复位寄存器 (DMA_CHENCLR)

Offset	Bit Position																															
0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								CHENCLR							
Access																									WO							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	CHENCLR	通道使能位复位	写 0 无任何效果，写 1 复位通道使能位 CHEN；读出恒为 0 写 1 时若本通道正在传输，则 DMA 将在完成当次数数据读写后强制终止传输  说明 每个 Bit 对应一个传输通道。	WO

16.5.15 传输源地址寄存器 (DMA_SARx) (x=0~7)

偏移地址 (Offset) : 0x40+0x40*x

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SAR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	SAR	传输源地址	设置传输源地址  说明 ● 传输数据宽度为 16-bit 时，即 DMA_CHCTLx.HSIZE=0b01，SAR[0]无效。 ● 传输数据宽度为 32-bit 时，即 DMA_CHCTLx.HSIZE=0b10/0b11，SAR[1:0]无效。	RW

16.5.16 传输目标地址寄存器 (DMA_DARx) (x=0~7)

偏移地址 (Offset) : 0x44+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DAR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DAR	传输目标地址	设置传输目标地址  说明 - 传输数据宽度为 16-bit 时, 即 DMA_CHCTLx.HSIZE=0b01, DAR[0]无效。 - 传输数据宽度为 32-bit 时, 即 DMA_CHCTLx.HSIZE=0b10/0b11, DAR[1:0]无效。	RW

16.5.17 数据控制寄存器 (DMA_DTCTLx) (x=0~7)

偏移地址 (Offset) : 0x48+0x40*x

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	CNT																Reserved								BLKSIZE							
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	CNT	传输次数	总的传输次数, 每次请求启动一个数据块的传输, 完成时传输次数计数器减 1, 当减到 0 时发生传输完成中断。如果设置为 0, 表示无限次传输, 每次启动请求传输一个数据块, 完成时传输次数计数器保持 0 不变, 不会产生传输完成中断。	RW
15:10	Reserved	保留	-	-
9:0	BLKSIZE	数据块的大小	设置数据块的大小, 最大可以配置 1024 个数据。每个数据的宽度由 DMA_CHCTLx.HSIZE 位决定。寄存器值设为 1 则每次传输 1 个数据, 设为 0 则每次传输 1024 个数据。	RW

16.5.18 重复区域大小寄存器 (DMA_RPTx) (x=0~7)

偏移地址 (Offset) : 0x4C+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				DRPT[9:0]								Reserved				SRPT[9:0]															
Access					RW												RW															

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:16	DRPT[9:0]	目标地址重复区域大小	设置目标地址重复区域大小 目标设备在每传输 DRPT 个数据后目标地址载入为 DMA_DARx 寄存器初始设定值。寄存器设为 10 则每传输 10 个数据后地址载入初始值，设为 0 则每传 1024 个数据后地址载入初始值。每个数据的宽度由 DMA_CHCTLx.HSIZE 决定。	RW
15:10	Reserved	保留	-	-
9:0	SRPT[9:0]	源地址重复区域大小	设置源地址重复区域大小 源设备在每传输 SRPT 个数据后源地址载入为 DMA_SARx 寄存器初始设定值。寄存器设为 10 则每传输 10 个数据后地址载入初始值，设为 0 则每传 1024 个数据后地址载入初始值。每个数据的宽度由 DMA_CHCTLx.HSIZE 决定。	RW

 **说明**

1. 该寄存器配置源地址和目标地址重复区域的大小。使用重复传输需要配置 DMA_CHCTLx 寄存器的 SRPTEN/DRPREN 位有效，并且配置 DMA_CHCTLx 寄存器的 SINC/DINC 位使地址更新方式为自增或自减，如果为固定则地址重复跳转无效。

2. 该寄存器在本通道的重置功能打开时无效。

16.5.19 重复区域大小寄存器 B (DMA_RPTBx) (x=0~7)

偏移地址 (Offset) : 0x4C+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				DRPTB[9:0]								Reserved				SRPTB[9:0]															
Access					RW												RW															

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:16	DRPTB[9:0]	目标地址重复区域大小	设置目标地址重复区域大小 目标设备在每传输 DRPTB 个数据块后目标地址载入 DMA_DARx 寄存器初始设定值。数据块大小由 DMA_DTCTLx.BLKSIZE 与 DMA_CHCTLx.HSIZE 决定。	RW
15:10	Reserved	保留	-	-
9:0	SRPTB[9:0]	源地址重复区域大小	设置源地址重复区域大小 源设备在每传输 SRPTB 个数据块后源地址载入 DMA_SARx 寄存器初始设定值。数据块大小由 DMA_DTCTLx.BLKSIZE 与 DMA_CHCTLx.HSIZE 决定。	RW

 **说明**

1. 该寄存器配置源地址和目标地址重复区域的大小。使用重复传输需要配置 DMA_CHCTLx 寄存器的 SRPTEN/DRPREN 位有效，并且配置 DMA_CHCTLx 寄存器的 SINC/DINC 位使地址更新方式为自增或自减，如果为固定则地址重复跳转无效。

2. 该寄存器只在本通道的重置功能打开时有效，替代寄存器 DMA_RPTx。通道重置功能关闭时无效。

16.5.20 源地址不连续传输控制寄存器 (DMA_SNSEQCTLx) (x=0~7)

偏移地址 (Offset) : 0x50+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SNSCNT[11:0]												SOFFSET[19:0]																			
Access	RW												RW																			

位/位域	标记	位名	功能描述	读写
31:20	SNSCNT[11:0]	源地址跳转的数据量	设置源地址跳转前传输的数据量大小。 源设备在每传输 SNSCNT 个数据后源地址按 SOFFSET 指定偏移量跳转。寄存器设为 10 则每传输 10 个数据后地址跳转，设为 0 则每传 4096 个数据后地址跳转。	RW
19:0	SOFFSET[19:0]	源地址跳转的地址偏移量	当不连续地址传输时，设置源地址跳转的偏移量。偏移量是相对于当前传输地址，即跳转前最后一次传输地址而言的。跳转的方向根据通道控制寄存器 DMA_CHCTLx.SINC 的值向前或向后跳转，详情请参见 不连续地址传输示意图 。当 DMA_CHCTLx.SINC 设为地址固定时，不连续地址传输无效。 跳转地址将根据数据的宽度 (DMA_CHCTLx.HSIZE) 设置的 bit 数和 SOFFSET 的值进行计算。 地址偏移量=SOFFSET × (DMA_CHCTLx.HSIZE(bit) / 8) 例如，当 SOFFSET 设为 10，数据宽度为 32-bit 时，地址偏移量为 10×4=40；若数据宽度为 16-bit 时，偏移量为 10×2=20；若数据宽度为 8-bit 时，偏移量为 10×1=10。 下次传输的源地址=当前传输的源地址±地址偏移量	RW

 **说明**

1. 使用源设备不连续传输需要配置 DMA_CHCTLx 寄存器的 SNSEQEN 位有效，并且配置 DMA_CHCTLx 寄存器的 SINC 位使地址更新方式为自增或自减。

2. 该寄存器在本通道的重置功能打开时无效。

16.5.21 源地址不连续传输控制寄存器 B (DMA_SNSEQCTLBx) (x=0~7)

偏移地址 (Offset) : 0x50+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SNSCNTB[11:0]												SNSDIST[19:0]																			
Access	RW												RW																			

位/位域	标记	位名	功能描述	读写
31:20	SNSCNTB[11:0]	源地址跳转的数据量	设置源地址跳转前传输的数据量大小。 源设备在每传输 SNSCNTB 个数据块后源地址按 SNSDIST 指定地址间距跳转。数据块大小由 DMA_DTCTLx.BLKSIZE 与 DMA_CHCTLx.HSIZE 决定。	RW
19:0	SNSDIST[19:0]	源不连续区域地址间距	当不连续地址传输时，设置源设备两个不连续区域的间距。跳转的方向根据通道控制寄存器 DMA_CHCTLx.SINC 的值向前或向后跳转，详情请参见 不连续式重置示意图 。当 DMA_CHCTLx.SINC 设为地址固定时，不连续地址传输无效。 地址间距将根据数据的宽度 (DMA_CHCTLx.HSIZE) 设置的 bit 数和 SNSDIST 的值进行计算。 地址间距= SNSDIST × (DMA_CHCTLx.HSIZE(bit) / 8) 例如，当 SNSDIST 设为 10，数据宽度为 32-bit 时，地址间距为 10×4=40；若数据宽度为 16-bit 时，间距为 10×2=20；若数据宽度为 8-bit 时，间距为 10×1=10。 下次传输的源地址=当前源不连续区域首地址±地址间距	RW

说明

1. 使用源设备不连续传输需要配置 DMA_CHCTLx 寄存器的 SNSEQEN 位有效，并且配置 DMA_CHCTLx 寄存器的 SINC 位使地址更新方式为自增或自减。

2. 该寄存器只在本通道的重置功能打开时有效，替代寄存器 DMA_SNSEQCTLx。通道重置功能关闭时无效。

16.5.22 目标地址不连续传输控制寄存器 (DMA_DNSEQCTLx) (x=0~7)

偏移地址 (Offset) : 0x54+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DNSCNT[11:0]												DOFFSET[19:0]																			
Access	RW												RW																			

位/位域	标记	位名	功能描述	读写
31:20	DNSCNT[11:0]	目标地址跳转的数据量	设置目标地址跳转前传输的数据量大小。 目标设备在每传输 DNSCNT 个数据后目标地址按 DOFFSET 指定偏移量跳转。寄存器设为 10 则每传输 10 个数据后地址跳转, 设为 0 则每传 4096 个数据后地址跳转。	RW
19:0	DOFFSET[19:0]	目标地址跳转的地址偏移量	当不连续地址传输时, 设置目标地址跳转的偏移量。偏移量是相对于当前传输地址, 即跳转前最后一次传输地址而言的。跳转的方向根据通道控制寄存器 DMA_CHCTLx.DINC 的值向前或向后跳转, 详情请参见 不连续地址传输示意图 。当 DMA_CHCTLx.DINC 设为地址固定时, 不连续地址传输将无效。 跳转地址将根据数据的宽度 (DMA_CHCTLx.HSIZE) 设置的 bit 数和 DOFFSET 的值进行计算。 地址偏移量=DOFFSET × (DMA_CHCTLx.HSIZE(bit) / 8) 例如, 当 DOFFSET 设为 10, 数据宽度为 32-bit 时, 地址偏移量为 10×4=40; 若数据宽度为 16-bit 时, 偏移量为 10×2=20; 若数据宽度为 8-bit 时, 偏移量为 10×1=10。 下次传输的目标地址=当前传输的目标地址±地址偏移量	RW

 **说明**

1. 使用目标设备不连续传输需要配置 DMA_CHCTLx 寄存器的 DNSEQEN 位有效, 并且配置 DMA_CHCTLx 寄存器的 DINC 位使地址更新方式为自增或自减。

2. 该寄存器在本通道的重置功能打开时无效。

16.5.23 目标地址不连续传输控制寄存器 B (DMA_DNSEQCTLBx) (x=0~7)

偏移地址 (Offset) : 0x54+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DNSCNTB[11:0]												DNSDIST[19:0]																			
Access	RW												RW																			

位/位域	标记	位名	功能描述	读写
31:20	DNSCNTB[11:0]	目标地址跳转的数据量	设置目标地址跳转前传输的数据量大小。 目标设备在每传输 SNSCNTB 个数据块后目标地址按 DNSDIST 指定地址间距跳转。数据块大小由 DMA_DTCTLx.BLKSIZE 与 DMA_CHCTLx.HSIZE 决定。	RW
19:0	DNSDIST[19:0]	目标不连续区域地址间距	当不连续地址传输时，设置目标设备两个不连续区域的间距。跳转的方向根据通道控制寄存器 DMA_CHCTLx.DINC 的值向前或向后跳转，详情请参见 不连续式重置示意图 。当 DMA_CHCTLx.DINC 设为地址固定时，不连续地址传输将无效。 地址间距将根据数据的宽度 (DMA_CHCTLx.HSIZE) 设置的 bit 数和 DNSDIST 的值进行计算。 地址间距=DNSDIST × (DMA_CHCTLx.HSIZE(bit) / 8) 例如，当 DNSDIST 设为 10，数据宽度为 32-bit 时，地址间距为 10×4=40；若数据宽度为 16-bit 时，间距为 10×2=20；若数据宽度为 8-bit 时，间距为 10×1=10。 下次传输的目标地址=当前目标不连续区域首地址±地址间距	RW

说明

1. 使用目标设备不连续传输需要配置 DMA_CHCTLx 寄存器的 DNSEQEN 位有效，并且配置 DMA_CHCTLx 寄存器的 DINC 位使地址更新方式为自增或自减。

2. 该寄存器只在本通道的重置功能打开时有效，替代寄存器 DMA_DNSEQCTLx。通道重置功能关闭时无效。

16.5.24 链指针寄存器 (DMA_LL Px) (x=0~7)

偏移地址 (Offset) : 0x58+0x40*x

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	LLP[31:2]																															Reserved
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:2	LLP[31:2]	链指针	连锁传输有效时, 设置下一次传输的描述符所在的地 址, 地址为字对齐, 即 LLP[1:0]固定为 0	RW
1:0	Reserved	保留	-	-

16.5.25 通道控制寄存器 (DMA_CHCTLx) (x=0~7)

偏移地址 (Offset) : 0x5C+0x40*x

Offset	Bit Position																																		
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00001000																																		
Name	Reserved																HPROT[3:2]	Reserved	IE	LLPRUN	LLPEN	HSIZE	DNSEQEN	SNSEQEN	DRPTEN	SRPTEN	DINC	SINC							
Access																	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW							

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:14	HPROT[3:2]	AHB 总线中的 HPROT 值	HPROT[3:2]位控制 AHB 总线的 HPROT[3:2]。 当使用 DMA 访问 DMC 或 SMC 时, 请将 HPROT[3:2] 设为 0b11; 其他情况请将 HPROT[3:2]设为 0b00。	RW
13	Reserved	保留	-	-
12	IE	中断使能位	配置该通道是否产生中断 ● 0b0: 该通道不产生中断 ● 0b1: 该通道产生中断	RW
11	LLPRUN	连锁传输模式选 择	连锁传输有效时, 设置当前传输完成时, 载入链指针 指向的新描述符后是否立即开始新描述符对应的传输 ● 0b0: 不立即传输, 等待下一个传输请求产生后开 始传输 ● 0b1: 新描述符载入后, 立即开始传输	RW
10	LLPEN	连锁传输使能	● 0b0: 连锁传输无效 ● 0b1: 连锁传输有效	RW

位/位域	标记	位名	功能描述	读写
9:8	HSIZE	传输数据的宽度	0b00: 8-bit 0b01: 16-bit 0b10/0b11: 32-bit	RW
7	DNSEQEN	目标地址不连续传输使能	0b0: 不允许不连续地址传输 0b1: 允许不连续地址传输	RW
6	SNSEQEN	源地址不连续传输使能	0b0: 不允许不连续地址传输 0b1: 允许不连续地址传输	RW
5	DRPTEN	目标地址重复传输功能使能位	设置是否允许目标地址重新载入初始值 0b0: 不重新载入初始值 0b1: 重新载入初始值, 进行地址重复传输	RW
4	SRPTEN	源地址重复传输功能使能位	设置是否允许源地址重新载入初始值 0b0: 不重新载入初始值 0b1: 重新载入初始值, 进行地址重复传输	RW
3:2	DINC	目标地址的更新方式	0b00: 固定 0b01: 递增 0b10/0b11: 递减	RW
1:0	SINC	源地址的更新方式	0b00: 固定 0b01: 递增 0b10/0b11: 递减	RW

16.5.26 传输源地址监视寄存器 (DMA_MONSARx) (x=0~7)

偏移地址 (Offset) : 0x60+0x40*x

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	SAR																																					
Access	RO																																					

位/位域	标记	位名	功能描述	读写
31:0	SAR	传输源地址	设置传输源地址  说明 - 传输数据宽度为 16-bit 时, 即 DMA_CHCTLx.HSIZE=0b01, SAR[0]无效。 - 传输数据宽度为 32-bit 时, 即 DMA_CHCTLx.HSIZE=0b10/0b11, SAR[1:0]无效。	RO



说明

传输源地址寄存器 (DMA_SARx) 在 DMA 传输前后保持不变, 而传输源地址监视寄存器 (DMA_MONSARx) 会在 DMA 每完成一个数据块传输后更新。DMA_MONSARx.SAR 的更新方式为: 按寄存器设置的固定/递增/递减/重复跳转/不连续跳转方式更新为下一次传输的地址。

16.5.27 传输目标地址监视寄存器 (DMA_MONDARx) (x=0~7)

偏移地址 (Offset) : 0x64+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RO DAR																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DAR	传输目标地址	设置传输目标地址 说明 ● 传输数据宽度为 16-bit 时，即 DMA_CHCTLx.HSIZE=0b01，DAR[0]无效。 ● 传输数据宽度为 32-bit 时，即 DMA_CHCTLx.HSIZE=0b10/0b11，DAR[1:0]无效。	RO

说明

传输目标地址寄存器 (DMA_DARx) 在 DMA 传输前后保持不变，而传输目标地址监视寄存器 (DMA_MONSARx) 会在 DMA 每完成一个数据块传输后更新。DMA_MONDARx.DAR 的更新方式为：按寄存器设置的固定/递增/递减/重复跳转/不连续跳转方式更新为下一次传输的地址。

16.5.28 数据控制监视寄存器 (DMA_MONDTCTLx) (x=0~7)

偏移地址 (Offset) : 0x68+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	CNT[15:0]																Reserved				BLKSIZE[9:0]											
Access	RO																				RO											

位/位域	标记	位名	功能描述	读写
31:16	CNT[15:0]	传输次数	剩余传输次数  说明 传输次数为 0 时 (DMA_DTCTLx.CNT=0x0000) , 该位域保持为 0。	RO
15:10	Reserved	保留	-	-
9:0	BLKSIZE[9:0]	数据块的大小	数据块的大小  说明 该位域为 0 时, 表示每次传输 1024 个数据。	RO

16.5.29 重复区域计数器监视寄存器 (DMA_MONRPTx) (x=0~7)

偏移地址 (Offset) : 0x6C+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved						DRPT[9:0]								Reserved						SRPT[9:0]											
Access							RO														RO											

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:16	DRPT[9:0]	目标地址重复区域大小	-	RO
15:10	Reserved	保留	-	-
9:0	SRPT[9:0]	源地址重复区域大小	-	RO

 **说明**

重复区域大小寄存器 (DMA_RPTx) 在 DMA 传输前后保持不变，而重复区域计数控制监视寄存器 (DMA_MONRPTx) 会在 DMA 每完成一个数据块传输后更新。

DMA_MONRPTx.SRPT[9:0]和 DMA_MONRPTx.DRPT[9:0]的更新方式为：

- 通道重置无效时：减去块大小，减到 0 时重新载入 DMA_RPTx 设定值。
- 通道重置有效时：减 1，减到 0 时重新载入初始值。

16.5.30 源地址不连续传输计数器监视寄存器 (DMA_MONSNSEQCTLx) (x=0~7)

偏移地址 (Offset) : 0x70+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SNSCNT[11:0]												SOFFSET[19:0]																			
Access	RO												RO																			

位/位域	标记	位名	功能描述	读写
31:20	SNSCNT[11:0]	源地址跳转的数据量	源地址跳转前传输的数据量大小	RO
19:0	SOFFSET[19:0]	源地址跳转的地址偏移量	源地址跳转的偏移量	RO

 **说明**

源设备不连续地址传输控制寄存器 (DMA_SNSEQCTLx) 在 DMA 传输前后保持不变，而源设备不连续传输计数器监视寄存器 (DMA_MONSNSEQCTLx) 会在 DMA 每完成一个数据块传输后更新。

DMA_MONSNSEQCTLx.SNSCNT[11:0]的更新方式为：

- 通道重置无效时：减去块大小，减到 0 时重新载入 DMA_SNSEQCTLx 初始值。
- 通道重置有效时：减 1，减到 0 时重新载入初始值。

16.5.31 目标地址不连续传输计数器监视寄存器 (DMA_MONDNSEQCTLx) (x=0~7)

偏移地址 (Offset) : 0x74+0x40*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DNSCNT[11:0]												DOFFSET[19:0]																			
Access	RO												RO																			

位/位域	标记	位名	功能描述	读写
31:20	DNSCNT[11:0]	目标地址跳转的数据量	目标地址跳转前传输的数据量大小	RO
19:0	DOFFSET[19:0]	目标地址跳转的地址偏移量	目标地址跳转的偏移量	RO

说明

目标设备不连续地址传输控制寄存器 (DMA_DNSEQCTLx) 在 DMA 传输前后保持不变，而目标设备不连续传输计数器监视寄存器 (DMA_MONDNSEQCTLx) 会在 DMA 每完成一个数据块传输后更新。

DMA_MONDNSEQCTLx.DNSCNT[11:0]的更新方式为：

- 通道重置无效时：减去块大小，减到 0 时重新载入 DMA_DNSEQCTLx 初始值。
- 通道重置有效时：减 1，减到 0 时重新载入初始值。

16.6 注意事项

DMA 的寄存器只支持 32-bit 读写，8/16-bit 读写操作无效。

17 电压比较器 (CMP)

17.1 简介

电压比较器 (Comparator, 以下简称 CMP) 是将两个模拟电压进行比较并且输出比较结果的外设模块。本产品搭载两组共 4 个比较通道：CMP1 和 CMP2、CMP3 和 CMP4。

17.2 主要特性

- 4 个比较通道可独立进行普通比较
- 同组的两个比较通道组合使用可实现最多 2 个窗口比较
- 每个比较通道的正/负端电压均有多个输入源 (IO、PGA、DAC) 供选择
- 迟滞电压可配置
- 噪声滤波器可以对比较器输出滤波, 3 种采样时钟可选
- 可使用定时器 PWM 进行比较器空白窗口控制
- 可在比较结果的变化边沿产生中断、触发其他外设以及唤醒 STOP 模式
- 比较结果可通过寄存器监视, 也可输出到外部管脚 VCOUT
- 比较结果可用于紧急刹车 (EMB) 控制事件

17.3 功能说明

17.3.1 功能框图

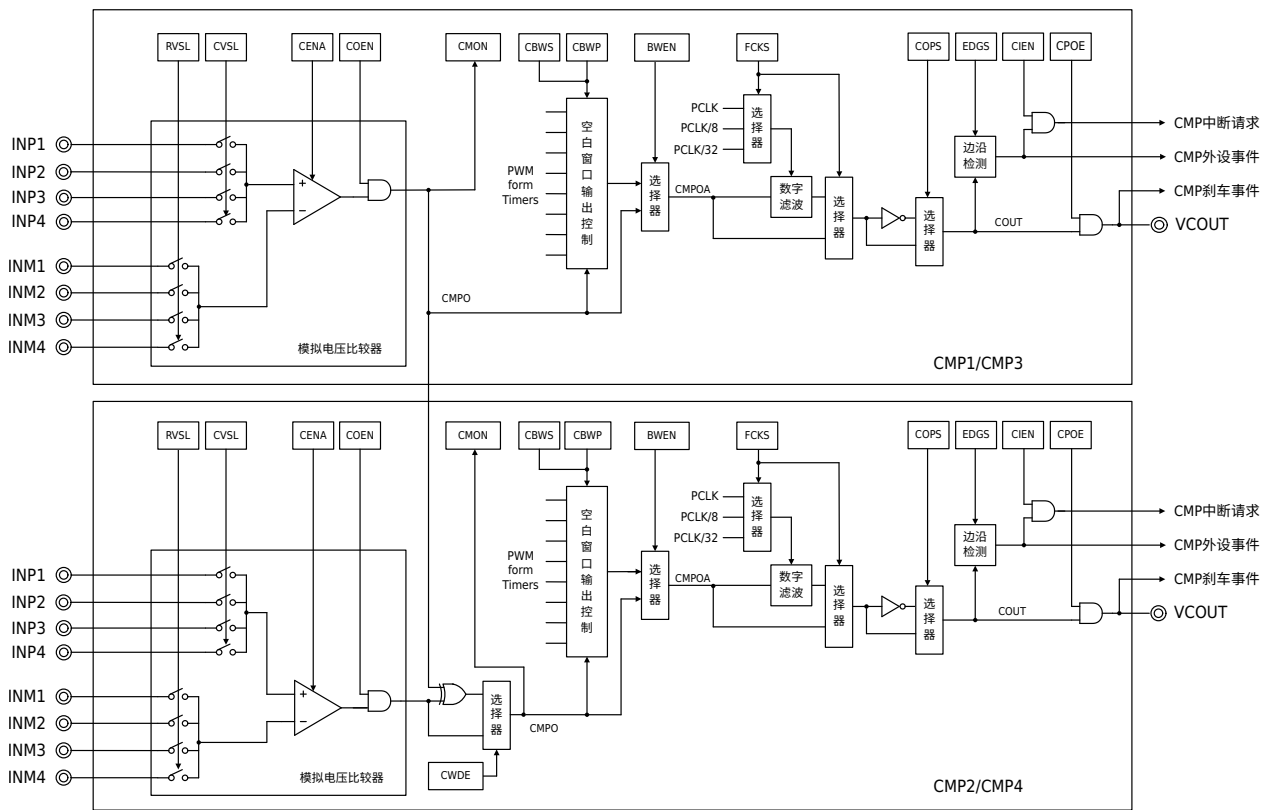


图 17-1 CMP 功能框图

表 17-1 CMP 引脚一览表

引脚名称	输入/输出	功能
AVCC	输入	模拟电源
AVSS	输入	模拟地
CMP1_INP2	输入	CMP1 正端模拟输入
CMP1_INP3	输入	CMP1 正端模拟输入
CMP1_INP4	输入	CMP1 正端模拟输入
CMP123_INM3	输入	CMP1~3 负端模拟输入
CMP1_INM4	输入	CMP1 负端模拟输入
CMP2_INP3	输入	CMP2 正端模拟输入
CMP2_INP4	输入	CMP2 正端模拟输入
CMP2_INM4	输入	CMP2 负端模拟输入
CMP3_INP2	输入	CMP3 正端模拟输入
CMP3_INP3	输入	CMP3 正端模拟输入
CMP3_INP4	输入	CMP3 正端模拟输入
CMP3_INM4	输入	CMP3 负端模拟输入
CMP4_INP3	输入	CMP4 正端模拟输入
CMP4_INP4	输入	CMP4 正端模拟输入
CMP4_INM3	输入	CMP4 负端模拟输入
CMP4_INM4	输入	CMP4 负端模拟输入
CMP1_VCOUT	输出	CMP1 的外部端口输出
CMP2_VCOUT	输出	CMP2 的外部端口输出
CMP3_VCOUT	输出	CMP3 的外部端口输出
CMP4_VCOUT	输出	CMP4 的外部端口输出
CMP_VCOUT	输出	CMP1~4 的外部端口输出

17.3.2 输入选择

CMP1~4 的模拟输入分别按照下表进行选择。

表 17-2 CMP 正端输入一览表

比较通道	CVSL=0b0001 (INP1)	CVSL=0b0010 (INP2)		CVSL=0b0100 (INP3)		CVSL=0b1000 (INP4)
CMP1	PGA1_BP	P2SL=0b001	PGA1	P3SL=0b01	CMP1_INP3	CMP1_INP4
		P2SL=0b010	PGA2	P3SL=0b10	CMP2_INP3	
		P2SL=0b100	CMP1_INP2	-	-	
CMP2	PGA2_BP	PGA2		CMP2_INP3		CMP2_INP4

比较通道	CVSL=0b0001 (INP1)	CVSL=0b0010 (INP2)		CVSL=0b0100 (INP3)		CVSL=0b1000 (INP4)
CMP3	PGA3_BP	P2SL=0b001	PGA3	P3SL=0b01	CMP3_INP3	CMP3_INP4
		P2SL=0b010	PGA4	P3SL=0b10	CMP4_INP3	
		P2SL=0b100	CMP3_INP2	-	-	
CMP4	PGA4_BP	PGA4		CMP4_INP3		CMP4_INP4

 **说明**
表中 PGA1_BP/PGA2_BP/PGA3_BP/PGA4_BP 分别为 PGA1~4 输出的未经放大的电压信号，使用时也要将 PGA 设为有效。

表 17-3 CMP 负端输入一览表

比较通道	RVSL=0b0001 (INM1)	RVSL=0b0010 (INM2)	RVSL=0b0100 (INM3)	RVSL=0b1000 (INM4)
CMP1	DA1O1	DA1O2	CMP123_INM3	CMP1_INM4
CMP2	DA1O1	DA1O2	CMP123_INM3	CMP2_INM4
CMP3	DA2O1	DA2O2	CMP123_INM3	CMP3_INM4
CMP4	DA2O1	DA2O2	CMP4_INM3	CMP4_INM4

17.3.3 普通比较

CMP1~4 可以独立进行普通比较。以 CMP1 为例，设定步骤如下：

- 步骤 1. PWC_FCG3.Bit3 写 0，并等待至少 2μs；
- 步骤 2. 使能 CMP1 模块功能（PWC_FCG3.CMP12 写 0）；
- 步骤 3. 设定 CMP_PMSR.CVSL，选择正端电压；
设定 CMP_PMSR.RVSL，选择负端电压；
- 步骤 4. 将 CMP_MDR.CENA 设为 1，启动比较器；
- 步骤 5. 等待比较器的稳定时间 t_{CMP} （具体数值请参见《数据手册》的“电气特性”）；
- 步骤 6. 设定 CMP_FIR.FCKS，设置数字滤波的采样周期；
设定 CMP_FIR.EDGS 和 CMP_FIR.CIEN，设置中断发生的边沿；
- 步骤 7. 设定 CMP_OCR.CPOE、CMP_OCR.COPS，设置 VCOUT 的输出；
- 步骤 8. 将 CMP_OCR.COEN 设为 1，允许比较器输出。

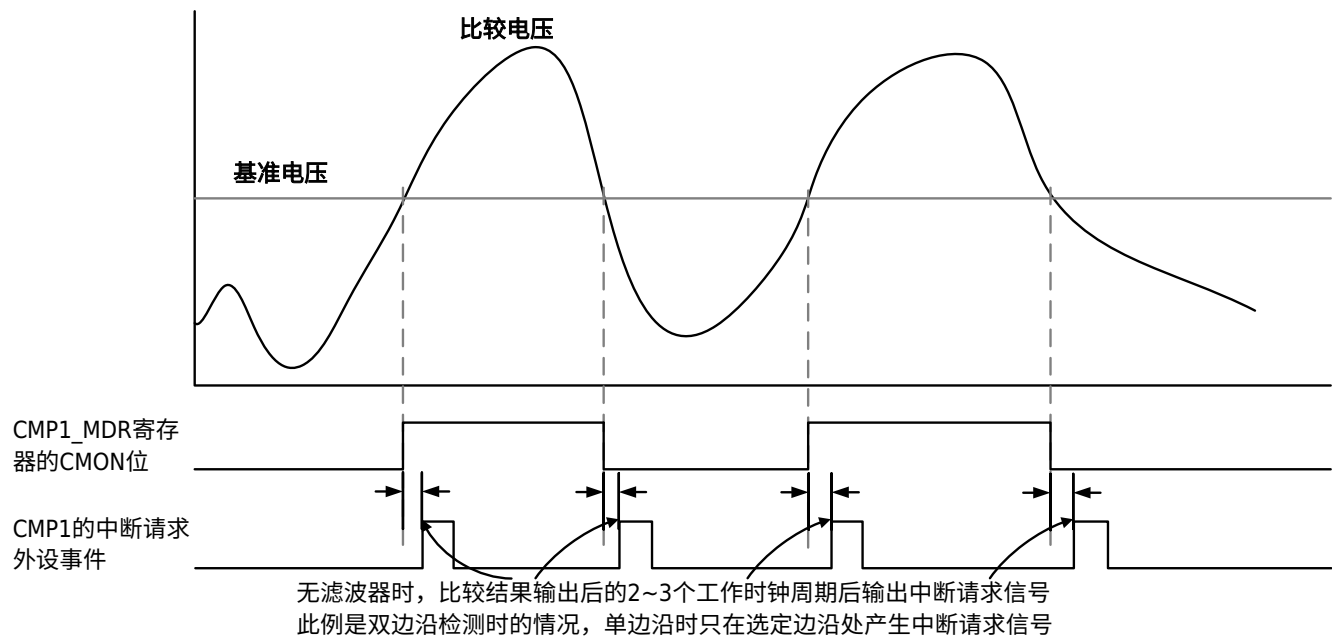


图 17-2 电压比较示意图

如上图所示，在正相输出时，CMP 允许输出后如果正端电压高于负端电压，CMP_MDR 寄存器的 CMON 位为 1；如果正端电压低于负端电压，则 CMON 位为 0。在反相输出时 CMON 的结果与正相输出时相反。

对于 CMP1 和 CMP3 来说，如果选择 INP2 或者 INP3 作为正端电压，那么还需要通过设定 CMP_VISR 选择电压输入源。具体请参见表 17-2。

17.3.4 窗口比较

CMP1 与 CMP2 或者 CMP3 与 CMP4 同时运行时可实现窗口比较。以 CMP1 与 CMP2 为例，设定步骤如下：

- 步骤 1. PWC_FCG3.Bit3 写 0，并等待至少 2 μ s；
- 步骤 2. 使能 CMP1 与 CMP2 模块功能（PWC_FCG3.CMP12 写 0）；
- 步骤 3. 设定 CMP1_PMSR.CVSL，选择正端电压；
设定 CMP1_PMSR.RVSL，选择的负端电压作为窗口下限；
- 步骤 4. 设定 CMP2_PMSR.CVSL，选择与 CMP1 相同的正端电压；
设定 CMP2_PMSR.RVSL，选择的负端电压作为窗口上限；
- 步骤 5. 将 CMP2_MDR.WDEN 设为 1，启动窗口比较；
- 步骤 6. 将 CMP1_MDR.CENA 和 CMP2_MDR.CENA 设为 1，启动比较器；
- 步骤 7. 等待稳定时间 t_{CMP} （具体数值请参见《数据手册》的“电气特性”）；
- 步骤 8. 设定 CMP2_FIR.FCKS，设置数字滤波器；
设定 CMP2_FIR.EDGS、CMP2_FIR.CIEN，设置中断和边沿检测条件；
- 步骤 9. 设定 CMP2_OCR.CPOE、CMP2_OCR.COPS，设置 VCOUT 的输出；
- 步骤 10. 将 CMP1_OCR.COEN、CMP2_OCR.COEN 设为 1，允许比较器输出。

窗口比较时，由 CMP2 完成对比较结果的监视、滤波、中断和输出等操作。

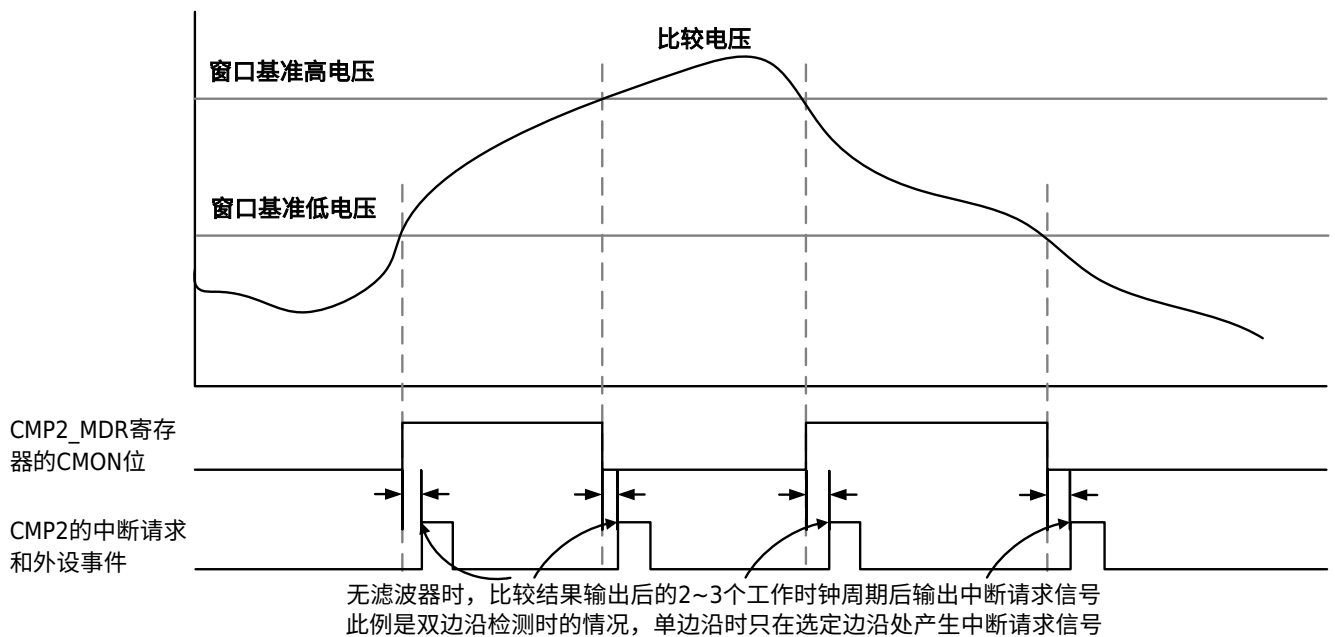


图 17-3 窗口比较示意图

如上图所示，在正相输出时，CMP 允许输出后如果正端电压处于两个负端电压组成的窗口内，CMP2_MDR 寄存器的 CMON 位为 1；如果正端电压处于两个负端电压组成的窗口外，则 CMON 位为 0。在反相输出时 CMON 的结果与正相输出时相反。

17.3.5 空白窗口

CMP 可将 Timer4、Timer6、TimerA 的输出作为空白窗口（Blank-Window）信号对比较结果 CMPOA 进行窗口输出控制，这为定时器 PWM 过零检测等应用场景提供了方便。

启动空白窗口输出 (CMP_OCR 寄存器的 BWEN 位为 1) 时, 通过 CMP_BWSR 寄存器的 CBWS 位选择一个或多个定时器 PWM 作为 Blank-Window 并通过 CMP_BWPR 寄存器的 CBWP 位为各 Blank-Window 选择有效电平。这样, 窗口输出 CMPOA 在 Blank-Window 有效时会被固定成由 CMP_OCR 寄存器的 BWOL 位设定的状态。

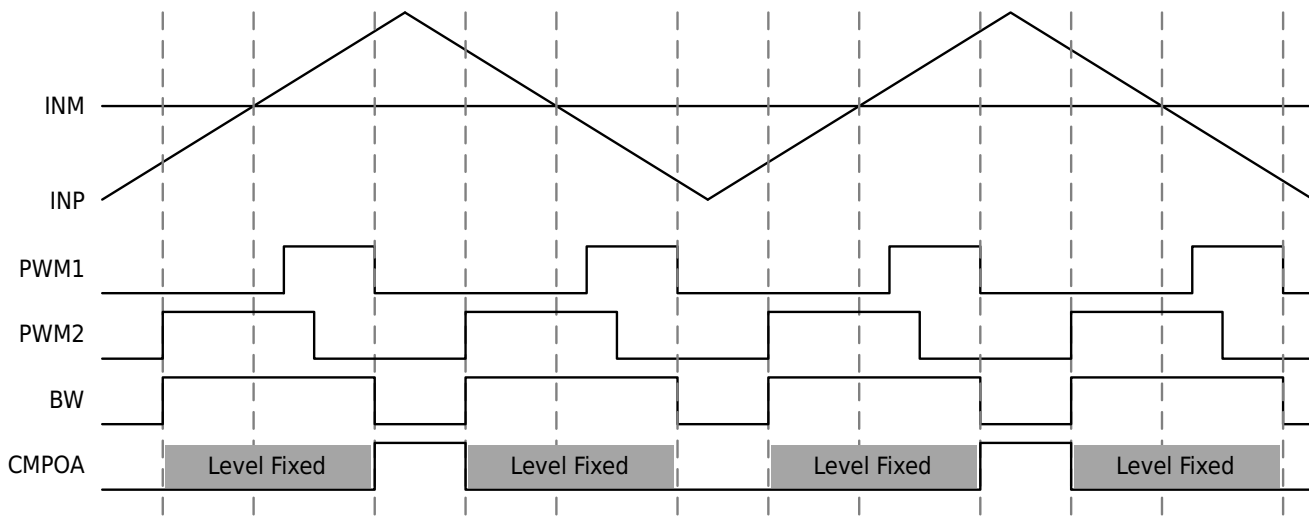


图 17-4 空白窗口输出示意图

如上图所示, 选择了两个 PWM 作为 Blank-Window 并且都是高电平有效, Blank-Window 有效时 CMPOA 的输出状态选择的是低电平。于是在 Blank-Window 信号为高电平时 CMPOA 被固定成了低电平, 而在 Blank-Window 信号为低电平时 CMPOA 则正常输出比较结果。

17.3.6 迟滞功能

为了避免噪声信号所引起的假输出, 每个比较通道都带有可配置的迟滞功能并通过 CMP_MDR 的 HYST 位选择迟滞电压。HYST 位设为 0b000 时迟滞功能关闭。

17.3.7 噪声滤波器

所有比较通道各包含一个噪声滤波器。噪声滤波器会在每个采样时钟周期对比较器输出 CMPOA 进行采样, 如果连续采样到三次相同的值, 则在下一个采样时钟周期将该值输出到 COUT, 否则 COUT 保持不变。采样时钟可以通过 CMP_FIR 寄存器的 FCKS 位选择, 当 FCKS 位设为 0b00 时噪声滤波器关闭。

17.3.8 外部管脚输出

可以将噪声滤波器输出的比较结果 COUT 输出到外部端口 VCOUT。通过 CMP_OCR 寄存器的 CPOE、COPS 位可以分别设定是否允许 VCOUT 输出以及输出极性 (正相输出或者反相输出)。复位后 VCOUT 对应的端口为通用端口且为输入状态, 因此还必须通过端口寄存器将其设定为 VCOUT 输出功能。

17.3.9 中断和事件

所有比较通道的结果均可以产生中断请求。使用时, 必须将 CMP_FIR 寄存器的 CIEN 位设置成 1 并通过 CMP_FIR 寄存器的 EDGS 位 (0b00 以外的值) 设定可以产生中断的比较结果的边沿。如果使用了噪声滤波和反相输出, 那么边沿检测的是经过噪声滤波和反相输出后的信号。

INTC_WKEN.CMP*WKEN=1 的时候, CMPO 可以唤醒低功耗停止模式, 系统将在 CMPO 由低变高时被唤醒并产生一次中断。

外设触发事件和中断请求的产生条件相同, 但必须通过目标外设的硬件触发源寄存器启动或关闭事件触发功能, 与 CIEN 位无关。

17.4 寄存器

17.4.1 寄存器总表

- CMP1 基地址：0x4004A000
- CMP2 基地址：0x4004A010
- CMP3 基地址：0x4004A400
- CMP4 基地址：0x4004A410

表 17-4 电压比较器 (CMP) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	CMP_MDR	比较器工作模式寄存器	8
0x01	CMP_FIR	比较器滤波和中断寄存器	8
0x02	CMP_OCR	比较器输出控制寄存器	8
0x03	CMP_PMSR	比较器正负端输入选择寄存器	8
0x04	CMP_BWSR	比较器输出空白窗口选择寄存器	16
0x06	CMP_BWPR	比较器输出空白窗口极性寄存器	16
0x08	CMP_VISR	比较器电压输入源选择寄存器 (仅 CMP1 和 CMP3 存在)	16

17.4.2 比较器工作模式寄存器 (CMP_MDR)

Offset	Bit Position							
0x00	7	6	5	4	3	2	1	0
Reset	0x00							
Name	CMON	HYST			Reserved		WDEN	CENA
Access	RO	RW					RW	RW

位/位域	标记	位名	功能描述	读写
7	CMON	比较结果监视	普通比较模式： 0b0：正端电压<负端电压 0b1：正端电压>负端电压 窗口比较模式（此位只对 CMP2 和 CMP4 有效）： 0b0：正端电压<下限负端电压，或者正端电压>上限负端电压 0b1：下限负端电压<正端电压<上限负端电压	RO
6:4	HYST	迟滞电压选择	0b000：0mV（无迟滞） 0b001：10mV 0b010：20mV 0b011：30mV 0b100：40mV 0b101：50mV 0b110：60mV 0b111：70mV  说明 上述均为典型值，误差范围请参见《数据手册》“电气特性”。	RW
3:2	Reserved	保留	-	-
1	WDEN	窗口比较选择	0b0：普通比较模式 0b1：窗口比较模式  说明 此位只存在于 CMP2 和 CMP4 中。设为窗口比较模式后，对窗口比较结果的滤波、边沿检测以及输出等控制均由 CMP2 和 CMP4 完成。	RW
0	CENA	比较器工作许可	0b0：电压比较器停止 0b1：电压比较器工作  说明 每次将 CENA 由 0 设为 1 后，需要等待约 300ns 的工作稳定时间才可以进行后续操作。	RW

17.4.3 比较器滤波和中断寄存器 (CMP_FIR)

Offset	Bit Position							
0x01	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved	CIEN	EDGS		Reserved		FCKS	
Access		RW	RW				RW	

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6	CIEN	比较器中断许可	0b0: 禁止比较器中断 0b1: 允许比较器中断	RW
5:4	EDGS	比较器输出边沿检测选择	0b00: 不检测比较器输出 COUT 的边沿 0b01: 检测比较器输出 COUT 的上升沿 0b10: 检测比较器输出 COUT 的下降沿 0b11: 检测比较器输出 COUT 的上升沿和下降沿  说明 改变 EDGS[1:0]可能会引起中断或者外设触发事件，因此请在中断禁止或者外设触发功能无效状态下设定该寄存器。寄存器设定后，请清除相应的中断标志。	RW
3:2	Reserved	保留	-	-
1:0	FCKS	比较器结果滤波采样选择	0b00: 不使用噪声滤波器 0b01: 使用噪声滤波器，通过 PCLK3 进行采样 0b10: 使用噪声滤波器，通过 PCLK3/8 进行采样 0b11: 使用噪声滤波器，通过 PCLK3/32 进行采样  说明 请在比较器输出禁止（即 CMP_OCR.COEN 为 0）时改写 FCKS[1:0]。FCKS[1:0]从 0b00 切换到其他值时，请使用经过 4 次采样后的滤波输出作为中断请求或外设触发事件。 改变 FCKS[1:0]可能会引起中断或者外设触发事件，因此请在中断禁止或者外设触发功能无效状态下设定该寄存器。寄存器设定后，请清除相应的中断标志。	RW

17.4.4 比较器输出控制寄存器 (CMP_OCR)

Offset	Bit Position							
0x02	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			BWOL	BWEN	CPOE	COPS	COEN
Access				RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4	BWOL	Blank-Window 输出电平	0b0: Blank-Window 有效期间 CMPOA 输出低电平 0b1: Blank-Window 有效期间 CMPOA 输出高电平	RW
3	BWEN	Blank-Window 使能	0b0: 禁止 0b1: 使能	RW
2	CPOE	VCOUT 输出允许	0b0: 禁止 VCOUT 输出 0b1: 允许 VCOUT 输出	RW
1	COPS	比较器输出极性选择	0b0: 比较器正相输出 0b1: 比较器反相输出  说明 请在比较器输出禁止（即 COEN 位为 0）时改写 COPS。改变 COPS 位可能会引起中断或者外设触发事件，因此请在中断禁止或者外设触发功能无效状态下设定该寄存器。寄存器设定后，请清除相应的中断标志。	RW
0	COEN	比较器输出允许	0b0: 禁止比较器输出（比较器输出固定成低电平） 0b1: 允许比较器输出	RW

17.4.5 比较器正负端输入选择寄存器 (CMP_PMSR)

Offset	Bit Position							
0x03	7	6	5	4	3	2	1	0
Reset	0x00							
Name	CVSL				RVSL			
Access	RW				RW			

位/位域	标记	位名	功能描述	读写
7:4	CVSL	正端输入选择	选择电压比较器的正端输入电压，所选输入详见 CMP 正端输入一览表： ● 0b0000：无正端电压输入 ● 0b0001：选择 INP1 作为正端电压 ● 0b0010：选择 INP2 作为正端电压 ● 0b0100：选择 INP3 作为正端电压 ● 0b1000：选择 INP4 作为正端电压 ● 其他：禁止设置  说明 请在比较器输出禁止（即 CMP_OCR.COEN 为 0）时改写 CVSL。每次改写 CVSL 后请等待 300ns 再允许比较器输出（即 CMP_OCR.COEN 设为 1）。改变 CVSL 可能会引起中断或者外设触发事件，因此请在中断禁止或者外设触发功能无效状态下设定该寄存器。寄存器设定后，请清除相应的中断标志。	RW
3:0	RVSL	负端输入选择	选择电压比较器的负端输入电压，所选输入详见 CMP 负端输入一览表： ● 0b0000：无负端电压输入 ● 0b0001：选择 INM1 作为负端电压 ● 0b0010：选择 INM2 作为负端电压 ● 0b0100：选择 INM3 作为负端电压 ● 0b1000：选择 INM4 作为负端电压 ● 其他：禁止设置  说明 请在比较器输出禁止（即 CMP_OCR.COEN 为 0）时改写 RVSL。每次改写 RVSL 后请等待 300ns 再允许比较器输出（即 CMP_OCR.COEN 设为 1）。改变 RVSL 可能会引起中断或者外设触发事件，因此请在中断禁止或者外设触发功能无效状态下设定该寄存器。寄存器设定后，请清除相应的中断标志。	RW

17.4.6 比较器电压输入源选择寄存器 (CMP_VISR) (仅 CMP1 和 CMP3 存在)

Offset	Bit Position															
0x08	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved										P3SL	Reserved	P2SL			
Access											RW		RW			

位/位域	标记	位名	功能描述	读写
15:6	Reserved	保留	-	-
5:4	P3SL	INP3 输入选择	CPM1: 选择 CMP1_PMSR.CVSL[3:0]为 0b0100 时的输入电压 0b00: 无电压输入 0b01/0b10: 所选输入详见 CMP 正端输入一览表 0b11: 禁止设置 CPM3: 选择 CMP3_PMSR.CVSL[3:0]为 0b0100 时的输入电压 0b00: 无电压输入 0b01/0b10: 所选输入详见 CMP 正端输入一览表 0b11: 禁止设置  注意 请在 CMP_PMSR.CVSL[2]为 0 时改写 P3SL。	RW
3	Reserved	保留	-	-
2:0	P2SL	INP2 输入选择	CPM1: 选择 CMP1_PMSR.CVSL[3:0]为 0b0010 时的输入电压 0b000: 无电压输入 0b001/0b010/0b100: 所选输入详见 CMP 正端输入一览表 - 其他: 禁止设置 CPM3: 选择 CMP3_PMSR.CVSL[3:0]为 0b0010 时的输入电压 0b00: 无电压输入 0b001/0b010/0b100: 所选输入详见 CMP 正端输入一览表 - 其他: 禁止设置  注意 请在 CMP_PMSR.CVSL[1]为 0 时改写 P2SL。	RW

17.4.7 比较器输出空白窗口选择寄存器 (CMP_BWSR)

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	CBWS15	CBWS14	CBWS13	CBWS12	CBWS11	CBWS10	CBWS9	CBWS8	CBWS7	CBWS6	CBWS5	CBWS4	CBWS3	CBWS2	CBWS1	CBWS0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:0	CBWSn (n=15~0)	窗口信号使能	Blank-Window 有效 (CMP_OCR.BWEN=1) 时选择用于 MASK 比较器输出的窗口信号, 各 Bit 对应的定时器 PWM 信号详见下表: 0b0: 不选择定时器 PWM 输出作为窗口信号 0b1: 选择定时器 PWM 输出作为窗口信号	RW

表 17-17 空白窗口 PWM 一览表

CBWS Bit	CMP1	CMP2	CMP3	CMP4
15	TMR4_1_OWL	TMR4_2_OWL	TMR4_3_OWL	TMR4_3_OWL
14	TMR4_1_OWH	TMR4_2_OWH	TMR4_3_OWH	TMR4_3_OWH
13	TMR4_1_OVL	TMR4_2_OVL	TMR4_3_OVL	TMR4_3_OVL
12	TMR4_1_OVH	TMR4_2_OVH	TMR4_3_OVH	TMR4_3_OVH
11	TMR4_1_OUL	TMR4_2_OUL	TMR4_3_OUL	TMR4_3_OUL
10	TMR4_1_OUH	TMR4_2_OUH	TMR4_3_OUH	TMR4_3_OUH
9	TMR6_4_PWMA	TMR6_8_PWMA	TMR6_4_PWMB	TMR6_8_PWMB
8	TMR6_3_PWMA	TMR6_7_PWMA	TMR6_3_PWMB	TMR6_7_PWMB
7	TMR6_2_PWMA	TMR6_6_PWMA	TMR6_2_PWMB	TMR6_6_PWMB
6	TMR6_1_PWMA	TMR6_5_PWMA	TMR6_1_PWMB	TMR6_5_PWMB
5	TMRA_2_PWM3	TMRA_4_PWM3	TMRA_3_PWM3	TMRA_4_PWM3
4	TMRA_2_PWM2	TMRA_4_PWM2	TMRA_3_PWM2	TMRA_4_PWM2
3	TMRA_2_PWM1	TMRA_4_PWM1	TMRA_3_PWM1	TMRA_4_PWM1
2	TMRA_1_PWM3	TMRA_3_PWM3	TMRA_1_PWM3	TMRA_2_PWM3
1	TMRA_1_PWM2	TMRA_3_PWM2	TMRA_1_PWM2	TMRA_2_PWM2
0	TMRA_1_PWM1	TMRA_3_PWM1	TMRA_1_PWM1	TMRA_2_PWM1

17.4.8 比较器输出空白窗口极性寄存器 (CMP_BWPR)

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	CBWP15	CBWP14	CBWP13	CBWP12	CBWP11	CBWP10	CBWP9	CBWP8	CBWP7	CBWP6	CBWP5	CBWP4	CBWP3	CBWP2	CBWP1	CBWP0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:0	CBWPn (n=15~0)	定时器窗口极性选择	设定 CMP_BWSR 选定的窗口信号的有效电平： ● 0b0：窗口信号为高电平时比较器输出关闭 ● 0b1：窗口信号为低电平时比较器输出关闭	RW

17.5 注意事项

17.5.1 模块停止功能

CMP 有模块停止功能，通过设定模块停止寄存器可将模块的数字部分关闭。

CMP1 和 CMP2 共用一个模块停止控制位 PWC_FCG3.Bit8。

CMP3 和 CMP4 共用一个模块停止控制位 PWC_FCG3.Bit9。

所有比较通道初始均为停止状态，设定模块工作时才可以访问各自的寄存器。相关寄存器设定请参见《电源控制 (PWC)》章节“动作模式与低功耗模式”。

17.5.2 模块停止时的功耗

CMP 在工作状态进入到模块停止状态时，比较器将继续保持工作，功耗等同于工作状态。如需进一步降低功耗，请将 CMP_MDR 寄存器的 CENA 位清 0。

17.5.3 停止低功耗模式时的功耗

芯片进入停止低功耗模式时，如果 CMP 处于工作状态，进入停止低功耗模式后将继续保持工作，功耗等同于进入停止低功耗模式之前的水平。如需进一步降低功耗，请在进入停止低功耗模式前将 CMP_MDR 寄存器的 CENA 位清 0。

18 模数转换模块 (ADC)

18.1 简介

12 位 ADC 是一种采用逐次逼近方式的模拟数字转换器。本 MCU 搭载 3 个 ADC 单元，单元 1 和 2 支持 16 个通道，单元 3 支持 20 个通道，可以转换来自外部引脚和芯片内部的模拟信号。模拟输入通道可以任意组合成一个序列，一个序列可以进行单次扫描转换，或连续扫描转换。支持对任意指定通道进行连续多次转换并对转换结果进行平均。ADC 模块还搭载模拟看门狗功能，对任意指定通道的转换结果进行监视，检测其是否超出用户设定的范围。

18.2 主要特性

- 高性能
 - ▶ 可配置 12 位、10 位和 8 位分辨率
 - ▶ ADC 数字接口时钟 PCLK4 和转换时钟 PCLK2（也称作 ADCLK）的频率比可设置为 1:1、2:1、4:1、8:1、1:2、1:4
PCLK2 可选与系统时钟 HCLK 异步的 PLL 时钟，此时频率比 PCLK4:PCLK2=1:1
PCLK2 频率最高支持 60MHz
 - ▶ 采样率：2.5Msps (PCLK2=60MHz，12 位，采样 11 周期，变换 13 周期)
 - ▶ 各通道采样时间独立编程
 - ▶ 各通道独立数据寄存器
 - ▶ 数据寄存器可配置左/右对齐方式
 - ▶ 连续多次转换平均功能
 - ▶ 过采样功能
 - ▶ 模拟看门狗，监视转换结果
 - ▶ 不使用时可以将 ADC 模块设定成停止状态
- 模拟输入通道
 - ▶ 最大 28 个外部模拟输入，单个 ADC 单元最大支持 20 通道
 - ▶ 2 个内部模拟信号：内部基准电压，VBAT 分压
- 转换开始条件
 - ▶ 软件设置转换开始
 - ▶ 周边外设同步触发转换开始
 - ▶ 外部引脚触发转换开始
- 转换模式
 - ▶ 2 个扫描序列 A、B，可任意指定单个或多个通道
 - ▶ 序列 A 单次扫描
 - ▶ 序列 A 连续扫描
 - ▶ 序列 A 数据缓冲
 - ▶ 双序列扫描，序列 A、B 独立选择触发源，序列 B 优先级高于 A
 - ▶ 协同工作模式（适用于具有两个或三个 ADC 的设备）
- 中断与事件信号输出
 - ▶ 序列 A 扫描结束中断和事件 ADC_EOCA
 - ▶ 序列 B 扫描结束中断和事件 ADC_EOCB
 - ▶ 模拟看门狗 0 比较中断和事件 ADC_CMP0

- ▶ 模拟看门狗 1 比较中断和事件 ADC_CMP1
- ▶ 上述的 4 个事件输出都可启动 DMA

18.3 功能说明

18.3.1 功能框图

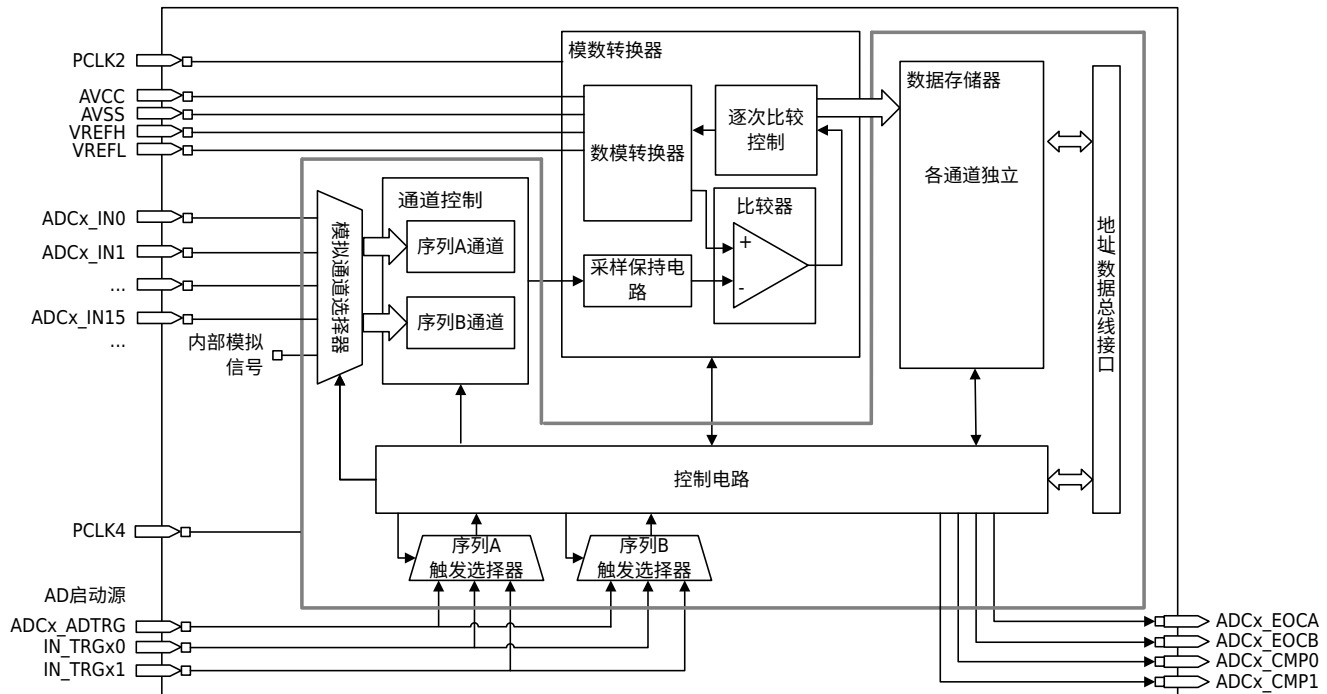


图 18-1 ADC 框图

本芯片搭载了 3 个 ADC 单元，每个单元的配置有所不同，具体参考下表：

表 18-1 各 ADC 单元规格

项目		单元 1 (ADC1)	单元 2 (ADC2)	单元 2 (ADC3)
电源		AVCC		
		AVSS/VREFL		
基准电压		VREFH		
模拟通道*1	CH0	ADC123_IN0	ADC123_IN0	ADC123_IN0
	CH1	ADC123_IN1	ADC123_IN1	ADC123_IN1
	CH2	ADC123_IN2	ADC123_IN2	ADC123_IN2
	CH3	ADC123_IN3	ADC123_IN3	ADC123_IN3
	CH4	ADC12_IN4	ADC12_IN4	ADC3_IN4
	CH5	ADC12_IN5	ADC12_IN5	ADC3_IN5
	CH6	ADC12_IN6	ADC12_IN6	ADC3_IN6
	CH7	ADC12_IN7	ADC12_IN7	ADC3_IN7
	CH8	ADC12_IN8	ADC12_IN8	ADC3_IN8

项目		单元 1 (ADC1)	单元 2 (ADC2)	单元 2 (ADC3)
模拟通道*1	CH9	ADC12_IN9	ADC12_IN9	ADC3_IN9
	CH10	ADC123_IN10	ADC123_IN10	ADC123_IN10
	CH11	ADC123_IN11	ADC123_IN11	ADC123_IN11
	CH12	ADC123_IN12	ADC123_IN12	ADC123_IN12
	CH13	ADC123_IN13	ADC123_IN13	ADC123_IN13
	CH14	ADC12_IN14	ADC12_IN14	ADC3_IN14
	CH15	ADC12_IN15/内部模拟信号	ADC12_IN15/内部模拟信号	ADC3_IN15/内部模拟信号
	CH16	-	-	ADC3_IN16
	CH17	-	-	ADC3_IN17
	CH18	-	-	ADC3_IN18
	CH19	-	-	ADC3_IN19
可编程增益放大器 PGA	PGA1/4	ADC123_IN0	ADC12_IN6	-
	PGA2	ADC123_IN1	-	-
	PGA3	ADC123_IN2	-	-
	COM	PGA123_VSS	PGA4_VSS	-
专用采样保持电路 SH	SH1	ADC123_IN0	-	-
	SH2	ADC123_IN1	-	-
	SH3	ADC123_IN2	-	-
硬件触发源	外部引脚	ADC1_ADTRG	ADC2_ADTRG	ADC3_ADTRG
	片内周边	IN_TRG10	IN_TRG20	IN_TRG30
		IN_TRG11	IN_TRG21	IN_TRG31



说明

ADC 中虚拟通道 CH0~CH15 与物理通道 ADCx_INy（实际的模拟输入源）可以设置寄存器进行自由映射，本表所示为复位后默认的映射关系。

18.3.2 ADC 时钟

ADC 模块需要使用 2 个时钟：数字接口时钟 PCLK4，模拟电路时钟 PCLK2。PCLK4 与 PCLK2 是同步关系，频率比率可设置为 1:1、2:1、4:1、8:1、1:2、1:4。
PCLK2 可以选择与 HCLK 异步的 PLL 时钟源，此时 PCLK4 与 PCLK2 相同，为同步同频率关系。



注意

PCLK2 的频率请设置在 1MHz~60MHz 以内。

18.3.3 通道选择

ADC 模块支持通道映射，即模块中的虚拟通道与实际的物理通道间的映射。虚拟通道是指 ADC 模块中假定的通道，如寄存器 ADC_CHSELRA 设置为 0x1 表示序列 A 选择转换 CH0，这个 CH0 就是虚拟通道，而寄存器 ADC_DR0，是虚拟通道 CH0 的转换结果寄存器。物理通道是指实际存在的模拟通道，包含外部引脚的模拟输入 ADCx_INy 以及内部基准电压等内部模拟信号。虚拟通道与物理通道的映射可以通过寄存器 ADC_CHMUXR 进行配置，具体参考寄存器描述。

 **说明**
本章中若无特别说明，通道 n 或 CHn 均表示虚拟通道。

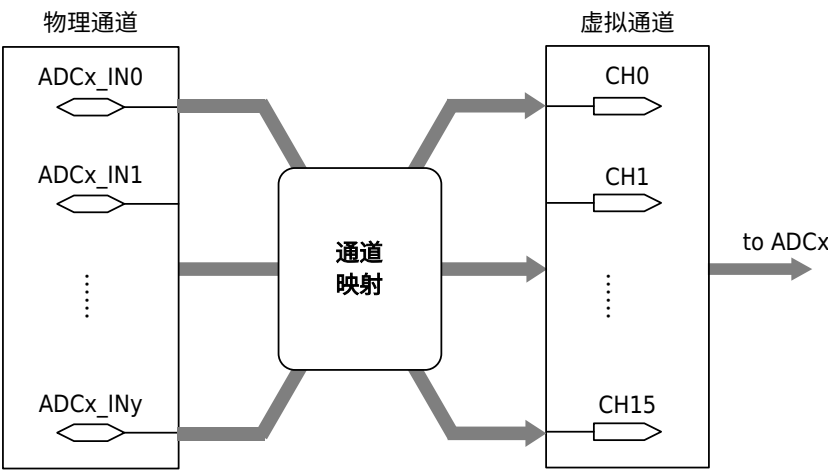


图 18-2 通道映射示意图

ADC 模块有多个通道，可配置为两个序列：序列 A、序列 B 进行转换。序列 A 和 B 配有独立的通道选择寄存器 ADC_CHSELRA、ADC_CHSELRB。寄存器每位代表一个通道，如 Bit0 位写 1 表示转换 CH0，写 0 表示不转换 CH0。两个序列可独立选择任意 1 个或多个通道进行转换。例如：ADC_CHSELRA 设置为 0x0055，ADCHSELRB 设置为 0x0002，则序列 A 的触发条件发生时，将依次转换 CH0、CH2、CH4 和 CH6 这 4 个通道。序列 B 的触发条件发生时，将转换 CH1 这一个通道。

其中，内部模拟信号与外部输入 ADCx_IN15 共用 CH15。设置 ADC_CHMUX 同时改变内部模拟信号和 ADCx_IN15 与 CHn 的映射关系。当需要转换内部模拟信号或 ADCx_IN15 时，需要先设置扩展通道寄存器 ADC_EXCHSELR，然后将 ADC_CHSELRA/B 中与之对应的虚拟通道位写 1（默认映射条件下为 Bit15）。当 ADC_EXCHSELR 设置为 0x0 时，表示转换外部模拟输入，当 ADC_EXCHSELR 设置为 0x1 时，表示转换内部模拟信号。2 个内部模拟信号的选择请参见《电源控制（PWC）》章节的 PWC_PWRC4 寄存器说明。

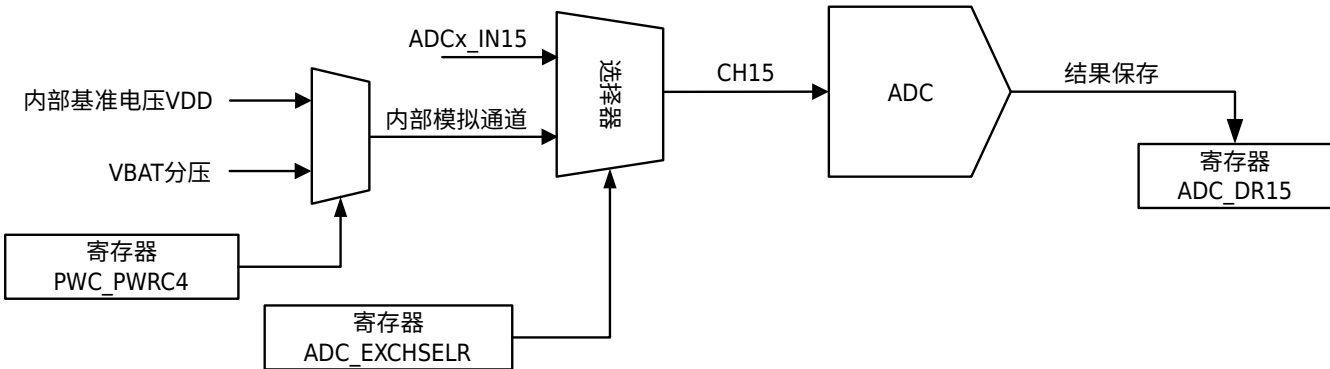


图 18-3 内部模拟信号选择

另外，大部分的物理通道可以输入到多个 ADC 单元，如 ADC123_IN0 可输入到 ADC1、ADC2、ADC3 三个单元，ADC12_IN4 可输入 ADC1、ADC2 两单元。结合多 ADC 协同工作模式以及通道映射功能，可以方便的对指定通道实现高采样率的转换。



注意

不要在序列 A 和 B 中选择相同的通道。对于不存在的通道，请不要设置相应的寄存器，保持其复位后的状态。

18.3.4 触发源选择

序列 A、序列 B 独立选择触发源。可选的触发源包括外部端口 ADCx_ADTRG，内部事件 IN_TRGx0/1。其中，端口 ADCx_ADTRG 下降沿输入有效。IN_TRGx0/1 由寄存器 ADC_TRGSEL0/1 设置，可以选择芯片内部丰富的事件源。此外，写寄存器 ADC_STR 可生成序列 A 软件触发信号，序列 A 软件触发只能在 ADC 处于待机状态时使用。软件触发与触发源选择寄存器 ADC_TRGSR 的设置无关。

18.3.5 序列 A 单次扫描模式

ADC 控制寄存器 ADC_CR0.MS 设为 0b000 选择序列 A 单次扫描模式。
该模式下，当寄存器 ADC_TRGSR 选择的序列 A 启动条件发生，或者 ADC_STR.STRT 位写 1 软件触发，ADC 启动，对序列 A 通道选择寄存器 ADC_CHSELRA 中选择的所有通道依次进行采样和转换，转换结果存入相应的数据寄存器 ADC_DR 中。ADC 转换过程中 ADC_STR.STRT 保持为 1，当所有通道转换结束后自动清 0，ADC 进入转换待机状态，等待下次触发条件的发生。
当所有通道转换结束时，序列 A 转换结束标志位 ADC_ISR.EOCADF 置 1，并产生序列 A 转换结束事件 ADC_EOCA，可以用此事件启动 DMA。若 ADC_ICR.EOCAIEN 为 1，中断许可的状态下，同时还产生序列 A 转换结束中断请求。

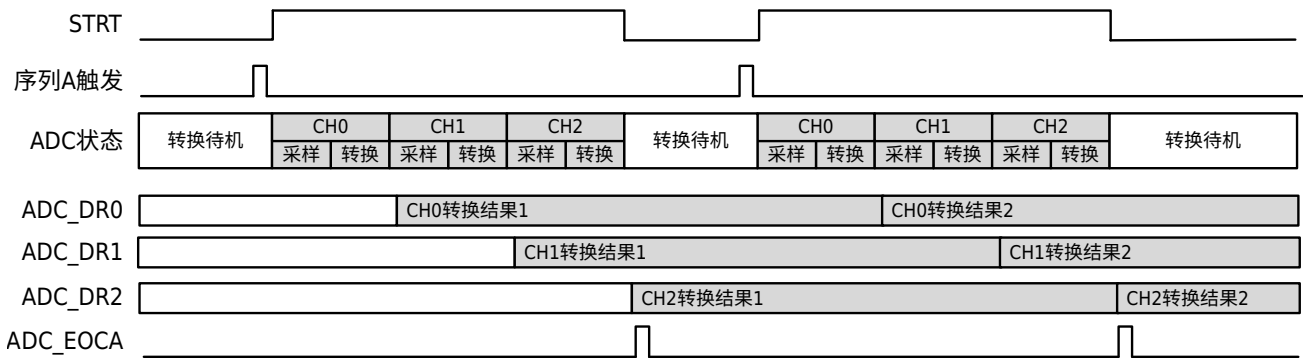


图 18-4 序列 A 单次扫描模式

序列 A 单次扫描模式的软件流程：

- 步骤 1. 确认 ADC_STR.STRT 为 0，ADC 处于转换待机状态；
 - 步骤 2. ADC 控制寄存器 ADC_CR0.MS 设为 0b000 选择序列 A 单次扫描模式；
 - 步骤 3. 设置序列 A 通道选择寄存器 ADC_CHSELRA；
 - 步骤 4. 设置采样时间寄存器 ADC_SSTR；
 - 步骤 5. ADC_STR.STRT 写 1 软件触发序列 A，或者设置寄存器 ADC_TRGSR 选择序列 A 触发条件；
 - 步骤 6. 查询序列 A 转换结束标志位 EOCADF；
 - 步骤 7. 读取各通道数据寄存器 ADC_DR；
 - 步骤 8. 写 0 清除 EOCADF 标志位，为下次转换准备。
- 上面的步骤 6~8 的 CPU 查询方式也可以替换成中断方式，利用 ADC_EOCA 中断来处理转换数据。或者利用 ADC_EOCA 事件启动 DMA 读取数据。

18.3.6 序列 A 连续扫描模式

ADC 控制寄存器 ADC_CR0.MS 设为 0b001 选择序列 A 连续扫描模式。

序列 A 连续扫描模式与序列 A 单次扫描模式动作类似，不同点在于，连续模式在所用通道转换结束后不是进入转换待机状态，而是重新开始转换序列 A。STRT 位也不会自动清 0。

当需要停止连续扫描时，对 STRT 位写 0，并读 STRT 确认为 0 以判断 ADC 进入转换待机状态。

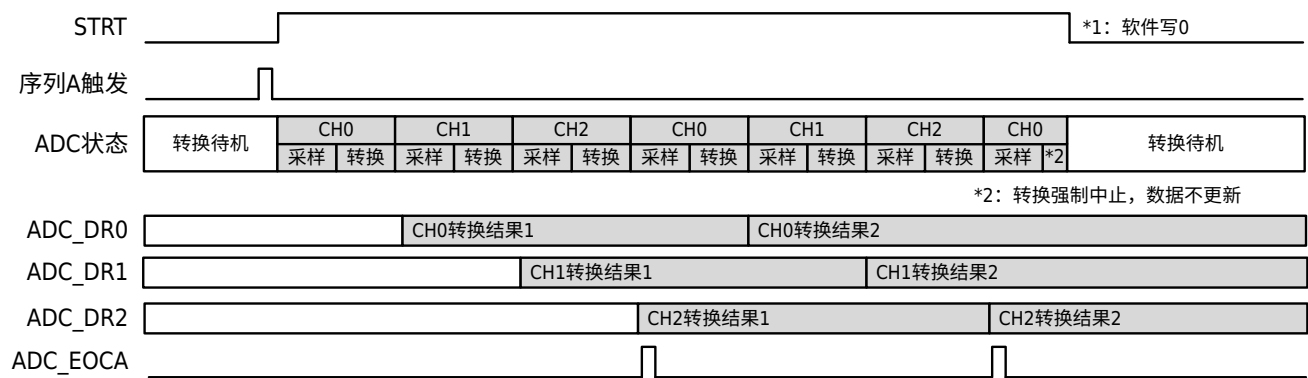


图 18-5 连续扫描

序列 A 连续扫描模式的软件流程：

- 步骤 1. 确认 ADC_STR.STRT 为 0，ADC 处于转换待机状态；
 - 步骤 2. ADC 控制寄存器 ADC_CR0.MS 设为 0b001 选择序列 A 连续扫描模式；
 - 步骤 3. 设置序列 A 通道选择寄存器 ADC_CHSELRA；
 - 步骤 4. 设置采样时间寄存器 ADC_SSTR；
 - 步骤 5. ADC_STR.STRT 写 1 软件触发序列 A，或者设置寄存器 ADC_TRGSR 选择序列 A 触发条件；
 - 步骤 6. 查询序列 A 转换结束标志位 EOCAF；
 - 步骤 7. 读取各通道数据寄存器 ADC_DR；
 - 步骤 8. 写 0 清除 EOCAF 标志位，为下次转换准备；
 - 步骤 9. 不需要继续转换时，对 STRT 位写 0，并读 STRT 确认为 0 以判断 ADC 进入转换待机状态。
- 上面的步骤 6~8 的查询方式也可以替换成中断方式，利用 ADC_EOCA 中断来处理转换数据。或者利用 ADC_EOCA 事件启动 DMA 读取数据。

 **注意**

由于是连续转换，每次扫描的间隔比较短，特别是只选择 1 个通道转换时。推荐使用 ADC_EOCA 事件启动 DMA 读取数据，避免查询方式下处理不及时导致数据丢失。

18.3.7 序列 A 数据缓冲模式

ADC 控制寄存器 ADC_CR0.MS 设为 0b100 选择序列 A 数据缓冲模式。

序列 A 数据缓冲模式与序列 A 单次扫描模式动作类似，不同点在于，数据缓冲模式在一个通道转换结束后不是直接开始下一个通道的转换，而是进入转换待机状态，直到启动条件再次发生才开始下一个通道的转换。

当所有通道转换结束时，序列 A 转换结束标志位 ADC_ISR.EOCAF 置 1，并产生序列 A 转换结束事件 ADC_EOCA，可以用此事件启动 DMA。若 ADC_ICR.EOCAIEN 为 1，中断许可的状态下，同时还产生序列 A 转换结束中断请求。

该模式下，可以用寄存器 ADC_CHSELRA 选择多个虚拟通道，再用通道映射功能，将这些虚拟通道映射到同一个物理通道，这样当所有通道转换结束后，可以依次从各个虚拟通道对应的 ADC_DR 寄存器中读取转换数据。

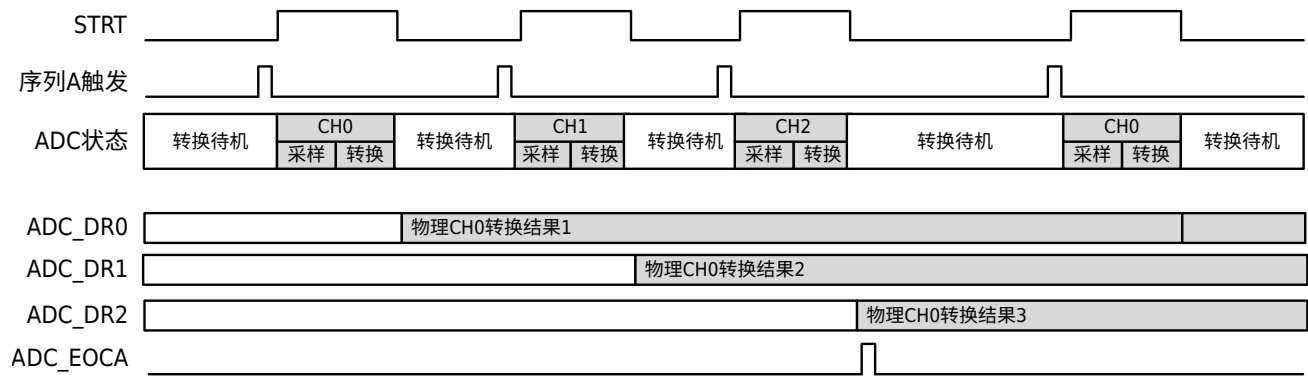


图 18-6 数据缓冲模式

序列 A 数据缓冲模式的软件流程：

- 步骤 1. 确认 ADC_STR.STRT 为 0，ADC 处于转换待机状态；
- 步骤 2. ADC 控制寄存器 ADC_CR0.MS 设为 0b100 选择序列 A 数据缓冲模式；
- 步骤 3. 设置序列 A 通道选择寄存器 ADC_CHSELRA；
- 步骤 4. 设置通道映射控制寄存器 ADC_CHMUXR；
- 步骤 5. 设置采样时间寄存器 ADC_SSTR；
- 步骤 6. ADC_STR.STRT 写 1 软件触发序列 A，或者设置寄存器 ADC_TRGSR 选择序列 A 触发条件；
- 步骤 7. 查询单次转换结束 ADC_STR.STRT=0；
- 步骤 8. 重复步骤 6~7 直到所有通道转换结束；
- 步骤 9. 查询序列 A 转换结束标志位 EOCAF；
- 步骤 10. 读取各通道数据寄存器 ADC_DR；
- 步骤 11. 写 0 清除 EOCAF 标志位，为下次转换准备。

上面的步骤 9~11 的查询方式也可以替换成中断方式，利用 ADC_EOCA 中断来处理转换数据。或者利用 ADC_EOCA 事件启动 DMA 读取数据。

18.3.8 双序列扫描模式

ADC 控制寄存器 ADC_CR0.MS 设为 0b010 或 0b011 选择双序列扫描模式，即序列 A 与序列 B 都可由各自选择的触发条件来启动扫描。

当 MS=0b010 时，序列 A 和 B 等效于两个独立的单次扫描序列。MS=0b011 序列 A 为连续扫描模式，B 为单次扫描模式。MS=0b110 序列 A 为数据缓冲模式，B 为单次扫描模式。

序列 A 由 ADC_TRGSR.TRGSELA 选择触发源，由 ADC_CHSELRA 选择转换的通道。序列 B 由 ADC_TRGSR.TRGSELB 选择触发源，由 ADC_CHSELRB 选择转换的通道。

当序列 A 全部通道转换结束时，序列 A 转换结束标志位 ADC_ISR.EOCAF 置 1，并产生序列 A 转换结束事件 ADC_EOCA，若 ADC_ISR.EOCAIEN 为 1，中断许可的状态下，同时产生序列 A 转换结束中断请求。当序列 B 全部通道转换结束时，序列 B 转换结束标志位 ADC_ISR.EOCBF 置 1，并产生序列 B 转换结束事件 ADC_EOCB，若 ADC_ISR.EOCBIEN 为 1，中断许可的状态下，同时产生序列 B 转换结束中断请求。

双序列扫描模式下，当序列 A 与序列 B 发生竞争时，序列 B 将会被优先处理，即序列 B 优先级高于序列 A。具体情况请参看下表。

表 18-2 序列 A 和 B 的各种竞争

ADC 转换	触发信号发生	处理方式	
		ADC_CR1.RSCHSEL=0	ADC_CR1.RSCHSEL=1
序列 A 转换过程中	序列 A 触发	触发信号无效	
	序列 B 触发	1. 序列 A 的转换被中断, 开始序列 B 转换 2. 序列 B 的转换完成后, 序列 A 从被中断的通道开始继续转换	1. 序列 A 的转换被中断, 开始序列 B 转换 2. 序列 B 的转换完成后, 序列 A 从第一个通道开始重新转换
序列 B 转换过程中	序列 A 触发	序列 B 全部通道转换完成后, 开始序列 A 转换	
	序列 B 触发	触发信号无效	
AD 空闲中, 序列 A, B 同时触发		序列 B 先启动, 全部通道转换完成后, 开始序列 A 转换	



图 18-7 双序列扫描模式 (序列 A 从被中断通道重新启动)

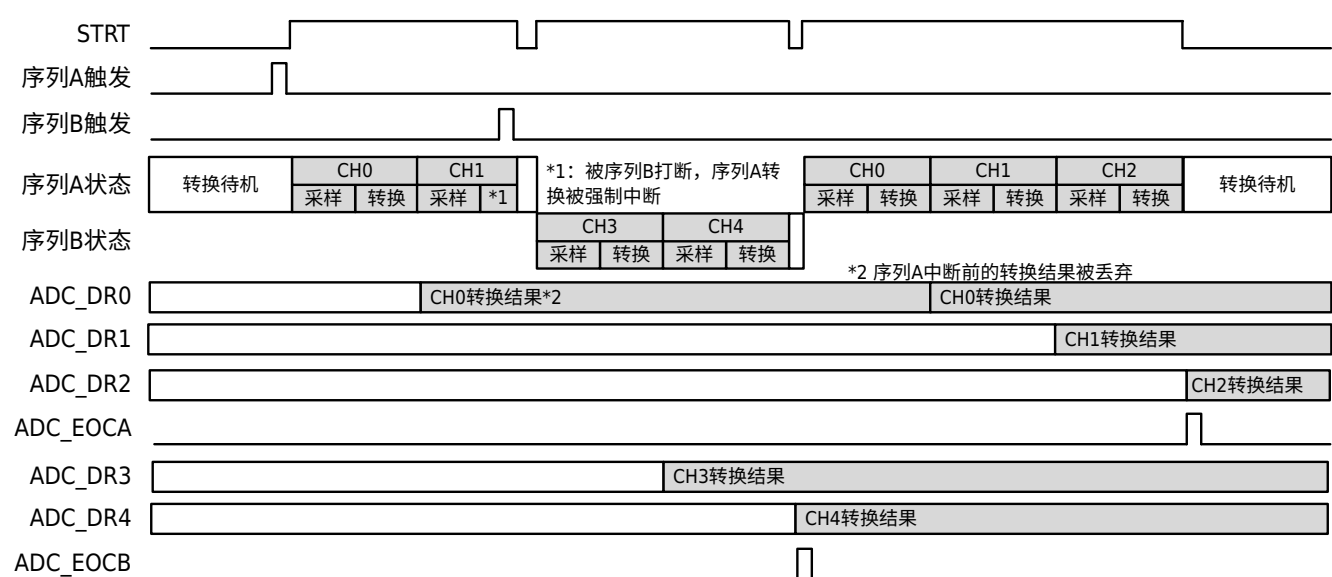


图 18-8 双序列扫描模式 (序列 A 从第一个通道重新启动)

双序列扫描模式的软件流程：

- 步骤 1. 确认 ADC_STR.STRT 为 0，ADC 处于转换待机状态；
- 步骤 2. ADC 控制寄存器 ADC_CR0.MS 设为 0b010 或 0b011 选择双序列扫描模式；
- 步骤 3. 设置寄存器 ADC_CR1.RSCHSEL 选择序列 A 被打断后启动方式；
- 步骤 4. 设置序列 A 通道选择寄存器 ADC_CHSELRA；
- 步骤 5. 设置序列 B 通道选择寄存器 ADC_CHSELRB；
- 步骤 6. 设置采样时间寄存器 ADC_SSTR；
- 步骤 7. 设置寄存器 ADC_TRGSR 选择序列 A 和 B 触发条件；
- 步骤 8. 通过查询 EOCAF、EOCBF，或者 ADC_EOCA、ADC_EOCB 中断，或者启动 DMA 在序列 A 或 B 转换结束后处理转换数据。



注意

不要在序列 A 和 B 中选择相同的通道。序列 A 和 B 不要选择相同的触发源。

18.3.9 模拟看门狗功能

模拟看门狗功能是指在通道的 ADC 转换结束时对转换结果进行比较，本 ADC 支持 2 个比较窗口：比较窗口 0、比较窗口 1。以比较窗口 0 为例，如下图所示，若转换结果在保护区域内，则生成看门狗比较中断和事件 ADC_CMP0。

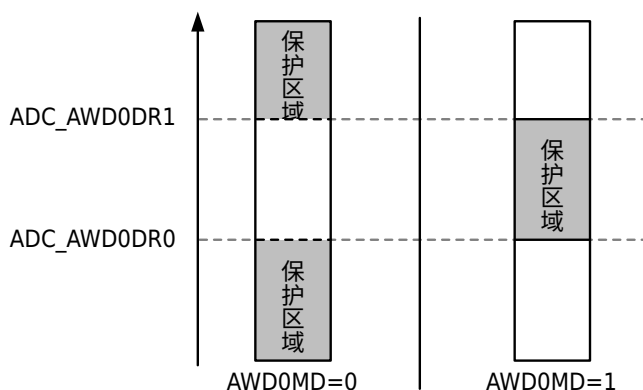


图 18-9 模拟看门狗保护区域（比较模式）

使用模拟看门狗功能的软件流程，以比较窗口 0 为例：

- 步骤 1. 设置阈值寄存器 AD_AWD0DR0、ADC_AWD0DR1；
- 步骤 2. 设置比较通道寄存器 ADC_AWD0CHSR，选择需要比较的通道；
- 步骤 3. 设置 ADC_AWDCR.AWD0MD 选择比较模式；
- 步骤 4. 设置 ADC_AWDCR.AWD0IEN 中断许可位；
- 步骤 5. 设置 ADC_AWDCR.AWD0EN 允许看门狗窗口 0 比较功能；
- 步骤 6. 根据前文，设置扫描模式，启动 ADC 进行转换；
- 步骤 7. 在 ADC_CMP0 中断中，或 ADC 转换结束后，查询比较状态寄存器 ADC_AWDSR.AWD0F，对比较结果进行做相应的处理。

比较窗口 1 的使用方法与比较窗口 0 相同。

两个比较窗口可以组合使用。当窗口组合功能有效时，在窗口 1 选的通道转换结束时，比较中断 ADC_CMP1 输出的不再是窗口 1 单独的比较结果，而是按照设置，对窗口 0 和 1 的比较结果进行逻辑或、逻辑与或者逻辑异或后产生的组合结果。

使用窗口组合比较功能的软件流程与窗口单独使用时相似，需要在设置好窗口 0 和 1 之后，追加设置 ADC_AWDCR.AWDCM 寄存器选择组合方式。

18.3.10 模拟输入的采样时间和转换时间

在单次扫描模式中，ADC 转换可选择软件设置，内部触发 IN_TRGx0/1 和外部引脚触发 ADCx_ADTRG 启动方式。在扫描转换延迟时间 t_D 后，ADC 模块才开始对模拟通道进行采样和转换，全部转换结束后经过转换结束延迟时间 t_{ED} 后进入待机状态，一次扫描才最终完成。连续扫描模式与单次扫描相似，只是在序列的第二次以及之后的启动时没有 t_D 时间。

单个通道的转换时间 $t_{CONV} = t_{SPL} + t_{CMP}$ 。其中 t_{SPL} 表示模拟输入的采样时间，可以根据输入阻抗设置寄存器 ADC_SSTR 调整采样周期数。 t_{CMP} 表示逐次比较时间，12 位精度 13 个 PCLK2，10 位精度 11 个 PCLK2，8 位精度 9 个 PCLK2。

一次扫描转换的时间 $t_{SCAN} = t_D + \sum t_{CONV} + t_{ED}$ 。其中 $\sum t_{CONV}$ 表示所有扫描通道的转换时间总和，由于可以独立设置采样时间 t_{SPL} ，各通道的转换时间 t_{CONV} 可以不同。

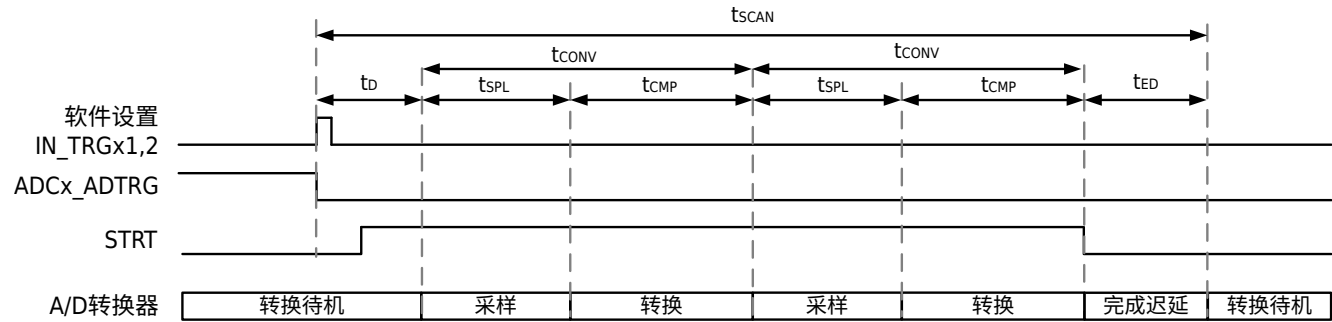


图 18-10 ADC 转换时间示意图

表 18-3 ADC 转换时间

标记		说明		条件			
				同步周边触发	异步周边触发 ⁽¹⁾	外部引脚触发	软件触发
t _D		扫描开始处理时间	ADC 空闲中，启动转换	4 PCLK2	3 PCLK4 + 4 PCLK2 + 1 PCLK4_SYNC	2 PCLK4 + 4 PCLK2	3 PCLK2
			序列 A 转换中被打断，启动序列 B 转换	5 PCLK2	4 PCLK4 + 5 PCLK2 + 1 PCLK4_SYNC	2 PCLK4 + 5 PCLK2	-
t _{CONV}	t _{SPL}	采样时间		ADC_SSTRx.SST x PCLK2			
	t _{CMP}	逐次转换时间	12 位分辨率	13 PCLK2			
			10 位分辨率	11 PCLK2			
			8 位分辨率	9 PCLK2			
t _{ED}		扫描完成处理时间		3 PCLK2			
t _{TD}		最小连续触发时间间隔		Σt _{CONV} + 6 PCLK2			

说明

- 异步周边触发是指 ADC 模块选择与系统时钟异步的 PLL 时钟动作时周边触发的情况。此时周边模块时钟与 ADC 模块时钟为异步关系。PCLK4_SYNC 表示 ADC 模块原来的同步时钟（即 CMU_SCFGR 设置的时钟），此时的 PCLK4、PCLK2 相同，均为异步的 PLL 时钟。

18.3.11 ADC 数据寄存器自动清除功能

当 ADC_CR0.CLREN 为 1，ADC 转换数据寄存器 ADC_DR 被 CPU 或者 DMA 读取后将自动被清除为 0x0000。

使用此功能能够检测到数据寄存器 ADC_DR 是否被更新。以下将举例说明。

- 当 ADC_CR0.CLREN 为 0，自动清除功能禁止的情况下，待测模拟量（0x0222）由于某种原因未被转换或结果未被更新到数据寄存器 ADC_DR 中，ADC_DR 寄存器继续保持前次转换值（0x0111）。ADC 转换

完成中断处理中将读取未被更新的 (0x0111) 。为检测 ADC 转换值是否有效，需要额外将以前的转换值存储到 RAM 中，通过对比转换结果来判断。

- 当 ADC_CR0.CLREN 为 1，自动清除功能许可的情况下，前次的转换结果 (0x0111) 被 CPU 或 DMA 读取后，ADC_DR 寄存器将自动被清除为 0x0000，此后进行 ADC 转换后，如果转换结果未被正确传送到 ADC_DR 寄存器中，ADC_DR 寄存器将保持 0x0000。这时如果在中断处理中读出了 0x0000，将很容易判断 ADC 转换数据是否正确被存储。

18.3.12 转换数据平均计算功能

ADC 转换平均计算功能是指对同一通道进行连续进行 2、4、8、16、32、64、128 或 256 次转换，并将转换结果进行平均后保存到数据寄存器的功能。使用平均计算功能可以去除一定的噪声成分使转换结果更加精确。

寄存器 ADC_CR0.AVCNT 设置连续转换的次数，寄存器 ADC_AVCHSELR 选择任意一个或多个需要平均的通道。

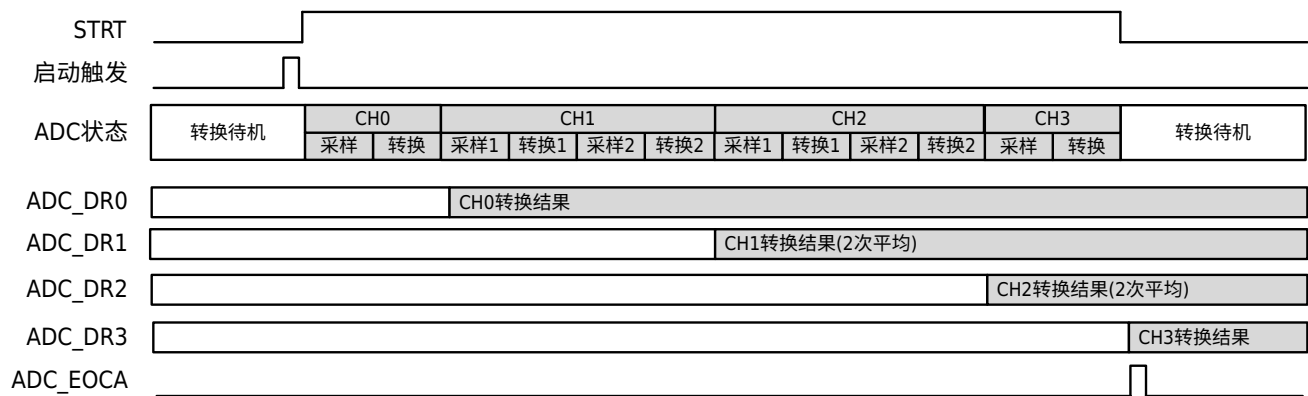


图 18-11 平均功能有效时的转换动作

上图中，序列 A 单次扫描模式，选择转换 CH0~3 这 4 个通道，其中 CH1、2 设置为 2 次平均模式。在扫描过程中，CH1、2 都会进行连续两次转换，并将平均后的结果保存至对应通道的数据寄存器 ADC_DR1、2 中。

18.3.13 过采样

通过设置寄存器 ADC_CR2.OVSMOD 为 1 和设置寄存器 ADC_AVCHSELR 选择任意一个或多个需要过采样的通道，使 AD 进入过采样模式，在该模式下，通过连续多次采样转换和结果移位，得到 16-bit 精度的转换结果。通过以下公式计算得到最终的转换结果：

$$AD\text{过采样结果} = \frac{1}{2^M} \times \sum_{n=1}^{n=N} AD\text{第}n\text{次转换结果}$$

采样率 N 通过寄存器 ADC_CR0.AVCNT 设置，可以设置 2 到 256 次采样。

除数系列 M 通过寄存器 ADC_CR2.OVSS 设置，最大可以设置向右移位 8 位。

18.3.14 可编程增益放大器 PGA

本产品搭载了可编程增益放大器 PGA，可以设置寄存器 ADC_PGACR，有效 PGA 电路并选择增益倍数，增益范围 x2~x32 可选择。此时，模拟输入先经过 PGA 电路进行放大，然后再输入到 ADC 模块进行转换。

使用 PGA 前请先设置寄存器 PWC_FCG3 的 Bit3 允许 PGA，并等待 2μs 的启动稳定时间。

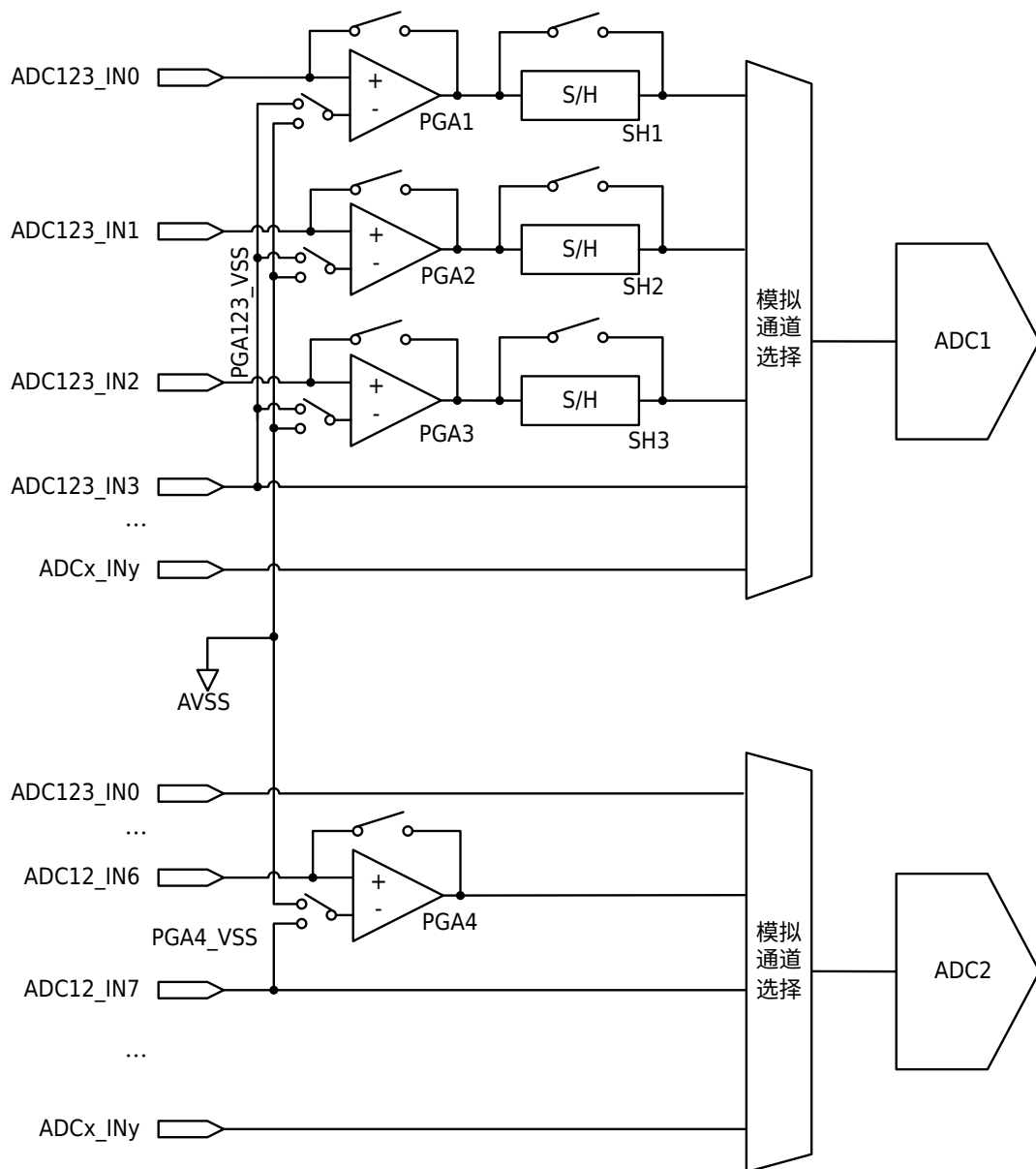


图 18-12 ADC1、ADC2 的 PGA 与 SH 通道示意图

如上图，ADC1 包含 PGA1~3 共 3 个 PGA 通道，分别与物理通道 ADC123_IN0~2 对应。ADC2 包含 PGA4 共 1 个 PGA 通道，与物理通道 ADC12_IN6 对应。

PGA 的参考地可以选择内部的 AVSS 也可以选择外部的 PGAx_VSS，当选择外部的 PGAx_VSS 时请将其接与 AVSS 相同的电压。

18.3.15 通道专用采样保持电路 SH

本产品搭载了 3 个专用的采样保持电路 (SH) 供 ADC1 的通道 CH0~2 使用。当专用采样保持电路有效时，每次序列启动时，先同时对所有 SH 有效的通道进行采样，然后再启动 ADC 开始依次对序列中的每个通道进行 ADC 转换。连续扫描模式时，序列在第二次以及之后的扫描启动时都会插入 SH 的采样时间。

可以设置寄存器 ADC_SHCR.SHSEL 位，有效 SH 电路，设置寄存器 ADC_SHCR.SHSST 修改 SH 电路的采样周期数。

使用 SH 前请先设置寄存器 PWC_FCG3 的 bit3 允许 SH，并等待 2us 的启动稳定时间。

参考下图，ADC1 包含 3 路 SH，可以与 PGA 组合使用，即先经过 PGA 放大，再由 SH 采样保持，输入到 ADC 进行转换。也可以无效 PGA，外部引脚输入直接由 SH 采样，再输入到 ADC 进行转换。

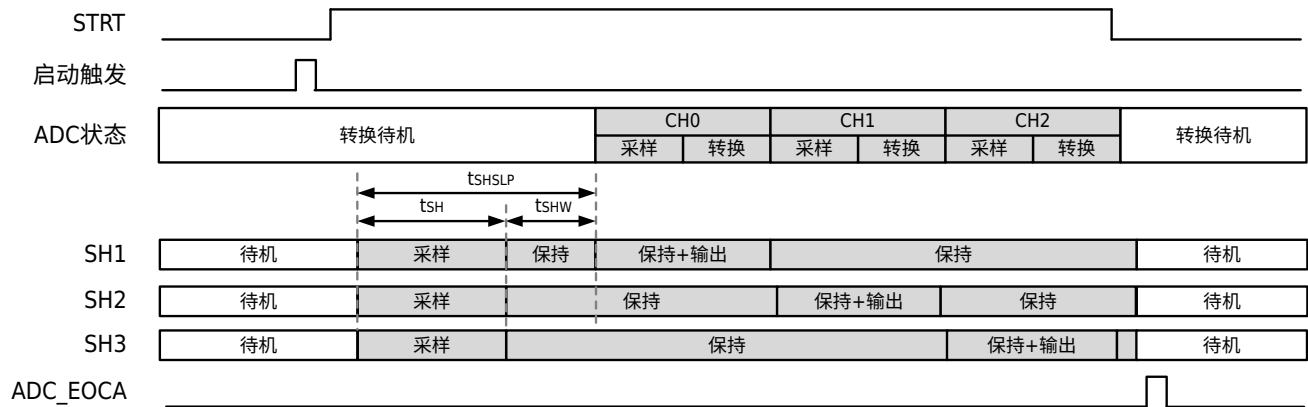


图 18-13 通道专用采样保持电路有效

如上图所示，CH0~2 的专用采样保持电路 SH1~3 均设置为有效。序列启动触发后，SH1~3 电路预先对通道 CH0~2 的模拟输入同时进行采样，即图中的 t_{SHSLP} 时间。然后对 SH1~3 保持的模拟电压进行 ADC 转换。其中， $t_{SHSLP}=t_{SH}+t_{SHW}$ 。 t_{SH} 表示专用采样保持电路采样时间，由寄存器 ADC_SHCR.SHSEL 设定，请设置在 $0.4\mu s$ 以上。 t_{SHW} 表示采样保持电路采样完成后的等待时间，为 12 个 PCLK2 周期。

**注意**

- 双序列扫描模式时，序列 A 中不要使用 SH。因为序列 A 可能会被序列 B 打断导致各专用采样保持电路的采样时间不一致而输出期待之外的转换结果。
- 寄存器 ADC_SHCR.SHSEL 写 1 后，SH 电路有效，对应的虚拟通道 CH0~2 固定映射为物理通道 ADC123_IN0~2，即当 SH 电路使能时，对应的虚拟通道不支持通道映射。

18.3.16 多 ADC 协同工作模式

在搭载有两个或三个 ADC 单元的芯片上，可以使用 ADC 协同工作模式。

在 ADC 协同工作模式下，ADC1 作为主控单元，通过 ADC1 的触发信号来同步 ADC2 和 ADC3 的转换。即 ADC2 和 ADC3 的序列 A 触发源选择寄存器 ADC_TRGSR.TRGSELA 设置无效。所有 ADC 单元均由 ADC1 的序列 A 触发源选择寄存器选定的触发源来触发。该模式下 ADC_STR.STRT 寄存器写 1 不会启动转换，即软件启动无效。

使用协同工作模式时，请禁止序列 B 动作，以免打乱同步。可以设置 ADC1、ADC2 两个 ADC 单元协同工作，也可以 ADC1、ADC2、ADC3 三个 ADC 单元协同工作。

ADC 可配置成以下四种协同工作模式：

- 单次并行触发模式
- 单次延迟触发模式
- 循环并行触发模式
- 循环延迟触发模式

单次并行触发模式

ADC1 的序列 A 触发条件同时触发处于协同工作模式的所有 ADC 单元，且只触发一次。

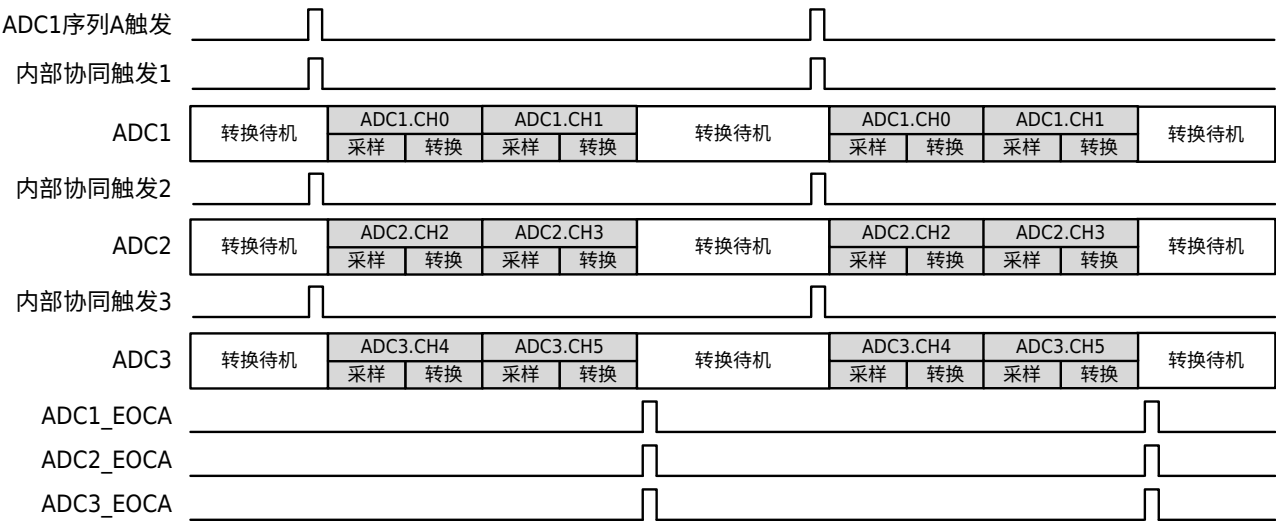


图 18-14 单次并行触发模式 (三个 ADC)



注意

禁止多个 ADC 同时对同一个模拟输入进行转换，一个模拟通道同一时间只能给一个 ADC 单元采样，否则精度不做保证，下同。

本模式的软件设定流程如下：

- 步骤 1.** 协同工作许可寄存器 `ADC_SYNCCR.SYNCEN` 写 0，确认协同工作无效。
- 步骤 2.** 设置 ADC1 单元：
- a. 确认 `ADC1_STR.STRT` 为 0，ADC1 处于转换待机状态
 - b. 设置控制寄存器 `ADC1_CR0.MS` 为 0b000（序列 A 单次扫描模式）或 0b001（序列 A 连续扫描模式）
 - c. 设置序列 A 通道选择寄存器 `ADC1_CHSELRA`
 - d. 设置采样时间寄存器 `ADC1_SSTR`
 - e. 设置序列 A 触发源选择寄存器 `ADC1_TRGSR`
- 步骤 3.** 设置 ADC2 单元：
- a. 确认 `ADC2_STR.STRT` 为 0，ADC2 处于转换待机状态
 - b. 设置控制寄存器 `ADC2_CR0.MS`，通道选择寄存器 `ADC2_CHSELRA`，通道采样时间寄存器 `ADC2_SSTR`
- 步骤 4.** 设置 ADC3 单元（三个 ADC 协同工作时）：
- a. 确认 `ADC3_STR.STRT` 为 0，ADC3 处于转换待机状态
 - b. 设置控制寄存器 `ADC3_CR0.MS`，通道选择寄存器 `ADC3_CHSELRA`，通道采样时间寄存器 `ADC3_SSTR`



说明

为保证 ADC2 与 ADC1 的同步工作，上述寄存器尽量与 ADC1 的寄存器设置相同的值。具体通道无需相同，只要保持通道数，以及对应通道的采样时间一致即可。



说明

同 ADC2，为保证 ADC3 与 ADC1 的同步工作，上述寄存器尽量与 ADC1 的寄存器设置相同的值。

- 步骤 5.** 设置协同模式控制寄存器 `ADC_SYNCCR.SYNCMD`，写 0b010（ADC1、ADC2 两个 ADC 协同工作）或者写 0b011（ADC1、ADC2、ADC3 三个 ADC 协同工作）；
- 步骤 6.** 协同工作许可寄存器 `ADC_SYNCCR.SYNCEN` 写 1，协同工作有效；
- 步骤 7.** 等待 ADC1 序列 A 触发源输入，在 ADC1、ADC2、ADC3 完成转换后处理结果。

单次延迟触发模式

ADC1 的序列 A 触发条件触发 ADC1 之后，经过设定的延迟后触发 ADC2 启动转换，再经过设定的延迟后触发 ADC3 启动转换，每个 ADC 单元只触发一次。

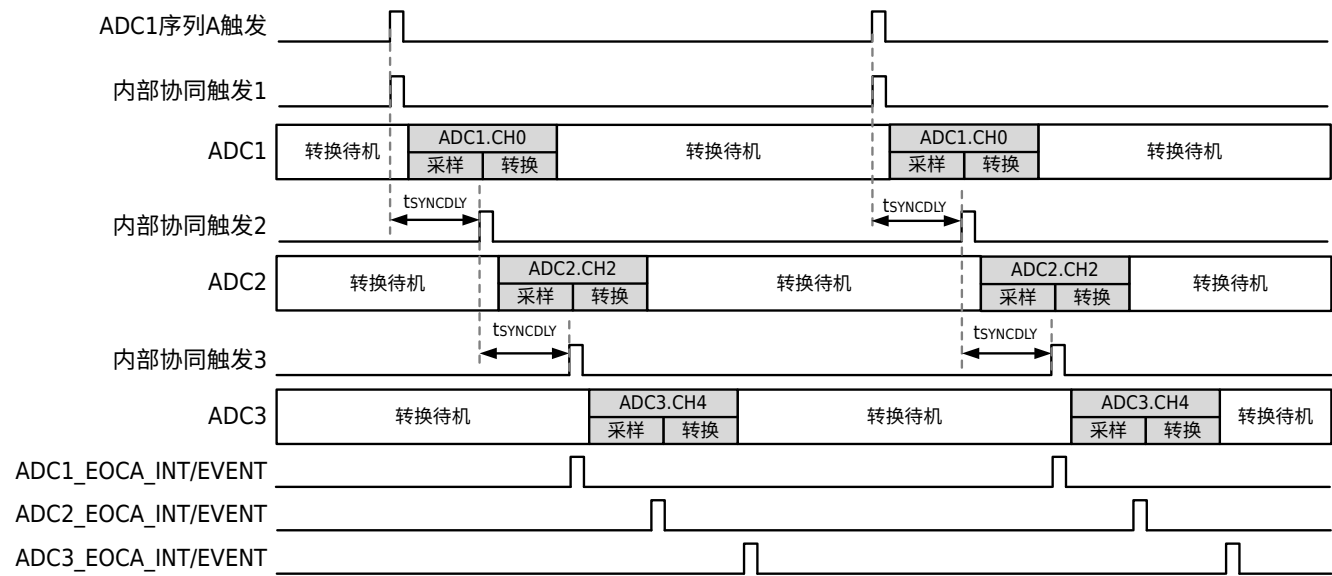


图 18-15 单次延迟触发模式（三个 ADC）

注意

- ADC1 序列 A 触发第一次输入后，ADC3 协同触发发生前，再次输入 ADC1 序列 A 触发将被忽略。
- 若各 ADC 单元转换的是同一模拟通道，需要错开采样时间，即延迟时间 $t_{SYNCDLY}$ 与通道开采样时间 t_{SPL} 需满足： $t_{SYNCDLY} > t_{SPL}$ 。

本模式的软件设定流程如下：

- 步骤 1.** 协同工作许可寄存器 ADC_SYNCCCR.SYNCEN 写 0，确认协同工作无效；
- 步骤 2.** 设置 ADC1、ADC2、ADC3 单元（参考单次平行模式）；
- 步骤 3.** 设置协同模式控制寄存器 ADC_SYNCCCR.SYNCMD，设置两个 ADC 的启动延迟；
- 步骤 4.** 设置协同模式控制寄存器 ADC_SYNCCCR.SYNCMD，写 0b000（ADC1、ADC2 两个 ADC 协同工作）或者写 0b001（ADC1、ADC2、ADC3 三个 ADC 协同工作）；
- 步骤 5.** 协同工作许可寄存器 ADC_SYNCCCR.SYNCEN 写 1，协同工作有效；
- 步骤 6.** 等待 ADC1 序列 A 触发源输入，在 ADC1、ADC2、ADC3 完成转换后处理结果。

循环并行触发模式

ADC1 的序列 A 触发条件同时触发处于协同工作模式的所有 ADC 单元，且之后每经过指定延迟之后会再次同时触发所有 ADC 单元。直至用户主动软件停止 ADC1 单元，或者禁止协同工作模式。

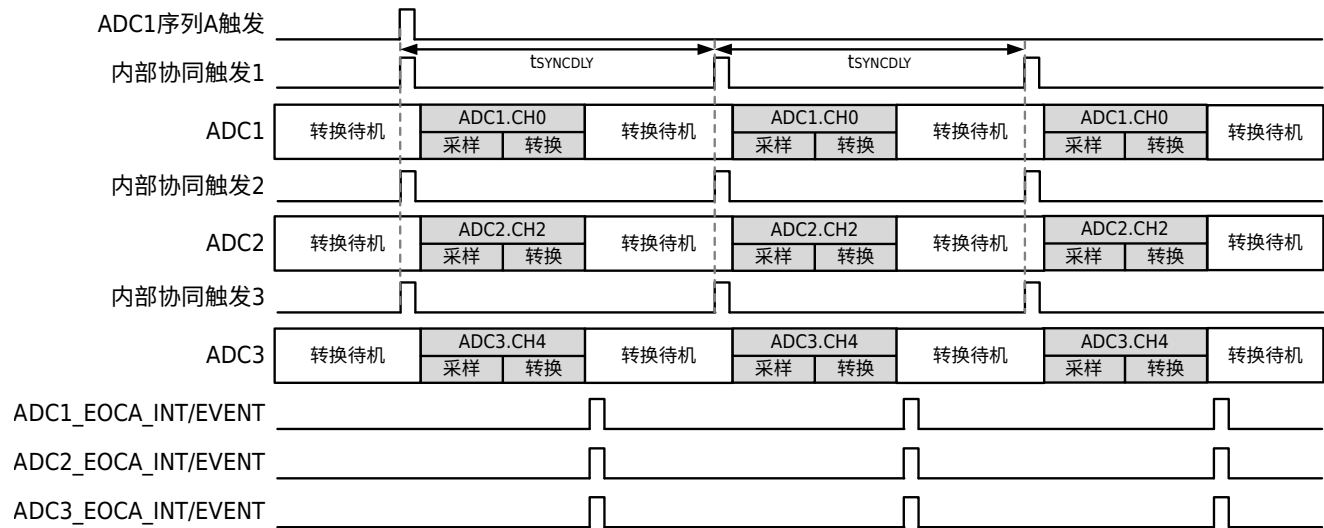


图 18-16 循环并行触发模式 (三个 ADC)



注意

延迟时间 $t_{SYNCDLY}$ 与一次扫描转换的时间 t_{SCAN} 需满足： $t_{SYNCDLY} > t_{SCAN}$

本模式的软件设定流程如下：

- 步骤 1. 协同工作许可寄存器 ADC_SYNCCR.SYNCEN 写 0，确认协同工作无效；
- 步骤 2. 设置 ADC1、ADC2、ADC3 单元，参考单次并行模式。ADC_CR0.MS 设置为 0b000（序列 A 单次扫描模式）；
- 步骤 3. 设置协同模式控制寄存器 ADC_SYNCCR.SYNCDLY，设置每次并行触发的延迟；
- 步骤 4. 设置协同模式控制寄存器 ADC_SYNCCR.SYNCMD，写 0b110（ADC1、ADC2 两个 ADC 协同工作）或者写 0b111（ADC1、ADC2、ADC3 三个 ADC 协同工作）；
- 步骤 5. 协同工作许可寄存器 ADC_SYNCCR.SYNCEN 写 1，协同工作有效；
- 步骤 6. 等待 ADC1 序列 A 触发源输入，在 ADC1、ADC2、ADC3 完成转换后处理结果。

循环延迟触发模式

ADC1 的序列 A 触发条件触发 ADC1 之后，每经过设定的延迟后，依次循环不断触发 ADC2、ADC3、ADC1、ADC2...，直至用户主动软件停止 ADC1 单元，或者禁止协同工作模式。

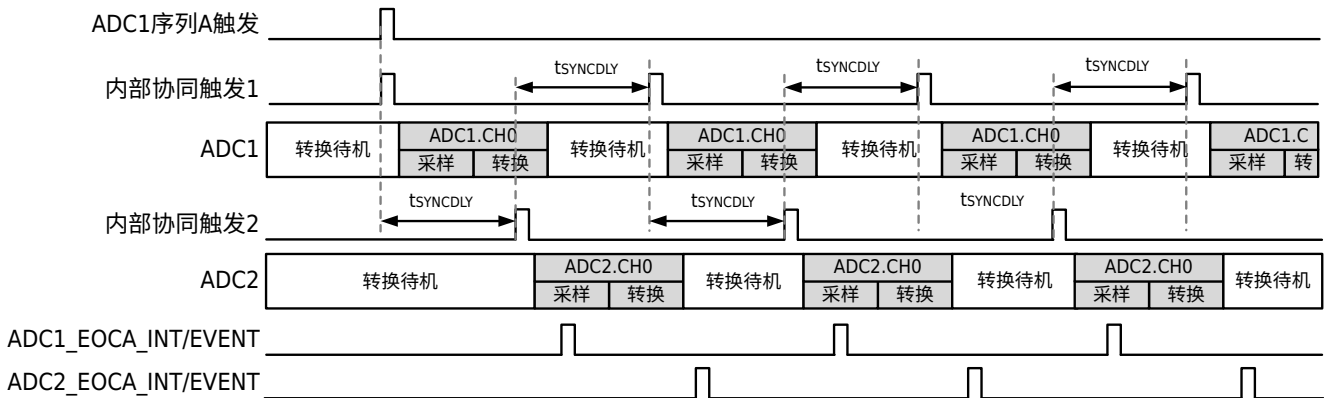


图 18-17 循环延迟触发模式 (两个 ADC)

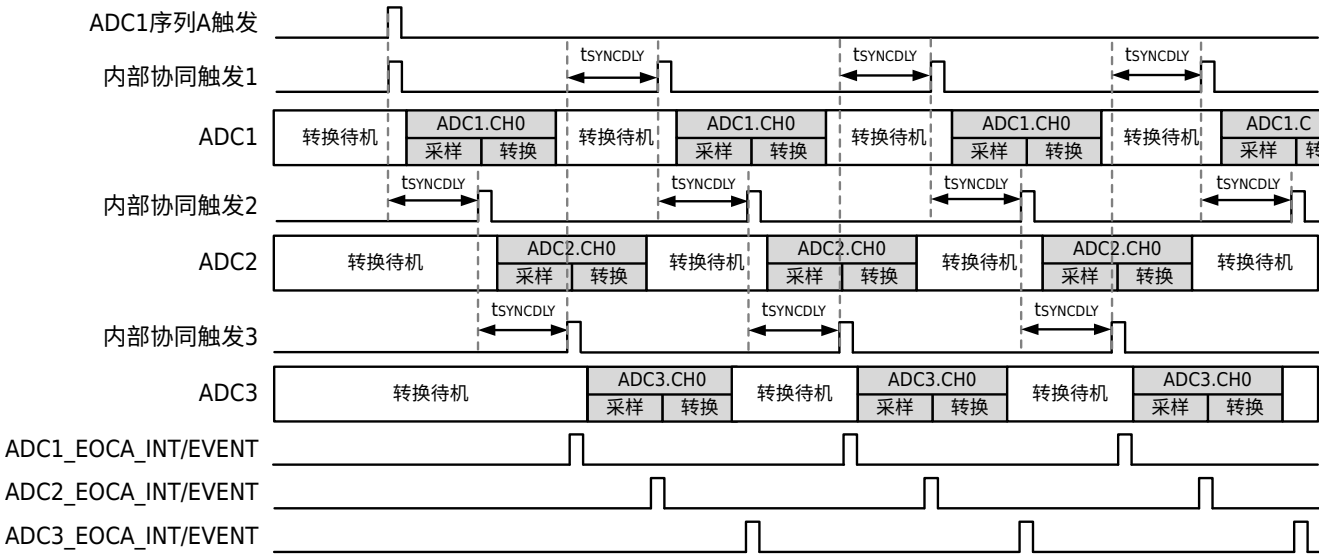


图 18-18 循环延迟触发模式 (三个 ADC)



注意

两个 ADC 协同工作时, 延迟时间 $t_{SYNCDELAY}$ 与一次扫描转换的时间 t_{SCAN} 需满足: $t_{SYNCDELAY} > t_{SCAN}/2$ 。三个 ADC 协同工作时需满足: $t_{SYNCDELAY} > t_{SCAN}/3$ 。同时, 如 ADC1、ADC2、ADC3 转换的是同一模拟通道, 还需要错开采样时间, 即 $t_{SYNCDELAY} > t_{SPL}$ 。

本模式的软件设定流程如下:

- 步骤 1. 协同工作许可寄存器 ADC_SYNCCCR.SYNCEN 写 0, 确认协同工作无效;
- 步骤 2. 设置 ADC1、ADC2、ADC3 单元, 参考循环并行触发模式;
- 步骤 3. 设置协同模式控制寄存器 ADC_SYNCCCR.SYNCDLY, 设置每次触发的延迟;
- 步骤 4. 设置协同模式控制寄存器 ADC_SYNCCCR.SYNCMD, 写 0b100 (ADC1、ADC2 两个 ADC 协同工作) 或者写 0b101 (ADC1、ADC2、ADC3 三个 ADC 协同工作) ;
- 步骤 5. 协同工作许可寄存器 ADC_SYNCCCR.SYNCEN 写 1, 协同工作有效;
- 步骤 6. 等待 ADC1 序列 A 触发源输入, 在 ADC1、ADC2、ADC3 完成转换后处理结果。

18.3.17 中断和事件

ADC 模块可以产生以下四种事件输出, 每个事件发生时, 若对应的中断许可寄存器设置为有效时, 同时输出中断申请。

- 序列 A 扫描结束 ADC_EOCA, 对应中断许可寄存器 ADC_ICR.EOCAIEN
- 序列 B 扫描结束 ADC_EOCB, 对应中断许可寄存器 ADC_ICR.EOCBIEN
- 模拟看门狗窗口 0 比较 ADC_CMP0, 对应中断许可寄存器 ADC_AWDCR.AWD0IEN
- 模拟看门狗窗口 1 比较 ADC_CMP1, 对应中断许可寄存器 ADC_AWDCR.AWD1IEN

上述四种事件输出, 可以启动其他片内周边模块, 包括启动 DMA 传送。利用 DMA 传送可以连续读取 ADC 转换结果, 不需要软件干预, 完全由硬件实现, 降低 CPU 的负荷。DMA 的设定请参见《DMA 控制器 (DMA)》章节相关描述。事件信号输出和中断使能位的控制无关, 只要条件发生就会输出。

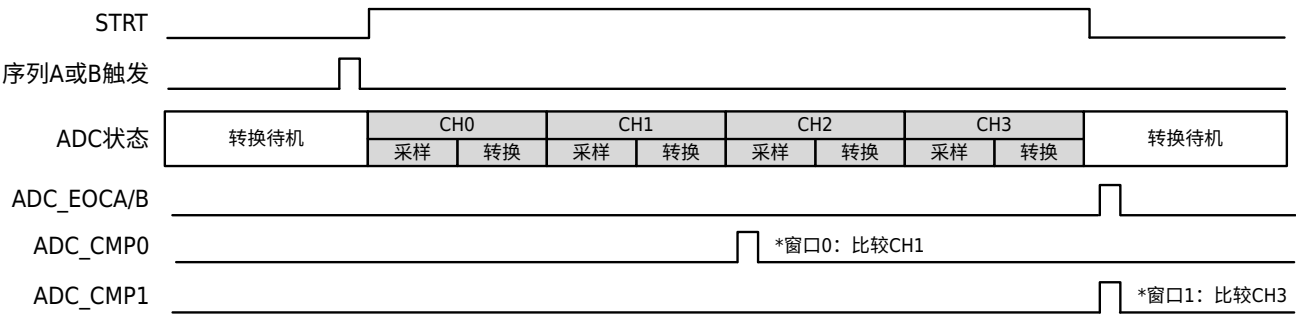


图 18-19 ADC 中断和事件输出时序

18.4 寄存器

18.4.1 寄存器总表

- ADC1 基地址：0x40040000
- ADC2 基地址：0x40040400
- ADC3 基地址：0x40040800

表 18-4 模数转换模块 (ADC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	ADC_STR	启动寄存器	8
0x02	ADC_CR0	控制寄存器 0	16
0x04	ADC_CR1	控制寄存器 1	16
0x06	ADC_CR2	控制寄存器 2	16
0x0A	ADC_TRGSR	转换开始触发寄存器	16
0x0C	ADC_CHSELRA	通道选择寄存器 A	32
0x10	ADC_CHSELRB	通道选择寄存器 B	32
0x14	ADC_AVCHSELR	平均通道选择寄存器	32
0x18	ADC_EXCHSELR	扩展通道选择寄存器	8
0x20+0x1*x	ADC_SSTRx	采样周期寄存器 (x=0~15)	8
0x30	ADC_SSTRL	采样周期寄存器	8
0x38	ADC_CHMUXR0	通道映射控制寄存器 0	16
0x3A	ADC_CHMUXR1	通道映射控制寄存器 1	16
0x3C	ADC_CHMUXR2	通道映射控制寄存器 2	16
0x3E	ADC_CHMUXR3	通道映射控制寄存器 3	16
0x44	ADC_ISR	中断状态寄存器	8
0x45	ADC_ICR	中断许可寄存器	8
0x46	ADC_ISCLRR	中断状态复位寄存器	8
0x4C	ADC_SYNCCR	协同模式控制寄存器	16
0x50+0x2*x	ADC_DRx	数据寄存器  说明 “x”表示通道，ADC1 和 ADC2 中 x=0~15， ADC3 中 x=0~19。	16
0xA0	ADC_AWDCR	模拟看门狗控制寄存器	16
0xA2	ADC_AWDSR	模拟看门狗状态寄存器	8

偏移地址	寄存器	描述	位宽
0xA3	ADC_AWDCLRR	模拟看门狗状态复位寄存器	8
0xA4+0x8*x+0x2*y	ADC_AWDxDRy	模拟看门狗窗口 x 阈值寄存器 (x=0~1, y=0~1)	16
0xA8+0x8*x	ADC_AWDxCHSR	模拟看门狗窗口 x 比较通道选择寄存器 (x=0~1)	8
0x1A	ADC_SHCR	采样保持电路控制寄存器	16
0xC0+0x1*x	ADC_PGACRx	可编程增益放大器控制寄存器 x (x=1~3)	8
0xC4	ADC_PGAVSSENr	可编程增益放大器参考地选择寄存器	8

18.4.2 启动寄存器 (ADC_STR)

Offset	Bit Position							
0x00	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							STRT
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	STRT	ADC 转换开始	● 0b0：停止转换 ● 0b1：开始转换 置 1 条件： 1. 软件设置 2. 选择的触发条件发生 3. ADC 转换中 清 0 条件： 1. 软件清 0 2. 转换结束后自动清 0  说明 ● STRT 为 0（ADC 空闲中）时写 1 产生软件触发，启动序列 A。 ● STRT 为 1（ADC 动作中）时写 1 无效。 ● STRT 为 1 时写 0 表示强制停止 ADC 转换。若 ADC_TRGSR 设置了 0x0 以外的值且不希望 ADC 再启动，请先将 ADC_TRGSR 设置为 0，再对 STRT 写 0。 ● STRT 为 0 时写 0 无效。	RW

18.4.3 控制寄存器 0 (ADC_CR0)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position															
0x02	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved					AVCNT			DFMT	CLREN	ACCSEL	Reserved	MS			
Access						RW			RW	RW	RW		RW			

位/位域	标记	位名	功能描述	读写
15:11	Reserved	保留	-	-
10:8	AVCNT	次数选择	0b000: 连续转换 2 次平均 0b001: 连续转换 4 次平均 0b010: 连续转换 8 次平均 0b011: 连续转换 16 次平均 0b100: 连续转换 32 次平均 0b101: 连续转换 64 次平均 0b110: 连续转换 128 次平均 0b111: 连续转换 256 次平均	RW
7	DFMT	数据格式	0b0: 转换数据右对齐 0b1: 转换数据左对齐 说明 过采样模式下该 Bit 位设置无效。	RW
6	CLREN	数据寄存器自动清除	0b0: 自动清除禁止 0b1: 自动清除许可 说明 CLREN 位设定后, 寄存器 ADC_DRx 将在 CPU、DMA 等读取后被自动清除。自动清除功能主要用于检测数据寄存器是否更新。	RW
5:4	ACCSEL	分辨率选择	0b00: 12 位分辨率 0b01: 10 位分辨率 0b10: 8 位分辨率 0b11: 禁止设置 说明 过采样模式下需要设置为 0b00。	RW
3	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
2:0	MS	模式选择	● 0b000: 序列 A 单次扫描模式, 序列 B 无效 ● 0b001: 序列 A 连续扫描模式, 序列 B 无效 ● 0b010: 序列 A 单次扫描模式, 序列 B 单次扫描模式 ● 0b011: 序列 A 连续扫描模式, 序列 B 单次扫描模式 ● 0b100: 序列 A 数据缓冲模式, 序列 B 无效 ● 0b110: 序列 A 数据缓冲模式, 序列 B 单次扫描模式 ● 其他: 禁止设置	RW

18.4.4 控制寄存器 1 (ADC_CR1)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												RSCHSEL	Reserved		
Access													RW			

位/位域	标记	位名	功能描述	读写
15:3	Reserved	保留	-	-
2	RSCHSEL	序列 A 重启通道选择	0b0: 被序列 B 中断后, 序列 A 重启时从被中断通道开始继续扫描 0b1: 被序列 B 中断后, 序列 A 重启时从被第一个通道开始重新扫描	RW
1:0	Reserved	保留	-	-

18.4.5 控制寄存器 2 (ADC_CR2)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved			OVSMOD		OVSS			Reserved							
Access				RW		RW										

位/位域	标记	位名	功能描述	读写
15:13	Reserved	保留	-	-
12	OVSMOD	过采样模式	0b0: 普通模式 0b1: 过采样模式	RW
11:8	OVSS	移位选择	0b0000: 不移位 0b0001: 向右移 1 位 0b0010: 向右移 2 位 0b0011: 向右移 3 位 0b0100: 向右移 4 位 0b0101: 向右移 5 位 0b0110: 向右移 6 位 0b0111: 向右移 7 位 0b1000: 向右移 8 位 - 其他: 禁止设置 说明 OVSMOD=0 (普通模式) 时, 该位设置无效。	RW
7:0	Reserved	保留	-	-

18.4.6 转换开始触发寄存器 (ADC_TRGSR)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position															
0x0A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TRGENB	Reserved					TRGSELB	TRGENA	Reserved					TRGSELA		
Access	RW						RW	RW						RW		

位/位域	标记	位名	功能描述	读写
15	TRGENB	序列 B 触发使能	0b0: 序列 B 片内或外部引脚触发禁止 0b1: 序列 B 片内或外部引脚触发许可 说明 选择外部引脚触发有效。如果 ADCx_ADTRG 由“High”变为“Low”，检测到下降沿，则扫描转换开始，请保持“Low”1.5*PCLK4 周期以上。	RW
14:10	Reserved	保留	-	-
9:8	TRGSELB	序列 B 触发条件选择	在序列 B 有效模式下 (ADC_CR0.MS[1]=1)，作为序列 B 的触发条件： 0b00: ADCx_ADTRG (x=1~3, 代表 ADC 单元编号) 0b01: IN_TRGx0 0b10: IN_TRGx1 0b11: IN_TRGx0 + IN_TRGx1 说明 只在序列 B 有效模式下有效，其他模式设定无效。两次触发的间隔必须大于或等于扫描周期 t_{SCAN}，若小于则触发无效。	RW
7	TRGENA	序列 A 触发使能	0b0: 序列 A 片内或外部引脚触发禁止 0b1: 序列 A 片内或外部引脚触发许可 说明 选择外部引脚触发有效。如果 ADCx_ADTRG 由“High”变为“Low”，检测到下降沿，则扫描转换开始，请保持“Low”1.5*PCLK4 周期以上。	RW
6:2	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
1:0	TRGSELA	序列 A 触发条件选择	序列 A 的触发条件： ● 0b00: ADCx_ADTRG (x=1~3, 代表 ADC 单元编号) ● 0b01: IN_TRGx0 ● 0b10: IN_TRGx1 ● 0b11: IN_TRGx0 + IN_TRGx1  说明 ADC 空闲中对 ADC_STR.STRT 写 1 软件触发，无视 TRGENA 和 TRGSELA 的设定，直接开始 ADC 转换。两次触发的间隔必须大于或等于扫描周期 t_{SCAN}，若小于则触发无效。	RW

18.4.7 通道选择寄存器 A (ADC_CHSELRA)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHSELA																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	CHSELA	转换通道选择	序列 A 的通道选择，每个 Bit 代表一个通道，CHSELA[x]代表通道 CHx，可选择任意组合。 0b0：未选择对应通道 0b1：选择对应通道 注意 请不要在序列 A 和序列 B 中选择相同的通道。 ADC1 和 ADC2 支持通道 0~15，ADC3 支持通道 0~19。不支持的通道需要设置为 0。	RW

18.4.8 通道选择寄存器 B (ADC_CHSELRB)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHSELB																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	CHSELB	转换通道选择	序列 B 的通道选择，每个 Bit 代表一个通道，CHSELB[x]代表通道 CHx，可选择任意组合。 0b0：未选择对应通道 0b1：选择对应通道 注意 只有在双序列扫描模式时才有效。请不要在序列 A 和序列 B 中选择相同的通道。 ADC1 和 ADC2 支持通道 0~15，ADC3 支持通道 0~19。不支持的通道需要设置为 0。	RW

18.4.9 平均通道选择寄存器 (ADC_AVCHSELR)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RW AVCHSEL																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	AVCHSEL	平均通道选择	每个 Bit 代表一个通道，AVCHSEL[x]代表通道 CHx，可选择任意组合。 ● 0b0：未选择对应通道 ● 0b1：选择对应通道  注意 当 AVCHSEL 与 ADC_CHSELRA 或 ADC_CHSELB 的对应的通道同时被选择时，则该通道在扫描时将连续执行设定次数 ADC 转换，并对转换结果进行平均计算后更新入数据寄存器。如果对应通道 AVCHSEL 未被设定，该通道将执行普通一次转换。 ADC1 和 ADC2 支持通道 0~15，ADC3 支持通道 0~19。不支持的通道需要设置为 0。	RW

18.4.10 扩展通道选择寄存器 (ADC_EXCHSELR)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

Offset	Bit Position							
0x18	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							EXCHSEL
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	EXCHSEL	扩展通道选择	扩展通道模拟输入源选择： 0b0：选择外部引脚 0b1：选择内部模拟信号 说明 内部模拟信号的设置请参见《电源控制（PWC）》章节相关描述。	RW

18.4.11 采样周期寄存器 (ADC_SSTRx) (x=0~15、L)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

偏移地址 (Offset) : 0x20+0x1*x (SSTR0~15) , 0x30 (SSTRL)

Offset	Bit Position							
-	7	6	5	4	3	2	1	0
Reset	0x0B							
Name	SST							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	SST	采样周期数	</	



说明

- 内部基准电压的采样时间不要小于 1 μ s。
- VBAT 分压的采样时间不要小于 4 μ s。
- 通道专用采样保持电路 SH 有效时, 对应通道采样时间不要小于 0.4 μ s。

18.4.12 通道映射控制寄存器 (ADC_CHMUXRx) (x=0~3)



说明

请在 ADC_STR.STRT 为 0 时设置本寄存器。

ADC_CHMUXR0 寄存器

Offset	Bit Position															
0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x3210															
Name	CH03MUX				CH02MUX				CH01MUX				CH00MUX			
Access	RW				RW				RW				RW			

ADC_CHMUXR1 寄存器

Offset	Bit Position															
0x3A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x7654															
Name	CH07MUX				CH06MUX				CH05MUX				CH04MUX			
Access	RW				RW				RW				RW			

ADC_CHMUXR2 寄存器

Offset	Bit Position															
0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xBA98															
Name	CH11MUX				CH10MUX				CH09MUX				CH08MUX			
Access	RW				RW				RW				RW			

ADC_CHMUXR3 寄存器

Offset	Bit Position															
0x3E	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFEDC															
Name	CH15MUX				CH14MUX				CH13MUX				CH12MUX			
Access	RW				RW				RW				RW			

位/位域	标记	位名	功能描述	读写
15:12	CHxMUX	通道 x 映射选择 (x=0~15)	选择 ADC 单元 CHx 映射对象，设定值选择详见 不同的 ADC 单元 CHx 映射关系表  注意 不存在的通道对应位读出时为 0，写入时写 0。 请不要设置到不存在的模拟输入上。	RW
11:8	CHxMUX			RW
7:4	CHxMUX			RW
3:0	CHxMUX			RW

表 18-30 不同的 ADC 单元 CHx 映射关系表

设定值	ADC1 映射对象	ADC2 映射对象	ADC3 映射对象
0x0	ADC123_IN0	ADC123_IN0	ADC123_IN0
0x1	ADC123_IN1	ADC123_IN1	ADC123_IN1
0x2	ADC123_IN2	ADC123_IN2	ADC123_IN2
0x3	ADC123_IN3	ADC123_IN3	ADC123_IN3
0x4	ADC12_IN4	ADC12_IN4	ADC3_IN4
0x5	ADC12_IN5	ADC12_IN5	ADC3_IN5
0x6	ADC12_IN6	ADC12_IN6	ADC3_IN6
0x7	ADC12_IN7	ADC12_IN7	ADC3_IN7
0x8	ADC12_IN8	ADC12_IN8	ADC3_IN8
0x9	ADC12_IN9	ADC12_IN9	ADC3_IN9
0xA	ADC123_IN10	ADC123_IN10	ADC123_IN10
0xB	ADC123_IN11	ADC123_IN11	ADC123_IN11
0xC	ADC123_IN12	ADC123_IN12	ADC123_IN12
0xD	ADC123_IN13	ADC123_IN13	ADC123_IN13
0xE	ADC12_IN14	ADC12_IN14	ADC3_IN14
0xF	ADC12_IN15/内部模拟信号	ADC12_IN15/内部模拟信号	ADC3_IN15/内部模拟信号

18.4.13 中断状态寄存器 (ADC_ISR)

Offset	Bit Position							
0x44	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			RO SASTPDF	Reserved		RO EOCBF	RO EOCAF
Access				RO			RO	RO

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4	SASTPDF	序列 A 被打断标志	● 0b0: 序列 A 未被打断 ● 0b1: 序列 A 被打断  说明 双序列扫描模式，序列 A 扫描过程中被高优先级的序列 B 打断时置 1。	RO
3:2	Reserved	保留	-	-
1	EOCBF	序列 B 转换完成标志	● 0b0: 序列 B 所选通道未扫描完成 ● 0b1: 序列 B 所选通道全部扫描完成  说明 序列 B 所选通道全部扫描完成后置 1。	RO
0	EOCAF	序列 A 转换完成标志	● 0b0: 序列 A 所选通道未扫描完成 ● 0b1: 序列 A 所选通道全部扫描完成  说明 序列 A 所选通道全部扫描完成后置 1。	RO

18.4.14 中断许可寄存器 (ADC_ICR)

Offset	Bit Position							
0x45	7	6	5	4	3	2	1	0
Reset	0x03							
Name	Reserved						EOCBIE	EOCAIE
Access							RW	RW

位/位域	标记	位名	功能描述	读写
7:2	Reserved	保留	-	-
1	EOCBIE	序列 B 转换完成中断使能	0b0: 中断禁止 0b1: 中断许可	RW
0	EOCAIE	序列 A 转换完成中断使能	0b0: 中断禁止 0b1: 中断许可	RW

18.4.15 中断状态复位寄存器 (ADC_ISCLRR)

Offset	Bit Position							
0x46	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			CLRSASTPDF	Reserved		CLREOCBF	CLREOCAF
Access				WO			WO	WO

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4	CLRSASTPDF	序列 A 被打断标志复位	写 1 复位 SASTPDF 状态位, 写 0 无效; 读出恒为 0	WO
3:2	Reserved	保留	-	-
1	CLREOCBF	序列 B 转换完成标志复位	写 1 复位 EOCBF 状态位, 写 0 无效; 读出恒为 0	WO
0	CLREOCAF	序列 A 转换完成标志复位	写 1 复位 EOCAF 状态位, 写 0 无效; 读出恒为 0	WO

18.4.16 协同模式控制寄存器 (ADC_SYNCRR)



说明

本寄存器只在主控 ADC（即 ADC1）中搭载，其他 ADC 单元中无此寄存器。

Offset	Bit Position															
0x4C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0C00															
Name	SYNCDLY								Reserved	SYNCDMD[2:0]			Reserved			SYNCEN
Access	RW									RW						RW

位/位域	标记	位名	功能描述	读写
15:8	SYNCDLY	同步延迟时间	延迟触发模式时，两个 ADC 的启动延迟时间 t_{SYNCDLY}。 0x00：禁止设置 0x01：$t_{\text{SYNCDLY}} = \text{PCLK2}$ 时钟周期*1 0xFF：$t_{\text{SYNCDLY}} = \text{PCLK2}$ 时钟周期*255 说明 请在 SYNCEN 为 0 时设置本寄存器。 根据各 ADC 的采样时间，转换时间，设置合理的延迟时间，避免多个 ADC 同时处于采样状态引起的误差增大，避免 ADC 还未转换结束就再次发生触发，导致同步失败。推荐设置如下： - 单次延迟触发模式：$t_{\text{SYNCDLY}} > t_{\text{SPL}}$ - 两个 ADC 循环延迟触发模式：$t_{\text{SYNCDLY}} > t_{\text{SPL}}$，且 $t_{\text{SYNCDLY}} > t_{\text{SCAN}}/2$ - 三个 ADC 循环延迟触发模式：$t_{\text{SYNCDLY}} > t_{\text{SPL}}$，且 $t_{\text{SYNCDLY}} > t_{\text{SCAN}}/3$ - 单次并行触发模式：本寄存器设置无效。 - 循环并行触发模式：$t_{\text{SYNCDLY}} > t_{\text{SCAN}}$	RW
7	Reserved	保留	-	-
6:4	SYNCDMD[2:0]	同步模式选择	SYNCDMD[2]: 0b0：单次触发 0b1：循环触发 SYNCDMD[1]: 0b0：延迟触发模式 0b1：并行触发模式 SYNCDMD[0]: 0b0：ADC1 和 ADC2 同步工作，ADC3 独立工作 0b1：ADC1，ADC2 和 ADC3 同步工作	RW

位/位域	标记	位名	功能描述	读写
6:4	SYNCMD[2:0]	同步模式选择	 说明 请在 SYNCEN 为 0 时设置本寄存器。 使用单次触发时，请将需要同步的 ADC 设置为序列 A 单次扫描，或序列 A 连续扫描模式。 使用循环触发模式时，请将 ADC 设置为序列 A 单次扫描模式。	RW
3:1	Reserved	保留	-	-
0	SYNCEN	同步模式许可	● 0b0：同步模式无效 ● 0b1：同步模式有效  说明 同步模式只支持序列 A。 在 SYNCEN 写 1 之前，请将参与同步的几个 ADC 的序列 B 关闭 (ADC_CR0.MS[1]=0)，并给序列 A 选择相同数目的通道，设置相同的通道采样时间 ADC_SSTRx。以避免各 ADC 扫描时间 t_{SCAN} 不一致，引起后续同步失败。 软件对 ADC1_STR.STRT 写 0 强制停止转换时，SYNCEN 自动清 0。	RW

18.4.17 ADC 数据寄存器 (ADC_DRx)

本寄存器是用于存储各通道 ADC 转换数据的只读寄存器。根据过采样模式、数据对齐方式和转换分辨率，转换结果数据存储方式有所不同。

偏移地址 (Offset)：0x50+0x2*x (“x”表示通道，ADC1 和 ADC2 中 x=0~15，ADC3 中 x=0~19)

过采样模式

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO AD[15:0]															
Access	RO															

数据右对齐-12 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved				RO AD[11:0]											
Access					RO											

数据右对齐-10 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved						RO AD[9:0]									
Access							RO									

数据右对齐-8 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved								RO AD[7:0]							
Access									RO							

数据左对齐-12 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO AD[11:0]												Reserved			
Access	RO															

数据左对齐-10 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO AD[9:0]										Reserved					
Access	RO															

数据左对齐-8 位分辨率

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO AD[7:0]								Reserved							
Access	RO															

18.4.18 模拟看门狗控制寄存器 (ADC_AWDCR)

Offset	Bit Position															
0xA0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved						AWDCM		Reserved	AWD1MD	AWD1IEN	AWD1EN	Reserved	AWD0MD	AWD0IEN	AWD0EN
Access							RW			RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:10	Reserved	保留	-	-
9:8	AWDCM	看门狗窗口组合选择	0b00: 窗口组合无效, ADC_CMP1 输出窗口 1 独立比较结果 0b01: 窗口组合有效, ADC_CMP1 输出窗口 0 与窗口 1 比较结果的逻辑或 0b10: 窗口组合有效, ADC_CMP1 输出窗口 0 与窗口 1 比较结果的逻辑与 0b11: 窗口组合有效, ADC_CMP1 输出窗口 0 与窗口 1 比较结果的逻辑异或  说明 使用窗口组合功能时需要窗口 0 和窗口 1 都使能, 即 AWD0EN 写 1, AWD1EN 写 1。若窗口 0 与窗口 1 选择的比较通道不相同, 请保证扫描转换过程中, 窗口 1 所选通道在窗口 0 所选通道之后转换。ADC_CMP1 中断或事件在窗口 1 所选通道转换结束时输出。	RW
7	Reserved	保留	-	-
6	AWD1MD	看门狗窗口 1 比较模式	窗口 1 保护区域选择: 0b0: 转换结果 < AWD1DR0, 或转换结果 > AWD1DR1 0b1: 转换结果 ≥ AWD1DR0, 且转换结果 ≤ AWD1DR1	RW
5	AWD1IEN	看门狗窗口 1 中断使能	0b0: 看门狗窗口 1 比较中断 ADC_CMP1 无效 0b1: 看门狗窗口 1 比较中断 ADC_CMP1 有效	RW
4	AWD1EN	看门狗窗口 1 比较功能使能	0b0: 看门狗窗口 1 比较功能无效 0b1: 看门狗窗口 1 比较功能有效	RW
3	Reserved	保留	-	-
2	AWD0MD	看门狗窗口 0 比较模式	窗口 0 保护区域选择: 0b0: 转换结果 < AWD0DR0, 或转换结果 > AWD0DR1 0b1: 转换结果 ≥ AWD0DR0, 且转换结果 ≤ AWD0DR1	RW
1	AWD0IEN	看门狗窗口 0 中断使能	0b0: 看门狗窗口 0 比较中断 ADC_CMP0 无效 0b1: 看门狗窗口 0 比较中断 ADC_CMP0 有效	RW
0	AWD0EN	看门狗窗口 0 比较功能使能	0b0: 看门狗窗口 0 比较功能无效 0b1: 看门狗窗口 0 比较功能有效	RW

18.4.19 模拟看门狗状态寄存器 (ADC_AWDSR)

Offset	Bit Position							
0xA2	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			AWDCMF	Reserved		AWD1F	AWD0F
Access				RO			RO	RO

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4	AWDCMF	看门狗窗口组合比较状态位	0b0: 窗口 0 与窗口 1 比较结果不满足组合条件 0b1: 窗口 0 与窗口 1 比较结果满足组合条件  说明 看门狗窗口组合比较功能有效（即 ADC_AWDCR.AWDCM[1:0]=0b01/0b10/0b11）时，窗口 1 所选通道转换结束时，窗口 0 与窗口 1 比较结果满足组合条件时置 1。	RO
3:2	Reserved	保留	-	-
1	AWD1F	看门狗窗口 1 比较状态位	0b0: 转换结果不满足比较条件 0b1: 转换结果满足比较条件  说明 窗口 1 所选通道转换结束时，转换结果满足比较条件时置 1。	RO
0	AWD0F	看门狗窗口 0 比较状态位	0b0: 转换结果不满足比较条件 0b1: 转换结果满足比较条件  说明 窗口 0 所选通道转换结束时，转换结果满足比较条件时置 1。	RO

18.4.20 模拟看门狗状态复位寄存器 (ADC_AWDSCCLR)

Offset	Bit Position							
0xA3	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			CLRAWDCMF	Reserved		CLRAW1F	CLRAW0F
Access				WO			WO	WO

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4	CLRAWDCMF	看门狗窗口组合比较状态位复位	写 1 复位 AWDCMF 状态位，写 0 无效；读出恒为 0	WO
3:2	Reserved	保留	-	-
1	CLRAW1F	看门狗窗口 1 比较状态位复位	写 1 复位 AWD1F 状态位，写 0 无效；读出恒为 0	WO
0	CLRAW0F	看门狗窗口 0 比较状态位复位	写 1 复位 AWD0F 状态位，写 0 无效；读出恒为 0	WO

18.4.21 模拟看门狗窗口 x 阈值寄存器 (ADC_AWDxDRy) (x=0~1, y=0~1)

偏移地址 (Offset) : 0xA4+0x8*x+0x2*y

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000(AWDxDR0);0xFFFF(AWDxDR1)															
Name	AWDDR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	AWDDR	比较数据	AWD0DR0 设置窗口 0 低阈值, AWD0DR1 设置窗口 0 高阈值。 AWD1DR0 设置窗口 1 低阈值, AWD1DR1 设置窗口 1 高阈值。 AWD0DR0、AWD0DR1、AWD1DR0、AWD1DR1 根据对齐方式 (数据右对齐或左对齐)、分辨率 (12 位、10 位或者 8 位) 会有差异。 ● 数据右对齐-12 位分辨率: 低 12 位[11:0]可用 ● 数据右对齐-10 位分辨率: 低 10 位[9:0]可用 ● 数据右对齐-8 位分辨率: 低 8 位[7:0]可用 ● 数据左对齐-12 位分辨率: 高 12 位[15:4]可用 ● 数据左对齐-10 位分辨率: 高 10 位[15:6]可用 ● 数据左对齐-8 位分辨率: 高 8 位[15:8]可用	RW

18.4.22 模拟看门狗窗口 x 比较通道选择寄存器 (ADC_AWDxCHSR) (x=0~1)

偏移地址 (Offset) : 0xA8+0x8*x

Offset	Bit Position							
-	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved			AWDCH				
Access				RW				

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4:0	AWDCH	看门狗比较通道选择	选择窗口 x 的比较通道 ● 0x00: CH0 (通道 0) ● 0x01: CH1 (通道 1) ● ● 0x0F: CH15 (通道 15) ● ● 0x13: CH19 (通道 19) ● 其他: 禁止设置  注意 请不要设置成不存在的通道。ADC1 和 ADC2 支持通道 0~15, ADC3 支持通道 0~19。	RW

18.4.23 专用采样保持电路控制寄存器 (ADC_SHCR)



说明

- 单元 1 (ADC1) 支持 SH1~3 共 3 个通道，参考[与物理通道映射关系](#)。寄存器 ADC_SHCR.SHSEL[0]对应 SH1，SHSEL[1]对应 SH2，SHSEL[2]对应 SH3。
- 单元 2 (ADC2) 不支持 SH。
- 单元 3 (ADC3) 不支持 SH。

Offset	Bit Position															
0x1A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0018															
Name	Reserved					SHSEL				SHSST						
Access						RW				RW						

位/位域	标记	位名	功能描述	读写
15:11	Reserved	保留	-	-
10:8	SHSEL	专用采样保持电路通道选择	对应 SH3~SH1: ● 0b0: 旁路专用采样保持电路 ● 0b1: 选通专用采样保持电路 说明 每个 Bit 对应一个通道。 在双序列扫描模式下，序列 A 中不要使用专用采样保持电路。因为序列 A 可能会被序列 B 打断导致各专用采样保持电路的采样时间不一致而输出期待之外的转换结果。	RW
7:0	SHSST	采样时间选择	● 0x00~0x03: 禁止设置 ● 0x04: PCLK2 时钟周期*4 ● ● 0xFF: PCLK2 时钟周期*255 说明 PCLK2 为 50MHz 工作时，1 周期时间为 20ns。初始的周期设定为 24 周期，当输入阻抗高采样时间不足或者 PCLK2 频率低时可以调整采样时间。采样时间不要少于 0.4μs。	RW

18.4.24 可编程增益放大器控制寄存器 (ADC_PGACRx) (x=1~3)



说明

- 单元 1 (ADC1) 支持 PGA1~3 共 3 个通道, 参考[与物理通道映射关系](#)。ADC1_PGACR1~3 寄存器分别对应控制 PGA1~3。
- 单元 2 (ADC2) 支持 PGA4 共 1 个通道, 参考[与物理通道映射关系](#)。ADC2_PGACR1 寄存器对应控制 PGA4。
- 单元 3 (ADC3) 不支持 PGA。

偏移地址 (Offset) : 0xC0+0x1*x

Offset	Bit Position							
-	7	6	5	4	3	2	1	0
Reset	0x00							
Name	PGAGAIN				PGACTL			
Access	RW				RW			

位/位域	标记	位名	功能描述	读写
7:4	PGAGAIN	增益设定	● 0b0000: × 2.000 ● 0b0001: × 2.133 ● 0b0010: × 2.286 ● 0b0011: × 2.667 ● 0b0100: × 2.909 ● 0b0101: × 3.200 ● 0b0110: × 3.556 ● 0b0111: × 4.000 ● 0b1000: × 4.571 ● 0b1001: × 5.333 ● 0b1010: × 6.400 ● 0b1011: × 8.000 ● 0b1100: × 10.667 ● 0b1101: × 16.000 ● 0b1110: × 32.000 ● 0b1111: 禁止设置	RW
3:0	PGACTL	放大器控制	● 0b0000: 放大器无效 ● 0b1110: 放大器有效, 信号按 PGAGAIN[3:0]设定值放大 ● 其他: 禁止设置	RW

18.4.25 可编程增益放大器参考地选择寄存器 (ADC_PGAVSSSEN)



说明

- 单元 1 (ADC1) 和单元 2 (ADC2) 支持 PGA。
- 单元 3 (ADC3) 不支持 PGA。

Offset	Bit Position							
0xC4	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved					PGAVSSEN		
Access						RW		

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2:0	PGAVSSEN	放大器参考地选择	● 0b0: 使用外部引脚 PGAx_VSS 作为 PGA 参考地 ● 0b1: 使用模拟地 AVSS 作为 PGA 参考地 其中: ● 单元 1 ADC1_PGAVSSSEN.PGAVSSEN[0]选择 PGA1 参考地 PGA123_VSS 或 AVSS ● 单元 1 ADC1_PGAVSSSEN.PGAVSSEN[1]选择 PGA2 参考地 PGA123_VSS 或 AVSS ● 单元 1 ADC1_PGAVSSSEN.PGAVSSEN[2]选择 PGA3 参考地 PGA123_VSS 或 AVSS ● 单元 2 ADC2_PGAVSSSEN.PGAVSSEN[0]选择 PGA4 参考地 PGA4_VSS 或 AVSS ● 单元 2 ADC2_PGAVSSSEN.PGAVSSEN[2:1]不存在	RW

18.5 注意事项

18.5.1 数据寄存器读取时注意事项

ADC 数据寄存器 ADC_DR 请以半字单位访问。请不要以字节单位访问数据寄存器。

18.5.2 扫描完成中断处理注意事项

当对同一通道进行连续两次扫描转换时，从第一次转换完成中断处理到第二次转完成中断处理期间，如果 CPU 没有及时将第一次的转换数据读取，那么第二次的转换数据会将第一次的转换数据覆盖。

18.5.3 模块停止与低功耗设定的注意事项

通过设定寄存器 PWC_FCG，可以设定 ADC 模块停止，降低功耗。ADC 初始状态为停止状态。需要 ADC 模块工作时请先设定 PWC_FCG 寄存器相应位取消停止，并等待 1 μ s 以后再启动 ADC 转换。

在设置模块停止前，请先确认 ADC 处于转换停止中，即 ADC_STR.STRT 位为 0。

在设置系统进入停止模式 (STOP) 前，请先将 ADC 设定为模块停止模式。

具体描述请参见《电源控制 (PWC)》章节“动作模式与低功耗模式”。

18.5.4 ADC 转换模拟通道输入的引脚设定

当芯片引脚设定为 ADC 模拟通道输入时，请先禁用对应引脚的数字功能 (PCRxy.DDIS)。具体描述请参见《通用 IO (GPIO)》。

18.5.5 噪声控制

为防止浪涌等异常电压破坏模拟输入引脚，推荐使用《数据手册》的“电气特性”章节所示保护电路。

19 数模转换器 (DAC)

19.1 简介

本 MCU 搭载了两个 12 位的数模转换器单元 DAC1 和 DAC2。每个 DAC 单元包含两个 DAC 转换通道，两个通道可以独立转换也可以同步转换。模拟电压输出范围有两档可设。每个转换通道配有输出放大器，可以在没有外部运放时直接驱动外部负载。独立管脚输入参考电压 VREFH 和 VREFL 可用来提高转换精度。

19.2 主要特性

- 两个 DAC 单元，每个单元有两个 DAC 转换通道
- 12 位转换数据可配置成左对齐或者右对齐格式
- 同一个 DAC 模块的两个转换通道可实现同步转换
- 转换数据计算单元 (DCU) 的数据可输出三角波和锯齿波
- 输出可用于电压比较器 (CMP) 的负端电压
- 输出配有放大功能，可直接驱动外部负载
- ADC 转换优先模式可减少 ADC 转换时的干扰
- 独立管脚输入参考电压 VREFH/VREFL

19.3 功能说明

19.3.1 功能框图

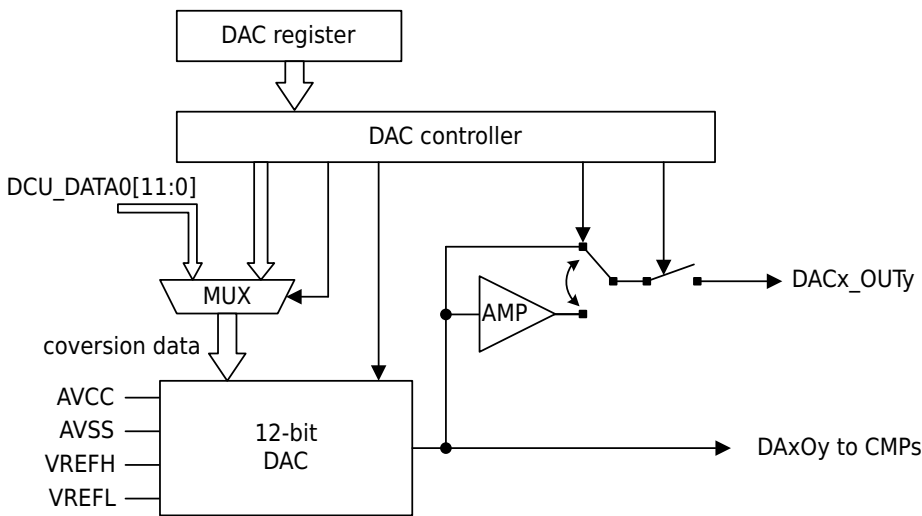


图 19-1 DAC 转换通道框图

表 19-1 DAC 引脚

引脚名称	输入/输出	功能
AVCC	输入	模拟电源
AVSS	输入	模拟地
VREFH	输入	参考电压
VREFL	输入	参考地
DACx_OUTy	输出	DAC 转换模拟输出 (x=1~2, y=1~2)

**说明**

本章节后文提到 x 均指单元编号, y 均指通道编号。

19.3.2 DAC 转换

每个转换通道都可以独立进行 DAC 转换。DAC_DACR.DAyE 设置成 1 时, 对应通道的 DAC 转换开始并从 DACx_OUTy 端口输出转换结果。如仅将转换结果作为电压比较器 CMP 的负端输入电压, 可通过 DAC_DAOCR 寄存器将 DACx_OUTy 输出关闭。DAC 转换动作时序如下:

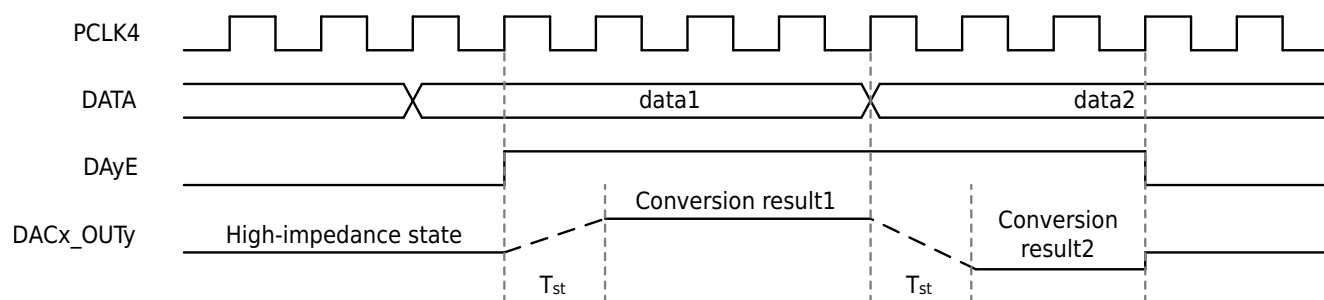


图 19-2 DAC 转换示意图

以使用 DAC1 通道 1 进行单通道转换为例, 对 DAC 转换过程进行介绍。

1. 设置 DAC 转换数据 (DAC_DADR1) 和数据格式 (DAC_DACR.DPSEL)。
2. 设置 DAC_DACR.DA1E 为 1, DAC 转换开始。经过转换时间 T_{st} 后, 转换结果从端口 DAC1_OUT1 输出并一直保持到 DAC_DADR1 的值被改写或者 DAC_DACR.DA1E 被设置成 0。输出的模拟电压值可由以下公式计算:

$$DAC_{output} = \frac{\text{Conversion}}{4096} \times VREFH$$

DAC_DACR.DA1E 为 1 时, 改写 DAC_DADR1 的值将触发一次新的 DAC 转换。同样地, 经过转换时间 T_{st} 后, 从端口 DAC1_OUT1 输出新的转换结果。

3. 设置 DAC_DACR.DA1E 位为 0, 关闭 DAC1 通道 1, DAC1_OUT1 输出高阻态。

19.3.3 同步转换

通过转换数据的同步更新可以实现同一个 DAC 单元的两个转换通道的同步转换。

以使用 DAC1 进行两通道同步转换为例, 对同步转换过程进行介绍。

1. 设置 DAC 转换数据 (DAC_DADR1、DAC_DADR2) 和数据格式 (DAC_DACR.DPSEL)。
2. 设置 DAC_DACR.DAE 为 1, 两个通道的 DAC 转换同步开始。经过转换时间 T_{st} 后, 转换结果分别从端口 DAC1_OUT1 和 DAC1_OUT2 输出并一直保持到 DADR 的值被改写, 或者 DAC_DACR.DAE 被设置成 0。
3. DAC_DACR.DAE 为 1 的状态下, 同时改写 DAC_DADR1 和 DAC_DADR2 的值将同步触发两个通道开始一次新的 DAC 转换。同样地, 经过转换时间 T_{st} 后, 从 DAC1_OUT1 和 DAC1_OUT2 输出新的转换结果。
4. 设置 DAC_DACR.DAE 为 0, 两个通道同步关闭, DAC1_OUT1 和 DAC1_OUT2 输出高阻态。

19.3.4 外部数据转换

DAC 除了可以转换 DAC_DADRx 中的数据, 还可以转换来自数据计算单元 (DCU) 的数据。DAC1 的通道 1 和通道 2 分别对应 DCU 的单元 1 和单元 2, DAC2 的通道 1 和通道 2 分别对应 DCU 的单元 3 和单元 4。设置 DAC_DACR.EXTDSLy 为 1 即选择转换来自相应 DCU 单元的数据寄存器 0 的低 12 位数据。

转换外部数据时禁止使用输出放大器和 ADC 转换优先模式。

19.3.5 ADC 转换优先模式

DAC 开始 DAC 转换时模拟电源上可能出现短暂的浪涌电流, 这会对正在进行中的 ADC 转换产生干扰。ADC 转换优先模式通过改变 DAC 转换数据的更新时序可以有效地避免这种干扰的发生。

设置 DAC_DAADPCR.ADPEN 为 1，DAC 进入 ADC 转换优先模式：

- 如果在 ADC 转换期间改写 DAC_DADRy 的值，DAC 转换并不会立即开始，而是要等到 ADC 转换完成后才开始。所以从改写 DAC_DADRy 到实际开始 DAC 转换最长需要等待一次 ADC 转换的时间，在此期间 DAC_DADRy 的值与模拟输出值并不一致。
- 如果改写 DAC_DADRy 时 ADC 处于停止状态 (ADC_STR.STRT 为 0)，DAC 转换则会在 2 个 ADCLK 周期后开始。

DAC_DAADPCR.ADCSLn (n=1~3) 用于选择 ADC 转换优先的 ADC 通道。ADC 转换优先模式动作时序如下：

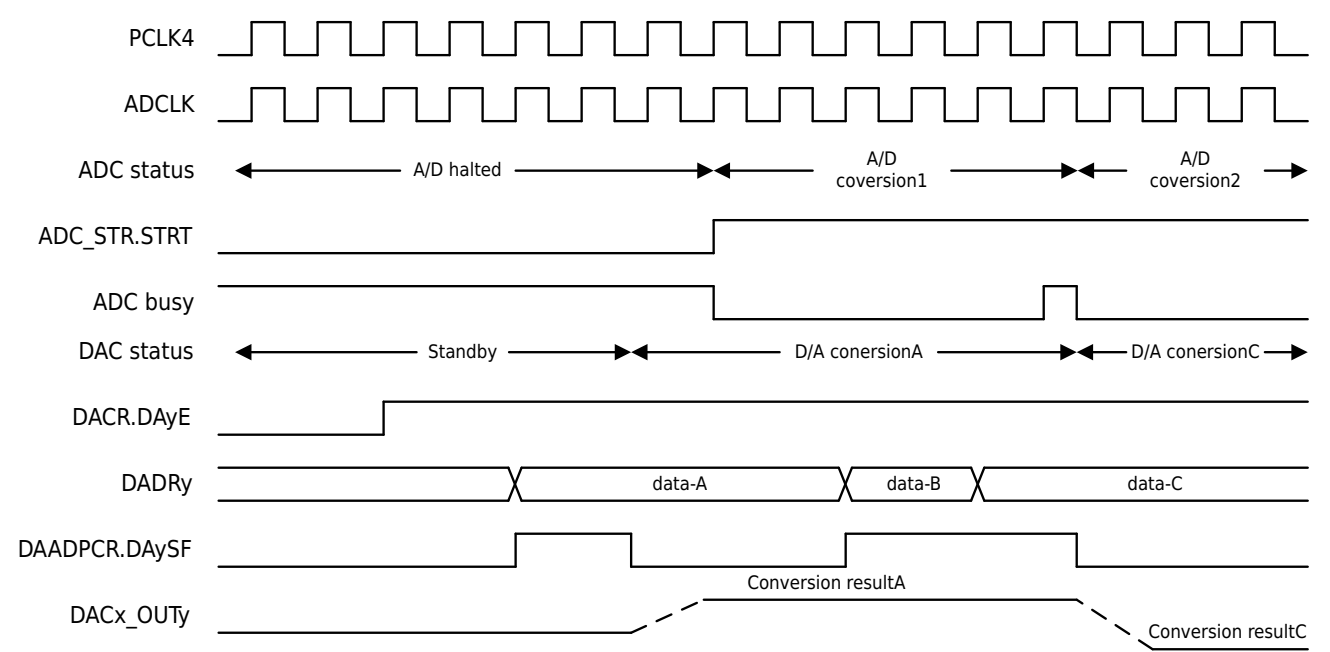


图 19-3 ADC 转换优先模式动作示意图

 **说明**

选择转换外部数据 (DAC_DACR.EXTDSLy=1) 时，ADC 转换优先模式无效。

下面以 DAC1 通道 1 为例，对 ADC 转换优先模式的设定步骤进行介绍。

1. 确认 ADC 处于停止状态，设置 DAC_DAADPCR.ADCSL1。
2. 确认 ADC 处于停止状态，设置 DAC_DACR.DA1E 为 1。
3. 将转换数据写入 DAC_DADR1。
 - 转换数据 A 被写入 DAC_DADR1 时，ADC_STR.STRT 为 0，ADC 处于停止状态，DAC 转换在 2 个 ADCLK 周期后开始。
 - 转换数据 B 被写入 DAC_DADR1 时，ADC_STR.STRT 为 1，ADC 正在进行 ADC 转换，DAC 转换要等到 ADC 转换完成后才会开始。而在 DAC 转换开始前转换数据 C 已被写入 DAC_DADR1，因此最终被转换的是转换数据 C，而不是转换数据 B。

为避免转换数据丢失，请先查看 DAC 转换状态寄存器 (DAC_DAADPCR.DaYSF)，确认当前数据转换完成后，再改写 DAC_DADRy 的值。

19.4 寄存器

19.4.1 寄存器总表

- DAC1 基地址：0x40041000
- DAC2 基地址：0x40041400

表 19-2 数模转换器 (DAC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	DAC_DADR1	数据寄存器 1	16
0x02	DAC_DADR2	数据寄存器 2	16
0x04	DAC_DACR	控制寄存器	16
0x06	DAC_DAADPCR	ADC 转换优先控制寄存器	16
0x1C	DAC_DAOCR	模拟输出控制寄存器	16

19.4.2 数据寄存器 (DAC_DADRy) (y=1~2)

该寄存器用于存放 DAC 变换的数据，每个通道对应有一个。只要允许 DAC 转换，DAC_DADRy 中的值就被转换成模拟电压并从模拟端口输出。对 DAC_DADR1 进行 32 位操作可实现该单元双通道转换数据同时设置。

偏移地址 (Offset)：0x00 (DAC_DADR1)，0x02 (DAC_DADR2)

数据右对齐 (DACR.DPSEL=0 时)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved				DADR											
Access																

位/位域	标记	位名	功能描述	读写
15:12	Reserved	保留	-	-
11:0	DADR	变换数据	变换数据设定值	RW

数据左对齐 (DACR.DPSEL=1 时)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	DADR												Reserved			
Access																

位/位域	标记	位名	功能描述	读写
15:4	DADR	变换数据	变换数据设定值	RW
3:0	Reserved	保留	-	-

19.4.3 控制寄存器 (DAC_DACR)

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved			EXTDSL2	EXTDSL1	DAAMP2	DAAMP1	DPSEL	Reserved					DA2E	DA1E	DAE
Access				RW	RW	RW	RW	RW						RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:13	Reserved	保留	-	-
12	EXTDSL2	通道 2 外部数据选择	0b0: 选择 DAC_DADR2 作为通道 2 的转换数据 0b1: 选择外部数据作为通道 2 的转换数据	RW
11	EXTDSL1	通道 1 外部数据选择	0b0: 选择 DAC_DADR1 作为通道 1 的转换数据 0b1: 选择外部数据作为通道 1 的转换数据	RW
10	DAAMP2	通道 2 输出放大器使能	0b0: 禁止通道 2 输出放大器 0b1: 启动通道 2 输出放大器	RW
9	DAAMP1	通道 1 输出放大器使能	0b0: 禁止通道 1 输出放大器 0b1: 启动通道 1 输出放大器	RW
8	DPSEL	数据寄存器格式选择	0b0: 右对齐格式 0b1: 左对齐格式	RW
7:3	Reserved	保留	-	-
2	DA2E	通道 2 使能	DAE=0 时此位设定有效: 0b0: 通道 2 禁止 0b1: 通道 2 使能	RW
1	DA1E	通道 1 使能	DAE=0 时此位设定有效: 0b0: 通道 1 禁止 0b1: 通道 1 使能	RW
0	DAE	通道总使能	0b0: 通道 1, 2 禁止 0b1: 通道 1, 2 使能	RW

 **说明**

- 该寄存器用于控制 DAC 转换和 DAC 输出，具体信息请参见“DAC 模拟输出控制寄存器”相关描述。
- DA1E 位为 0 并且 DAE 位也为 0 时，通道 1 停止 DAC 转换，这时不管 DAAMP1 是什么值，通道 1 的模拟输出被禁止，端口呈现高阻态。DA2E 位为 0 并且 DAE 位也为 0 时，通道 2 停止 DAC 转换，这时不管 DAAMP2 是什么值，通道 2 的模拟输出被禁止，端口呈现高阻态。
- 如果选择 ADC 转换优先模式，请在 ADC 停止（ADC_STR.STRT=0）的状态下设定 DA2E，DA1E 和 DAE 位。同时，为了避免 ADC 被意外启动，请将 ADC 的触发选择设定为软件触发。

19.4.4 模拟输出控制寄存器 (DAC_DAOCR)

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	DAODIS2	DAODIS1	Reserved													
Access	RW	RW														

位/位域	标记	位名	功能描述	读写
15	DAODIS2	DACx_OUT2 端口输出控制	● 0b0: 禁止端口输出模拟电压 ● 0b1: 允许端口输出模拟电压	RW
14	DAODIS1	DACx_OUT1 端口输出控制	● 0b0: 禁止端口输出模拟电压 ● 0b1: 允许端口输出模拟电压	RW
13:0	Reserved	保留	-	-

表 19-11 DAC 转换和模拟输出控制

DAE	DAyE	DAAMP	DAODIS	DAC 转换	输出放大器	DACx_OUTy 输出	DAO 输出
0b0	0b0	-	-	停止	停止	高阻态	高阻态
	0b1	0b0	0b0	启动	停止	普通输出	普通输出
			0b1	启动	停止	高阻态	普通输出
		0b1	0b0	启动	启动	放大输出	普通输出
			0b1	启动	启动	高阻态	普通输出
0b1	0b0	0b0	0b0	启动	停止	普通输出	普通输出
			0b1	启动	停止	高阻态	普通输出
		0b1	0b0	启动	启动	放大输出	普通输出
			0b1	启动	启动	高阻态	普通输出
	0b1	0b0	0b0	启动	停止	普通输出	普通输出
			0b1	启动	停止	高阻态	普通输出
		0b1	0b0	启动	启动	放大输出	普通输出
			0b1	启动	启动	高阻态	普通输出

19.4.5 ADC 转换优先控制寄存器 (DAC_DAADPCR)

Offset	Bit Position															
0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	ADPEN	Reserved					DA2SF	DA1SF	Reserved					ADCSL3	ADCSL2	ADCSL1
Access	RW						RO	RO						RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	ADPEN	ADC 转换优先模式选择	0b0: ADC 转换优先模式无效 0b1: ADC 转换优先模式有效	RW
14:10	Reserved	保留	-	-
9	DA2SF	通道 2 数据更新状态	本标志位只在 ADC 转换优先模式时变化: 0b0: 通道 2 数据更新完成 0b1: 通道 2 数据更新中	RO
8	DA1SF	通道 1 数据更新状态	本标志位只在 ADC 转换优先模式时变化: 0b0: 通道 1 数据更新完成 0b1: 通道 1 数据更新中	RO
7:3	Reserved	保留	-	-
2	ADCSL3	ADC_3 优先选择	0b0: 不选择 ADC_3 作为 ADC 转换优先对象 0b1: 选择 ADC_3 作为 ADC 转换优先对象	RW
1	ADCSL2	ADC_2 优先选择	0b0: 不选择 ADC_2 作为 ADC 转换优先对象 0b1: 选择 ADC_2 作为 ADC 转换优先对象	RW
0	ADCSL1	ADC_1 优先选择	0b0: 不选择 ADC_1 作为 ADC 转换优先对象 0b1: 选择 ADC_1 作为 ADC 转换优先对象	RW

 说明

- 该寄存器用于 ADC 转换优先模式的控制。
- ADPEN 设定成 1 时 ADC 转换优先模式有效，请在设定 ADPEN 的同时选择作为优先对象的 ADC 单元。
- DA1SF、DA2SF 为 DA 转换状态标志位，反映的是当前该通道数据寄存器中的数据是否已被转换。

19.5 注意事项

19.5.1 模块停止功能的设置

可以使用模块停止控制寄存器设定 DAC 有效或者停止。系统复位后 DAC 都是停止的，需要解除模块停止状态后才可以访问寄存器。具体描述请参见《电源控制 (PWC)》章节“动作模式与低功耗模式”。

19.5.2 模块停止状态下的 DAC 动作

如果 DAC 是在 DAC 转换时进入模块停止状态的，模拟输出将被保持，这时流过模拟电源的电流与 DAC 转换时相同。如果需要进一步减小模块停止状态下的功耗，请将 DAC_DACR 的 DAyE 和 DAE 位全部设为 0。

19.5.3 停止低功耗模式下的 DAC 动作

如果系统是在 DAC 转换时进入停止模式 (Stop Mode) 的，模拟输出将被保持，这时流过模拟电源的电流与 DAC 转换时相同。如果需要进一步减小停止模式下的功耗，请将 DAC_DACR 的 DAyE 和 DAE 位全部设为 0。

19.5.4 掉电低功耗模式下的 DAC 动作

如果系统是在 DAC 转换时进入掉电模式 (Power Down Mode) 的，模拟输出将被置为高阻态。

19.5.5 使用输出放大器

使用输出放大器时请使用以下的初始化程序。

- 步骤 1. 给 DAC_DAD_{Ry} 写全 0。
- 步骤 2. 设置 DAC_DACR.DAAMP_y 为 1。
- 步骤 3. 设置 DAC_DACR.DAE 或者 DAyE 为 1。
- 步骤 4. 等待 3μs 后将变换数据写入 DAC_DAD_{Ry}。

关闭 DAC 可以使放大器停止工作。若要再次使用放大器，请重复步骤 1~4。

20 温度传感器（OTS）

20.1 简介

温度传感器（On-chip Temperature Sensor，以下简称 OTS）可以获取芯片内部的温度，以支持系统的可靠性操作。OTS 提供了一组与温度相关的数字量，通过计算可以得到温度值。不使用时可以通过模块停止功能将其关闭以减少系统功耗。

20.2 功能说明

20.2.1 功能框图

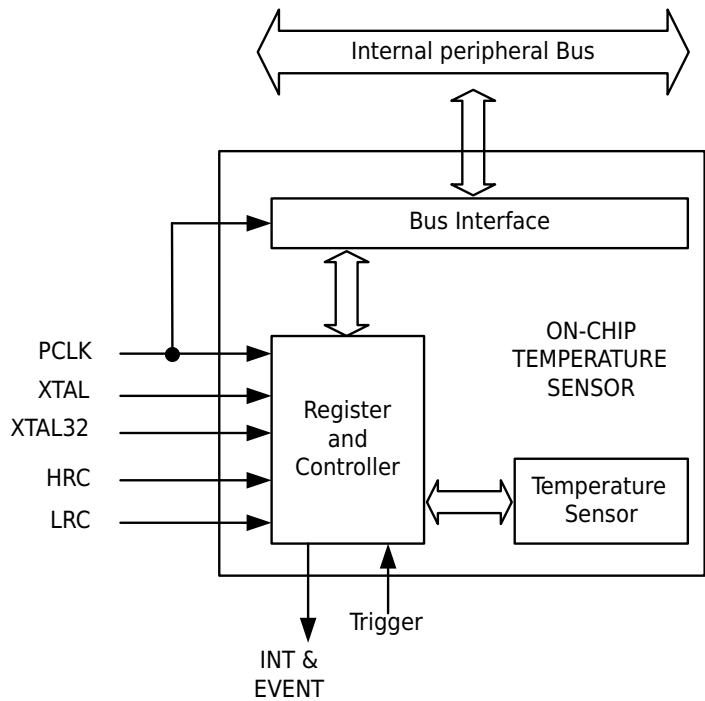


图 20-1 OTS 功能框图

20.2.2 使用方法

使用 OTS 获取芯片内部温度之前，先要对其进行定标，且测量精度与定标方式有关，请参考下述定标实验的说明进行定标。使用时请先关闭模块停止功能并启动内部低速时钟 LRC，同时请根据使用情况选择启动内部高速时钟 HRC、外部高速时钟 XTAL 以及外部低速时钟 XTAL32。将寄存器 OTS_CTL.OTSST 设置成“1”后测温开始，测温完成后 OTSST 位会自动清零。确认 OTSST 为“0”后，读取寄存器 OTS_DR1、OTS_DR2 以及 OTS_ECR 中的温度参数，并利用以下计算公式求得温度值。

$$T = K \times (1.7/D1 - 1/D2) \times E_{hrc} + M$$

参数说明：

- T：温度（℃）
- K：温度斜率（通过定标实验确定）
- D1：温度数据 1（从寄存器 OTS_DR1 读取）
- D2：温度数据 2（从寄存器 OTS_DR2 读取）
- E_{hrc} ：HRC 频率误差补偿数据（从寄存器 OTS_ECR 读取）
- M：温度偏移量（通过定标实验确定）

寄存器 OTS_CTL.OTSCK 用来选择测温时钟。选择 HRC 动作时，其频率误差可能会影响到最终计算出的温度的精度。为了消除这一误差，请在测温前启动 XTAL32 并在定标和计算温度时使用 E_{hrc} 。选择 XTAL 动作时请忽略 E_{hrc} 并将其示为常量 1。

定标实验：在两个确定的温度下进行定标实验，计算 K 和 M。

$$K = (T2 - T1) / (A2 - A1)$$

$$M = T1 - K \times A1 = T2 - K \times A2$$

参数说明：

- T1：实验温度 1
- T2：实验温度 2
- A1： $(1.7/D1_{T1} - 1/D2_{T1}) \times E_{hrcT1}$
D1_{T1}, D1_{T1}, E_{hrcT1} 分别是在温度 T1 下测到的 D1, D2, E_{hrc}。
- A2： $(1.7/D1_{T2} - 1/D2_{T2}) \times E_{hrcT2}$
D1_{T2}, D1_{T2}, E_{hrcT2} 分别是在温度 T2 下测到的 D1, D2, E_{hrc}。

作为参考，芯片内部预置了三组温度数据，可任意选择两个温度点的数据进行定标。由于测试环境温度会有偏差，预置数据仅供参考使用。

表 20-1 OTS 预置温度数据

测试温度	读取方式		测试条件
	温度数据 1（D1）	温度数据 2（D2）	
T _A = -40℃	OTS_PDR3[15:0]	OTS_PDR3[31:16]	XTAL=8MHz
T _A = 25℃	OTS_PDR1[15:0]	OTS_PDR1[31:16]	XTAL=8MHz
T _A = 125℃	OTS_PDR2[15:0]	OTS_PDR2[31:16]	XTAL=8MHz

E_{hrc} 是为了消除 HRC 本身的误差对测量温度影响的一个校准值，可以从寄存器 OTS_ECR 中读取，使用方法如下：

表 20-2 E_{hrc} 使用及设定方法

动作时钟	定标实验（计算 K,M）		温度测量
(OTSCK)	使用预置温度数据	重新获取温度数据	(计算 T)
XTAL	E _{hrc} =1	E _{hrc} =1	E _{hrc} =1
HRC	E _{hrc} =f _{hrc} (MHz)/0.032768	E _{hrc} =OTS_ECR[15:0]	E _{hrc} =OTS_ECR[15:0]

 说明

f_{hrc} 为 HRC 的标准频率，即 16MHz 或者 20MHz。

寄存器 OTS_CTL.TSSTP 用来选择测温完成后是否关闭模拟温度传感器。TSSTP 初始值为“0”，表示在一次测温完成后将模拟温度传感器置于开启状态，这样在下一次测温时将自动跳过模拟温度传感器由关闭到开启时的稳定时间。如需在每次测温后关闭模拟温度传感器，请将 TSSTP 设置成“1”。

可以通过其他的外设事件触发开始测温，使用时请将触发源的触发目标设置成 OTS。测温完成时也可以产生触发其他的外设启动的事件，使用时请设定寄存器 OTS_TRG 选择触发目标。使用测温完成中断时请将寄存器 OTS_CTL.OTSIE 设置为“1”。

20.3 寄存器

20.3.1 寄存器总表

OTS 预置数据寄存器基地址: 0x40010600

表 20-3 OTS 预置数据寄存器偏移地址

偏移地址	寄存器	描述	位宽
0xE0	OTS_PDR1	OTS 预置数据寄存器 1	32
0xF4	OTS_PDR2	OTS 预置数据寄存器 2	32
0xF8	OTS_PDR3	OTS 预置数据寄存器 3	32

OTS 基地址: 0x4004A800

表 20-4 温度传感器 (OTS) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	OTS_CTL	OTS 控制寄存器	16
0x02	OTS_DR1	OTS 数据寄存器 1	16
0x04	OTS_DR2	OTS 数据寄存器 2	16
0x06	OTS_ECR	OTS 误差补偿寄存器	16

20.3.2 OTS 控制寄存器 (OTS_CTL)

Offset	Bit Position															
0x00	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												TSSTP	OTSIE	OTSCK	OTSST
Access													RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3	TSSTP	关闭模拟温度传感器	选择测温结束后是否自动关闭模拟温度传感器 ● 0b0：不关闭模拟温度传感器 ● 0b1：关闭模拟温度传感器	RW
2	OTSIE	中断使能位	● 0b0：禁止测温结束中断请求 ● 0b1：允许测温结束中断请求	RW
1	OTSCK	时钟选择位	● 0b0：选择外部高速时钟 (XTAL) 动作 ● 0b1：选择内部高速时钟 (HRC) 动作	RW
0	OTSST	测温开始位	● 0b0：停止测温 ● 0b1：开始测温  说明 置“1”条件： 1. 软件置“1” 2. 硬件触发置“1” 清“0”条件： 1. 软件清“0” 2. 测温结束后自动清“0”	RW

20.3.3 OTS 数据寄存器 1 (OTS_DR1)

Offset	Bit Position															
0x02	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TSDC															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	TSDC	温度数据 D1	温度数据 D1，测温完成后自动更新  说明 请确认 OTS_CTL.OTSST 为“0”后再读取。	RO

20.3.4 OTS 数据寄存器 2 (OTS_DR2)

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TSDC															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	TSDC	温度数据 D2	温度数据 D2，测温完成后自动更新  说明 请确认 OTS_CTL.OTSST 为“0”后再读取。	RO

20.3.5 OTS 误差补偿寄存器 (OTS_ECR)

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TSEC															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	TSEC	误差系数	误差系数 E_{hrc} ，测温完成后自动更新  说明 请确认 OTS_CTL.OTSST 为“0”后再读取。	RO

20.3.6 OTS 预置温度数据寄存器 1 (OTS_PDR1)

Offset	Bit Position																															
0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	TSPD2																TSPD1															
Access	RO																RO															

位/位域	标记	位名	功能描述	读写
31:16	TSPD2	预置温度数据 D2	预置温度数据，供定标时使用。 该数据为 8MHz 时测得（测试温度 25℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D2 = TSPD2 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO
15:0	TSPD1	预置温度数据 D1	预置温度数据，供定标时使用。 该数据为 8MHz 时测得（测试温度 25℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D1 = TSPD1 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO

20.3.7 OTS 预置温度数据寄存器 2 (OTS_PDR2)

Offset	Bit Position																															
0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	TSPD2																TSPD1															
Access	RO																RO															

位/位域	标记	位名	功能描述	读写
31:16	TSPD2	预置温度数据 D2	预置温度数据，供定标时使用。 该数据为 8MHz 时测得（测试温度 125℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D2 = TSPD2 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO
15:0	TSPD1	预置温度数据 D1	预置温度数据，供定标时使用。 该数据为 8MHz 时测得（测试温度 125℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D1 = TSPD1 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO

20.3.8 OTS 预置温度数据寄存器 3 (OTS_PDR3)

Offset	Bit Position																															
0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	TSPD2																TSPD1															
Access	RO																RO															

位/位域	标记	位名	功能描述	读写
31:16	TSPD2	预置温度数据 D2	预置温度数据，供定时使用。 该数据为 8MHz 时测得（测试温度-40℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D2 = TSPD2 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO
15:0	TSPD1	预置温度数据 D1	预置温度数据，供定时使用。 该数据为 8MHz 时测得（测试温度-40℃），使用前请先与用由 OTSCK 选择的动作时钟的实际频率进行换算。 $D1 = TSPD1 \times f_{OTSCK}/8$ ， f_{OTSCK} 为实际动作时钟的频率，单位 MHz。	RO

21 高级控制定时器 (Timer6)

21.1 简介

高级控制定时器 6 (Timer6) 是一个 32/16 位计数宽度的高性能定时器，能在各种复杂应用场景中提供丰富、灵活的搭配组合和各种中断、事件、PWM 输出。该定时器支持锯齿波和三角波两种波形模式，可生成各种 PWM 波形（单边对齐独立 PWM、双边对称独立 PWM、双边对称互补 PWM、双边非对称 PWM 等）；单元间可实现软件同步和硬件同步（同步启动、停止、清零、刷新等）；各基准值寄存器支持缓存功能（单级缓存和双级缓存）；支持脉宽测量和周期测量；支持 2 相正交编码计数和 3 相正交编码计数；支持 EMB 控制；单元 1~单元 5、单元 7 支持 PTO 输出。本系列产品中搭载 8 个单元的 Timer6（单元 1~单元 4 为 32-bit 定时器；单元 5~单元 8 为 16-bit 定时器）。

21.2 主要特性

Timer6 基本的功能及特性如下表所示。

表 21-1 Timer6 的基本功能及特性

波形模式	锯齿波（递加、递减计数）
	三角波（递加递减计数）
基本功能	捕获输入
	软件同步
	硬件同步
	缓存功能
	脉宽测量
	周期测量
	正交编码计数
	通用 PWM 输出
	EMB 控制
	PTO 输出
中断输出	计数比较匹配中断
	计数周期匹配中断
	死区时间错误中断
事件输出	计数比较匹配事件
	计数周期匹配事件

21.3 功能说明

21.3.1 Timer6 框图

Timer6 的基本框图如下图所示。框图中所示“<t>”表示单元编号，即“<t>”为 1~8，本章节后文提到“<t>”时均指单元编号，不再赘述。

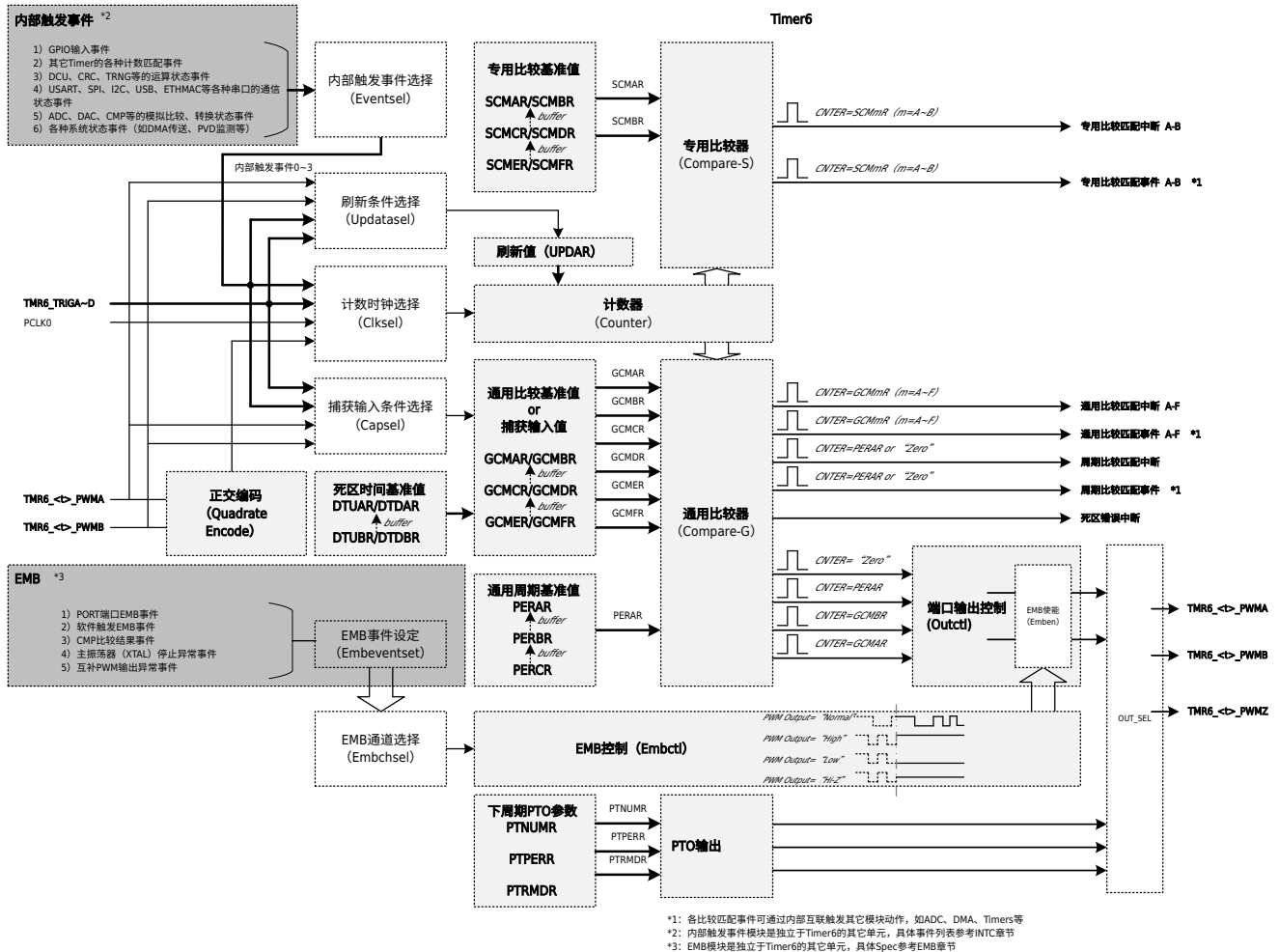


图 21-1 Timer6 基本框图

表 21-2 所示为 Timer6 的输入输出端口列表。

表 21-2 Timer6 端口列表

端口名	方向	功能
TMR6_<t>_PWMA	输入/输出	● 正交编码计数时钟输入端口或捕获输入端口或比较输出端口或 PTO AB 相输出端口 ● 硬件启动、停止、清零、刷新条件输入端口
TMR6_<t>_PWMB		
TMR6_TRIGA	输入/输出	● 硬件计数时钟输入端口或捕获输入端口 ● 硬件启动、停止、清零、刷新条件输入端口 ● PTO Z 相输出端口
TMR6_TRIGB		
TMR6_TRIGC		
TMR6_TRIGD		

21.3.2 基本动作

21.3.2.1 波形模式

Timer6 有 2 种基本计数波形模式，锯齿波模式和三角波模式。两种波形模式的基本波形如下所示。

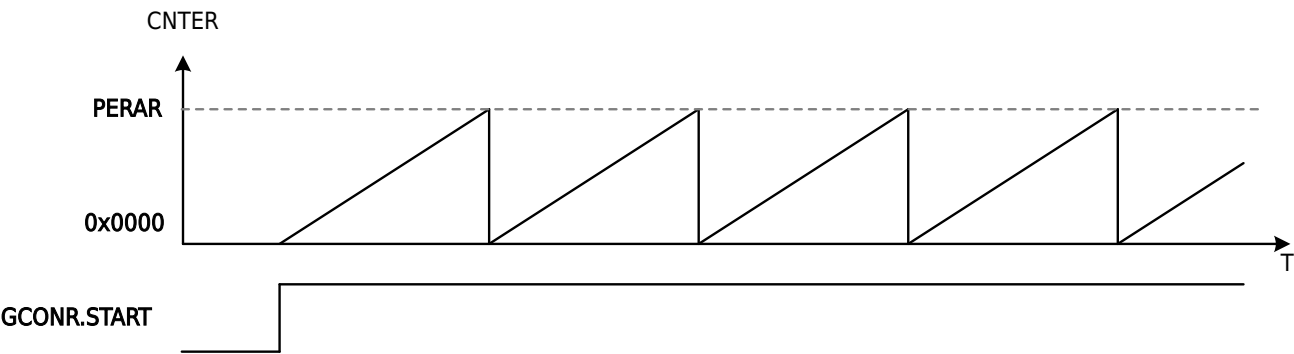


图 21-2 锯齿波波形 (递加计数)

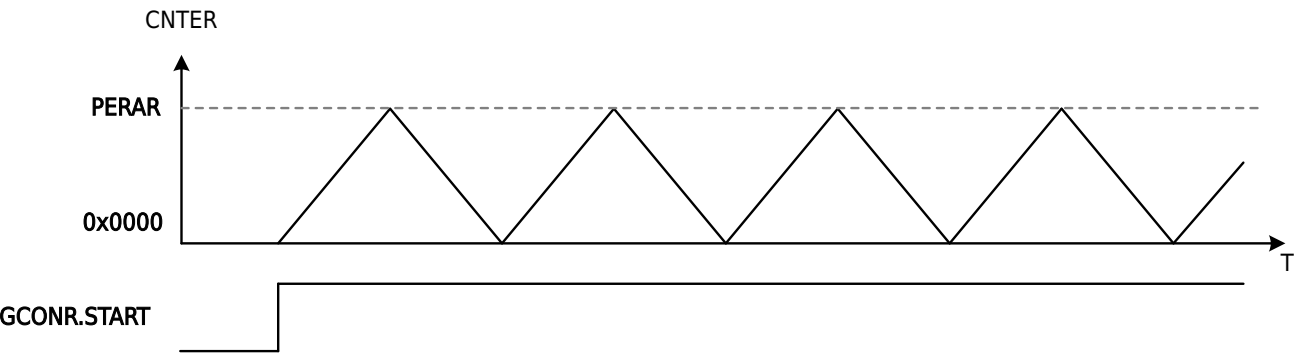


图 21-3 三角波波形

21.3.2.2 比较输出

每个单元的 Timer6 有 2 个比较输出端口 (TMR6_<t>_PWMA、TMR6_<t>_PWMB)，可在计数值与比较基准值比较匹配时输出指定的电平。GCMAR、GCMBR 寄存器对应了 TMR6_<t>_PWMA、TMR6_<t>_PWMB 的计数比较基准值。当定时器的计数值和 GCMAR 相等时，TMR6_<t>_PWMA 或 TMR6_<t>_PWMB 端口输出指定电平。

TMR6_<t>_PWMA 端口的计数开始时的电平、计数停止时的电平、计数溢出时的电平、计数比较匹配时的电平等，可通过端口控制寄存器 (PCNAR) 的 PCNAR.STACA、PCNAR.STPCA、PCNAR.OVFCA、PCNAR.UDFCA、PCNAR.CMAU<D>A、PCNAR.FORCA 位设定。

TMR6_<t>_PWMB 端口的计数开始时的电平、计数停止时的电平、计数溢出时的电平、计数比较匹配时的电平等，可通过端口控制寄存器 (PCNBR) 的 PCNBR.STACB、PCNBR.STPCB、PCNBR.OVFCEB、PCNBR.UDFCEB、PCNBR.CMAU<D>B、PCNBR.FORCB 位设定。下图为比较输出的动作例。

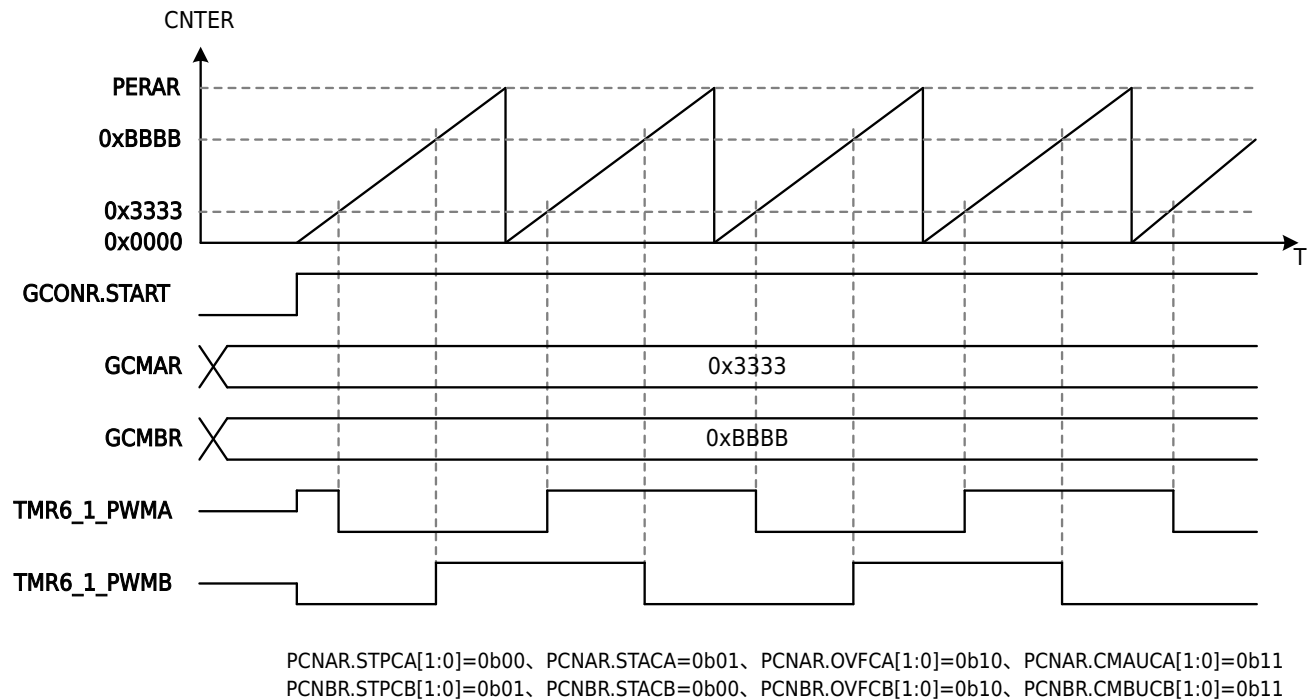


图 21-4 比较输出动作

21.3.2.3 捕获输入

每个单元都具有捕获输入功能，具备 2 组捕获输入寄存器（GCMAR、GCMBR），用于保存捕获到的当前计数值。设定端口控制寄存器（PCNAR）的 CAPMDA位为 1，捕获输入功能变为有效。当设定了对应的捕获输入条件且该条件有效时，当前的计数值就被保存到相应的捕获寄存器（GCMAR、GCMBR）中。各单元的每组捕获输入的条件可以是内部触发事件输入 0~3、TMR6_TRIGA~D 的端口输入、TMR6_<t>_PWMA 或 TMR6_<t>_PWMB 的端口输入以及 TMR6_<t>_XOR（TMR6_<t>_PWMA 与 TMR6_<t>_PWMB 的端口输入异或）等，具体的条件选择可通过硬件捕获事件选择寄存器（HCPAR、HCPBR）来设定。下图为捕获输入的动作例。

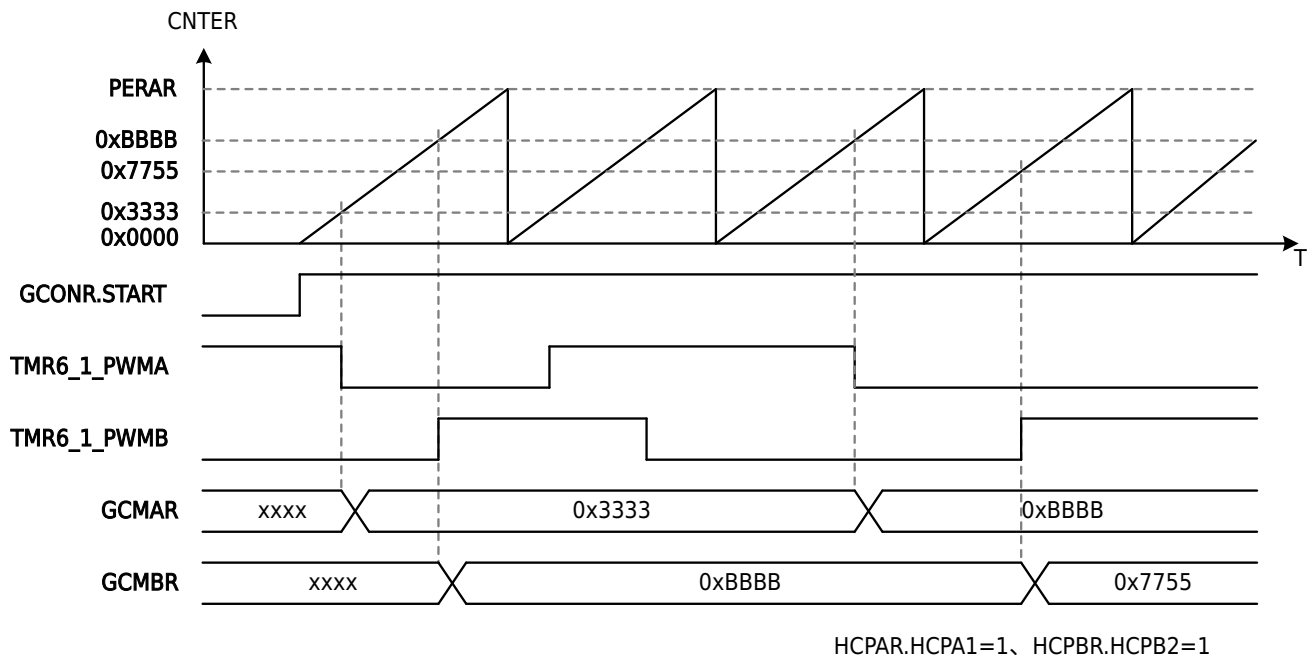


图 21-5 捕获输入动作

21.3.3 时钟源选择

Timer6 的计数时钟可以有以下几种选择：

1. PCLK0 的 1、2、4、8、16、32、64、128、256、512、1024 分频 (GCONR.CKDIV[3:0]设定)
2. 内部触发事件输入 0~3 (HCUPR[11:8]或 HCDOR[11:8]设定)
3. TMR6_TRIGA~D 的端口输入 (HCUPR[23:16]或 HCDOR[23:16]设定)
4. TMR6_<t>_PWMA 和 TMR6_<t>_PWMB 的正交编码关系输入 (HCUPR[7:0]或 HCDOR[7:0]设定)

计数时钟源选择 1 时为软件计数模式，计数时钟源选择 2、3、4 时为硬件计数模式。上述描述可以看到，2、3、4 时钟互相独立，可分别设定有效或无效，并且当选择 2、3、4 时钟时，1 时钟自动无效。



说明

硬件计数模式时请将 GCONR.CKDIV 设定为 0。

21.3.4 计数方向

Timer6 的定时器计数方向可通过软件方式改变。不同波形模式时，改变计数方向的方法略有不同。

21.3.4.1 锯齿波计数方向

锯齿波模式时，计数方向可在定时器计数中或停止时设定。

在递增计数中时，设定 GCONR.DIR=0b0（递减计数），则定时器计数到上溢后变为递减计数模式；在递减计数中时，设定 GCONR.DIR=0b1（递增计数），则定时器计数到下溢后变为递增计数模式。

在计数器停止时，设定 GCONR.DIR 位的同时设定 GCONR.FODIR=0b1，可立即将 GCONR.DIR 的设定反映到计数方向中，启动计数后按照设定的 GCONR.DIR 计数；设定 GCONR.DIR 位的同时设定 GCONR.FODIR=0b0，计数开始后直至上溢或下溢时，GCONR.DIR 的设定反映到计数中。

21.3.4.2 三角波计数方向

三角波模式时，计数过程中设定计数方向无效，计数至计数峰点或计数谷点时，自动改变计数方向。

在计数器停止时，设定 GCONR.DIR 位的同时设定 GCONR.FODIR=0b1，可立即将 GCONR.DIR 的设定反映到计数方向中，启动计数后按照设定的 GCONR.DIR 计数；设定 GCONR.DIR 位的同时设定 GCONR.FODIR=0b0，计数开始后直至上溢或下溢时，自动改变计数方向，设定 GCONR.DIR 位无效。

21.3.5 计数器刷新

Timer6 具有一个通用刷新值寄存器 (UPDAR)，可以在计数停止或计数中时，对通用计数值寄存器 (CNTERR) 的计数值实时进行刷新。

计数值的刷新条件由硬件刷新事件选择寄存器 (HUPDR) 或软件同步刷新控制寄存器 (SUPDR) 的对应位设定。当设定的刷新事件有效时，通用计数值寄存器 (CNTERR) 的值更新为通用刷新值寄存器 (UPDAR) 中指定的值。下图是锯齿波模式在计数中的硬件刷新动作例。

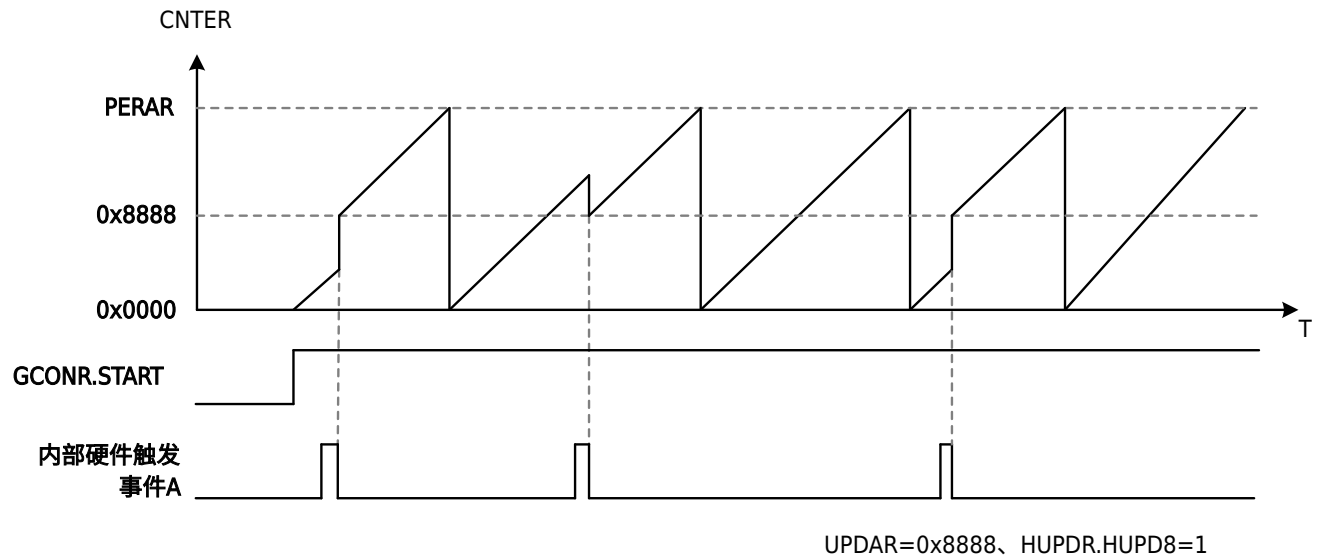


图 21-6 硬件刷新动作

21.3.6 数字滤波

TMR6_<t>_PWMA、TMR6_<t>_PWMB、TMR6_TRIGA~D 端口输入都具有数字滤波功能。TMR6_<t>_PWMA、TMR6_<t>_PWMB 通过设定滤波控制寄存器 (FCNCR) 的相关使能位开启对应端口的滤波功能，滤波有效时的滤波基准时钟也可通过滤波控制寄存器 (FCNCR) 设定；TMR6_TRIGA~D 端口是一组单元间共用的端口通过设定滤波控制寄存器 (FCNTR) 的相关使能位开启对应端口的滤波功能，滤波有效时的滤波基准时钟也可通过滤波控制寄存器 (FCNTR) 设定。

在滤波采样基准时钟采样到端口上 3 次一致的电平时，该电平被当作有效电平传送到模块内部；小于 3 次一致的电平会被当作外部干扰滤掉，不传送到模块内部。其动作例如下图所示。

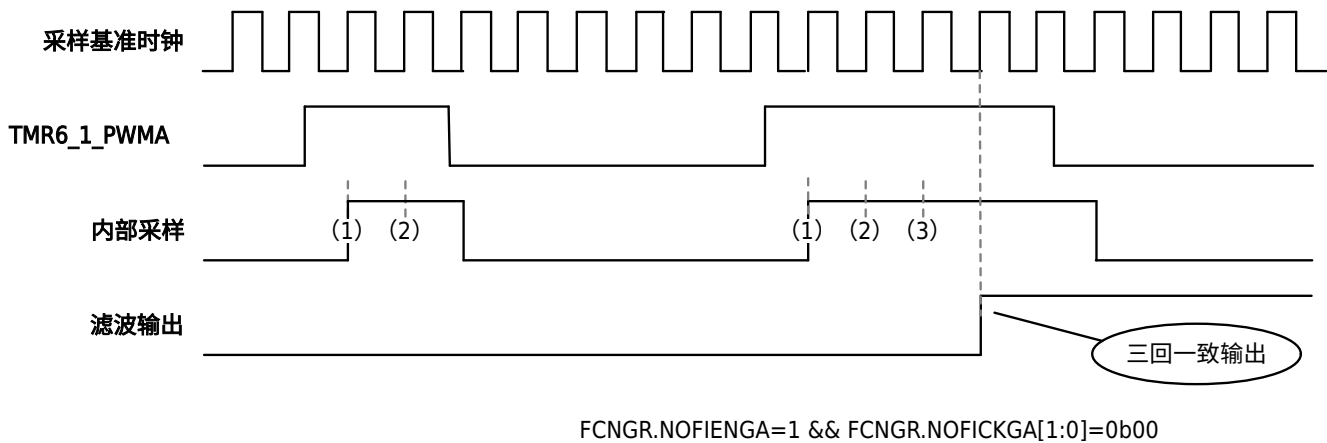


图 21-7 捕获输入端口的滤波功能

21.3.7 软件同步

软件同步启动

各单元可通过设定软件同步启动控制寄存器 (SSTAR) 的相关位，实现目标单元计数器 (CNTNR) 的同步启动。

另外，产品配置了计数同步使能寄存器 TMR_SYNENRx (x=1~2)，用于同步启动产品所搭载的所有 Timer (包括 Timer0、Timer2、Timer4、Timer6、TimerA)。使用方法和寄存器说明参考《通用定时器 (Timer0)》章节。

软件同步停止

各单元可通过设定软件同步停止控制寄存器 (SSTPR) 的相关位, 实现目标单元计数器 (CNTER) 的同步停止。

软件同步清零

各单元可通过设定软件同步清零控制寄存器 (SCLRR) 的相关位, 实现目标单元计数器 (CNTER) 的同步清零。

软件同步刷新

各单元可通过设定软件同步刷新控制寄存器 (SUPDR) 的相关位, 实现目标单元计数器 (CNTER) 的同步刷新。

如下图所示, 设定 SSTAR.SSTA1=SSTAR.SSTA2=SSTAR.SSTA3=1, 即可实现单元 1~3 的软件同步启动; 设定 SSTPR.SSTP1=SSTPR.SSTP2=SSTPR.SSTP3=1, 即可实现单元 1~3 的软件同步停止。

软件同步动作相关寄存器 (SSTAR、SSTPR、SCLRR、SUPDR) 是一组独立于单元外、各个单元间共用的寄存器, 这组寄存器的各个位只在写 1 时有效, 写 0 无效。在读取 SSTAR 寄存器时, 会读出各个单元的定时器状态 (计数停止或计数中), 在读取 SSTPR、SCLRR 或 SUPDR 时, 会读出零。

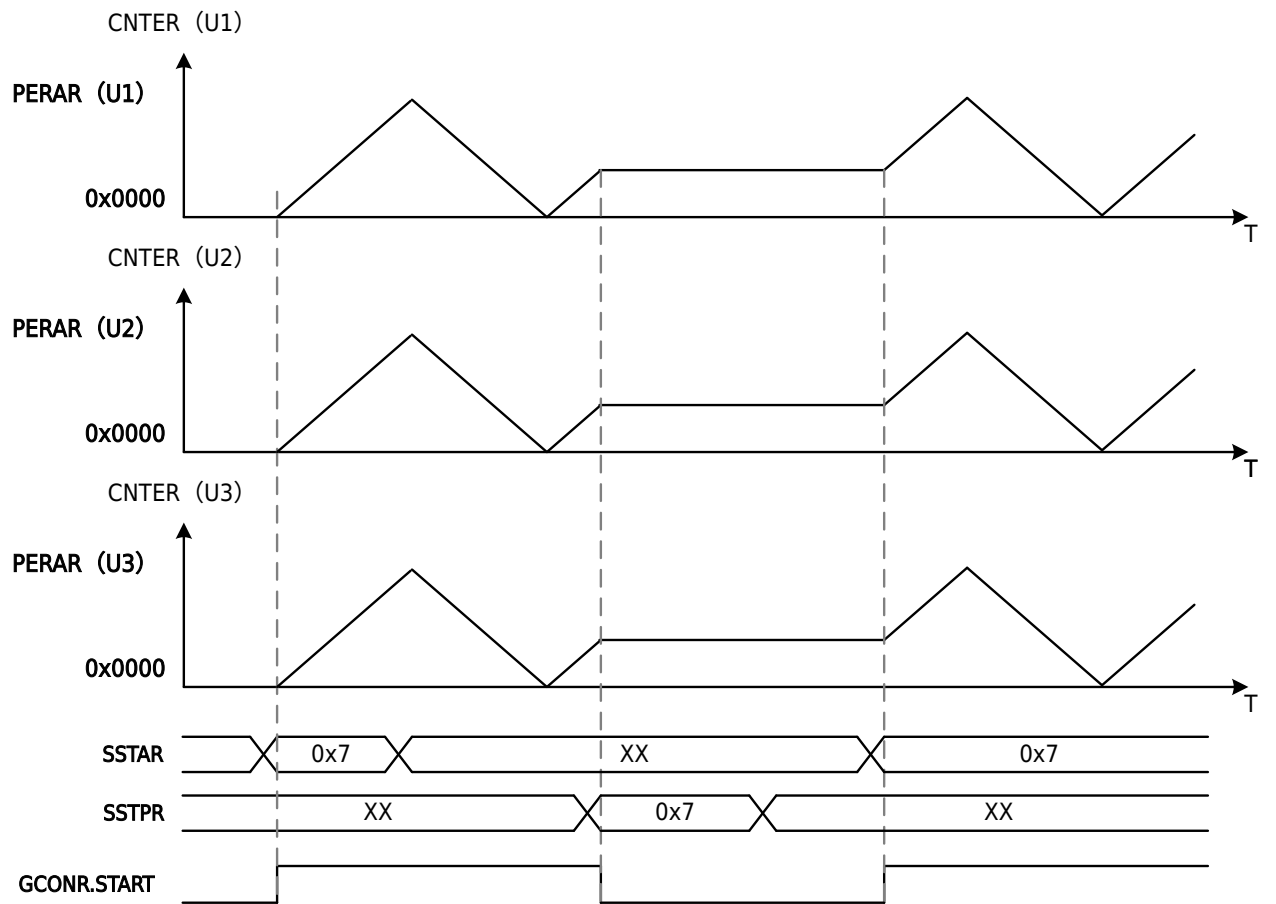


图 21-8 软件同步动作

21.3.8 硬件同步

每个单元除独立拥有 2 个通用输入端口 (TMR6_<t>_PWMA、TMR6_<t>_PWMB) 外, 还共同拥有 4 个通用外部触发输入端口 (TMR6_TRIGA、TMR6_TRIGB、TMR6_TRIGC、TMR6_TRIGD) 及 4 个内部触发事件输入条件, 可实现单元间的硬件同步动作。

内部触发事件的事件源可通过硬件触发事件选择寄存器中对应的编号设定来选择, 具体的寄存器内容详见《自动运行系统 (AOS) 》的“Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3)”章节, 具体

的事件对应关系请参考《中断控制器 (INTC)》章节。使用内部触发功能时，需要先将功能时钟控制寄存器 0 (PWC_FCG0) 的外围电路触发功能使能位置 1。

硬件同步启动

各单元均可选择用硬件方式启动定时器，选择相同硬件启动条件的单元即可在启动条件有效时实现同步启动。具体的硬件启动条件由硬件启动事件选择寄存器 (HSTAR) 的设定来决定。

硬件同步停止

各单元均可选择用硬件方式停止定时器，选择相同硬件停止条件的单元即可在停止条件有效时实现同步停止。具体的硬件停止条件由硬件停止事件选择寄存器 (HSTPR) 的设定来决定。

硬件同步清零

各单元均可选择用硬件方式清零定时器，选择相同硬件清零条件的单元即可在清零条件有效时实现同步清零。具体的硬件清零条件由硬件清零事件选择寄存器 (HCLRR) 的设定来决定。

硬件同步刷新

各单元均可选择用硬件方式刷新定时器，选择相同硬件刷新条件的单元即可在刷新条件有效时实现同步刷新。具体的硬件刷新条件由硬件刷新事件选择寄存器 (HUPDR) 的设定来决定。

硬件同步捕获输入

各单元均可选择用硬件方式实现捕获输入功能，选择相同捕获输入功能条件的单元即可在捕获输入功能条件有效时实现同步捕获输入。具体的硬件捕获输入功能条件由硬件捕获事件选择寄存器 (HCPAR、HCPBR) 的设定来决定。

硬件同步计数

各单元均可选择用硬件输入作为 CLOCK 进行计数，选择相同硬件计数条件的单元即可在硬件计数时钟有效时实现同步计数。具体的硬件计数条件由硬件递加事件选择寄存器 (HCUPR) 和硬件递减事件选择寄存器 (HCDOR) 的设定来决定。

选择硬件同步计数功能时，只是选择了外部输入时钟源，不影响定时器的启动、停止、清零动作。定时器的启动、停止、清零等还需要单独设定。

下图是单元 1~3 的硬件同步动作例。

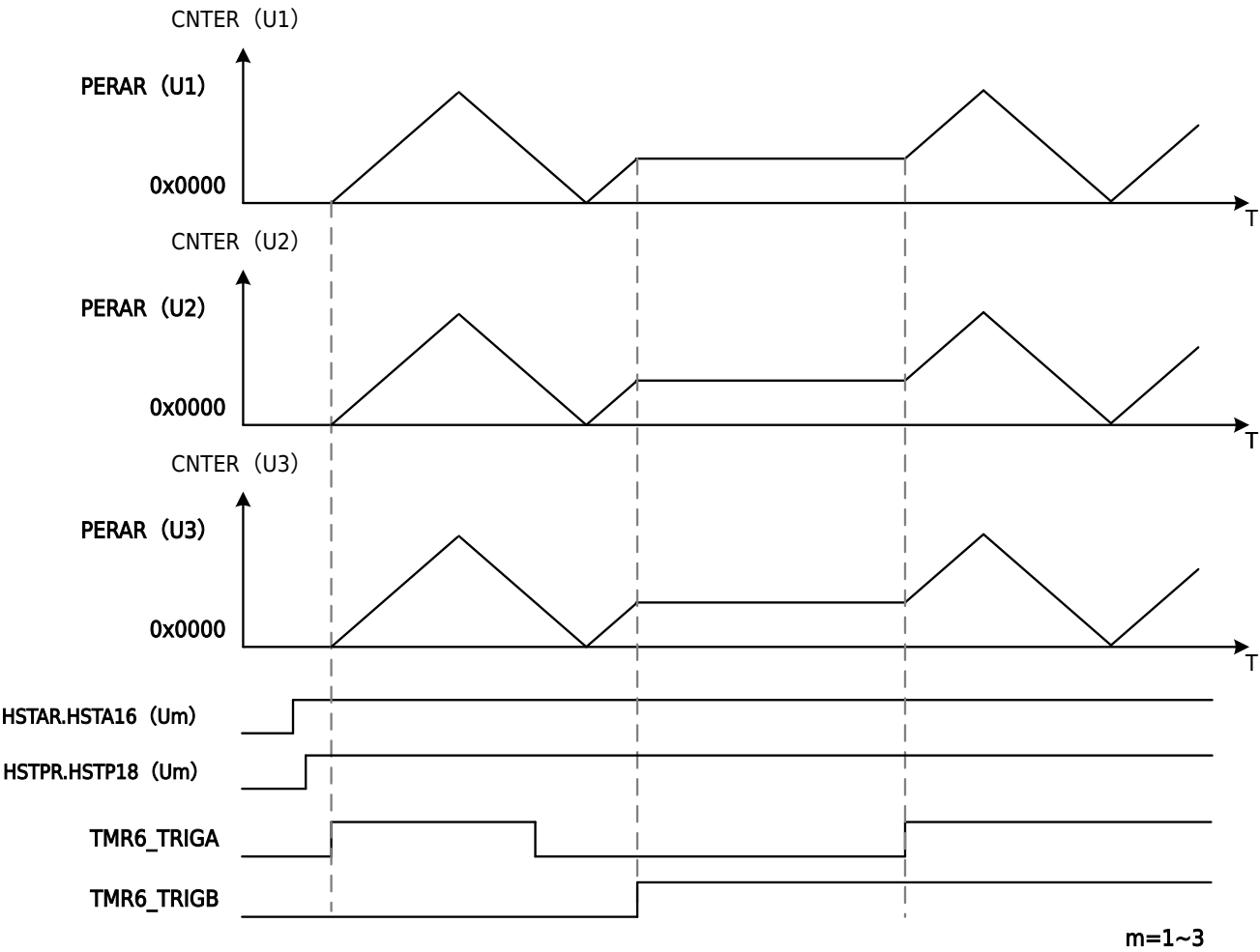


图 21-9 硬件同步动作

21.3.9 脉宽测量

在使用 TMR6_TRIGA~D 端口、TMR6_<t>_PWMA~B 端口的硬件触发相关功能（参考“硬件同步”）时，每个单元都可以实现 2 路独立的脉宽测量功能。

例如，将计数器的硬件启动条件设为 TMR6_TRIGA 的上升沿，硬件清零条件、停止条件和 GCMAR 寄存器的捕获输入条件均设为 TMR6_TRIGA 的下降沿，就可以实现连续的脉冲宽度测量。对应动作如下图所示。

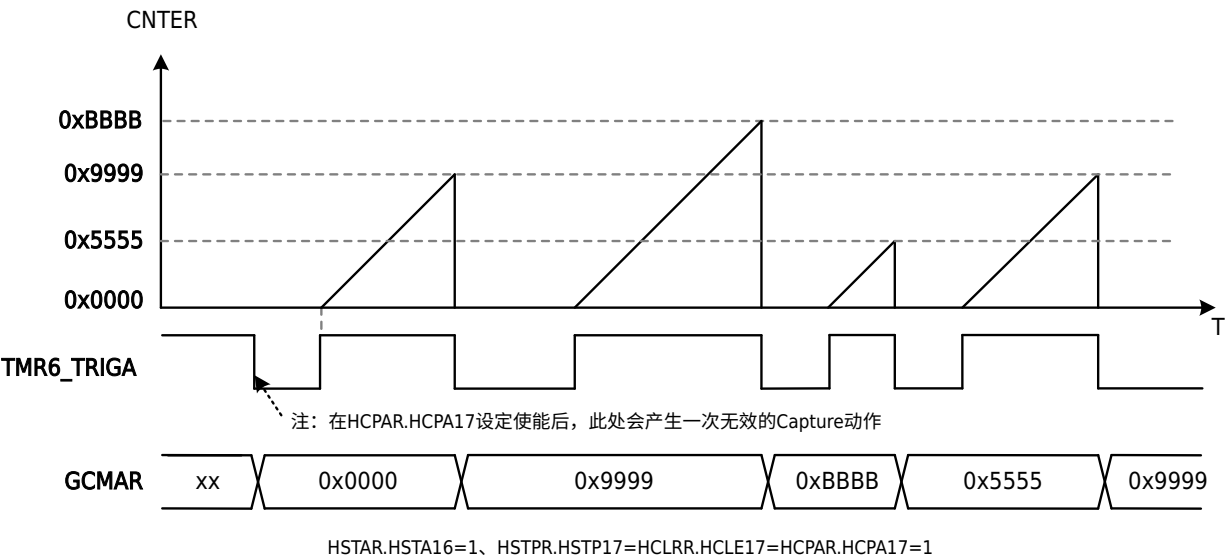


图 21-10 脉宽测量

21.3.10 周期测量

在使用 TMR6_TRIGA~D 端口、TMR6_<t>_PWMA~B 端口的硬件触发相关功能（参考“[硬件同步](#)”）时，每个单元都可以实现 2 路独立的周期测量功能。

例如，将计数器的硬件启动条件、硬件清零条件、GCMBR 寄存器的捕获输入条件均设为 TMR6_TRIGB 的上升沿，就可以实现连续的周期测量。对应动作如下图所示。

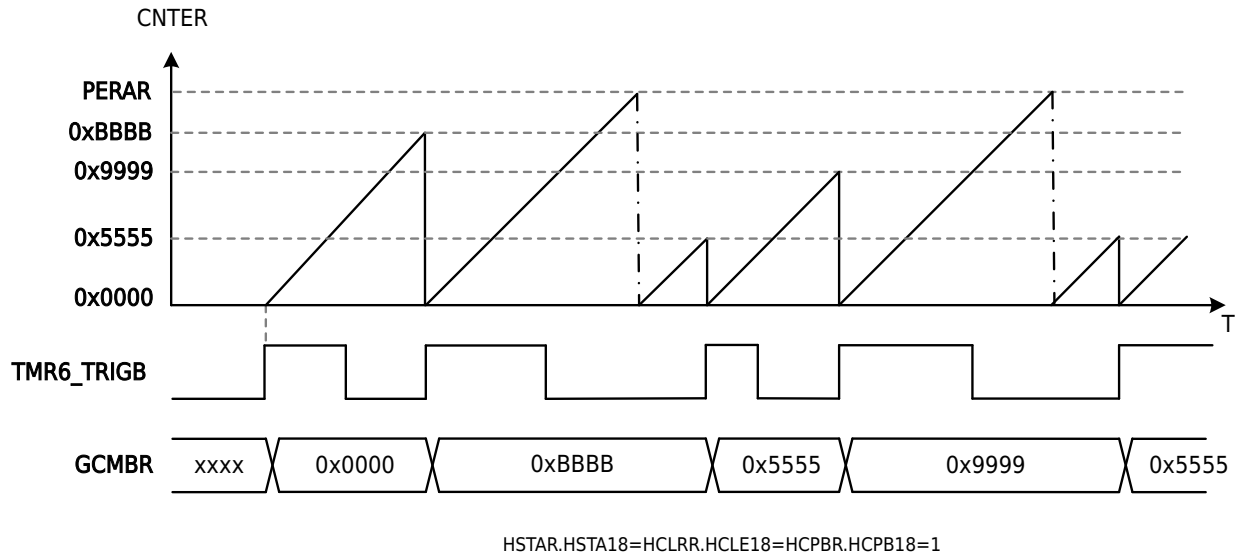


图 21-11 周期测量

21.3.11 缓存功能

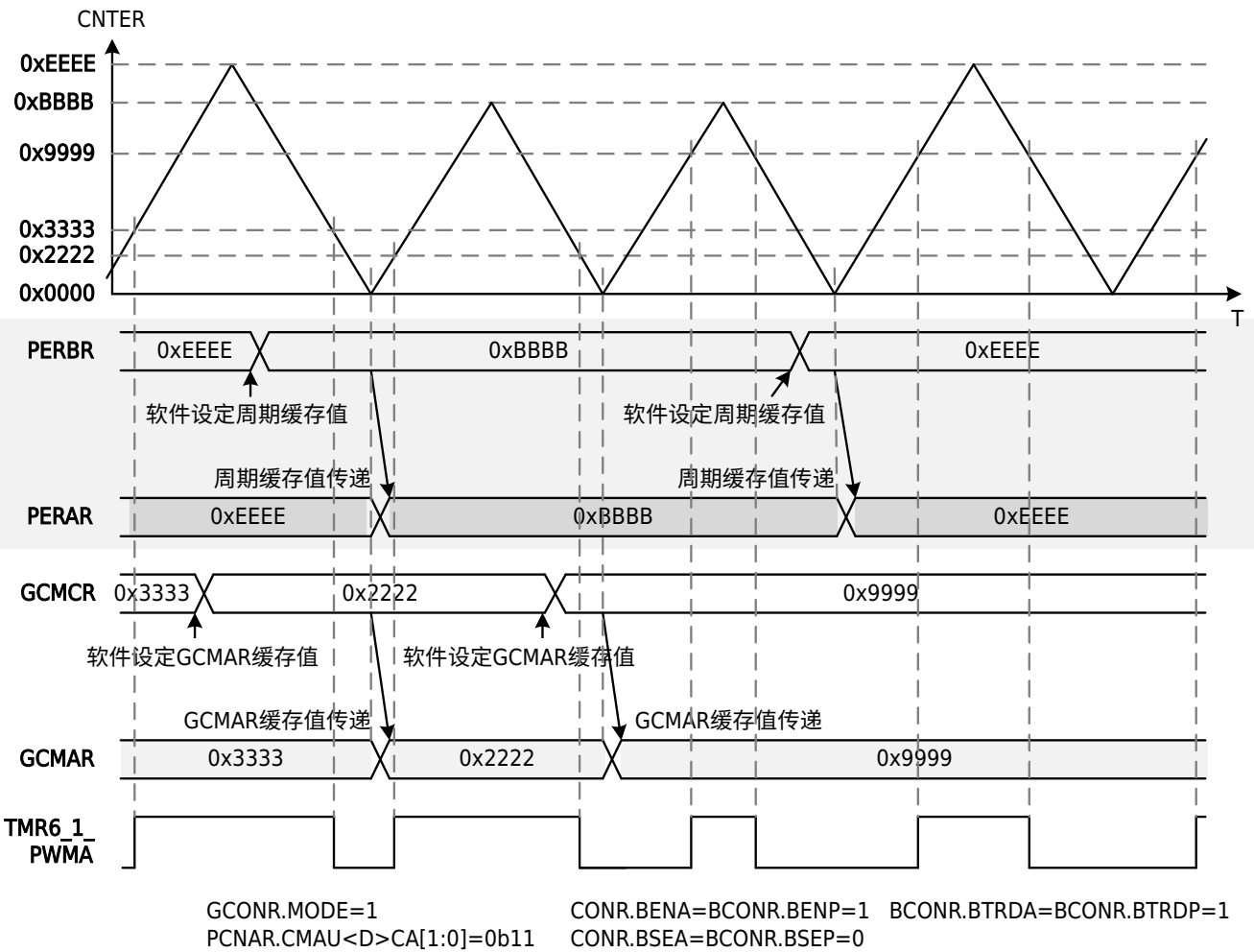
Timer6 的通用周期基准值、通用比较基准值、专用比较基准值、死区时间基准值等都具有缓存功能，可在计数期间硬件实现周期变化、占空比变化、死区变化等。周期基准值、通用比较基准值和专用比较基准值等有单缓存、双缓存功能，死区时间基准值有单缓存功能。

21.3.11.1 单缓存动作

单缓存动作是指通过设定缓存控制寄存器（BCONR.BENA<P><SPA><SPB>=1、BCONR.BSEA<P><SPA><SPB>=0）、死区控制寄存器（DCONR.DTBENU<D>=1），在缓存传送时间点，选择发生以下事件：

1. 通用周期基准值缓存寄存器（PERBR）的值自动传送到通用周期基准值寄存器（PERAR）中
2. 通用比较基准值缓存寄存器（GCMCR、GCMDBR）的值自动传送到通用比较基准值寄存器（GCMAR、GCMBR）中（比较输出时）
3. 通用比较基准值寄存器（GCMAR、GCMBR）的值自动传送到通用比较基准值缓存寄存器（GCMCR、GCMDBR）中（捕获输入时）
4. 专用比较基准值缓存寄存器（SCMCR、SCMDR）的值自动传送到专用比较基准值寄存器（SCMAR、SCMBR）中
5. 死区时间基准值缓存寄存器（DTUBR、DTDBR）的值自动传送到死区时间基准值寄存器（DTUAR、DTDAR）中

下图是单元 1 比较输出动作（PCNAR.CAPMDA=0）时、通用比较基准值寄存器的单缓存方式的时序图。从图中可以看到，在计数期间改变通用比较基准值寄存器（GCMAR）的值可以调整输出占空比，改变通用周期基准值寄存器（PERAR）的值可以调整输出周期。



21.3.11.2 双缓存动作

双缓存动作是指通过设定缓存控制寄存器（BCONR.BENA<P><SPA><SPB>=1、BCONR.BSEA<P><SPA><SPB>=1），在缓存传送时间点，选择发生以下事件：

1. 通用周期基准值缓存寄存器（PERBR）的值自动传送到通用周期基准值寄存器（PERAR）中、通用周期基准值双缓存寄存器（PERCR）的值自动传送到通用周期基准值缓存寄存器（PERBR）中
2. 通用比较基准值缓存寄存器（GCMCR、GCMDR）的值自动传送到通用比较基准值寄存器（GCMAR、GCMBR）、通用比较基准值双缓存寄存器（GCMER、GCMFR）的值自动传送到通用比较基准值缓存寄存器（GCMCR、GCMDR）中（比较输出时）
3. 通用比较基准值缓存寄存器（GCMCR、GCMDR）的值自动传送到通用比较基准值双缓存寄存器（GCMER、GCMFR）中、通用比较基准值寄存器（GCMAR、GCMBR）的值自动传送到通用比较基准值缓存寄存器（GCMCR、GCMDR）中（捕获输入时）
4. 专用比较基准值缓存寄存器（SCMCR、SCMDR）的值自动传送到专用比较基准值寄存器（SCMAR、SCMBR）中、专用比较基准值双缓存寄存器（SCMER、SCMFR）的值自动传送到专用比较基准值缓存寄存器（SCMCR、SCMDR）中

下图是内部触发事件 0 触发捕获输入时、双缓存方式的时序图。

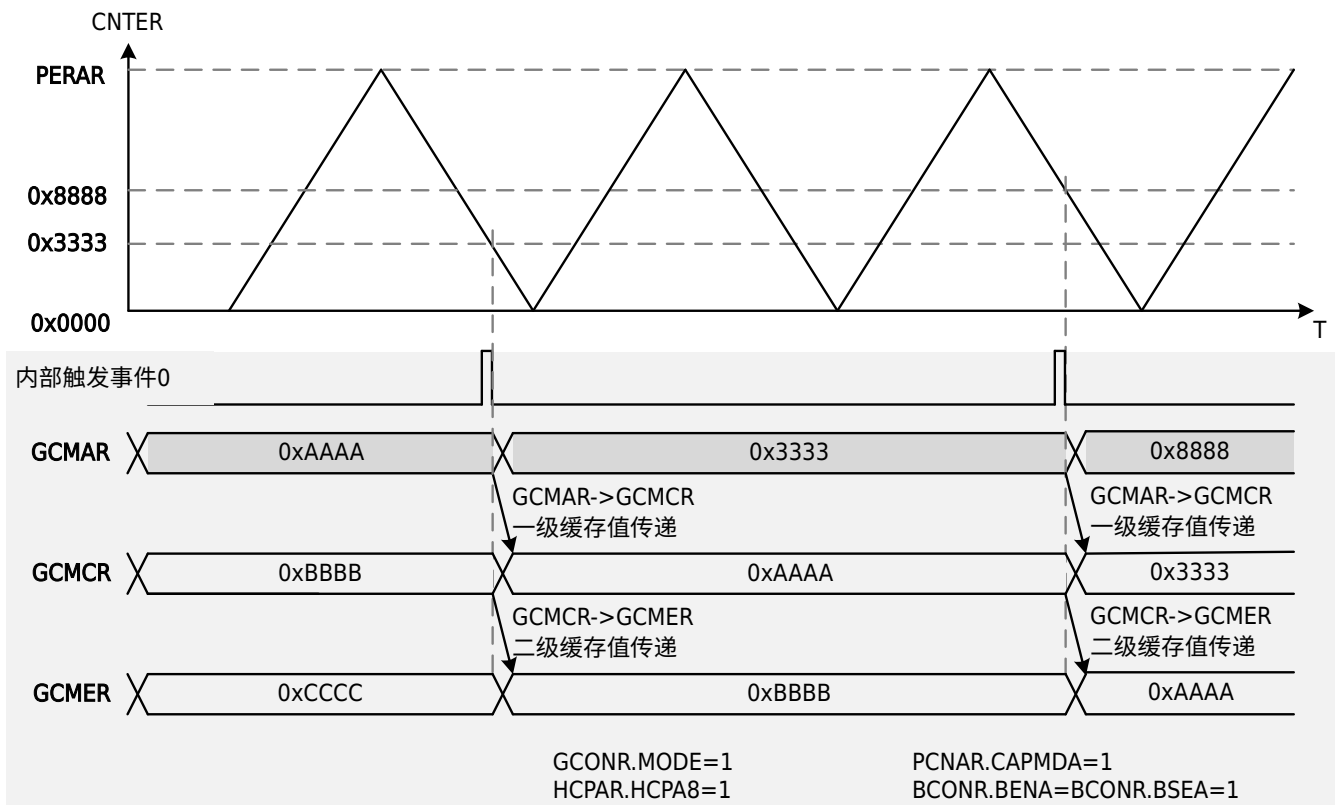


图 21-13 双缓存方式捕获输入时序

21.3.11.3 缓存传送节点

比较输出（锯齿波）缓存传送

在缓存功能有效（BCONR.BENA<P><SPA><SPB>=DCONR.DTBENU<D>=1）&&比较输出计数（PCNAR.CAPMDA=0）&&锯齿波计数模式（GCONR.MODE=0）时，通用周期基准值、通用比较基准值、专用比较基准值、死区时间基准值的缓存传送发生在递加计数上溢点或递减计数下溢点。

下图是锯齿波递加计数时的缓存动作。

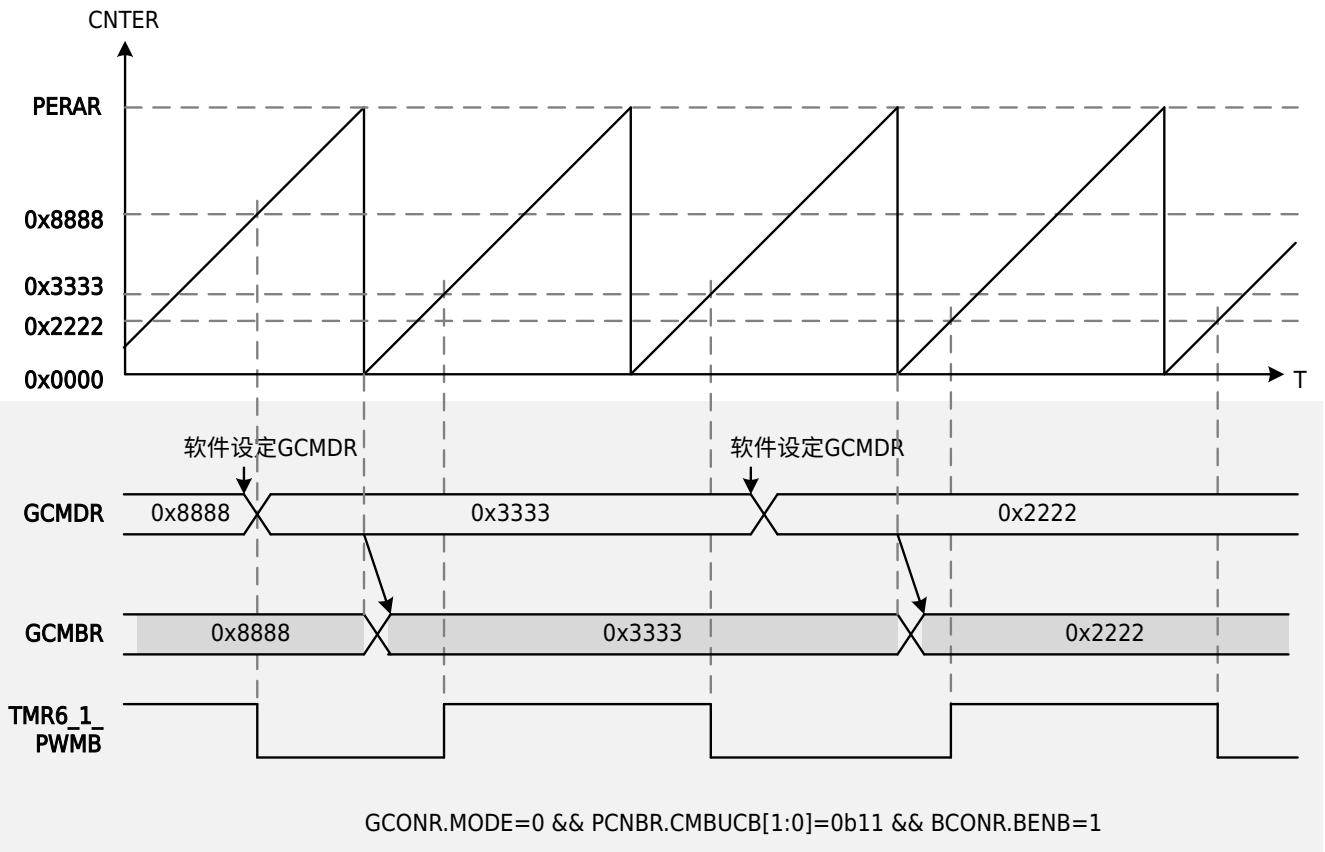


图 21-14 锯齿波模式时计数缓存动作



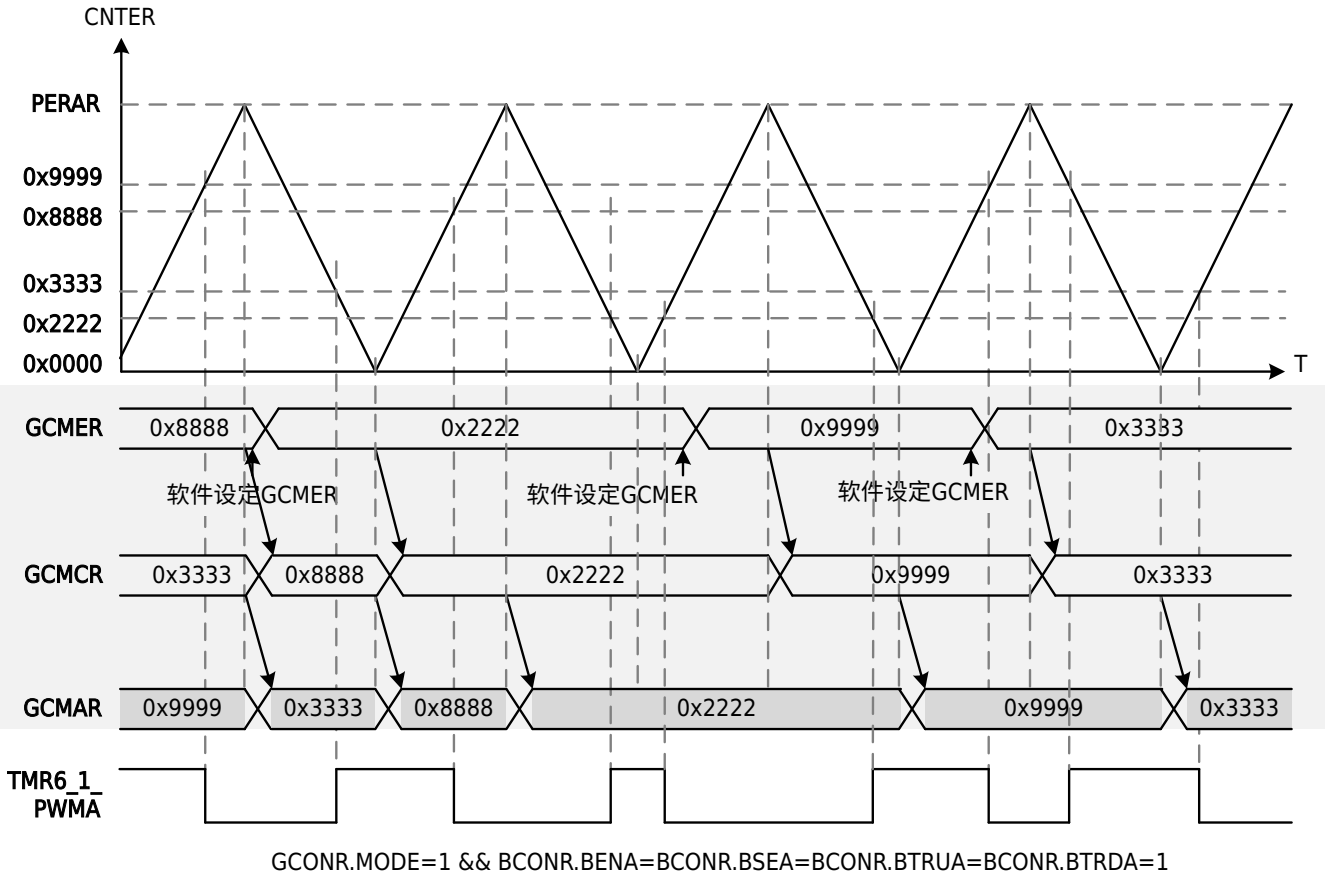
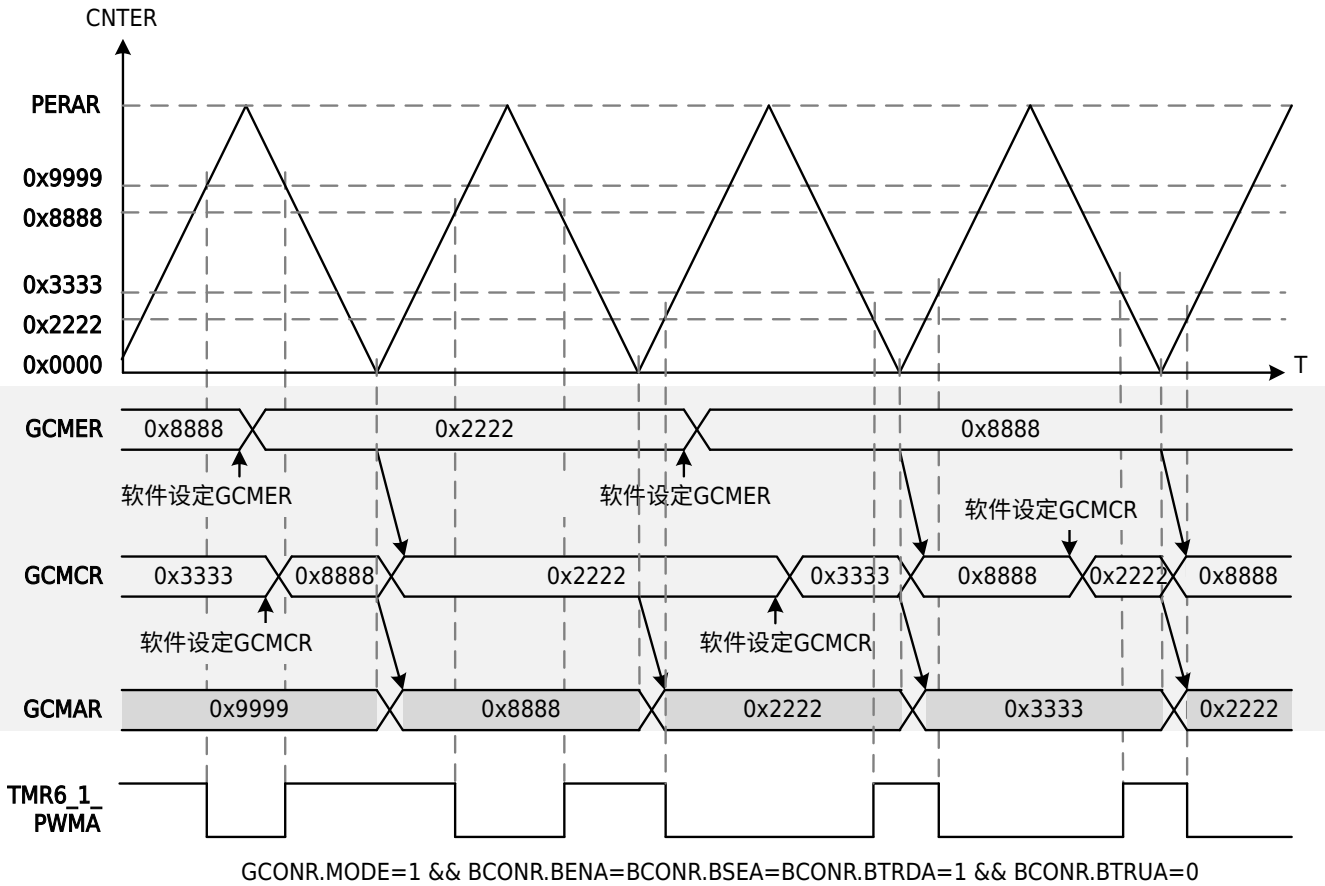
说明

- 在锯齿波计数模式时，若产生清零动作，也视为一次计数溢出。各基准值会根据相应的缓存动作设定状况（单缓存、双缓存等）发生一次缓存传送。
- 在硬件计数模式时，若产生清零动作，也视为一次计数溢出。通用周期基准值、通用比较基准值会根据相应的缓存动作设定状况（单缓存、双缓存、上溢传送、下溢传送等）发生一次缓存传送，其它基准值不发生缓存传送。

比较输出（三角波）缓存传送

在缓存功能有效（BCONR.BENA<P><SPA><SPB>=DCONR.DTBENU<D>=1）&&比较输出计数（PCNAR.CAPMDA=0）&&三角波计数模式（GCONR.MODE=1）时，通用周期基准值、通用比较基准值、专用比较基准值、死区时间基准值的缓存传送时间点由对应的寄存器控制位决定。当 BCONR.BTRUA<P><SPA><SPB>=1 或 DCONR.DTBTRU=1 时，在计数器计数到三角波的峰点时，发生对应的缓存传送；当 BCONR.BTRDA<P><SPA><SPB>=1 或 DCONR.DTBTRD=1 时，在计数器计数到三角波的谷点时，发生对应的缓存传送。

图 21-15 是三角波计数到谷点时的缓存动作，图 21-16 是三角波计数到峰点和谷点时均发生缓存动作的示意图。



捕获输入缓存传送

在捕获输入动作 (PCNAR.CAPMDA=1) 有效时, 通用比较基准值支持缓存功能。若缓存设定有效 (BCONR.BENA=1), 在捕获输入动作点会发生一次缓存传送。捕获输入动作可选择单缓存功能或双缓存功能 (由 BCONR.BSEA设定)。

21.3.12 通用 PWM 输出

21.3.12.1 单边对齐独立 PWM 输出

在锯齿波计数模式 (GCONR.MODE=0) 时, 每个单元的 2 个端口 TMR6_<t>_PWMA、TMR6_<t>_PWMB 都能独立的输出 PWM 波。在计数溢出点若设定同样的电平变化, 则可实现单边对齐独立 PWM 输出。

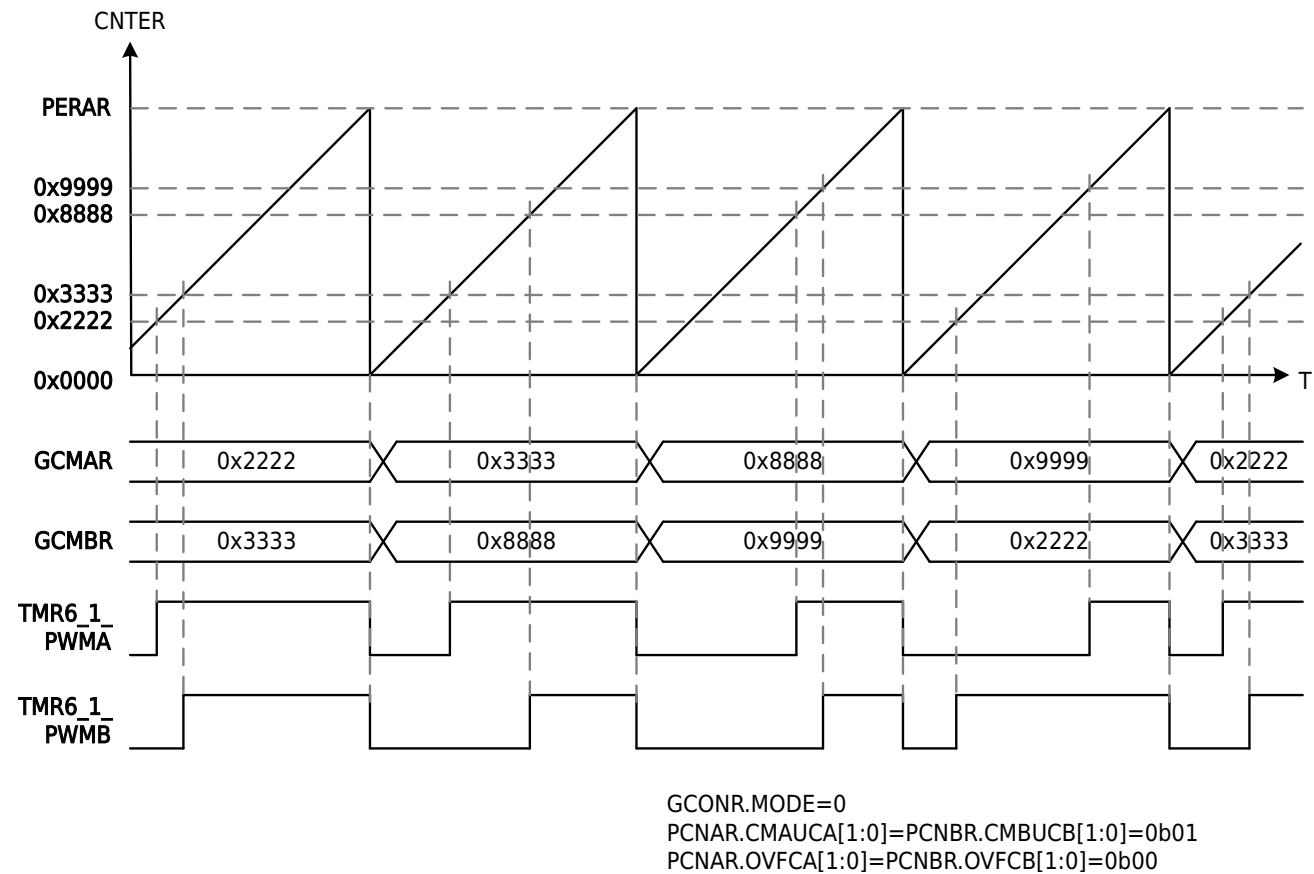


图 21-17 单边对齐独立 PWM

21.3.12.2 双边对称独立 PWM 输出

在三角波计数模式 (GCONR.MODE=1) 时, 每个单元的 2 个端口 TMR6_<t>_PWMA、TMR6_<t>_PWMB 都能独立的输出 PWM 波。在计数比较匹配点若设定电平变化、在计数峰点和谷点设定电平不变化, 则可实现双边对称独立 PWM 输出。下图是 TMR6_<t>_PWMA 端口和 TMR6_<t>_PWMB 端口实现双边对称独立输出 PWM。

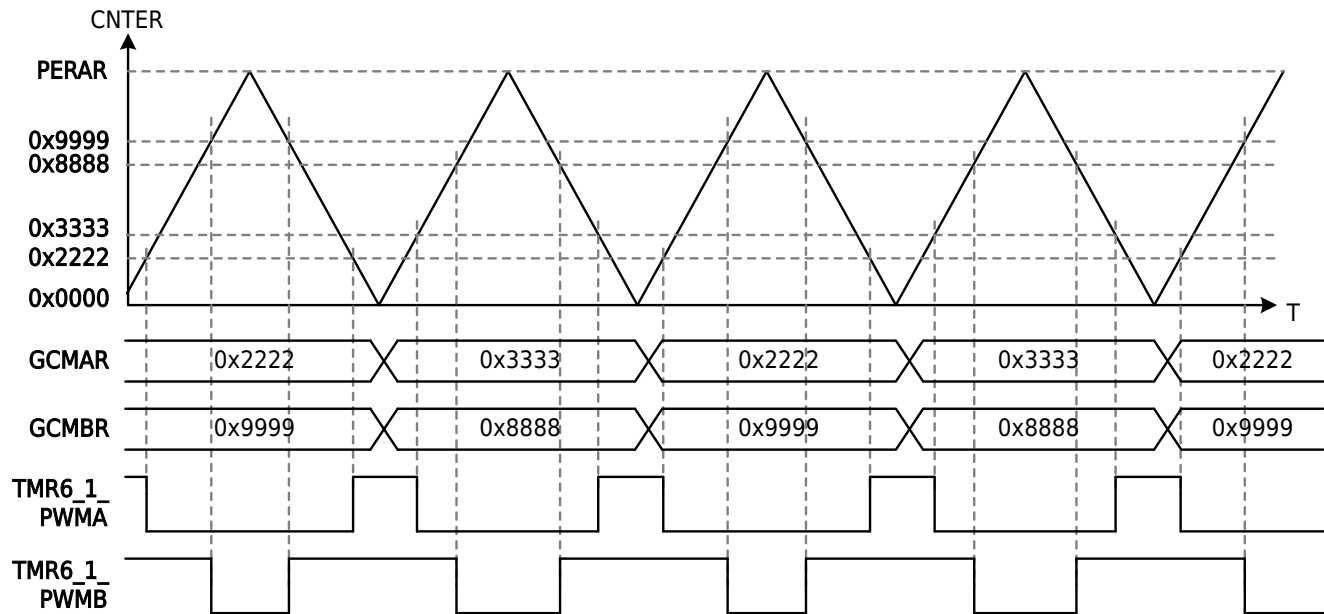


图 21-18 双边对称独立 PWM

21.3.12.3 双边对称互补 PWM 输出

在三角波计数模式 (GCONR.MODE=1) 时, 设定好端口在计数启动、比较匹配、计数溢出时的电平状态, 可实现 TMR6_<t>_PWMA 和 TMR6_<t>_PWMB 端口上输出一对互补 PWM 波形。

根据 GCMBR 基准值的赋值方式不同, 双边对称互补 PWM 的输出可分为“软件设定 GCMBR 互补 PWM 输出”和“硬件设定 GCMBR 互补 PWM 输出”。

软件设定 GCMBR 互补 PWM 输出

软件设定是指在三角波模式下, 用于 TMR6_<t>_PWMB 端口波形输出的通用比较基准值寄存器 (GCMBR) 直接由 CPU 等写入, 与 GCMAR 的值没有直接关系。

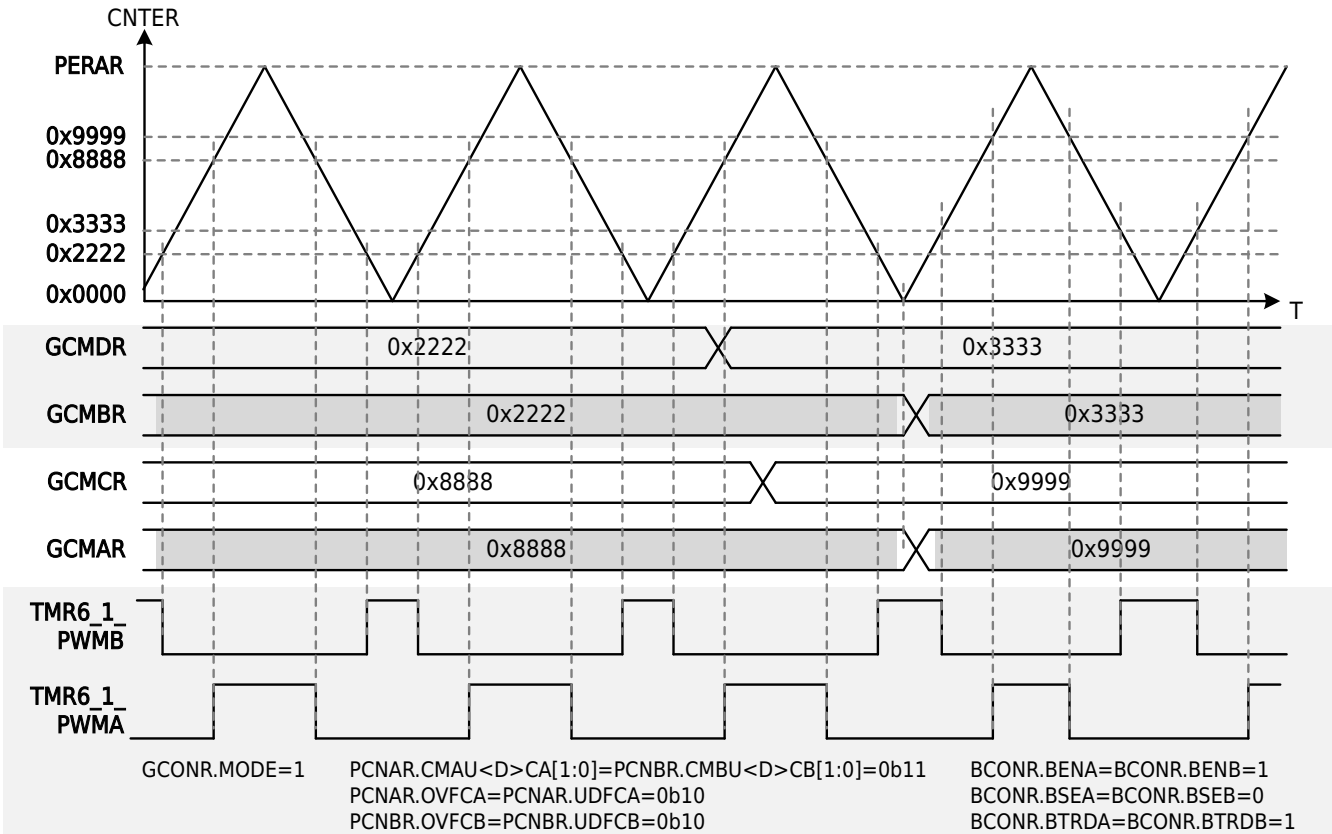


图 21-19 软件设定 GCMBR 互补 PWM 波输出

硬件设定 GCMBR 互补 PWM 输出

硬件设定是指在三角波模式下，用于 TMR6_<t>_PWMB 端口波形输出的通用比较基准值寄存器（GCMBR）的值由通用比较基准值寄存器（GCMAR）和死区时间基准值寄存器（DTU<D>AR）的值运算决定。

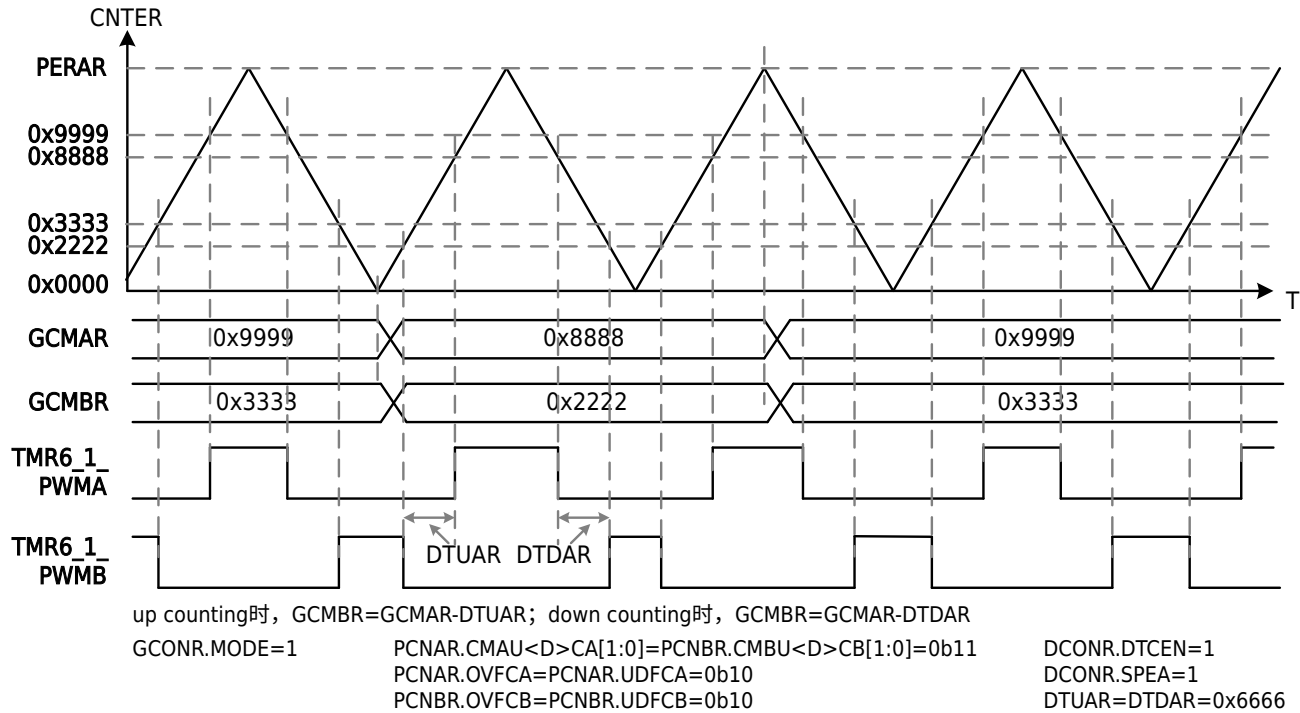


图 21-20 硬件设定 GCMBR 互补 PWM 波输出

21.3.12.4 双边非对称 PWM 输出

在三角波计数模式（GCONR.MODE=1）时，每个单元的 2 个端口 TMR6_<t>_PWMA、TMR6_<t>_PWMB 都能独立的输出 PWM 波，且每个端口都能根据通用比较基准值寄存器（GCMAR、GCMBR）的基准进行对应的电平输出变化。若在计数器的向上计数期间和向下计数期间，TMR6_<t>_PWMA 的端口电平变化分别由 GCMAR、GCMBR 基准值的比较结果控制，则在 TMR6_<t>_PWMA 端口上可实现双边非对称 PWM 输出。下图是 TMR6_<t>_PWMA 端口的双边非对称 PWM 输出。

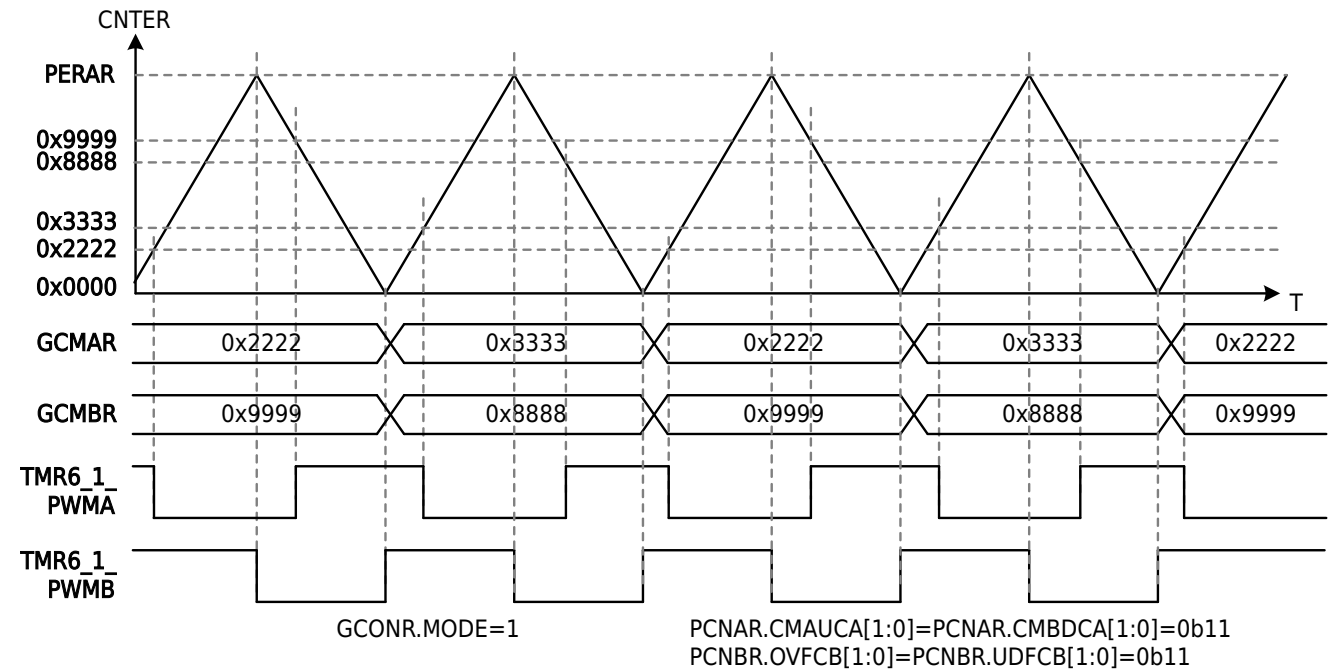


图 21-21 双边非对称 PWM 输出

21.3.12.5 单元间多相 PWM 输出

每个单元的 TMR6_<t>_PWMA、TMR6_<t>_PWMB 端口都能输出 2 相 PWM 波，多个单元间组合，同时结合软件、硬件同步动作就可实现多相 PWM 波输出。图 21-22 是单元 1、单元 2 组合输出单边对齐独立 PWM；图 21-23 是单元 1、单元 2 组合输出双边对称互补 PWM。

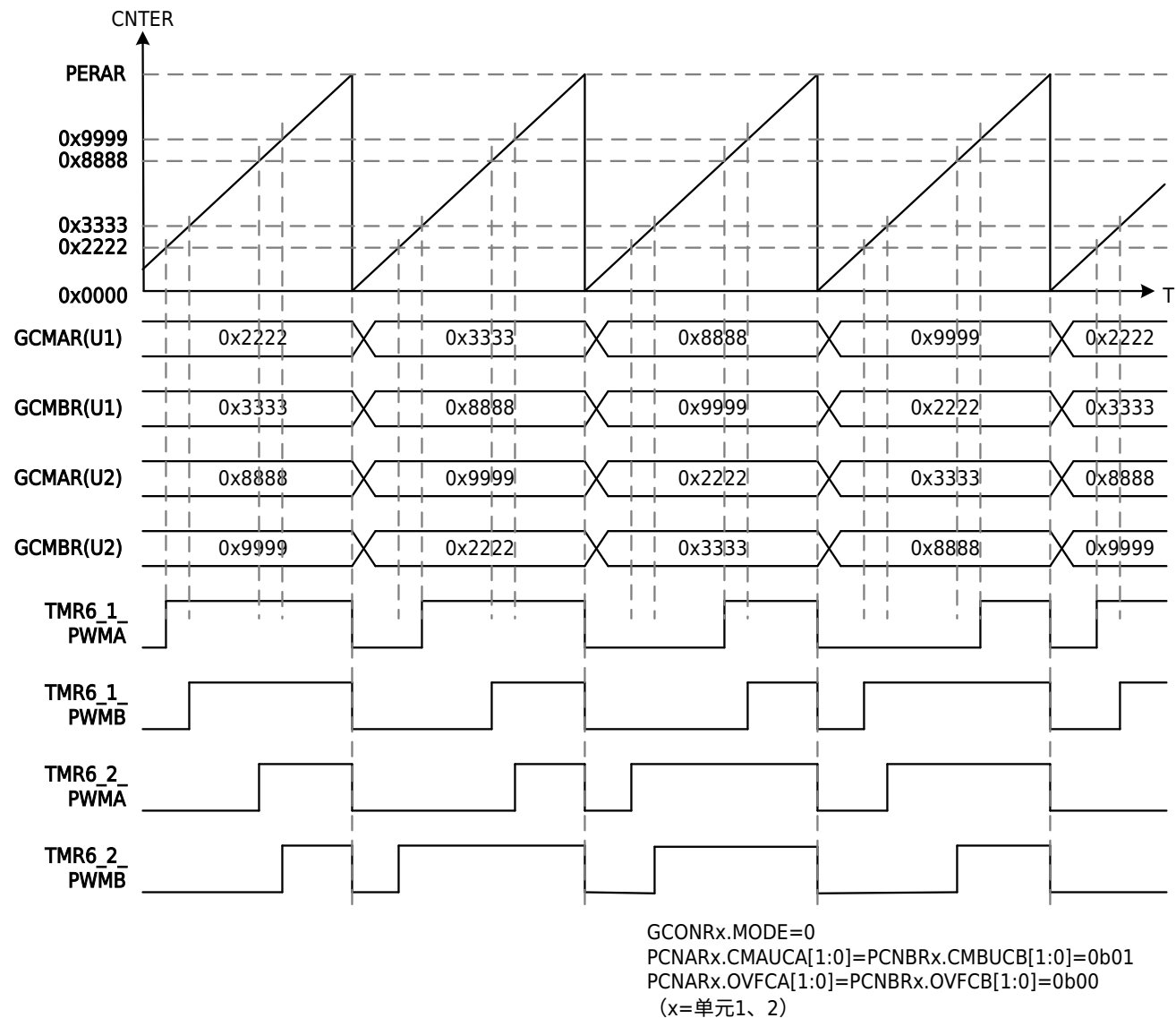


图 21-22 单边对齐独立 PWM

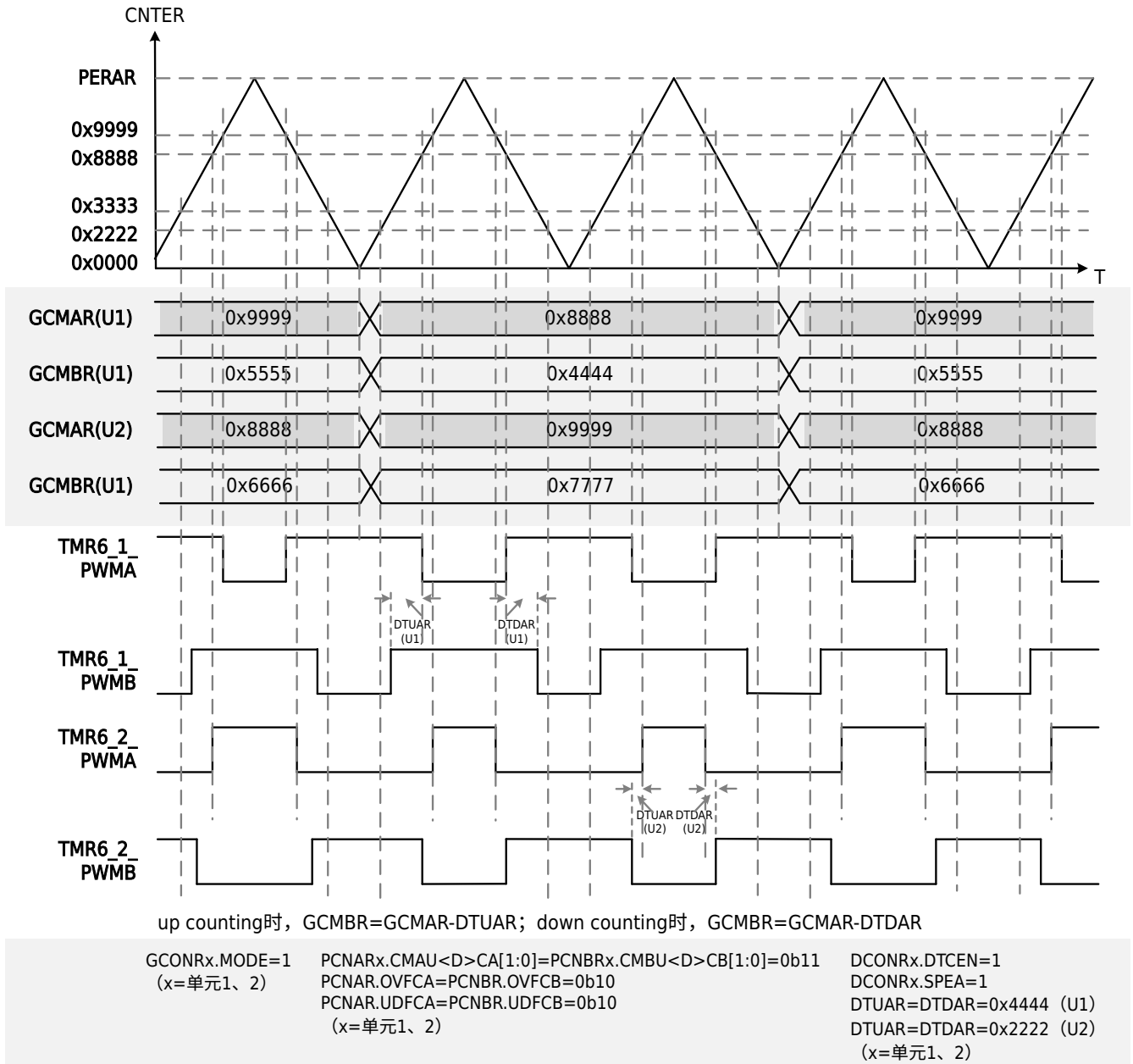


图 21-23 带死区时间双边对称互补 PWM

21.3.13 正交编码计数

将 TMR6_<t>_PWMA 输入看作 AIN 输入、TMR6_<t>_PWMB 输入看作 BIN 输入、TMR6_TRIGA~D 中的任意一个输入看作 ZIN 输入，Timer6 就可以实现三路输入的正交编码计数。

一个单元的 AIN、BIN 单独动作可以实现位置计数模式；两个单元的 AIN、BIN、ZIN 组合动作可以实现公转计数模式，一个单元用于位置计数，一个单元用于公转计数。

公转计数模式时，Timer6 的 8 个单元可任意组合选择做位置计数单元或公转计数单元，但在 Z 相屏蔽功能有效时组合固定，单元 1、2 组合，单元 1 作为位置计数单元，单元 2 作为公转计数单元，分别实现位置计数和公转计数；单元 3、4 组合，单元 3 作为位置计数单元，单元 4 作为公转计数单元，分别实现位置计数和公转计数；单元 5、6 组合，单元 5 作为位置计数单元，单元 6 作为公转计数单元，分别实现位置计数和公转计数；单元 7、8 组合，单元 7 作为位置计数单元，单元 8 作为公转计数单元，分别实现位置计数和公转计数。

AIN 和 BIN 的计数条件是通过设定硬件递加事件选择寄存器（HCUPR）和硬件递减事件选择寄存器（HCDOR）中 TMR6_<t>_PWMA 和 TMR6_<t>_PWMB 的正交关系实现；ZIN 的输入动作通过设定位置单元

的硬件清零事件选择寄存器 (HCLRR) 实现位置计数单元的位置定时器清零、通过设定公转单元的硬件递加事件选择寄存器 (HCUPR) 实现公转计数单元的公转定时器计数。

21.3.13.1 位置计数模式

正交编码位置计数模式，是指根据 AIN、BIN 的输入实现基本计数功能、相位差计数功能和方向计数功能。

基本计数

基本计数动作是根据 AIN 或 BIN 端口的输入时钟进行计数。

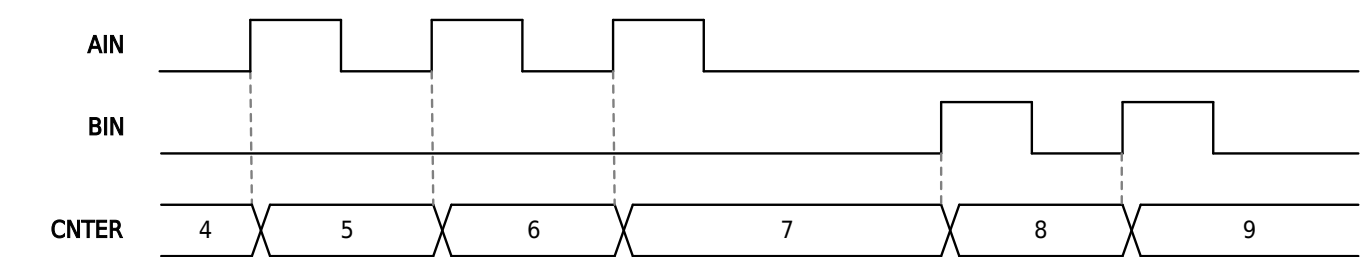
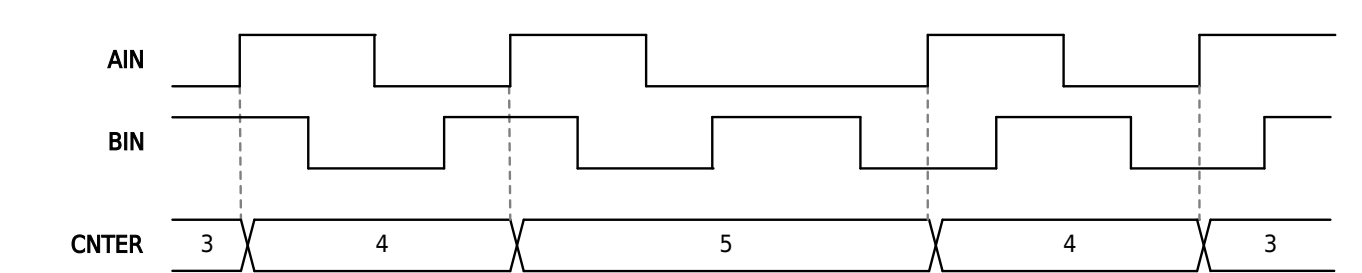


图 21-24 位置模式-基本计数

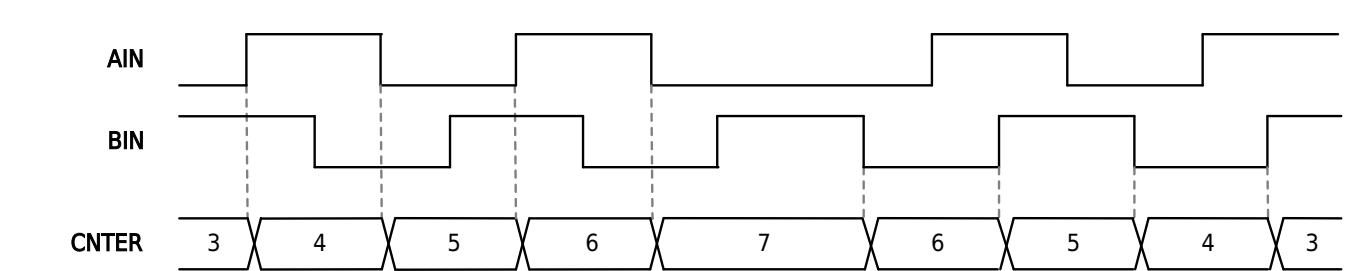
相位差计数

相位差计数是指根据 AIN 和 BIN 的相位关系进行计数。根据设定的不同，可以实现 1 倍计数、2 倍计数、4 倍计数等。



HCUPR.HCUP6=0b1 && HCDOR.HCDO4=0b1

图 21-25 位置计数模式-相位差计数 (1 倍计数)



HCUPR.HCUP6=HCUPR.HCUP5=0b1 && HCDOR.HCDO2=HCDOR.HCDO1=0b1

图 21-26 位置计数模式-相位差计数 (2 倍计数)

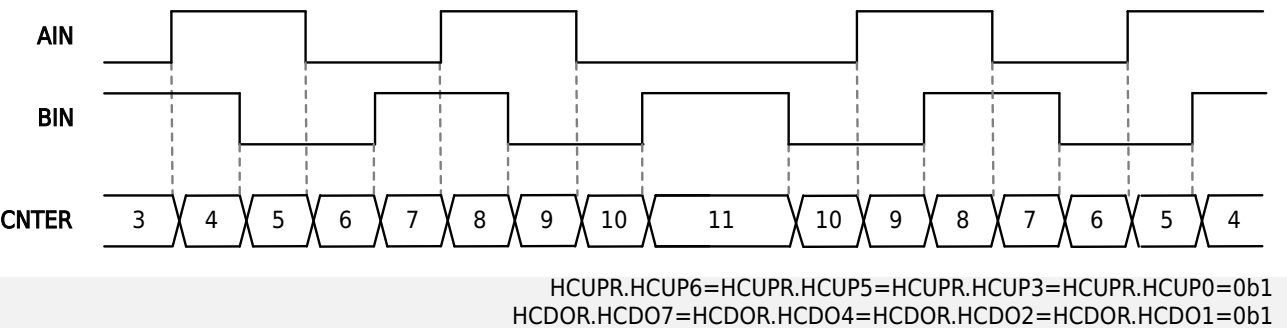


图 21-27 位置计数模式-相位差计数（4 倍计数）

方向计数

方向计数是指将 AIN 的输入状态设定为方向控制，将 BIN 的输入作为时钟计数。

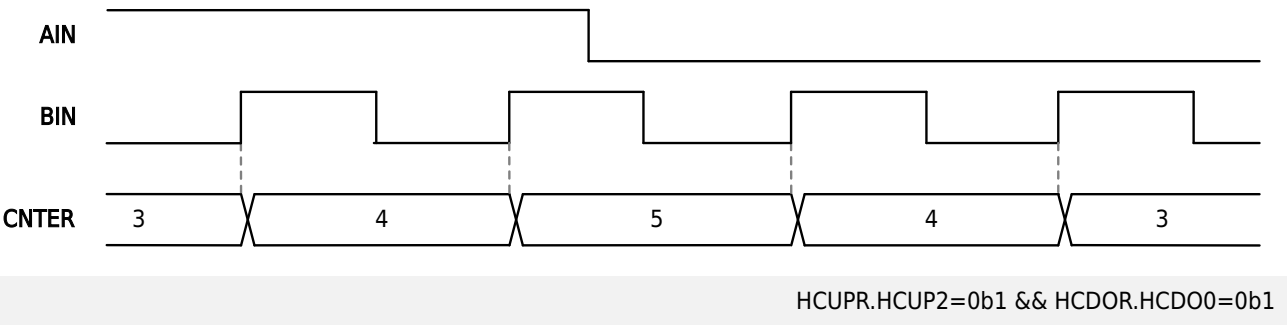


图 21-28 位置计数模式-方向计数

21.3.13.2 公转计数模式

正交编码公转计数模式，是指在 AIN、BIN 计数的基础上，加入 ZIN 的输入事件以实现对公转圈数等的判断。公转计数模式时根据公转定时器的计数方式，可实现 Z 相计数功能、位置溢出计数功能和混合计数功能。

Z 相计数

Z 相计数是指根据 ZIN 的输入，公转计数单元进行计数，同时将位置计数单元清零的计数动作。

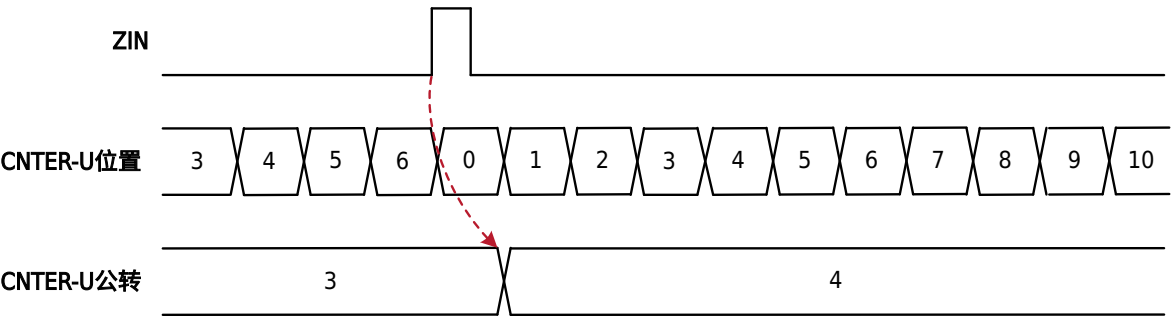


图 21-29 公转计数模式-Z 相计数

位置溢出计数

位置溢出计数是指位置计数单元计数发生上溢或下溢时，产生一个溢出事件，从而触发公转计数单元的计数器进行一次计数（在该计数方式时 ZIN 的输入不进行公转计数单元的计数动作和位置计数单元的清零动作）。

位置计数单元的溢出事件通过内部触发事件接口选通实现公转计数单元计数，即可实现位置溢出计数。公转计数单元的硬件递加（递减）事件选择寄存器（HCUPR 或 HCDOR）的递加（递减）事件选择 Bit8~Bit11 中

的 1 位，同时将对应的硬件触发事件选择寄存器 (TMR6_TRGSELx, x=0~3) 中的事件编号设定为位置计数单元的上溢或下溢事件。具体的寄存器内容详见《自动运行系统 (AOS) 》章节“Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3) ”，具体的事件对应关系请参考《中断控制器 (INTC) 》章节。

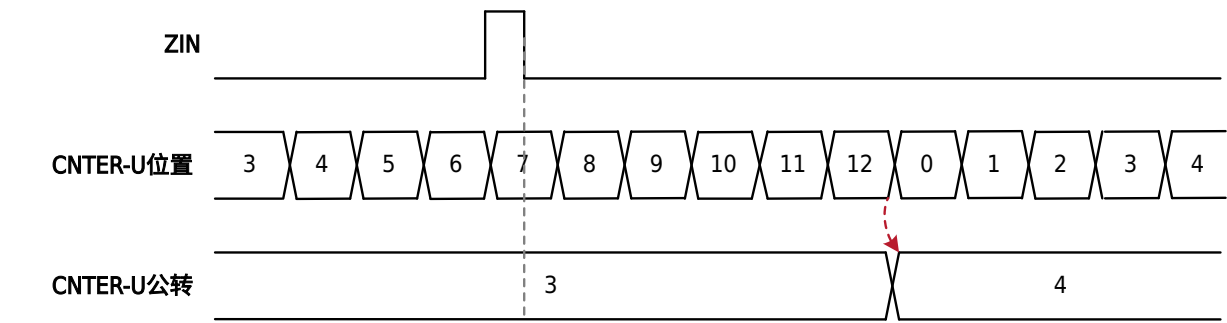


图 21-30 公转计数模式-位置溢出计数

混合计数

混合计数是指上述 Z 相计数和位置溢出计数两种计数方式合并起来的计数动作，其实现方式也是上述两种计数方式的组合。

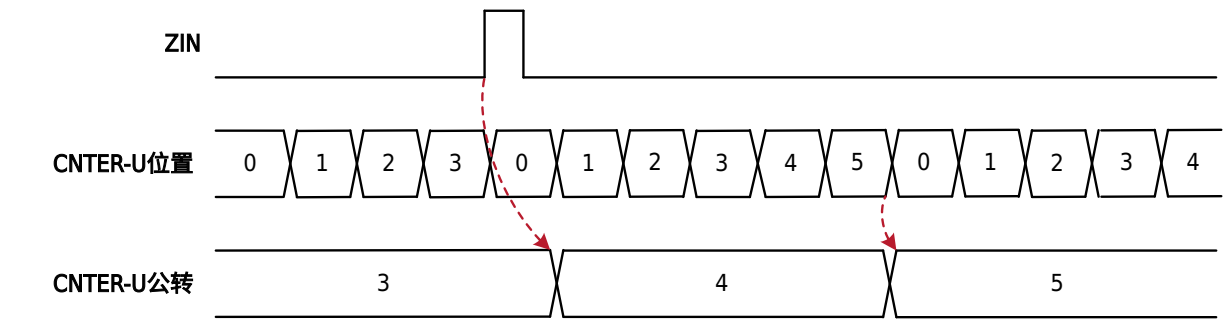


图 21-31 公转计数模式-混合计数

21.3.13.3 Z 相动作屏蔽

在公转计数模式的 Z 相计数功能或混合计数功能时，可以设定在位置定时器的上溢点或下溢点后的几个周期内 (GCONR.ZMSKVAL[0:1]设定)，将 ZIN 的有效输入屏蔽，不进行公转计数单元的计数和位置计数单元的清零。

位置计数单元的通用控制寄存器 (GCONR) 的 GCONR.ZMSKPOS 为 1 时，位置计数单元的 Z 相屏蔽功能使能，Z 相屏蔽的周期数由 GCONR.ZMSKVAL 设定；公转计数单元的通用控制寄存器 (GCONR) 的 GCONR.ZMSKREV 为 1 时，公转计数单元的 Z 相屏蔽功能使能。

图 20-33 是公转计数模式混合计数时，在位置计数单元计数上溢后的 4 个计数周期内有 ZIN 相输入时，ZIN 相输入的动作无效，即公转计数单元不计数、位置计数单元不清零；之后再来的 ZIN 相输入正常动作。

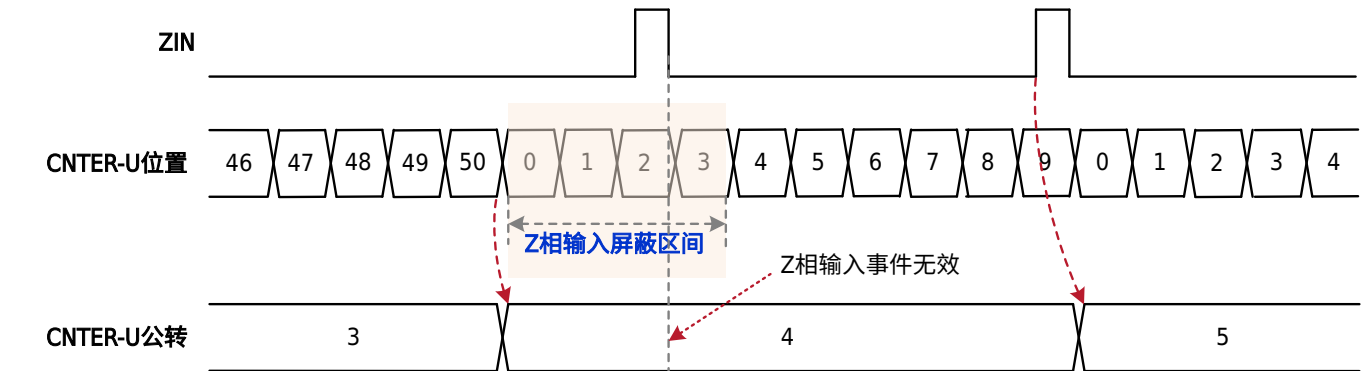


图 21-32 公转计数模式-混合计数 Z 相屏蔽动作例 1

下图是公转计数模式混合计数时，在位置计数单元计数上溢后的第 3 个周期，计数方向发生变化，此时设定的 4 个周期的屏蔽周期变为无效（实际 ZIN 相屏蔽功能维持了 3 个周期），开始向下计数。在位置计数单元发生计数下溢后，ZIN 相屏蔽功能重新开启，维持 4 个周期后变为无效。在 ZIN 相屏蔽期间，ZIN 相的输入功能无效，即公转计数单元不计数、位置计数单元不清零；之后再来的 ZIN 相输入正常动作。

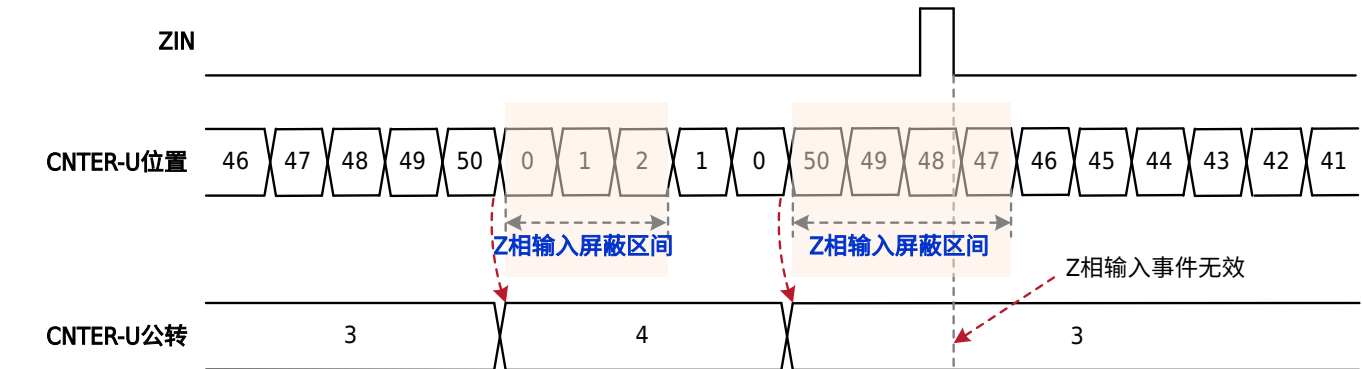


图 21-33 公转计数模式-混合计数 Z 相屏蔽动作例 2

21.3.14 周期间隔响应

Timer6 的 2 个专用比较基准值寄存器（SCMAR、SCMBR），在计数比较匹配时可分别输出专用比较匹配中断 A 信号、专用比较匹配中断 B 信号到 INTC 产生对应的中断；同时可分别输出专用比较匹配事件 A 信号、专用比较匹配事件 B 信号，用于和其他模块关联动作，多用于启动 ADC 等。

该中断和事件的请求信号可以每间隔几个周期后产生一次有效的请求信号，即实现周期间隔响应。该功能通过设定有效周期寄存器（VPERR）的 VPERR.PCNT[1:0]位和 VPERR.SPPERIA/B 位使能。设定 VPERR.PCNTS[2:0]位来指定每隔多少个周期请求信号有效一次，其他周期内即使计数值和专用比较基准值寄存器 SCMAR 或 SCMBR 的值相等，也不会输出有效的请求信号。

该功能有效后，各波形模式下的周期匹配中断和周期匹配事件也只在专用比较匹配中断和事件输出的有效周期里（下图中 STFLR.VPERNUM=0 的周期）输出。下图是周期间隔有效请求信号的动作例。

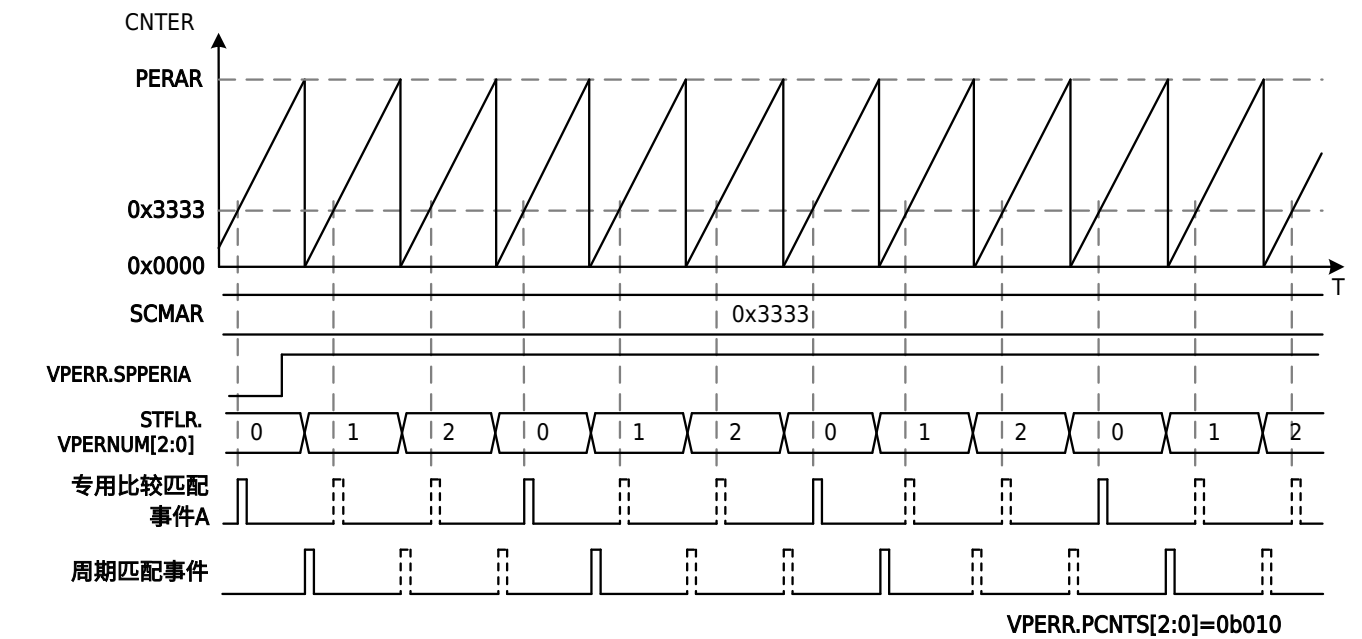


图 21-34 周期间隔有效请求信号动作

21.3.15 EMB 控制

Timer6 可以对端口的输出状态进行保护控制，在发生异常时将端口状态固定成预先设定好的安全状态。所有单元具有 4 个共用的端口输出控制接口，每个单元通过端口控制寄存器 (PCNAR.EMBR) 的设定选择要使用的 EMB 事件连接接口，这个接口连接 EMB 模块输出的 1 组 EMB 事件。同时接口上选通的异常状况事件可从 EMB 侧设定（参见 EMB 章节），当这些接口上监测到异常状况时，可以实现对通用 PWM 输出的控制。

端口在正常输出期间，若监测到从 EMB 过来的 EMB 事件，则端口的输出状态可变为预先设定好的状态。通用 PWM 输出端口在 EMB 异常事件发生时，端口状态可以变为输出高阻态、输出低电平或输出高电平（根据 PCNAR.EMBCAR 的设定决定）。例如，若 PCNAR.EMBCA=0b01 设定时，则在 TMR6_<t>_PWMA 端口正常输出期间，若产生 EMB 事件，则 TMR6_<t>_PWMA 端口上输出变为高阻态。

在 EMB 事件无效后（从 EMB 模块连接到 Timer6 的异常事件消失，信号变为正常电平），PWM 端口的输出可以自动恢复到正常的输出。此时，可以通过端口控制寄存器的设定选择立即恢复 PWM 正常输出

（PCNAR.EMBR=0b00；这种方式称之为 One Shot 方式释放）或在计数到下一个溢出点之后再恢复 PWM 的正常输出（PCNAR.EMBR=0b01、0b10、0b11；这种方式称之为 Cycle By Cycle 方式释放）。

21.3.16 PTO 功能

将 TMR6_<t>_PWMA、TMR6_<t>_PWMB 作为 AB 相输出、TMR6_TRIGA~D 中的任意一个看作 Z 相输出，Timer6 就可以实现 PTO ABZ 三相输出。

PTO 算法周期通过 TMR6_PERAR 来设置，通过配置 GCMAR/GCMBR 设定下周期 TMR6_PTNUMR 和 TMR6_PTPERR 写入时间点，ICONR.INTENA/INTENB 为 1，设置计数模式为锯齿波模式 GCONR.MODE 为 0，且计数方向为向上 GCMONR.DIR 为 1。TMR6_PTCTLR.PTOENA 控制 TMR6 进入 PTO 模式。

TMR6_PTCTLR.AOUTENA、TMR6_PTCTLR.BOUTENA、TMR6_PTCTLR.ZOUTENA 3 个使能开关为控制 ABZ 相的输出。

21.3.16.1 AB 相输出

AB 相的输出相位差与当算法周期脉冲个数 TMR6_PTNUMR.Bit31 相关。

正转

TMR6_PTNUMR[31]为 0 表示正转，AB 分别相按照 PTAOPR[3:0]和 PTBOPR[3:0]的 Bit0->Bit1->Bit2->Bit3->Bit0->Bit1...循环输出。

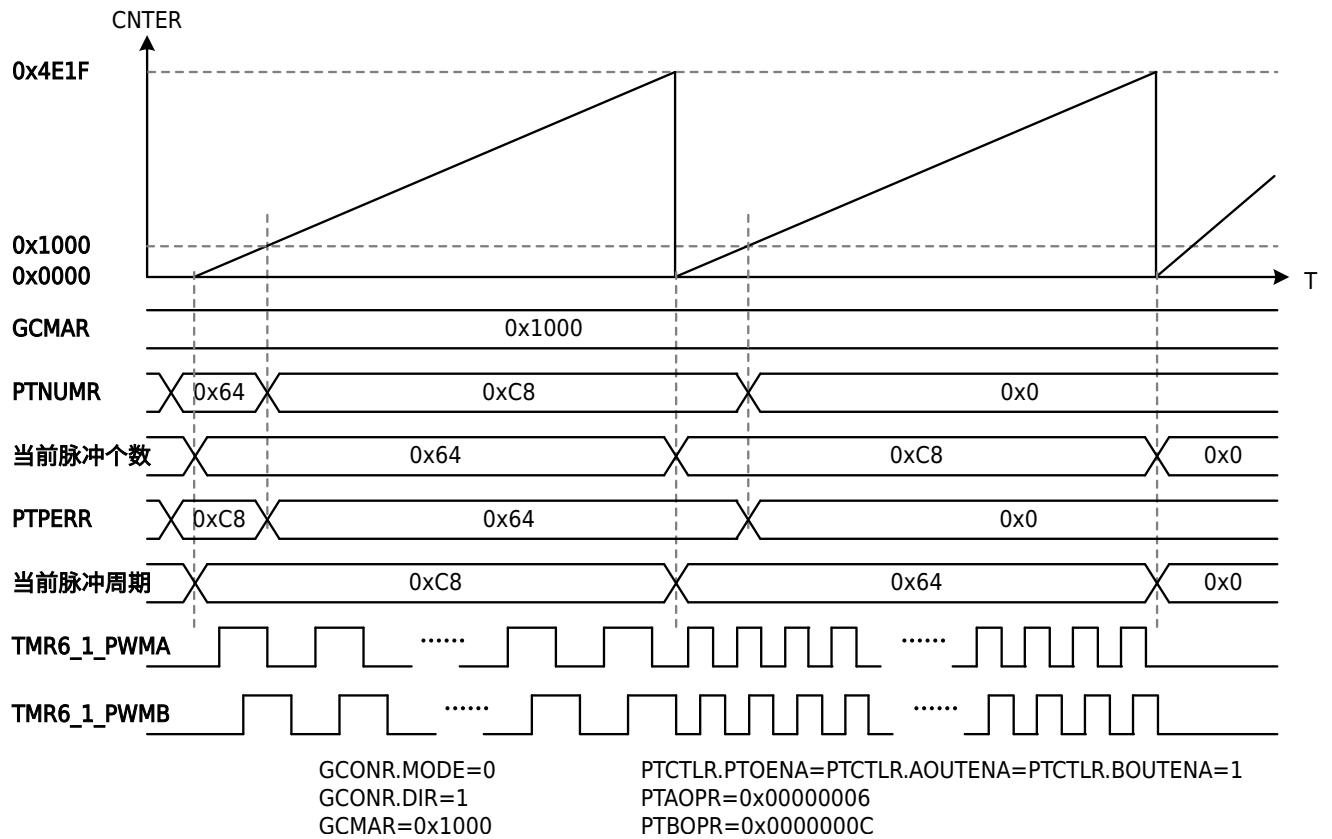


图 21-35 正转 AB 输出时序

反转

TMR6_PTNUMR[31]为 1 表示反转，AB 相分别按照 PTAOPR[3:0]和 PTBOPR[3:0]的 Bit3->Bit2->Bit1->Bit0->Bit3->Bit2...循环输出。

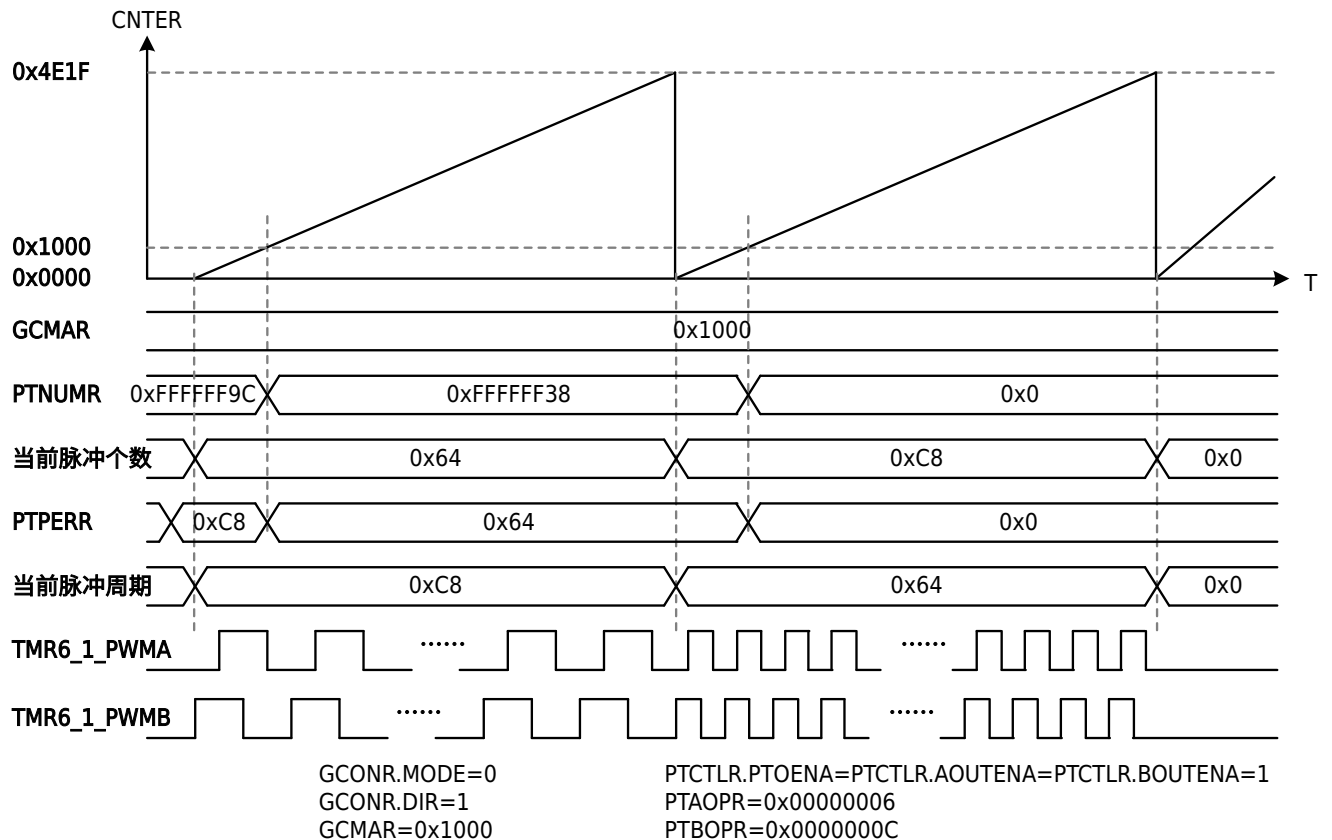


图 21-36 反转 AB 输出时序

正转切反转

当前算法周期结束若下周期旋转方向反向，正转切反转时 AB 相分别按照 PTAOPR[3:0]和 PTBOPR[3:0]的 Bit0->Bit1->Bit2->Bit3->Bit3->Bit2->Bit1->Bit0...循环输出，同理反转切正转时 AB 相分别按照 PTAOPR[3:0]和 PTBOPR[3:0]的 Bit3->Bit2->Bit1->Bit0->Bit0->Bit1->Bit2->Bit3...循环输出。

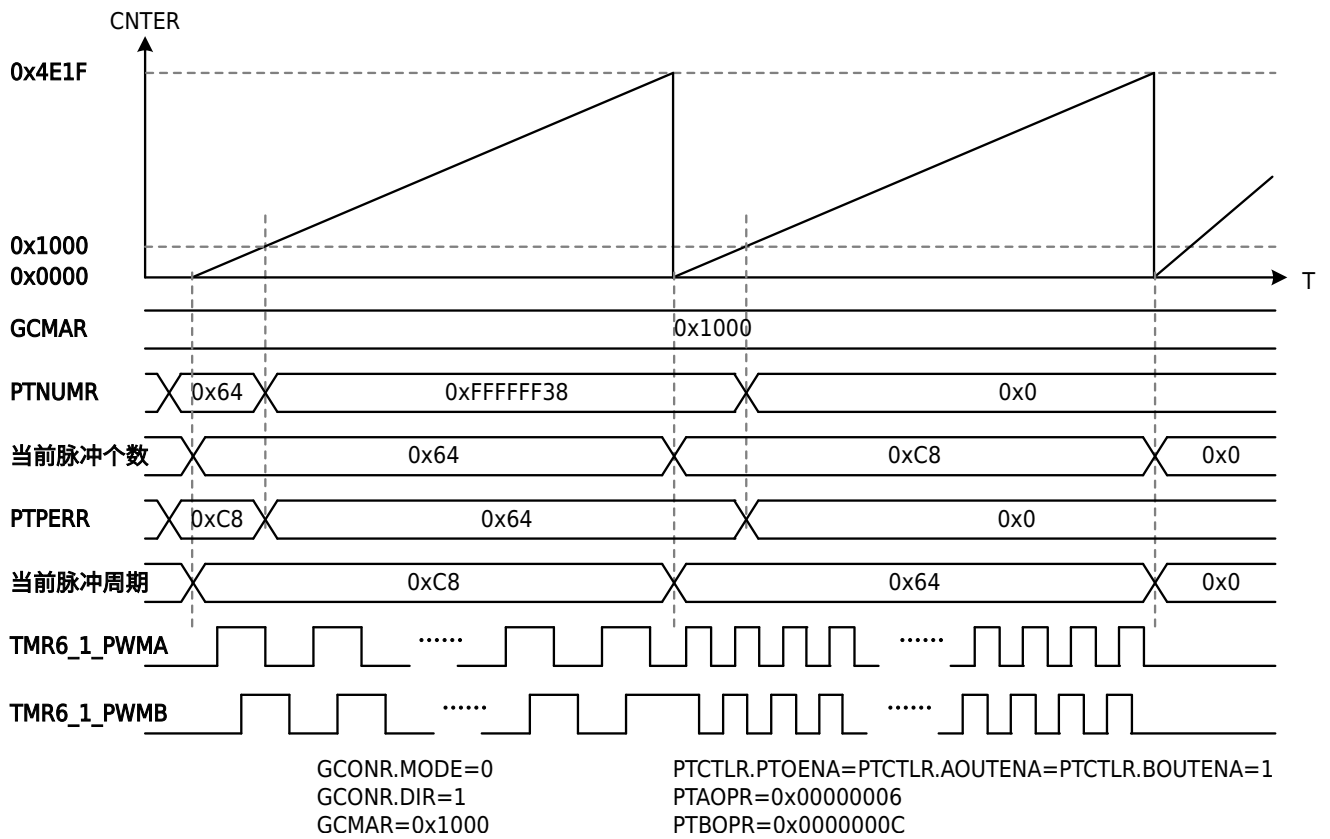


图 21-37 正转切反转

21.3.16.2 Z 相输出

Z 相输出 (无压缩)

TMR6_PTZPER 寄存器设定整圈脉冲个数，位置计数器达到 TMR6_PTZPER-1 时下个时钟周期自动归零。Z 相的输出位置相关，正转时位置计数器达到 TMR6_PTZOPSR 时 Z 输出高，达到 TMR6_PTZOPER 时 Z 输出低，如下图所示。

理论上 TMR6_PTZOPSR、TMR6_PTZOPER 要小于 TMR6_PTZPER，否则 Z 相将无输出。

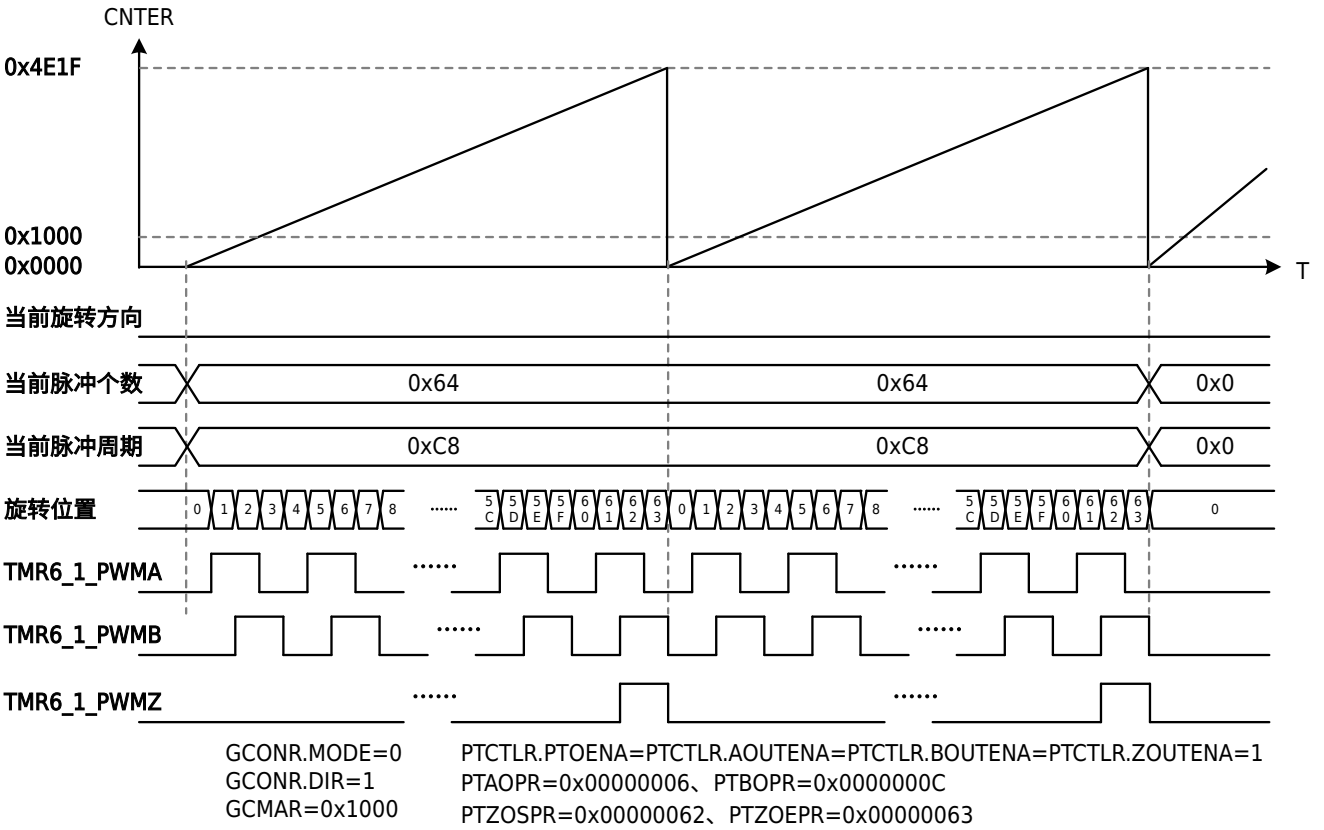


图 21-38 正转 Z 输出 (PTZOSPR<PTZOEPR)

反转时位置计数器达到 TMR6_PTZOPER 时 Z 输出高，达到 TMR6_PTZOPSR 时 Z 输出低，如下图所示。

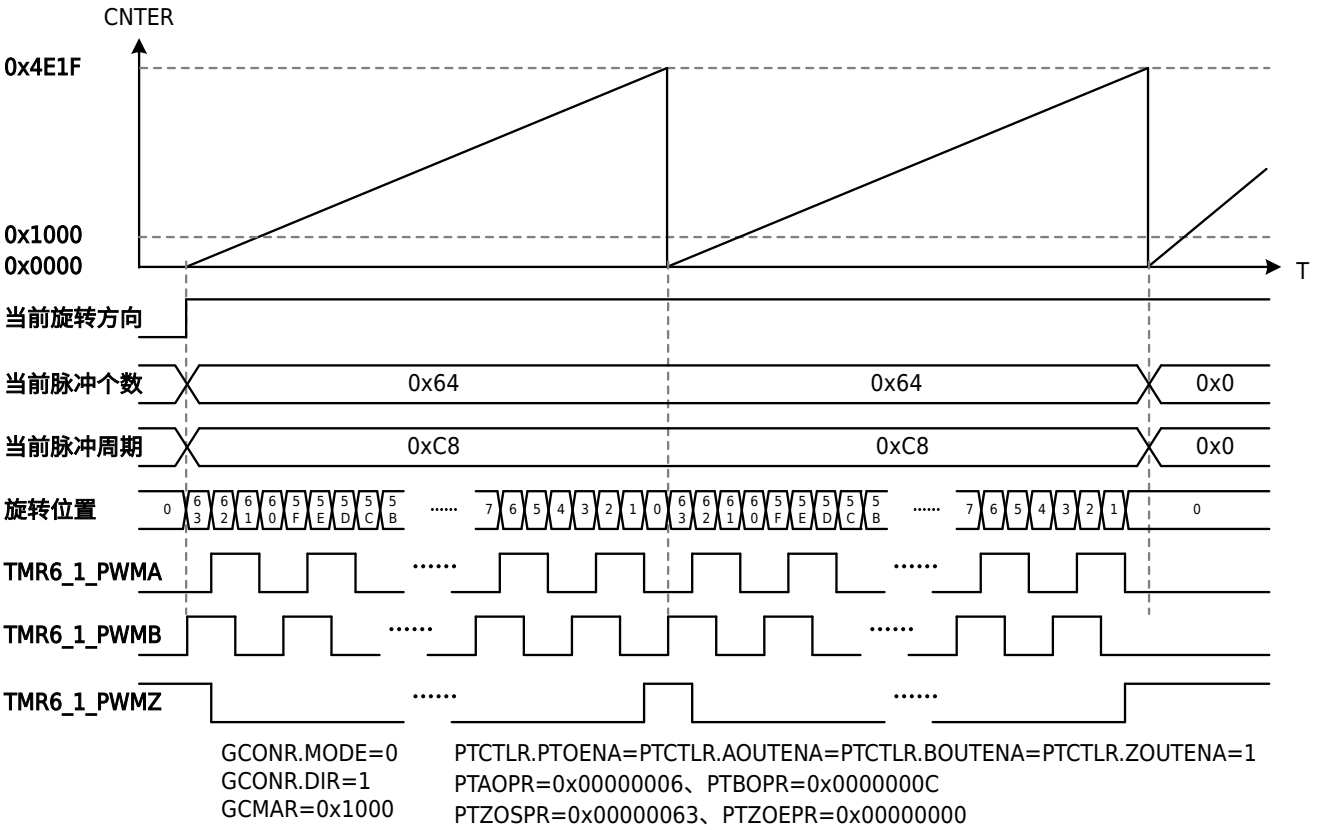


图 21-39 反转 Z 输出 (PTZOSPR>PTZOEPR)

若当前算法周期为正转，下一算法周期将反转且当前算法周期最后一个脉冲位置等于 PTZSOPR，Z 相输出如下图所示。

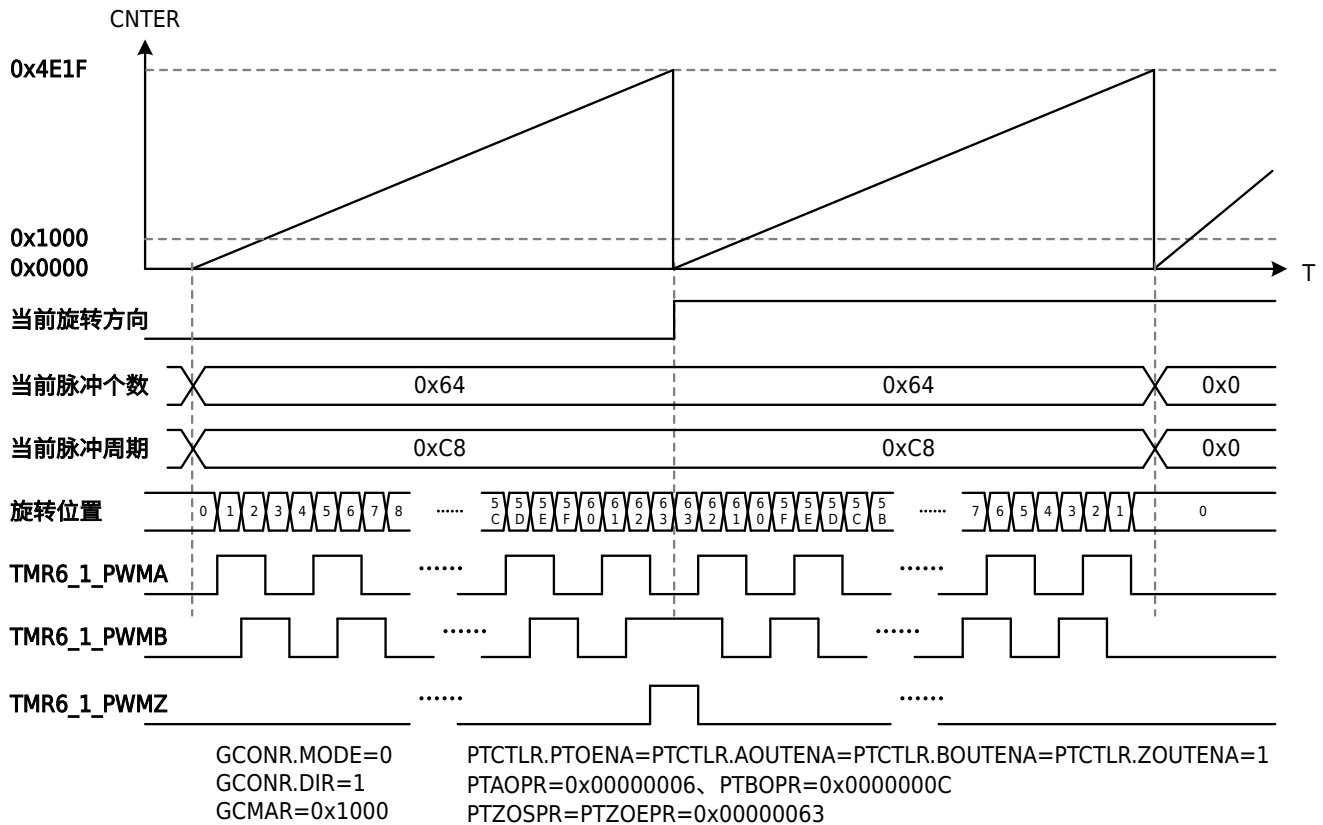


图 21-40 正转切反转 Z 输出 (PTZOSPR=PTZOEPR)

Z 相输出 (压缩)

Z 相输出支持 CLK 级压缩，指定 Z 输出位置后可配置 TMR6_PTZCMPBR.ZCMPB 寄存器来设定压缩范围，再配置寄存器 TMR6_PTZCMPBR.ZCMPBENA 打开 CLK 压缩使能。

由于 TMR6 PTPER 值是动态的，TMR6 PTZCMPBR.ZCMPB 在配置时应满足如下约束：

若 PTZOSPR、PTZOEPR 相等时应满足 $\text{TMR6\_PTZCMPBR.ZCMPB} < (\text{TMR6\_PTPER}_{\min}/2)$ 约束，否则压缩功能失效；若 PTZOSPR、PTZOEPR 不相等时，则无此限制。

压缩功能如下图所示。

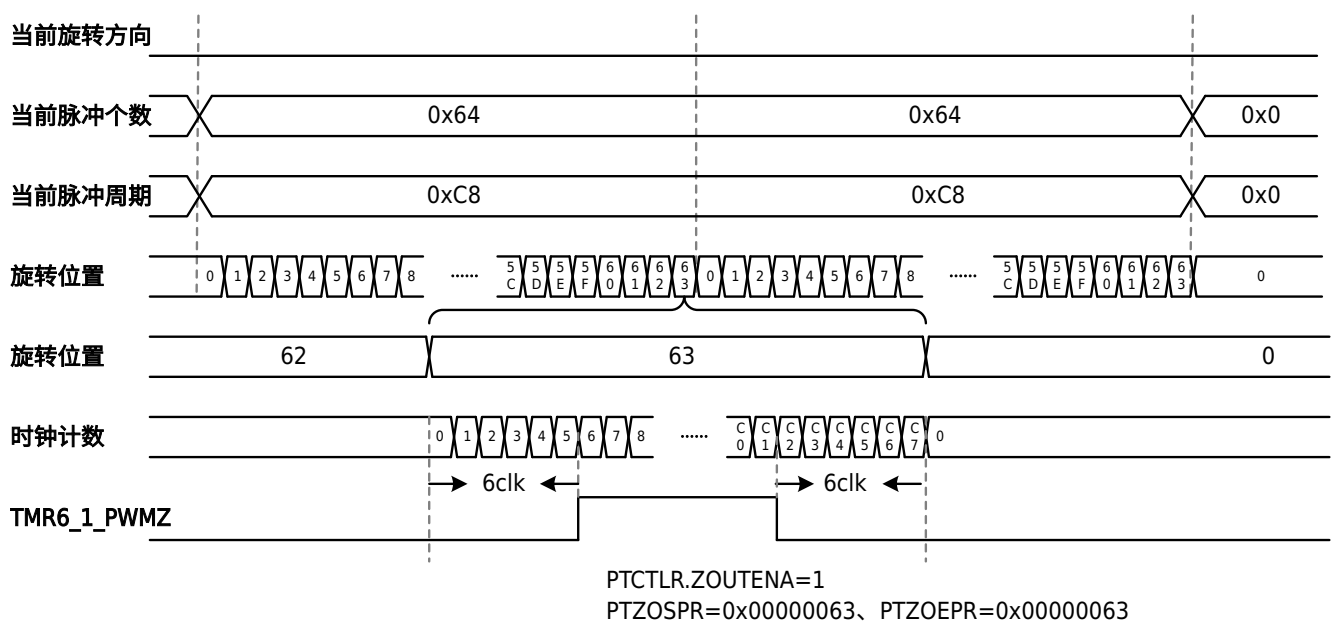


图 21-41 正转 Z 相输出压缩

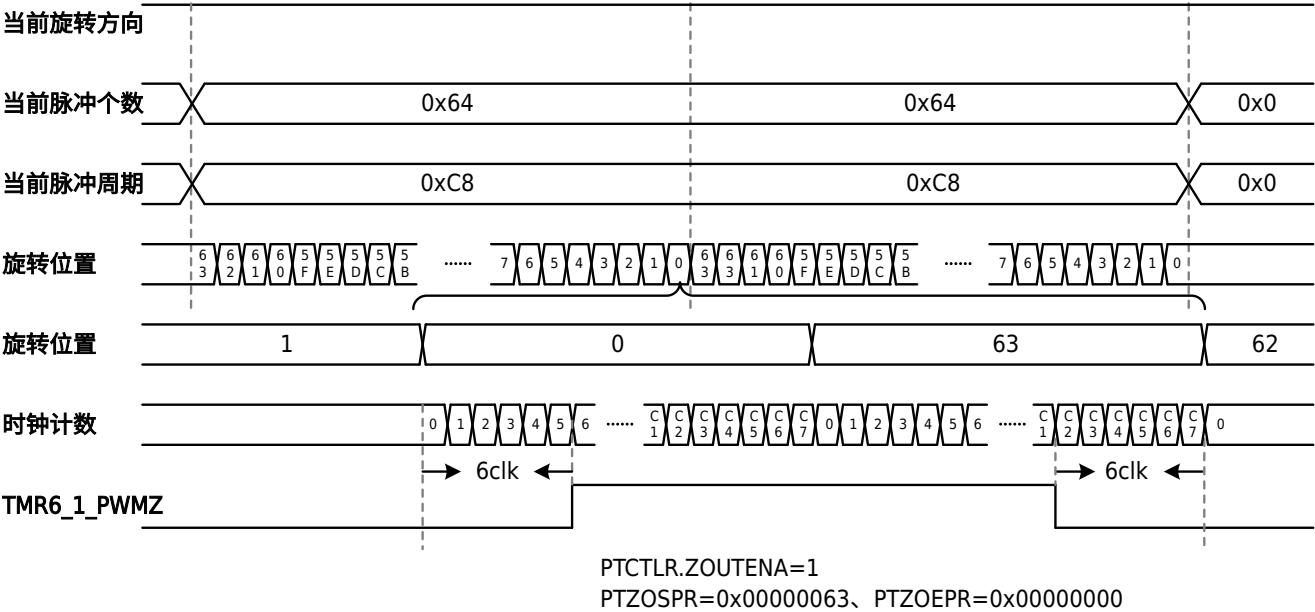


图 21-42 反转 Z 相输出压缩

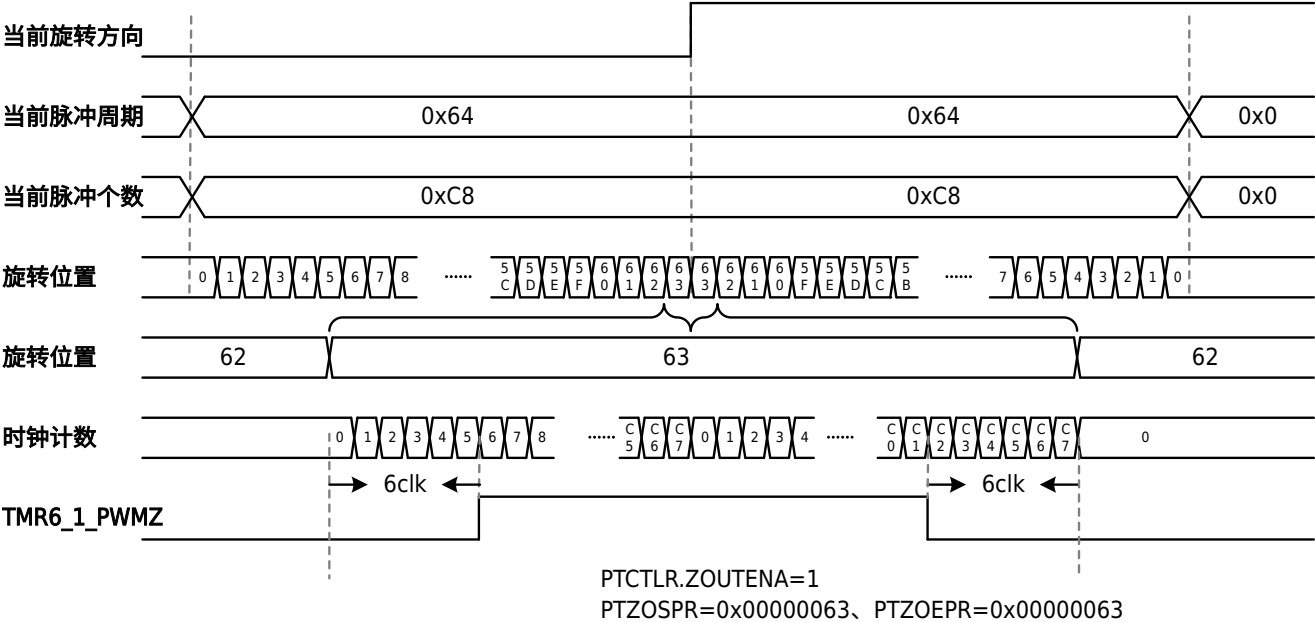


图 21-43 正转切反转 Z 相输出压缩

21.3.16.3 余数处理

由于脉冲周期=算法周期/脉冲个数，因此在软件计算脉冲周期时会出现有余数的情况，若舍去余数则下一算法周期结束前，AB 相将保持最后的输出状态。如下图所示，算法周期（0x41EF）/脉冲个数（0x 12C）=脉冲周期（0x 42），余数（0x C8）。若不补偿余数，则算法周期最后 200 个时钟周期 AB 将保持。

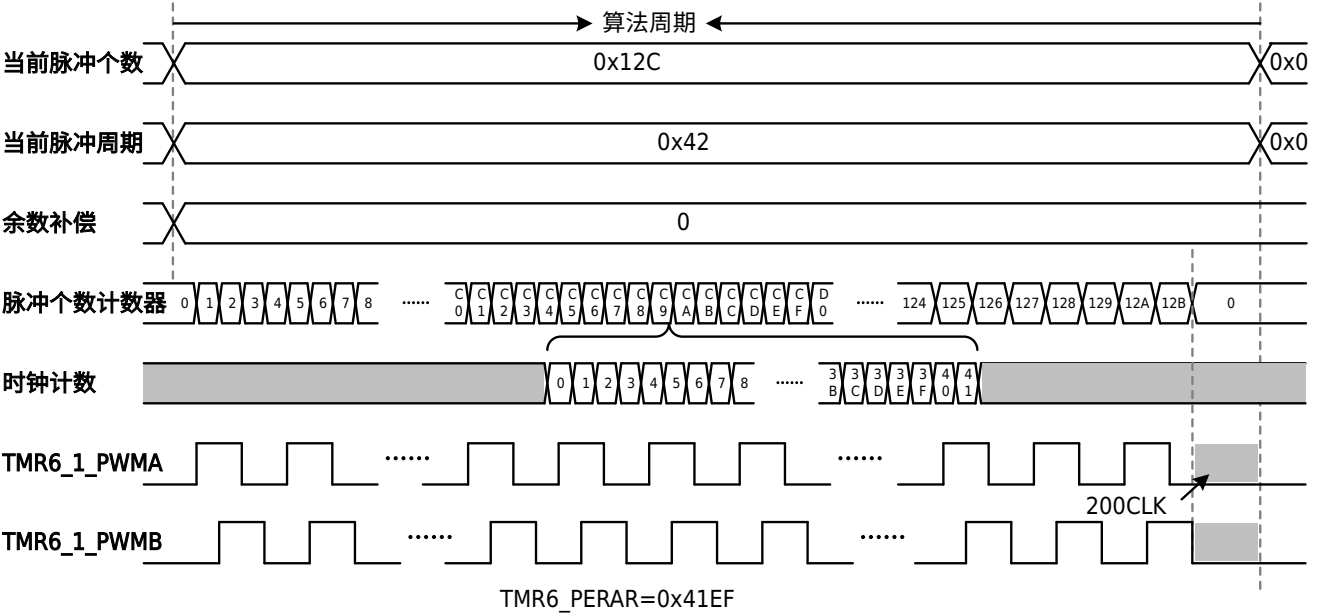


图 21-44 AB 输出（无余数补偿）

如下图所示，通过 TMR6_PTRMDR 将余数下发到硬件，前 200 个脉冲周期为 0x 42，后 100 个脉冲周期为 0x 41，将余数补充到前 200 个脉冲，每个脉冲补偿 1 个时钟周期，而不会出现上图中算法周期末尾 AB 输出停止后还有 200 时钟周期的空当期。

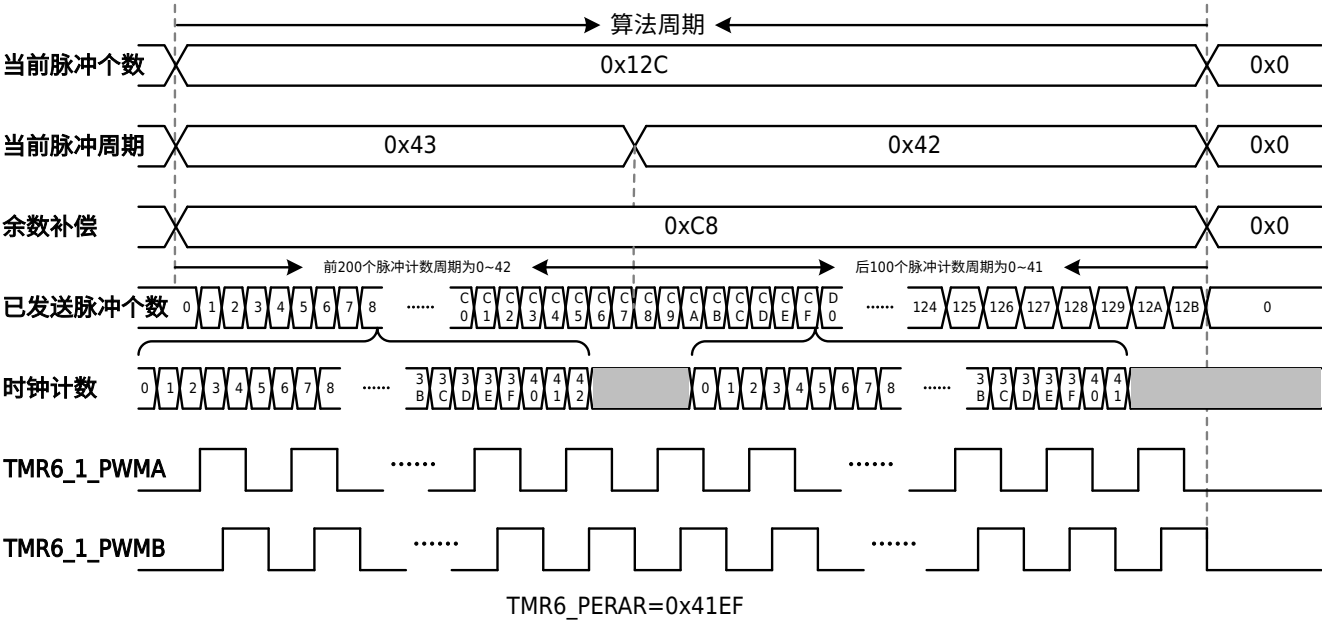


图 21-45 AB 输出（余数补偿）

21.3.16.4 位置记忆

PTPCPR、PTPCTR 为只读寄存器，分别记录了位置信息、ABZ 端口电平、旋转方向等信息。若需要复位或其他会导致 PTO 信息丢失的操作，可先读取 PTPCPR、PTPCTR 寄存器并保存，待操作完成后再将保存的信息对应写入 PTPSPR、PTPSTR，可将位置信息、ABZ 端口电平信息恢复到复位前。此时再次按之前的配置 PTO 功能，写入 TMR6_GCONR.START 为 1 后 ABZ 输出、位置可按保存的信息继续工作。

21.3.17 功能汇总表

Timer6 的锯齿波模式和三角波 A、B 模式下，主要功能的汇总表如下所示。

表 21-3 不同模式下的功能对比表

PWM 输出功能				锯齿波	三角波	相关主要寄存器
独立 PWM 输出	端口状态控制	启动时		支持	支持	PCNAR.STACA
		停止时		支持	支持	PCNAR.STPCA
		上溢时		支持	支持	PCNAR.OVFCA
		下溢时		支持	支持	PCNAR.UDFCA
		计数匹配时（Up Counting）		支持	支持	PCNAR.CMAUCA
		计数匹配时（Down Counting）		支持	支持	PCNAR.CMADCA
	缓存传送	周期基准值	单缓存	支持	支持	PERBR->PERAR
			双缓存	支持	支持	PERCR->PERBR PERBR->PERAR
		比较基准值	单缓存	支持	支持	GCMDR->GCMBR GCMCR->GCMAR
			双缓存	支持	支持	GCMFR->GCMDR GCMER->GCMCR GCMDR->GCMBR GCMCR->GCMAR
	EMB			支持	支持	PCNAR.EMBCA
互补 PWM 输出	端口状态控制	启动时		支持	支持	PCNAR.STACA
		停止时		支持	支持	PCNAR.STPCA
		上溢时		支持	支持	PCNAR.OVFCA
		下溢时		支持	支持	PCNAR.UDFCA
		计数匹配时（Up Counting）		支持	支持	PCNAR.CMAUCA
		计数匹配时（Down Counting）		支持	支持	PCNAR.CMADCA
	缓存传送	周期基准值	单缓存	支持	支持	PERBR->PERAR
			双缓存	支持	支持	PERCR->PERBR PERBR->PERAR
		比较基准值	单缓存	支持	支持	GCMDR->GCMBR GCMCR->GCMAR
			双缓存	支持	支持	GCMFR->GCMDR GCMER->GCMCR GCMDR->GCMBR GCMCR->GCMAR
		死区基准值	单缓存	支持	支持	DTUBR->DTUAR DTDBR->DTDAR
	无死区 PWM 输出			支持	支持	GCMAR=GCMBR
	带死区 PWM 输出		软件方式	支持	支持	GCMAR≠GCMBR
			硬件方式	不支持	支持	GCMBR=GCMAR-DTUAR GCNBR=GCMAR-DTDAR
	EMB			支持	支持	PCNAR.EMBCA

21.3.18 中断和事件

21.3.18.1 中断

Timer6 含有 6 个通用计数比较匹配中断（含 2 个捕获输入中断）、2 个专用计数比较匹配中断、2 个计数周期匹配中断、1 个死区时间错误中断。

计数比较匹配中断

通用比较基准值寄存器 (GCMAR~GCMFR) 共计 6 个，可分别与计数值比较产生比较匹配。计数比较匹配时，状态标志寄存器 (STFLR) 中的 STFLR.CMAF~STFLR.CMFF 位分别会被置为 1。此时若设定中断控制寄存器 (ICONR) 的 INTENA~INTENF 中相应位为 1 使能中断，则对应的中断请求 (TMR6_<t>_GCMA~F) 也会被触发。

在硬件捕获事件选择寄存器 (HCPAR、HCPBR) 选择的捕获输入有效条件产生时，捕获输入动作发生。此时若设置中断控制寄存器 (ICONR) 的 INTENA 或 INTENB 位为 1 使能中断，则对应的中断请求 (TMR6_<t>_GCMA~B) 被触发。

2 个专用比较基准值寄存器 (SCMAR~SCMBR) 也可分别与计数值比较产生比较匹配。计数比较匹配时，状态标志寄存器 (STFLR) 中的 STFLR.CMSPAF~CMSPBF 位分别会被置为 1。此时若设定中断控制寄存器 (ICONR) 的 INTENSAU<D>或 INTENSBU<D>中相应位为 1 使能中断，则对应的中断请求 (TMR6_<t>_SCMA~B) 也会被触发。

计数周期匹配中断

锯齿波递加计数至上溢点、锯齿波递减计数至下溢点、三角波计数至谷点或三角波计数至峰点时，状态标志寄存器 (STFLR) 的 STFLR.OVFF 或 STFLR.UDFF 位会被置为 1。此时若设定中断控制寄存器 (ICONR) 的 ICONR.INTENOVF 或 ICONR.INTENUDF 位使能中断，则在对应的时间点可触发计数周期匹配中断 (TMR6_<t>_GOVF 和 TMR6_<t>_GUDF)。

死区时间错误中断

死区时间基准值寄存器 (DTU<D>AR) 的值加载到通用比较基准值寄存器 (GCMBR) 中时，若超过周期限制，则会产生死区时间错误，状态标志寄存器 (STFLR) 的 STFLR.DTEF 位会被置为 1。此时若设置中断控制寄存器 (ICONR) 的 INTENDTE 位使能中断，则会在该时刻触发死区时间错误中断 (TMR6_<t>_GDTE)。

21.3.18.2 事件

在时钟计数过程中，若产生周期匹配事件（锯齿波的上溢点或下溢点、三角波的计数峰点或谷点）、通用计数比较匹配事件、专用计数比较匹配事件时，会产生相应的事件输出信号，用于选择触发别的模块，如 ADC、DMA 等。

下图是单元 1 的通用比较匹配中断 A~F&&事件 A~F、专用比较匹配中断 A~B&&事件 A~B、周期匹配中断 &&事件的动作例。

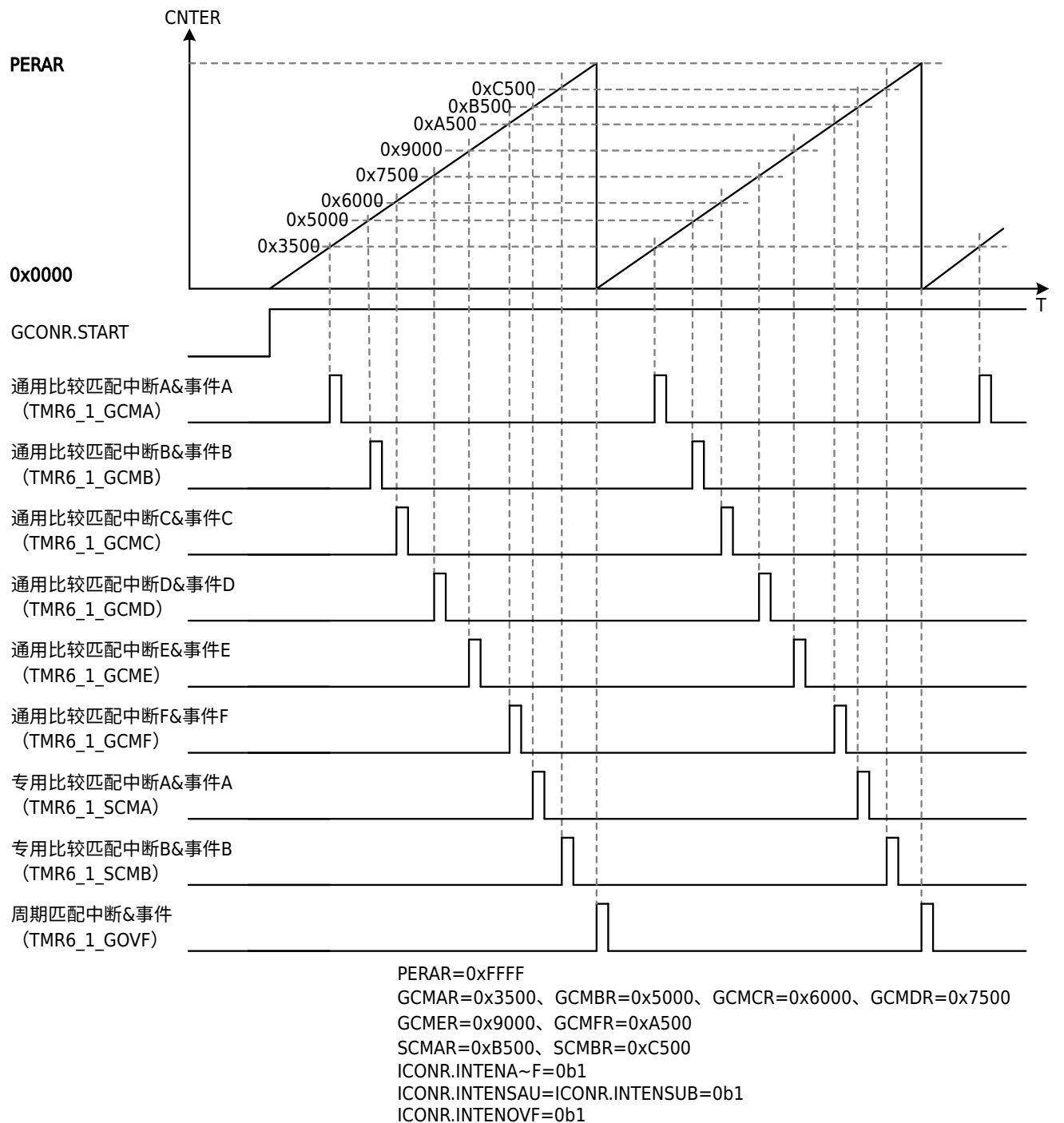


图 21-46 锯齿波模式时中断&事件输出例

21.4 操作示例

下面描述几种典型应用情况下，Timer6 相关寄存器的基本设定，供用户参考。

21.4.1 基本计数及中断动作

- 步骤 1. 设定通用周期基准值 (PERAR) ；
- 步骤 2. 设定需要的比较基准值，包括通用比较基准值 (GCMAR~GCMFR)、专用比较基准值 (SCMAR~SCMBR) 等；
- 步骤 3. 设定需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、计数匹配中断 (ICONR.INTENA~F、ICONR.INTENSAU、ICONR.INTENSAD、ICONR.INTENSBU、ICONR.INTENSBD) 等；
- 步骤 4. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ；
- 步骤 5. 设定波形模式 (GCONR.MODE) ；
- 步骤 6. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ；
- 步骤 7. 启动计数器 (GCONR.START=1) 。

21.4.2 比较输出及中断动作

- 步骤 1. 设定通用周期基准值 (PERAR) ；
- 步骤 2. 设定各通道的比较基准值，包括通用比较基准值 A (GCMAR)、通用比较基准值 B (GCMBR) ；
- 步骤 3. 设定需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、计数匹配中断 (ICONR.INTENA~B) 等；
- 步骤 4. 设定各通道在不同计数状态时的端口输出状态 (参考 PCNAR 或 PCNBR 的 Bit17~Bit0 相关控制) ；
- 步骤 5. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ；
- 步骤 6. 设定波形模式 (GCONR.MODE) ；
- 步骤 7. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ；
- 步骤 8. 设定各通道比较输出模式 (PCNAR.CAPMDA=0、PCNBR.CAPMDB=0) ；
- 步骤 9. 设定各通道输出使能 (PCNAR.OUTENA=1、PCNBR.OUTENB=1) ；
- 步骤 10. 启动计数器 (GCONR.START=1) 。

21.4.3 捕获输入及中断动作

- 步骤 1. 设定通用周期基准值 (PERAR) ；
- 步骤 2. 设定需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、捕获输入中断 (ICONR.INTENA~B) 等；
- 步骤 3. 设定各通道的捕获输入外部条件 (参考 HCPAR 或 HCPBR 的全部有效控制位。该有效控制位互相独立，可以同时选择多个均作为某通道的捕获输入条件) ；
- 步骤 4. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ；
- 步骤 5. 设定波形模式 (GCONR.MODE) ；
- 步骤 6. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ；
- 步骤 7. 设定捕获输入模式 (PCNAR.CAPMDA=1、PCNBR.CAPMDB=1) ；
- 步骤 8. 启动计数器 (GCONR.START=1) ；
- 步骤 9. 等待捕获输入条件产生，读取对应通道的捕获输入值 (GCMAR 或 GCMBR) 或等待相应的中断产生。

21.4.4 缓存传送动作 (周期基准值)

- 步骤 1. 设定需要的通用周期基准值 (PERAR、PERBR、PERCR) ；

- 步骤 2. 设定单、双缓存传送方式 (BCONR.BSEP) ;
- 步骤 3. 设定缓存传送时间点 (BCONR.BTRUP、BCONR.BTRDP, 这两个控制位互相独立, 可以同时选择, 均作为缓存传送时间点) (该设定仅在三角波模式时有效, 在锯齿波模式时无效) ;
- 步骤 4. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ;
- 步骤 5. 设定波形模式 (GCONR.MODE) ;
- 步骤 6. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ;
- 步骤 7. 设定缓存功能有效 (BCONR.BENP=1) ;
- 步骤 8. 启动计数器 (GCONR.START=1) ;
- 步骤 9. 等待对应的缓存传送时间点, 发生缓存动作 (PERBR->PERAR (BCONR.BSEP=0 时)、PERCR->PERBR->PERAR (BCONR.BSEP=1 时)) 。

21.4.5 缓存传送动作 (通用比较基准值)

- 步骤 1. 设定需要的通用比较基准值 (GCMAR、GCMCR、GCMER、GCMBR、GCMDBR、GCMFR) ;
- 步骤 2. 设定各通道单、双缓存传送方式 (BCONR.BSEA、BCONR.BSEB) ;
- 步骤 3. 设定各通道缓存传送时间点 (BCONR.BTRUA、BCONR.BTRDA、BCONR.BTRUB、BCONR.BTRDB, 每个通道的两个控制位互相独立, 可以同时选择, 均作为缓存传送时间点) (该设定仅在三角波模式时有效, 在锯齿波模式时无效) ;
- 步骤 4. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ;
- 步骤 5. 设定波形模式 (GCONR.MODE) ;
- 步骤 6. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ;
- 步骤 7. 设定各通道缓存功能有效 (BCONR.BENA=1、BCONR.BENB=1) ;
- 步骤 8. 启动计数器 (GCONR.START=1) ;
- 步骤 9. 等待各通道所设定的对应缓存传送时间点, 发生缓存动作 (GCMCR->GCMAR (BCONR.BSEA=0 时)、GCMER->GCMCR->GCMAR (BCONR.BSEA=1 时)、GCMDBR->GCMBR (BCONR.BSEB=0 时)、GCMFR->GCMDBR->GCMBR (BCONR.BSEB=1 时)) 。

21.4.6 缓存传送动作 (专用比较基准值)

- 步骤 1. 设定需要的专用比较基准值 (SCMAR、SCMCR、SCMER、SCMBR、SCMDR、SCMFR) ;
- 步骤 2. 设定各通道单、双缓存传送方式 (BCONR.BSESPA、BCONR.BSESPB) ;
- 步骤 3. 设定各通道缓存传送时间点 (BCONR.BTRUSPA、BCONR.BTRDSPA、BCONR.BTRUSPB、BCONR.BTRDSPB, 每个通道的两个控制位互相独立, 可以同时选择, 均作为缓存传送时间点) (该设定仅在三角波模式时有效, 在锯齿波模式时无效) ;
- 步骤 4. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ;
- 步骤 5. 设定波形模式 (GCONR.MODE) ;
- 步骤 6. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ;
- 步骤 7. 设定各通道缓存功能有效 (BCONR.BENSPA=1、BCONR.BENSPB=1) ;
- 步骤 8. 启动计数器 (GCONR.START=1) ;
- 步骤 9. 等待各通道所设定的对应缓存传送时间点, 发生缓存动作 (SCMCR->SCMAR (BCONR.BSESPA=0 时)、SCMER->SCMCR->SCMAR (BCONR.BSESPA=1 时)、SCMDR->SCMBR (BCONR.BSESPB=0 时)、SCMFR->SCMDR->SCMBR (BCONR.BSESPB=1 时)) 。

21.4.7 缓存传送动作 (死区基准值)

- 步骤 1. 设定需要的死区时间基准值 (DTUAR、DTUBR、DTDAR、DTDBR) ;
- 步骤 2. 设定缓存传送时间点 (DCONR.DTBTRU、DCONR.DTBTRD, 这两个控制位互相独立, 可以同时选择, 均作为缓存传送时间点) (该设定仅在三角波模式时有效, 在锯齿波模式时无效) ;

- 步骤 3. 设定内部计数时钟分频 (GCONR.CKDIV[3:0]) ;
- 步骤 4. 设定波形模式 (GCONR.MODE) ;
- 步骤 5. 设定计数方向 (仅在锯齿波模式 GCONR.MODE=0 时需要设定) ;
- 步骤 6. 设定缓存功能有效 (DCONR.DTBENU=1、DCONR.DTBEND=1) ;
- 步骤 7. 设定硬件死区功能有效 (DCONR.DTCEN=1) ;
- 步骤 8. 启动计数器 (GCONR.START=1) ;
- 步骤 9. 等待对应缓存传送时间点, 发生缓存动作 (DTUBR->DTUAR、DTDBR->DTDAR) 。

21.4.8 同步启动动作 (软件方式)

- 步骤 1. 参考"基本计数及中断动作"章节的步骤 1~6, 对需要同步启动的各个单元做设定;
- 步骤 2. 同步启动计数器 (设定 SSTAR 寄存器的对应位为 1, 每个单元对应一个寄存器位) 。

21.4.9 同步启动动作 (硬件方式)

- 步骤 1. 设定通用周期基准值 (PERAR) ;
- 步骤 2. 设定需要的比较基准值, 包括通用比较基准值 (GCMAR~GCMFR)、专用比较基准值 (SCMAR~SCMBR) 等;
- 步骤 3. 设定需要的中断使能位, 包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUF)、计数匹配中断 (ICONR.INTENA~F、ICONR.INTENSAU、ICONR.INTENSAD、ICONR.INTENSBUS、ICONR.INTENSBD) 等;
- 步骤 4. 设定硬件启动条件 (通过 HSTAR.HSTAx 选择, x=8~31) ;
- 步骤 5. 设定硬件启动使能 (HSTAR.STAS=1) ;
- 步骤 6. 重复上述 1~5 步骤对需要同步启动的各个单元做设定 (需要同步启动的各个单元中, 步骤 4 的设定要一致) ;
- 步骤 7. 等待设定的触发事件产生, 确认各个单元的计数器同步启动。

21.4.10 正交编码计数动作 (2 相)

- 步骤 1. 设定通用周期基准值 (PERAR) ;
- 步骤 2. 设定需要的比较基准值, 包括通用比较基准值 (GCMAR~GCMFR)、专用比较基准值 (SCMAR~SCMBR) 等;
- 步骤 3. 设定需要的中断使能位, 包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUF)、计数匹配中断 (ICONR.INTENA~F、ICONR.INTENSAU、ICONR.INTENSAD、ICONR.INTENSBUS、ICONR.INTENSBD) 等;
- 步骤 4. 设定需要的硬件向上计数条件 (通过 HCUPR.HCUPx 选择, x=0~7) ;
- 步骤 5. 设定需要的硬件向下计数条件 (通过 HCDOR.HCDOx 选择, x=0~7) ;
- 步骤 6. 启动计数器 (GCONR.START=1) ;
- 步骤 7. 等待设定的正交编码计数事件产生, 确认计数器正常计数。

21.4.11 正交编码计数动作 (3 相)

- 步骤 1. 参考"正交编码计数动作 (2 相)"章节的步骤 1~5, 对位置计数单元做设定;
- 步骤 2. 设定位置计数单元的硬件清零条件 (通过 HCLRR.HCLR_x 选择, x=16~31) ;
- 步骤 3. 设定位置计数单元硬件清零使能 (HCLRR.CLES=1) ;
- 步骤 4. 设定公转计数单元的通用周期基准值 (PERAR) ;
- 步骤 5. 设定公转计数单元的比较基准值, 包括通用比较基准值 (GCMAR~GCMFR)、专用比较基准值 (SCMAR~SCMBR) 等;

- 步骤 6.** 设定公转计数单元需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、计数匹配中断 (ICONR.INTENA~F、ICONR.INTENSAU、ICONR.INTENSAD、ICONR.INTENSBU、ICONR.INTENSBD) 等；
- 步骤 7.** 设定公转计数单元的硬件向上计数条件 1 (ZIN 相输入) (通过 HCUPR.HCUPx 选择, x=16~31, 此处的设定事件应和步骤 2 中位置计数单元设定的事件一致)；
- 步骤 8.** 设定公转计数单元的硬件向上计数条件 2 (位置计数单元的上溢事件输入) (通过 HCUPR.HCUP8 选择内部硬件触发事件 0)；
- 步骤 9.** 设定公转计数单元的硬件向下计数条件 (位置计数单元的下溢事件输入) (通过 HCDOR.HCDO9 选择内部硬件触发事件 1)；
- 步骤 10.** 设定 TMR6_TRGSEL0 中的触发源编号为位置计数单元的计数上溢事件 (寄存器内容详见《自动运行系统 (AOS)》章节“寄存器-Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3)”，上溢事件编号请参见《中断控制器 (INTC)》章节)；
- 步骤 11.** 设定 TMR6_TRGSEL1 中的触发源编号为位置计数单元的计数下溢事件 (寄存器内容详见《自动运行系统 (AOS)》章节“寄存器-Timer6 硬件触发事件选择寄存器 (TMR6_TRGSELx) (x=0~3)”，上溢事件编号请参见《中断控制器 (INTC)》章节)；
- 步骤 12.** 启动公转计数单元计数器 (GCONR.START=1)；
- 步骤 13.** 启动位置计数单元计数器 (GCONR.START=1)；
- 步骤 14.** 等待设定的 AIN、BIN、ZIN 相计数事件产生，确认计数器正常计数。

21.4.12 单路 PWM 输出

参考“[比较输出及中断动作](#)”章节步骤 1~10 的设定 (每个单元内部的 2 个 PWM 通道 TMR6_<t>_PWMA 和 TMR6_<t>_PWMB 的输出状态均可独立设定，形成 2 个互不相关的单路 PWM 输出)。

21.4.13 互补 PWM 输出 (软件死区)

- 步骤 1.** 设定通用周期基准值 (PERAR)；
- 步骤 2.** 设定通用比较基准值 A (GCMAR)、通用比较基准值 B (GCMBR)；
- 步骤 3.** 设定需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、计数匹配中断 (ICONR.INTENA~B) 等；
- 步骤 4.** 设定在不同计数状态时的端口输出状态 (参考 PCNAR 和 PCNBR 的 Bit17~Bit0 相关控制，结合 GCMAR 和 GCMBR 的设定值，需要保证 2 个 PWM 输出之间形成互补死区)；
- 步骤 5.** 设定内部计数时钟分频 (GCONR.CKDIV[3:0])；
- 步骤 6.** 设定波形模式为三角波模式 (GCONR.MODE=1)；
- 步骤 7.** 设定比较输出模式 (PCNAR.CAPMDA=0、PCNBR.CAPMDB=0)；
- 步骤 8.** 设定输出使能 (PCNAR.OUTENA=1、PCNBR.OUTENB=1)；
- 步骤 9.** 启动计数器 (GCONR.START=1)。

21.4.14 互补 PWM 输出 (硬件死区)

- 步骤 1.** 设定通用周期基准值 (PERAR)；
- 步骤 2.** 设定通用比较基准值 A (GCMAR)、死区时间基准值 (DTUAR、DTDAR)；
- 步骤 3.** 设定需要的中断使能位，包括计数上溢中断 (ICONR.INTENOVF)、计数下溢中断 (ICONR.INTENUDF)、计数匹配中断 (ICONR.INTENA~B)、死区错误中断 (ICONR.INTENDTE) 等；
- 步骤 4.** 设定在不同计数状态时的端口输出状态 (参考 PCNAR 和 PCNBR 的 Bit17~Bit0 相关控制，结合 GCMAR、DTUAR 和 DTDAR 的设定值，需要保证 2 个 PWM 输出之间形成互补死区)；
- 步骤 5.** 设定内部计数时钟分频 (GCONR.CKDIV[3:0])；
- 步骤 6.** 设定波形模式为三角波模式 (GCONR.MODE=1)；

- 步骤 7. 设定各通道比较输出模式 (PCNAR.CAPMDA=0、PCNBR.CAPMDB=0) ；
- 步骤 8. 设定各通道输出使能 (PCNAR.OUTENA=1、PCNBR.OUTENB=1) ；
- 步骤 9. 设定硬件死区功能有效 (DCONR.DTCEN=1) ；
- 步骤 10. 启动计数器 (GCONR.START=1) 。

21.4.15 EMB 监控及中断动作

- 步骤 1. 参考"互补 PWM 输出 (软件死区)"章节的步骤 1~8 或"互补 PWM 输出 (硬件死区)"章节步骤 1~9, 对互补 PWM 输出动作进行设定；
- 步骤 2. 设定 EMB 事件发生时 PWM 端口的状态 (PCNAR.EMBCA、PCNBR.EMBCB) (根据系统应用的不同, 选择相应的保护状态) ；
- 步骤 3. 设定 EMB 事件变为无效时 PWM 端口恢复正常输出的时间点 (PCNAR.EMBRA、PCNBR.EMBRB) ；
- 步骤 4. 设定选择从 EMB 模块输入的 EMB 事件源 (PCNAR.EMBSA、PCNBR.EMBSB) ；
- 步骤 5. 设定 EMB 模块的相关寄存器 (包括 EMB 中断许可寄存器 (EMB_INTEN0~3)、EMB 控制寄存器 1/2 (EMB_CTL1/2_0~3)、EMB 控制 PWM 输出释放方式选择寄存器 (EMB_RLSSEL0~3) 等) ；
- 步骤 6. 启动计数器 (GCONR.START=1) , EMB 模块实时监控系统状态。

21.5 寄存器

21.5.1 寄存器总表

- Timer6_1 基地址：0x40018000
- Timer6_2 基地址：0x40018400
- Timer6_3 基地址：0x40018800
- Timer6_4 基地址：0x40018C00
- Timer6_5 基地址：0x40019000
- Timer6_6 基地址：0x40019400
- Timer6_7 基地址：0x40019800
- Timer6_8 基地址：0x40019C00



说明

Timer6 为定时器名称，Timer6_<t> (<t>：1~8) 为该定时器单元。

表 21-4 高级控制定时器 (Timer6) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	TMR6_CNTER	通用计数值寄存器	32
0x0004	TMR6_UPDAR	通用刷新值寄存器	32
0x0040	TMR6_PERAR	通用周期基准值寄存器 A	32
0x0044	TMR6_PERBR	通用周期基准值寄存器 B	32
0x0048	TMR6_PERCR	通用周期基准值寄存器 C	32
0x0080	TMR6_GCMAR	通用比较基准值寄存器 A	32
0x0084	TMR6_GCMBR	通用比较基准值寄存器 B	32
0x0088	TMR6_GCMCR	通用比较基准值寄存器 C	32
0x008C	TMR6_GCMDR	通用比较基准值寄存器 D	32
0x0090	TMR6_GCMER	通用比较基准值寄存器 E	32
0x0094	TMR6_GCMFR	通用比较基准值寄存器 F	32
0x00C0	TMR6_SCMAR	专用比较基准值寄存器 A	32
0x00C4	TMR6_SCMBR	专用比较基准值寄存器 B	32
0x00C8	TMR6_SCMCR	专用比较基准值寄存器 C	32
0x00CC	TMR6_SCMDR	专用比较基准值寄存器 D	32
0x00D0	TMR6_SCMER	专用比较基准值寄存器 E	32
0x00D4	TMR6_SCMFR	专用比较基准值寄存器 F	32
0x0100	TMR6_DTUAR	死区时间基准值寄存器 UA	32
0x0104	TMR6_DTDAR	死区时间基准值寄存器 DA	32

偏移地址	寄存器	描述	位宽
0x0108	TMR6_DTUBR	死区时间基准值寄存器 UB	32
0x010C	TMR6_DTDBR	死区时间基准值寄存器 DB	32
0x0140	TMR6_GCONR	通用控制寄存器	32
0x0144	TMR6_ICONR	中断控制寄存器	32
0x0148	TMR6_BCONR	缓存控制寄存器	32
0x014C	TMR6_DCONR	死区控制寄存器	32
0x0154	TMR6_PCNAR	端口控制寄存器 A	32
0x0158	TMR6_PCNBR	端口控制寄存器 B	32
0x015C	TMR6_FCNGR	滤波控制寄存器	32
0x0160	TMR6_VPERR	有效周期寄存器	32
0x0164	TMR6_STFLR	状态标志寄存器	32
0x0180	TMR6_HSTAR	硬件启动事件选择寄存器	32
0x0184	TMR6_HSTPR	硬件停止事件选择寄存器	32
0x0188	TMR6_HCLRR	硬件清零事件选择寄存器	32
0x018C	TMR6_HUPDR	硬件刷新事件选择寄存器	32
0x0190	TMR6_HCPAR	硬件捕获事件选择寄存器 A	32
0x0194	TMR6_HCPBR	硬件捕获事件选择寄存器 B	32
0x0198	TMR6_HCUPR	硬件递加事件选择寄存器	32
0x019C	TMR6_HCDOR	硬件递减事件选择寄存器	32
0x0200	TMR6_PTNUMR	PTO 脉冲个数寄存器	32
0x0204	TMR6_PTPERR	PTO 脉冲周期寄存器	16
0x0208	TMR6_PTRMDR	PTO 算法周期余数寄存器	32
0x020C	TMR6_PTCTLR	PTO 模式控制寄存器	16
0x0220	TMR6_PTAOPR	PTO A 相输出位置寄存器	16
0x0228	TMR6_PTBOPR	PTO B 相输出位置寄存器	16
0x0240	TMR6_PTZOSPR	PTO Z 相输出起始位置寄存器	32
0x0248	TMR6_PTZOEPR	PTO Z 相输出结束位置寄存器	32
0x0264	TMR6_PTZCMPBR	PTO Z 相输出脉宽压缩 B 寄存器	32
0x0268	TMR6_PTZPERR	PTO 整圈脉冲个数寄存器	32
0x0280	TMR6_PTPSPR	PTO 脉冲初始位置寄存器	32

偏移地址	寄存器	描述	位宽
0x0284	TMR6_PTPSTR	PTO 脉冲初始端口寄存器	32
0x0290	TMR6_PTPCPR	PTO 脉冲当前位置寄存器	32
0x0294	TMR6_PTPCTR	PTO 脉冲当前端口寄存器	32

Timer6 公共寄存器基地址：0x40018300

表 21-5 Timer6 公共寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00EC	TMR6_FCNTNTR	公共滤波控制寄存器	32
0x00F0	TMR6_SSTAR	公共软件同步启动控制寄存器	32
0x00F4	TMR6_SSTPR	公共软件同步停止控制寄存器	32
0x00F8	TMR6_SCLLR	公共软件同步清零控制寄存器	32
0x00FC	TMR6_SUPDR	公共软件同步刷新控制寄存器	32

说明

- 单元 1~单元 4 单元的通用计数值寄存器、通用刷新值寄存器及各基准值寄存器的位宽均为 32 位，各复位值与单元 5~单元 8 单元不一样。其中，通用计数值寄存器、通用刷新值寄存器的复位值为 0x00000000；各基准值寄存器的复位值为 0xFFFFFFFF。
- 触发选择寄存器（TMR6_TRGSEL0~3）是 4 个独立于单元的寄存器，为 8 个单元 Timer6 所共有。
- 软件同步寄存器（TMR6_SSTAR、TMR6_SSTPR、TMR6_SCLLR、TMR6_SUPDR）是 4 个独立于单元的寄存器，为 8 个单元 Timer6 所共有。
- 滤波控制寄存器（TMR6_FCNTNTR）是 1 个独立于单元的寄存器，为 8 个单元 Timer6 所共有。

21.5.2 通用计数值寄存器（TMR6_CNTER）

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNT	计数值	当前定时器的计数值  说明 锯齿波向上计数模式，CNT 初始值设置为 0；锯齿波向下计数模式，CNT 初始值设置为周期值；三角波模式计数 CNT 初始值设置为 1。	RW

21.5.3 通用刷新值寄存器（TMR6_UPDAR）

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																UPDA															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	UPDA	刷新值	设定要更新进定时器的刷新值	RW

21.5.4 通用周期基准值寄存器 (TMR6_PERmR) (m=A~C)

偏移地址 (Offset) : 0x0040 (TMR6_PERAR) , 0x0044 (TMR6_PERBR) , 0x0048 (TMR6_PERCR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																PER															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	PER	计数周期值	设定每轮计数的计数周期值及对应缓存值	RW

21.5.5 通用比较基准值寄存器 (TMR6_GCMmR) (m=A~F)

偏移地址 (Offset) : 0x0080 (TMR6_GCMAR) , 0x0084 (TMR6_GCMBR) , 0x0088 (TMR6_GCMCR) , 0x008C (TMR6_GCMDR) , 0x0090 (TMR6_GCMER) , 0x0094 (TMR6_GCMFR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																GCM															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	GCM	计数比较基准值	通用比较基准值设定, 与计数值相等时匹配信号有效	RW

21.5.6 专用比较基准值寄存器 (TMR6_SCMmR) (m=A~F)

偏移地址 (Offset) : 0x00C0 (TMR6_SCMAR) , 0x00C4 (TMR6_SCMBR) , 0x00C8 (TMR6_SCMCR) , 0x00CC (TMR6_SCMDR) , 0x00D0 (TMR6_SCMER) , 0x00D4 (TMR6_SCMFR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																SCM															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	SCM	专用比较基准值	设定比较基准值及缓存值	RW

21.5.7 死区时间基准值寄存器 (TMR6_DTmnR) (m=U/D, n=A/B)

偏移地址 (Offset) : 0x0100 (TMR6_DTUAR) , 0x0104 (TMR6_DTDAR) , 0x0108 (TMR6_DTUBR) , 0x010C (TMR6_DTDBR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																DT															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	DT	死区时间值	死区时间设定值及缓存值	RW

21.5.8 通用控制寄存器 (TMR6_GCONR)

Offset	Bit Position																															
0x0140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved												ZMSKVAL	ZMSKPOS	ZMSKREV	Reserved							FODIR	OVSTP	CKDIV			Reserved	MODE	DIR	START	
Access													RW	RW	RW								RW	RW	RW				RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19:18	ZMSKVAL	Z 相输入屏蔽周期数	0b00: Z 相输入屏蔽功能无效 0b01: 位置计数上溢后或下溢后的 4 个计数周期内的 Z 相输入被屏蔽 0b10: 位置计数上溢后或下溢后的 8 个计数周期内的 Z 相输入被屏蔽 0b11: 位置计数上溢后或下溢后的 16 个计数周期内的 Z 相输入被屏蔽	RW
17	ZMSKPOS	Z 相输入位置定时器选择	0b0: Z 相输入时该单元作为位置定时器, 在屏蔽周期期间内位置定时器清零功能正常动作 0b1: Z 相输入时该单元作为位置定时器, 在屏蔽周期期间内位置定时器清零功能被屏蔽	RW
16	ZMSKREV	Z 相输入公转定时器选择	0b0: Z 相输入时该单元作为公转定时器, 在屏蔽周期期间内公转定时器计数功能正常动作 0b1: Z 相输入时该单元作为公转定时器, 在屏蔽周期期间内公转定时器计数功能被屏蔽	RW
15:10	Reserved	保留	-	-
9	FODIR	强制修改计数方向	0b0: 锯齿波模式上/下溢点, 更新计数方向为 DIR; 三角波模式在上/下溢点自动切换方向, 设定 DIR 无效 0b1: 立即更新计数方向为 DIR  说明 该功能在计数器停止时有效。	RW
8	OVSTP	计数溢出停止控制	0b0: 计数器在计数上溢或下溢后, 继续计数 0b1: 计数器在计数上溢或下溢后, 停止计数	RW

位/位域	标记	位名	功能描述	读写
7:4	CKDIV	计数时钟选择	● 0b0000: PCLK0 ● 0b0001: PCLK0/2 ● 0b0010: PCLK0/4 ● 0b0011: PCLK0/8 ● 0b0100: PCLK0/16 ● 0b0101: PCLK0/32 ● 0b0110: PCLK0/64 ● 0b0111: PCLK0/128 ● 0b1000: PCLK0/256 ● 0b1001: PCLK0/512 ● 0b1010: PCLK0/1024 ● 其他: 禁止设置	RW
3	<i>Reserved</i>	<i>保留</i>	-	-
2	MODE	计数模式	● 0b0: 锯齿波模式 ● 0b1: 三角波模式  说明 PTO 模式下, 设置为锯齿波模式。	RW
1	DIR	计数方向	● 0b0: 递减计数 ● 0b1: 递加计数  说明 PTO 模式下, 设置为递加计数。	RW
0	START	定时器启动	● 0b0: 定时器关闭 ● 0b1: 定时器启动  说明 该位在软件停止条件或硬件停止条件有效时, 会自动变为 0。	RW

21.5.9 中断控制寄存器 (TMR6_ICONR)

Offset	Bit Position																																			
0x0144	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved												INTENSBD	INTENSBU	INTENSAD	INTENSAU	Reserved								INTENDTE	INTENUDF	INTENOVF	INTENF	INTENE	INTEND	INTENC	INTENB	INTENA			
Access													RW	RW	RW	RW									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19	INTENSBD	专用向下计数中断使能 B	中断条件：向下计数期间，SCMBR 寄存器和计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
18	INTENSBU	专用向上计数中断使能 B	中断条件：向上计数期间，SCMBR 寄存器和计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
17	INTENSAD	专用向下计数中断使能 A	中断条件：向下计数期间，SCMAR 寄存器和计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
16	INTENSAU	专用向上计数中断使能 A	中断条件：向上计数期间，SCMAR 寄存器与计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
15:9	Reserved	保留	-	-
8	INTENDTE	死区时间错误中断使能	中断条件：死区时间错误 ● 0b0：禁止 ● 0b1：使能	RW
7	INTENUDF	下溢中断使能	中断条件：计数器计数下溢 ● 0b0：禁止 ● 0b1：使能	RW
6	INTENOVF	上溢中断使能	中断条件：计数器计数上溢 ● 0b0：禁止 ● 0b1：使能	RW
5	INTENF	计数匹配中断使能 F	中断条件：GCMFR 寄存器与计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
4	INTENE	计数匹配中断使能 E	中断条件：GCMER 寄存器与计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW

位/位域	标记	位名	功能描述	读写
3	INTEND	计数匹配中断使能 D	中断条件：GCMDR 寄存器与计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
2	INTENC	计数匹配中断使能 C	中断条件：GCMCR 寄存器与计数值 (CNTERR) 相等 ● 0b0：禁止 ● 0b1：使能	RW
1	INTENB	计数匹配中断使能 B	中断条件：GCMBR 寄存器与计数值 (CNTERR) 相等，或发生输入捕获 ● 0b0：禁止 ● 0b1：使能	RW
0	INTENA	计数匹配中断使能 A	中断条件：GCMAR 寄存器与计数值 (CNTERR) 相等，或发生输入捕获 ● 0b0：禁止 ● 0b1：使能	RW

21.5.10 缓存控制寄存器 (TMR6_BCONR)

Offset	Bit Position																																				
0x0148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved								BTRDSPB	BTRUSPB	BSESPB	BENSPB	BTRDSPA	BTRUSPA	BSESPA	BENSPA	Reserved								BTRDP	BTRUP	BSEP	BENP	BTRDB	BTRUB	BSEB	BENB	BTRDA	BTRUA	BSEA	BENA	
Access									RW	RW	RW	RW	RW	RW	RW	RW									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	BTRDSPB	专用比较基准值缓存传送时间设定 DB	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
22	BTRUSPB	专用比较基准值缓存传送时间设定 UB	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
21	BSESPB	专用比较基准值缓存传送选择 B	0b0: 单缓存传送 (SCMDR->SCMBR) 0b1: 双缓存传送 (SCMFR->SCMDR->SCMBR)	RW
20	BENSPB	专用比较基准值缓存传送 B	0b0: 缓存传送无效 0b1: 缓存传送使能	RW
19	BTRDSPA	专用比较基准值缓存传送时间设定 DA	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
18	BTRUSPA	专用比较基准值缓存传送时间设定 UA	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
17	BSESPA	专用比较基准值缓存传送选择 A	0b0: 单缓存传送 (SCMCR->SCMAR) 0b1: 双缓存传送 (SCMER->SCMCR->SCMAR)	RW

位/位域	标记	位名	功能描述	读写
16	BENSPA	专用比较基准值缓存传送 A	0b0: 缓存传送无效 0b1: 缓存传送使能	RW
15:12	Reserved	保留	-	-
11	BTRDP	周期值缓存传送时间设定 D	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
10	BTRUP	周期值缓存传送时间设定 U	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
9	BSEP	周期值缓存传送选择	0b0: 单缓存传送 (PERBR->PERAR) 0b1: 双缓存传送 (PERCR->PERBR->PERAR)  说明 传送时间点跟计数模式无关, 只在锯齿波的上溢点、下溢点或三角波的波谷。	RW
8	BENP	周期值缓存传送	0b0: 缓存传送无效 0b1: 缓存传送使能	RW
7	BTRDB	通用比较值缓存传送时间设定 DB	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
6	BTRUB	通用比较值缓存传送时间设定 UB	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
5	BSEB	通用比较值缓存传送选择 B	比较输出功能时: 0b0: 单缓存传送 (GCMDR->GCMBR) 0b1: 双缓存传送 (GCMFR->GCMDR->GCMBR) 捕获输入功能时: 0b0: 单缓存传送 (GCMBR->GCMDR) 0b1: 双缓存传送 (GCMBR->GCMDR->GCMFR)	RW

位/位域	标记	位名	功能描述	读写
4	BENB	通用比较值缓存 传送 B	0b0: 缓存传送无效 0b1: 缓存传送使能	RW
3	BTRDA	通用比较值缓存 传送时间设定 DA	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
2	BTRUA	通用比较值缓存 传送时间设定 UA	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
1	BSEA	通用比较值缓存 传送选择 A	比较输出功能时: 0b0: 单缓存传送 (GCMCR->GCMAR) 0b1: 双缓存传送 (GCMER->GCMCR->GCMAR) 捕获输入功能时: 0b0: 单缓存传送 (GCMAR->GCMCR) 0b1: 双缓存传送 (GCMAR->GCMCR->GCMER)	RW
0	BENA	通用比较值缓存 传送 A	0b0: 缓存传送无效 0b1: 缓存传送使能	RW

21.5.11 死区控制寄存器 (TMR6_DCONR)

Offset	Bit Position																															
0x014C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								DTBTRD	DTBTRU	DTBEND	DTBENU	Reserved	SEPA	DTCEN	
Access																									RW	RW	RW	RW		RW	RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	DTBTRD	死区时间值缓存 传送时间设定 D	0b0: 三角波计数模式计数到谷点时, 死区缓存值不传送 0b1: 三角波计数模式计数到谷点时, 发生一次死区缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
6	DTBTRU	死区时间值缓存 传送时间设定 U	0b0: 三角波计数模式计数到峰点时, 死区缓存值不传送 0b1: 三角波计数模式计数到峰点时, 发生一次死区缓存值传送  说明 锯齿波模式时, 不需要该位控制。	RW
5	DTBEND	死区时间值缓存 传送 D	0b0: 缓存传送无效 0b1: 缓存传送使能 (DTDBR->DTDAR)	RW
4	DTBENU	死区时间值缓存 传送 U	0b0: 缓存传送无效 0b1: 缓存传送使能 (DTUBR->DTUAR)	RW
3:2	Reserved	保留	-	-
1	SEPA	分离设定	0b0: DTUAR 和 DTDAR 分别设定 0b1: DTUAR 和 DTDAR 的值自动相等	RW
0	DTCEN	死区功能	0b0: 死区功能无效 0b1: 死区功能有效	RW

21.5.12 端口控制寄存器 A (TMR6_PCNAR)

Offset	Bit Position															
0x0154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x00000000															
Name	CAPMDA	Reserved	OUTENA	Reserved	EMBSA	EMBRA	EMBCA	Reserved	FORCA	CMBDCA	CMBUCA	CMADCA	CMAUCA	UDFCA	OVFCA	STPCA
Access	RW		RW		RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	CAPMDA	功能模式选择 A	0b0: 比较输出功能 0b1: 捕获输入功能	RW
30:29	Reserved	保留	-	-
28	OUTENA	输出使能 A	0b0: Timer6 功能时的 TMR6_<t>_PWMA 端口输出无效 0b1: Timer6 功能时的 TMR6_<t>_PWMA 端口输出有效	RW
27:26	Reserved	保留	-	-
25:24	EMBSA	EMB 事件时端口状态选择 A	0b00: 选择 EMB 的 Group0 控制事件有效 0b01: 选择 EMB 的 Group1 控制事件有效 0b10: 选择 EMB 的 Group2 控制事件有效 0b11: 选择 EMB 的 Group3 控制事件有效	RW
23:22	EMBRA	EMB 释放方式选择 A	0b00: EMBSA 选择的 EMB 控制事件无效时, 立即释放 TMR6_<t>_PWMA 端口 (One Shot) 0b01: EMBSA 选择的 EMB 控制事件无效时, 等到计数器计数到上溢时释放 TMR6_<t>_PWMA 端口 (Cycle By Cycle 1) 0b10: EMBSA 选择的 EMB 控制事件无效时, 等到计数器计数到下溢时释放 TMR6_<t>_PWMA 端口 (Cycle By Cycle 2) 0b11: EMBSA 选择的 EMB 控制事件无效时, 等到计数器计数到上溢或下溢时释放 TMR6_<t>_PWMA 端口 (Cycle By Cycle 3)	RW
21:20	EMBCA	EMB 事件时端口状态设定 A	0b00: 发生 EMBSA 选择的 EMB 控制事件时, TMR6_<t>_PWMA 端口正常输出 0b01: 发生 EMBSA 选择的 EMB 控制事件时, TMR6_<t>_PWMA 端口输出高阻态 0b10: 发生 EMBSA 选择的 EMB 控制事件时, TMR6_<t>_PWMA 端口输出低电平 0b11: 发生 EMBSA 选择的 EMB 控制事件时, TMR6_<t>_PWMA 端口输出高电平	RW
19:18	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
17:16	FORCA	强制端口状态设定 A	0b0x: 禁止设置 0b10: 下周期开始, TMR6_<t>_PWMA 端口输出设定为低电平 0b11: 下周期开始, TMR6_<t>_PWMA 端口输出设定为高电平  说明 下周期是指硬件计数模式或锯齿波计数到上溢点或下溢点、三角波计数到谷点。 该寄存器位可用于实现 PWM 输出占空比 0%或 100% 的控制。	RW
15:14	CMBDCA	向下计数&&比较值 B 匹配时端口状态设定 A	0b00: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为低电平 0b01: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为高电平 0b10: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出保持先前状态 0b11: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW
13:12	CMBUCA	向上计数&&比较值 B 匹配时端口状态设定 A	0b00: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为低电平 0b01: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为高电平 0b10: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出保持先前状态 0b11: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW
11:10	CMADCA	向下计数&&比较值 A 匹配时端口状态设定 A	0b00: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为低电平 0b01: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为高电平 0b10: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出保持先前状态 0b11: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW

位/位域	标记	位名	功能描述	读写
9:8	CMAUCA	向上计数&&比较值 A 匹配时端口状态设定 A	● 0b00: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为低电平 ● 0b01: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为高电平 ● 0b10: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出保持先前状态 ● 0b11: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW
7:6	UDFCA	计数下溢端口状态设定 A	● 0b00: 计数下溢时, TMR6_<t>_PWMA 端口输出设定为低电平 ● 0b01: 计数下溢时, TMR6_<t>_PWMA 端口输出设定为高电平 ● 0b10: 计数下溢时, TMR6_<t>_PWMA 端口输出保持先前状态 ● 0b11: 计数下溢时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW
5:4	OVFCA	计数上溢端口状态设定 A	● 0b00: 计数上溢时, TMR6_<t>_PWMA 端口输出设定为低电平 ● 0b01: 计数上溢时, TMR6_<t>_PWMA 端口输出设定为高电平 ● 0b10: 计数上溢时, TMR6_<t>_PWMA 端口输出保持先前状态 ● 0b11: 计数上溢时, TMR6_<t>_PWMA 端口输出设定为反转电平	RW
3:2	STPCA	计数停止端口状态设定 A	● 0b00: 计数停止时, TMR6_<t>_PWMA 端口输出设定为低电平 ● 0b01: 计数停止时, TMR6_<t>_PWMA 端口输出设定为高电平 ● 0b10: 计数停止时, TMR6_<t>_PWMA 端口输出保持先前状态 ● 0b11: 计数停止时, TMR6_<t>_PWMA 端口输出保持先前状态	RW
1:0	STACA	计数开始端口状态设定 A	● 0b00: 计数开始时, TMR6_<t>_PWMA 端口输出设定为低电平 ● 0b01: 计数开始时, TMR6_<t>_PWMA 端口输出设定为高电平 ● 0b10: 计数开始时, TMR6_<t>_PWMA 端口输出保持先前状态 ● 0b11: 计数开始时, TMR6_<t>_PWMA 端口输出保持先前状态	RW

21.5.13 端口控制寄存器 B (TMR6_PCNBR)

Offset	Bit Position																																								
0x0158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
Reset	0x00000000																																								
Name	CAPMDB	Reserved	OUTENB	Reserved	EMBSB			EMBRB			EMBCB			Reserved	FORCB			CMBDCB			CMBUCB			CMADCB			CMAUCB			UDFCB			OVFCB			STPCB			STACB		
Access	RW		RW		RW			RW			RW				RW			RW			RW			RW			RW			RW			RW			RW			RW		

位/位域	标记	位名	功能描述	读写
31	CAPMDB	功能模式选择 B	0b0: 比较输出功能 0b1: 捕获输入功能	RW
30:29	Reserved	保留	-	-
28	OUTENB	输出使能 B	0b0: Timer6 功能时的 TMR6_<t>_PWMB 端口输出无效 0b1: Timer6 功能时的 TMR6_<t>_PWMB 端口输出有效	RW
27:26	Reserved	保留	-	-
25:24	EMBSB	EMB 事件时端口状态选择 B	0b00: 选择 EMB 的 Group0 控制事件有效 0b01: 选择 EMB 的 Group1 控制事件有效 0b10: 选择 EMB 的 Group2 控制事件有效 0b11: 选择 EMB 的 Group3 控制事件有效	RW
23:22	EMBRB	EMB 释放方式选择 B	0b00: EMBSB 选择的 EMB 控制事件无效时, 立即释放 TMR6_<t>_PWMB 端口 (One Shot) 0b01: EMBSB 选择的 EMB 控制事件无效时, 等到计数器计数到上溢时释放 TMR6_<t>_PWMB 端口 (Cycle By Cycle 1) 0b10: EMBSB 选择的 EMB 控制事件无效时, 等到计数器计数到下溢时释放 TMR6_<t>_PWMB 端口 (Cycle By Cycle 2) 0b11: EMBSB 选择的 EMB 控制事件无效时, 等到计数器计数到上溢或下溢时释放 TMR6_<t>_PWMB 端口 (Cycle By Cycle 3)	RW
21:20	EMBCB	EMB 事件时端口状态设定 B	0b00: 发生 EMBSB 选择的 EMB 控制事件时, TMR6_<t>_PWMB 端口正常输出 0b01: 发生 EMBSB 选择的 EMB 控制事件时, TMR6_<t>_PWMB 端口输出高阻态 0b10: 发生 EMBSB 选择的 EMB 控制事件时, TMR6_<t>_PWMB 端口输出低电平 0b11: 发生 EMBSB 选择的 EMB 控制事件时, TMR6_<t>_PWMB 端口输出高电平	RW
19:18	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
17:16	FORCB	强制端口状态设定 B	0b0x: 禁止设置 0b10: 下周期开始, TMR6_<t>_PWMB 端口输出设定为低电平 0b11: 下周期开始, TMR6_<t>_PWMB 端口输出设定为高电平  说明 下周期是指硬件计数模式或锯齿波计数到上溢点或下溢点、三角波计数到谷点。 该寄存器位可用于实现 PWM 输出占空比 0%或 100% 的控制。	RW
15:14	CMBDCB	向下计数&&比较值 B 匹配时端口状态设定 B	0b00: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为低电平 0b01: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为高电平 0b10: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出保持先前状态 0b11: 在向下计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW
13:12	CMBUCB	向上计数&&比较值 B 匹配时端口状态设定 B	0b00: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为低电平 0b01: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为高电平 0b10: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出保持先前状态 0b11: 在向上计数期间, 定时器计数值与 GCMBR 相等时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW
11:10	CMADCB	向下计数&&比较值 A 匹配时端口状态设定 B	0b00: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为低电平 0b01: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为高电平 0b10: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出保持先前状态 0b11: 在向下计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW

位/位域	标记	位名	功能描述	读写
9:8	CMAUCB	向上计数&&比较值 A 匹配时端口状态设定 B	● 0b00: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为低电平 ● 0b01: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为高电平 ● 0b10: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出保持先前状态 ● 0b11: 在向上计数期间, 定时器计数值与 GCMAR 相等时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW
7:6	UDFCB	计数下溢端口状态设定 B	● 0b00: 计数下溢时, TMR6_<t>_PWMB 端口输出设定为低电平 ● 0b01: 计数下溢时, TMR6_<t>_PWMB 端口输出设定为高电平 ● 0b10: 计数下溢时, TMR6_<t>_PWMB 端口输出保持先前状态 ● 0b11: 计数下溢时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW
5:4	OVFCB	计数上溢端口状态设定 B	● 0b00: 计数上溢时, TMR6_<t>_PWMB 端口输出设定为低电平 ● 0b01: 计数上溢时, TMR6_<t>_PWMB 端口输出设定为高电平 ● 0b10: 计数上溢时, TMR6_<t>_PWMB 端口输出保持先前状态 ● 0b11: 计数上溢时, TMR6_<t>_PWMB 端口输出设定为反转电平	RW
3:2	STPCB	计数停止端口状态设定 B	● 0b00: 计数停止时, TMR6_<t>_PWMB 端口输出设定为低电平 ● 0b01: 计数停止时, TMR6_<t>_PWMB 端口输出设定为高电平 ● 0b10: 计数停止时, TMR6_<t>_PWMB 端口输出保持先前状态 ● 0b11: 计数停止时, TMR6_<t>_PWMB 端口输出保持先前状态	RW
1:0	STACB	计数开始端口状态设定 B	● 0b00: 计数开始时, TMR6_<t>_PWMB 端口输出设定为低电平 ● 0b01: 计数开始时, TMR6_<t>_PWMB 端口输出设定为高电平 ● 0b10: 计数开始时, TMR6_<t>_PWMB 端口输出保持先前状态 ● 0b11: 计数开始时, TMR6_<t>_PWMB 端口输出保持先前状态	RW

21.5.14 滤波控制寄存器 (TMR6_FCNGR)

Offset	Bit Position																																	
0x015C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																										NOFICKGB		NOFIENGB		Reserved	NOFICKGA		NOFIENGA
Access																											RW		RW			RW		RW

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:5	NOFICKGB	滤波采样基准时钟选择 GB	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
4	NOFIENGB	捕获输入端口滤波 GB	0b0: 该单元 TMR6_<t>_PWMB 输入端口滤波功能无效 0b1: 该单元 TMR6_<t>_PWMB 输入端口滤波功能使能	RW
3	Reserved	保留	-	-
2:1	NOFICKGA	滤波采样基准时钟选择 GA	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
0	NOFIENGA	捕获输入端口滤波 GA	0b0: 该单元 TMR6_<t>_PWMA 输入端口滤波功能无效 0b1: 该单元 TMR6_<t>_PWMA 输入端口滤波功能使能	RW

21.5.15 有效周期寄存器 (TMR6_VPERR)

Offset	Bit Position																																				
0x0160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved											PCNTS		PCNTE		Reserved									SPPERIB		SPPERIA		Reserved								
Access												RW		RW											RW		RW										

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:18	PCNTS	有效周期选择	0b000: 有效周期选择功能无效 0b001: 每隔 1 个周期有效一次 0b010: 每隔 2 个周期有效一次 0b011: 每隔 3 个周期有效一次 0b100: 每隔 4 个周期有效一次 0b101: 每隔 5 个周期有效一次 0b110: 每隔 6 个周期有效一次 0b111: 每隔 7 个周期有效一次	RW
17:16	PCNTE	有效周期计数条件选择	0b00: 有效周期选择功能无效 0b01: 锯齿波计数上、下溢点或三角波波谷做为计数条件 0b10: 锯齿波计数上、下溢点或三角波波峰做为计数条件 0b11: 锯齿波计数上、下溢点或三角波波谷、波峰做为计数条件	RW
15:10	Reserved	保留	-	-
9	SPPERIB	专用信号有效周期选择 B	0b0: 有效周期选择功能无效 0b1: 有效周期选择功能使能	RW
8	SPPERIA	专用信号有效周期选择 A	0b0: 有效周期选择功能无效 0b1: 有效周期选择功能使能	RW
7:0	Reserved	保留	-	-

21.5.16 状态标志寄存器 (TMR6_STFLR)

Offset	Bit Position																																
0x0164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x80000000																																
Name	DIRF	Reserved			CMBF2	CMAF2	Reserved			VPERNUM			Reserved					CMSBDF	CMSBUF	CMSADF	CMSAUF	DTEF	UDFF	OVFF	CMFF	CMEF	CMDF	CMCF	CMBF	CMAF			
Access	RO				RW	RW				RO									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	DIRF	计数方向	0b0: 递减计数 0b1: 递加计数	RO
30:28	Reserved	保留	-	-
27	CMBF2	再次捕获 B	0b0: 计数匹配 B 标志 CMBF 为 1 后未发生 TMR6_<t>_PWMB 捕获完成动作 0b1: 计数匹配 B 标志 CMBF 为 1 后再次发生 TMR6_<t>_PWMB 捕获完成动作	RW
26	CMAF2	再次捕获 A	0b0: 计数匹配 A 标志 CMAF 为 1 后未发生 TMR6_<t>_PWMA 捕获完成动作 0b1: 计数匹配 A 标志 CMAF 为 1 后再次发生 TMR6_<t>_PWMA 捕获完成动作	RW
25:24	Reserved	保留	-	-
23:21	VPERNUM	周期次数	有效周期选择功能使能时, 计数后的周期次数	RO
20:13	Reserved	保留	-	-
12	CMSBDF	向下计数专用比较基准值匹配 B	0b0: 向下计数时, SCMBR 寄存器的值与计数值不相等 0b1: 向下计数时, SCMBR 寄存器的值与计数值相等	RW
11	CMSBUF	向上计数专用比较基准值匹配 B	0b0: 向上计数时, SCMBR 寄存器的值与计数值不相等 0b1: 向上计数时, SCMBR 寄存器的值与计数值相等	RW
10	CMSADF	向下计数专用比较基准值匹配 A	0b0: 向下计数时, SCMAR 寄存器的值与计数值不相等 0b1: 向下计数时, SCMAR 寄存器的值与计数值相等	RW
9	CMSAUF	向上计数专用比较基准值匹配 A	0b0: 向上计数时, SCMAR 寄存器的值与计数值不相等 0b1: 向上计数时, SCMAR 寄存器的值与计数值相等	RW
8	DTEF	死区时间错误	0b0: 未发生死区时间错误 0b1: 发生死区时间错误	RW
7	UDFF	下溢匹配	0b0: 未发生锯齿波下溢或三角波计数到谷点 0b1: 发生锯齿波下溢或三角波计数到谷点	RW

位/位域	标记	位名	功能描述	读写
6	OVFF	上溢匹配	0b0: 未发生锯齿波上溢或三角波计数到峰点 0b1: 发生锯齿波上溢或三角波计数到峰点	RW
5	CMFF	计数匹配 F	0b0: GCMFR 寄存器的值与计数值不相等 0b1: GCMFR 寄存器的值与计数值相等	RW
4	CMEF	计数匹配 E	0b0: GCMEF 寄存器的值与计数值不相等 0b1: GCMEF 寄存器的值与计数值相等	RW
3	CMDF	计数匹配 D	0b0: GCMDR 寄存器的值与计数值不相等 0b1: GCMDR 寄存器的值与计数值相等	RW
2	CMCF	计数匹配 C	0b0: GCMCR 寄存器的值与计数值不相等 0b1: GCMCR 寄存器的值与计数值相等	RW
1	CMBF	计数匹配 B	0b0: GCMBR 寄存器的值与计数值不相等, 且未发生 TMR6_<t>_PWMB 捕获完成动作 0b1: GCMBR 寄存器的值与计数值相等, 或发生 TMR6_<t>_PWMB 捕获完成动作	RW
0	CMAF	计数匹配 A	0b0: GCMAR 寄存器的值与计数值不相等, 且未发生 TMR6_<t>_PWMA 捕获完成动作 0b1: GCMAR 寄存器的值与计数值相等, 或发生 TMR6_<t>_PWMA 捕获完成动作	RW

21.5.17 硬件启动事件选择寄存器 (TMR6_HSTAR)

Offset	Bit Position																																	
0x0180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved								HSTA23	HSTA22	HSTA21	HSTA20	HSTA19	HSTA18	HSTA17	HSTA16	Reserved					HSTA11	HSTA10	HSTA9	HSTA8	STAS	Reserved			HSTA3	HSTA2	HSTA1	HSTA0	
Access									RW	RW	RW	RW	RW	RW	RW	RW						RW	RW	RW	RW	RW	STAS				RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HSTA23	硬件启动条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
22	HSTA22	硬件启动条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
21	HSTA21	硬件启动条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
20	HSTA20	硬件启动条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
19	HSTA19	硬件启动条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
18	HSTA18	硬件启动条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
17	HSTA17	硬件启动条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
16	HSTA16	硬件启动条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
15:12	Reserved	保留	-	-
11	HSTA11	硬件启动条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
10	HSTA10	硬件启动条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW

位/位域	标记	位名	功能描述	读写
9	HSTA9	硬件启动条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
8	HSTA8	硬件启动条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
7	STAS	硬件启动使能	● 0b0：硬件启动无效 ● 0b1：硬件启动有效  说明 硬件启动有效时，SSTAR 的设置无效。	RW
6:4	Reserved	保留	-	-
3	HSTA3	硬件启动条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
2	HSTA2	硬件启动条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
1	HSTA1	硬件启动条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
0	HSTA0	硬件启动条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW

21.5.18 硬件停止事件选择寄存器 (TMR6_HSTPR)

Offset	Bit Position																																
0x0184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved								HSTP23	HSTP22	HSTP21	HSTP20	HSTP19	HSTP18	HSTP17	HSTP16	Reserved					HSTP11	HSTP10	HSTP9	HSTP8	STPS	Reserved			HSTP3	HSTP2	HSTP1	HSTP0
Access									RW	RW	RW	RW	RW	RW	RW	RW						RW	RW	RW	RW	RW				RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HSTP23	硬件停止条件 23	条件: TMR6_TRIGD 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
22	HSTP22	硬件停止条件 22	条件: TMR6_TRIGD 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
21	HSTP21	硬件停止条件 21	条件: TMR6_TRIGC 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
20	HSTP20	硬件停止条件 20	条件: TMR6_TRIGC 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
19	HSTP19	硬件停止条件 19	条件: TMR6_TRIGB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
18	HSTP18	硬件停止条件 18	条件: TMR6_TRIGB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
17	HSTP17	硬件停止条件 17	条件: TMR6_TRIGA 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
16	HSTP16	硬件停止条件 16	条件: TMR6_TRIGA 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
15:12	Reserved	保留	-	-
11	HSTP11	硬件停止条件 11	条件: 内部硬件触发事件 3 有效 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
10	HSTP10	硬件停止条件 10	条件: 内部硬件触发事件 2 有效 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW

位/位域	标记	位名	功能描述	读写
9	HSTP9	硬件停止条件 9	条件：内部硬件触发事件 1 有效 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW
8	HSTP8	硬件停止条件 8	条件：内部硬件触发事件 0 有效 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW
7	STPS	硬件停止使能	0b0：硬件停止无效 0b1：硬件停止有效  说明 硬件停止有效时，SSTPR 的设定无效。	RW
6:4	Reserved	保留	-	-
3	HSTP3	硬件停止条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW
2	HSTP2	硬件停止条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW
1	HSTP1	硬件停止条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW
0	HSTP0	硬件停止条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 0b0：条件匹配时，硬件停止无效 0b1：条件匹配时，硬件停止有效	RW

21.5.19 硬件清零事件选择寄存器 (TMR6_HCLRR)

Offset	Bit Position																																
0x0188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved								HCLE23	HCLE22	HCLE21	HCLE20	HCLE19	HCLE18	HCLE17	HCLE16	Reserved					HCLE11	HCLE10	HCLE9	HCLE8	CLES	Reserved			HCLE3	HCLE2	HCLE1	HCLE0
Access									RW	RW	RW	RW	RW	RW	RW	RW						RW	RW	RW	RW	RW				RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HCLE23	硬件清零条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
22	HCLE22	硬件清零条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
21	HCLE21	硬件清零条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
20	HCLE20	硬件清零条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
19	HCLE19	硬件清零条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
18	HCLE18	硬件清零条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
17	HCLE17	硬件清零条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
16	HCLE16	硬件清零条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
15:12	Reserved	保留	-	-
11	HCLE11	硬件清零条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
10	HCLE10	硬件清零条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW

位/位域	标记	位名	功能描述	读写
9	HCLE9	硬件清零条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
8	HCLE8	硬件清零条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
7	CLES	硬件清零使能	● 0b0：硬件清零无效 ● 0b1：硬件清零有效  说明 硬件清零有效时，SCLRR 的设定无效。	RW
6:4	Reserved	保留	-	-
3	HCLE3	硬件清零条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
2	HCLE2	硬件清零条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
1	HCLE1	硬件清零条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
0	HCLE0	硬件清零条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW

21.5.20 硬件刷新事件选择寄存器 (TMR6_HUPDR)

Offset	Bit Position																																										
0x018C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0											
Reset	0x00000000																																										
Name	Reserved								HUPD23	HUPD22	HUPD21	HUPD20	HUPD19	HUPD18	HUPD17	HUPD16	Reserved								HUPD11	HUPD10	HUPD9	HUPD8	UPDS		Reserved								HUPD3	HUPD2	HUPD1	HUPD0	
Access									RW	RW	RW	RW	RW	RW	RW	RW									RW	RW	RW	RW	RW											RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HUPD23	硬件刷新条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
22	HUPD22	硬件刷新条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
21	HUPD21	硬件刷新条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
20	HUPD20	硬件刷新条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
19	HUPD19	硬件刷新条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
18	HUPD18	硬件刷新条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
17	HUPD17	硬件刷新条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
16	HUPD16	硬件刷新条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
15:12	Reserved	保留	-	-
11	HUPD11	硬件刷新条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
10	HUPD10	硬件刷新条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW

位/位域	标记	位名	功能描述	读写
9	HUPD9	硬件刷新条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
8	HUPD8	硬件刷新条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
7	UPDS	硬件刷新使能	● 0b0：硬件刷新无效 ● 0b1：硬件刷新有效  说明 硬件刷新有效时，SUPDR 的设定无效。	RW
6:4	Reserved	保留	-	-
3	HUPD3	硬件刷新条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
2	HUPD2	硬件刷新条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
1	HUPD1	硬件刷新条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW
0	HUPD0	硬件刷新条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件刷新无效 ● 0b1：条件匹配时，硬件刷新有效	RW

21.5.21 硬件捕获事件选择寄存器 A (TMR6_HCPAR)

Offset	Bit Position																																			
0x0190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved						HCPA25	HCPA24	HCPA23	HCPA22	HCPA21	HCPA20	HCPA19	HCPA18	HCPA17	HCPA16	Reserved						HCPA11	HCPA10	HCPA9	HCPA8	Reserved						HCPA3	HCPA2	HCPA1	HCPA0
Access							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW							RW	RW	RW	RW							RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	HCPA25	硬件捕获 A 条件 25	条件：内部信号 TMR6_<t>_XOR 上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
24	HCPA24	硬件捕获 A 条件 24	条件：内部信号 TMR6_<t>_XOR 上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
23	HCPA23	硬件捕获 A 条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
22	HCPA22	硬件捕获 A 条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
21	HCPA21	硬件捕获 A 条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
20	HCPA20	硬件捕获 A 条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
19	HCPA19	硬件捕获 A 条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
18	HCPA18	硬件捕获 A 条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
17	HCPA17	硬件捕获 A 条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
16	HCPA16	硬件捕获 A 条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
15:12	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
11	HCPA11	硬件捕获 A 条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
10	HCPA10	硬件捕获 A 条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
9	HCPA9	硬件捕获 A 条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
8	HCPA8	硬件捕获 A 条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
7:4	Reserved	保留	-	-
3	HCPA3	硬件捕获 A 条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
2	HCPA2	硬件捕获 A 条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
1	HCPA1	硬件捕获 A 条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW
0	HCPA0	硬件捕获 A 条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 A 无效 ● 0b1：条件匹配时，硬件捕获 A 有效	RW



说明

内部信号 TMR6_<t>_XOR 为 TMR6_<t>_PWMA 与 TMR6_<t>_PWMB 进行异或逻辑运算后得到的信号。

21.5.22 硬件捕获事件选择寄存器 B (TMR6_HCPBR)

Offset	Bit Position																																			
0x0194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved						HCPB25	HCPB24	HCPB23	HCPB22	HCPB21	HCPB20	HCPB19	HCPB18	HCPB17	HCPB16	Reserved						HCPB11	HCPB10	HCPB9	HCPB8	Reserved						HCPB3	HCPB2	HCPB1	HCPB0
Access							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW							RW	RW	RW	RW							RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	HCPB25	硬件捕获 B 条件 25	条件：内部信号 TMR6_<t>_XOR 上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
24	HCPB24	硬件捕获 B 条件 24	条件：内部信号 TMR6_<t>_XOR 上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
23	HCPB23	硬件捕获 B 条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
22	HCPB22	硬件捕获 B 条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
21	HCPB21	硬件捕获 B 条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
20	HCPB20	硬件捕获 B 条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
19	HCPB19	硬件捕获 B 条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
18	HCPB18	硬件捕获 B 条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
17	HCPB17	硬件捕获 B 条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
16	HCPB16	硬件捕获 B 条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
15:12	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
11	HCPB11	硬件捕获 B 条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
10	HCPB10	硬件捕获 B 条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
9	HCPB9	硬件捕获 B 条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
8	HCPB8	硬件捕获 B 条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
7:4	Reserved	保留	-	-
3	HCPB3	硬件捕获 B 条件 3	条件：TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
2	HCPB2	硬件捕获 B 条件 2	条件：TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
1	HCPB1	硬件捕获 B 条件 1	条件：TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW
0	HCPB0	硬件捕获 B 条件 0	条件：TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获 B 无效 ● 0b1：条件匹配时，硬件捕获 B 有效	RW



说明

内部信号 TMR6_<t>_XOR 为 TMR6_<t>_PWMA 与 TMR6_<t>_PWMB 进行异或逻辑运算后得到的信号。

21.5.23 硬件递加事件选择寄存器 (TMR6_HCUPR)

Offset	Bit Position																																			
0x0198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved								HCUP23	HCUP22	HCUP21	HCUP20	HCUP19	HCUP18	HCUP17	HCUP16	Reserved								HCUP11	HCUP10	HCUP9	HCUP8	HCUP7	HCUP6	HCUP5	HCUP4	HCUP3	HCUP2	HCUP1	HCUP0
Access									RW	RW	RW	RW	RW	RW	RW	RW									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HCUP23	硬件递加条件 23	条件: TMR6_TRIGD 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
22	HCUP22	硬件递加条件 22	条件: TMR6_TRIGD 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
21	HCUP21	硬件递加条件 21	条件: TMR6_TRIGC 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
20	HCUP20	硬件递加条件 20	条件: TMR6_TRIGC 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
19	HCUP19	硬件递加条件 19	条件: TMR6_TRIGB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
18	HCUP18	硬件递加条件 18	条件: TMR6_TRIGB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
17	HCUP17	硬件递加条件 17	条件: TMR6_TRIGA 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
16	HCUP16	硬件递加条件 16	条件: TMR6_TRIGA 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
15:12	Reserved	保留	-	-
11	HCUP11	硬件递加条件 11	条件: 内部硬件触发事件 3 有效 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
10	HCUP10	硬件递加条件 10	条件: 内部硬件触发事件 2 有效 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW

位/位域	标记	位名	功能描述	读写
9	HCUP9	硬件递加条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
8	HCUP8	硬件递加条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
7	HCUP7	硬件递加条件 7	条件：TMR6_<t>_PWMB 端口为高电平时，TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
6	HCUP6	硬件递加条件 6	条件：TMR6_<t>_PWMB 端口为高电平时，TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
5	HCUP5	硬件递加条件 5	条件：TMR6_<t>_PWMB 端口为低电平时，TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
4	HCUP4	硬件递加条件 4	条件：TMR6_<t>_PWMB 端口为低电平时，TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
3	HCUP3	硬件递加条件 3	条件：TMR6_<t>_PWMA 端口为高电平时，TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
2	HCUP2	硬件递加条件 2	条件：TMR6_<t>_PWMA 端口为高电平时，TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
1	HCUP1	硬件递加条件 1	条件：TMR6_<t>_PWMA 端口为低电平时，TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
0	HCUP0	硬件递加条件 0	条件：TMR6_<t>_PWMA 端口为低电平时，TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW



说明

硬件计数模式时请将 GCONR.CKDIV 设定为 0。

21.5.24 硬件递减事件选择寄存器 (TMR6_HCDOR)

Offset	Bit Position																																				
0x019C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved								HCDO23	HCDO22	HCDO21	HCDO20	HCDO19	HCDO18	HCDO17	HCDO16	Reserved								HCDO11	HCDO10	HCDO9	HCDO8	HCDO7	HCDO6	HCDO5	HCDO4	HCDO3	HCDO2	HCDO1	HCDO0	
Access									RW	RW	RW	RW	RW	RW	RW	RW									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	HCDO23	硬件递减条件 23	条件：TMR6_TRIGD 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
22	HCDO22	硬件递减条件 22	条件：TMR6_TRIGD 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
21	HCDO21	硬件递减条件 21	条件：TMR6_TRIGC 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
20	HCDO20	硬件递减条件 20	条件：TMR6_TRIGC 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
19	HCDO19	硬件递减条件 19	条件：TMR6_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
18	HCDO18	硬件递减条件 18	条件：TMR6_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
17	HCDO17	硬件递减条件 17	条件：TMR6_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
16	HCDO16	硬件递减条件 16	条件：TMR6_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
15:12	Reserved	保留	-	-
11	HCDO11	硬件递减条件 11	条件：内部硬件触发事件 3 有效 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
10	HCDO10	硬件递减条件 10	条件：内部硬件触发事件 2 有效 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW

位/位域	标记	位名	功能描述	读写
9	HCDO9	硬件递减条件 9	条件：内部硬件触发事件 1 有效 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
8	HCDO8	硬件递减条件 8	条件：内部硬件触发事件 0 有效 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
7	HCDO7	硬件递减条件 7	条件：TMR6_<t>_PWMB 端口为高电平时，TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
6	HCDO6	硬件递减条件 6	条件：TMR6_<t>_PWMB 端口为高电平时，TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
5	HCDO5	硬件递减条件 5	条件：TMR6_<t>_PWMB 端口为低电平时，TMR6_<t>_PWMA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
4	HCDO4	硬件递减条件 4	条件：TMR6_<t>_PWMB 端口为低电平时，TMR6_<t>_PWMA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
3	HCDO3	硬件递减条件 3	条件：TMR6_<t>_PWMA 端口为高电平时，TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
2	HCDO2	硬件递减条件 2	条件：TMR6_<t>_PWMA 端口为高电平时，TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
1	HCDO1	硬件递减条件 1	条件：TMR6_<t>_PWMA 端口为低电平时，TMR6_<t>_PWMB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
0	HCDO0	硬件递减条件 0	条件：TMR6_<t>_PWMA 端口为低电平时，TMR6_<t>_PWMB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW



说明

硬件计数模式时请将 GCONR.CKDIV 设定为 0。

21.5.25 PTO 脉冲个数寄存器 (TMR6_PTNUMR)

Offset	Bit Position																															
0x0200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PTNUM																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTNUM	脉冲个数	算法周期需要发送的脉冲个数  说明 PTNUM 有符号数，PTNUM[31]指示旋转方向： 0b0：AB 相按照 Bit0->Bit1>Bit2>Bit3>Bit0...输出 0b1：AB 相按照 Bit3->Bit2>Bit1>Bit0>Bit3...输出	RW

21.5.26 PTO 脉冲周期寄存器 (TMR6_PTPERR)

Offset	Bit Position															
0x0204	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PTPER															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PTPER	脉冲周期	脉冲周期：PTPER = 算法周期 / (PTNUM + 1)	RW

21.5.27 PTO 算法周期余数寄存器 (TMR6_PTRMDR)

Offset	Bit Position																															
0x0208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PTRMD																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTRMD	算法周期余数	算法周期余数：PTPER = 算法周期 % (PTNUM + 1)	RW

21.5.28 PTO 模式控制寄存器 (TMR6_PTCTLR)

Offset	Bit Position															
0x020C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							WORKF	Reserved				ZOUTENA	BOUTENA	AOUTENA	PTOENA
Access								RW					RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	WORKF	工作标志	0b0: PTO 处于停止状态 0b1: PTO 处于工作状态	RW
7:4	Reserved	保留	-	-
3	ZOUTENA	Z 相输出使能	0b0: 禁止 0b1: 使能	RW
2	BOUTENA	B 相输出使能	0b0: 禁止 0b1: 使能	RW
1	AOUTENA	A 相输出使能	0b0: 禁止 0b1: 使能	RW
0	PTOENA	PTO 模式使能	0b0: PTO 模式不使能, 默认为 PWM 输出方式 0b1: PTO 模式使能, 端口输出 ABZ 脉冲串	RW

21.5.29 PTO A 相输出位置寄存器 (TMR6_PTAOPR)

Offset	Bit Position															
0x0220	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												PTAOP			
Access													RW			

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3:0	PTAOP	A 相输出位置	每 Bit 代表该相数位置输出高/低电平	RW

21.5.30 PTO B 相输出位置寄存器 (TMR6_PTBOPR)

Offset	Bit Position															
0x0228	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												PTBOP			
Access													RW			

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3:0	PTBOP	B 相输出位置	每 Bit 代表该相数位置输出高/低电平	RW

21.5.31 PTO Z 相输出开始位置寄存器 (TMR6_PTZOSPR)

Offset	Bit Position																															
0x0240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	PTZOSP																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTZOSP	Z 相输出起始位置	Z 相输出高电平起始脉冲位置	RW

21.5.32 PTO Z 相输出结束位置寄存器 (TMR6_PTZOEPR)

Offset	Bit Position																															
0x0248	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	PTZOEP																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTZOEP	Z 相输出结束位置	Z 相输出高电平结束脉冲位置	RW

21.5.33 PTO Z 相输出脉宽压缩 B 寄存器 (TMR6_PTZCMPBR)

Offset	Bit Position																																
0x0264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																ZCMPBENA	ZCMPB															
Access																	RW	RW															

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	ZCMPBENA	Z 相 B 压缩使能	0b0: 禁止 0b1: 使能	RW
15:0	ZCMPB	Z 相 B 压缩时钟个数	设置 Z 输出上升沿滞后拉高和下降沿提前拉低时钟个数	RW

21.5.34 PTO 整圈脉冲个数寄存器 (TMR6_PTZPERR)

Offset	Bit Position																															
0x0268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PTZPER																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTZPER	整圈脉冲个数	一圈脉冲个数设置	RW

21.5.35 PTO 脉冲初始位置寄存器 (TMR6_PTPSPR)

Offset	Bit Position																															
0x0280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PTPSP																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PTPSP	初始位置	设置初始位置	RW

21.5.36 PTO 脉冲初始端口寄存器 (TMR6_PTPSTR)

Offset	Bit Position																															
0x0284	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														ABSTP				Reserved				SDIRIND	ZPORTST	BPORTST	APORTST						
Access															RW								RW	RW	RW	RW						

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:8	ABSTP	起始相位	启动前 AB 脉冲相位	RW
7:4	Reserved	保留	-	-
3	SDIRIND	初始旋转方向	0b0: AB 相按照 Bit0->Bit1>Bit2>Bit3>Bit0...输出 0b1: AB 相按照 Bit3->Bit2>Bit1>Bit0>Bit3...输出	RW
2	ZPORTST	Z 相端口初始状态	0b0: 设置 Z 端口输出电平为低 0b1: 设置 Z 端口输出电平为高	RW
1	BPORTST	B 相端口初始状态	0b0: 设置 B 端口输出电平为低 0b1: 设置 B 端口输出电平为高	RW
0	APORTST	A 相端口初始状态	0b0: 设置 A 端口输出电平为低 0b1: 设置 A 端口输出电平为高	RW

21.5.37 PTO 脉冲当前位置寄存器 (TMR6_PTPCPR)

Offset	Bit Position																															
0x0290	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PTPCP																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	PTPCP	当前位置	读取当前位置	RO

21.5.38 PTO 脉冲当前端口寄存器 (TMR6_PTPCTR)

Offset	Bit Position																																			
0x0294	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																RO ABCRP				Reserved				RO	CDIRIND	RO	ZPORTCR	RO	BPORTCR	RO	APORTCR				
Access																	RO												RO		RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:8	ABCRP	当前相位	当前 AB 脉冲相位	RO
7:4	Reserved	保留	-	-
3	CDIRIND	当前旋转方向	0b0: AB 相按照 Bit0->Bit1>Bit2>Bit3>Bit0...输出 0b1: AB 相按照 Bit3->Bit2>Bit1>Bit0>Bit3...输出	RO
2	ZPORTCR	Z 相端口当前状态	0b0: 当前 Z 端口输出电平为低 0b1: 当前 Z 端口输出电平为高	RO
1	BPORTCR	B 相端口当前状态	0b0: 当前 B 端口输出电平为低 0b1: 当前 B 端口输出电平为高	RO
0	APORTCR	A 相端口当前状态	0b0: 当前 A 端口输出电平为低 0b1: 当前 A 端口输出电平为高	RO

21.5.39 公共滤波控制寄存器 (TMR6_FCNT)

Offset	Bit Position																																						
0x00EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x00000000																																						
Name	Reserved																NOFICKTD	NOFIENTD	Reserved	NOFICKTC	NOFIENTC	Reserved	NOFICKTB	NOFIENTB	Reserved	NOFICKTA	NOFIENTA												
Access																	RW	RW		RW	RW		RW	RW		RW	RW		RW	RW		RW	RW		RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14:13	NOFICKTD	滤波采样基准时钟选择 TD	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
12	NOFIENTD	捕获输入端口滤波 TD	0b0: TMR6_TRIGD 端口输入滤波功能无效 0b1: TMR6_TRIGD 端口输入滤波功能使能	RW
11	Reserved	保留	-	-
10:9	NOFICKTC	滤波采样基准时钟选择 TC	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
8	NOFIENTC	捕获输入端口滤波 TC	0b0: TMR6_TRIGC 端口输入滤波功能无效 0b1: TMR6_TRIGC 端口输入滤波功能使能	RW
7	Reserved	保留	-	-
6:5	NOFICKTB	滤波采样基准时钟选择 TB	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
4	NOFIENTB	捕获输入端口滤波 TB	0b0: TMR6_TRIGB 端口输入滤波功能无效 0b1: TMR6_TRIGB 端口输入滤波功能使能	RW
3	Reserved	保留	-	-
2:1	NOFICKTA	滤波采样基准时钟选择 TA	0b00: PCLK0 0b01: PCLK0/4 0b10: PCLK0/16 0b11: PCLK0/64	RW
0	NOFIENTA	捕获输入端口滤波 TA	0b0: TMR6_TRIGA 端口输入滤波功能无效 0b1: TMR6_TRIGA 端口输入滤波功能使能	RW

21.5.40 公共软件同步启动控制寄存器 (TMR6_SSTAR)

Offset	Bit Position																																							
0x00F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved																								SSTA8		SSTA7		SSTA6		SSTA5		SSTA4		SSTA3		SSTA2		SSTA1	
Access																									RW		RW		RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SSTA8	单元 8 软件启动	写 1 启动定时器单元 8，写 0 无效 读取： ● 0b0：定时器单元 8 未启动 ● 0b1：定时器单元 8 已启动	RW
6	SSTA7	单元 7 软件启动	写 1 启动定时器单元 7，写 0 无效 读取： ● 0b0：定时器单元 7 未启动 ● 0b1：定时器单元 7 已启动	RW
5	SSTA6	单元 6 软件启动	写 1 启动定时器单元 6，写 0 无效 读取： ● 0b0：定时器单元 6 未启动 ● 0b1：定时器单元 6 已启动	RW
4	SSTA5	单元 5 软件启动	写 1 启动定时器单元 5，写 0 无效 读取： ● 0b0：定时器单元 5 未启动 ● 0b1：定时器单元 5 已启动	RW
3	SSTA4	单元 4 软件启动	写 1 启动定时器单元 4，写 0 无效 读取： ● 0b0：定时器单元 4 未启动 ● 0b1：定时器单元 4 已启动	RW
2	SSTA3	单元 3 软件启动	写 1 启动定时器单元 3，写 0 无效 读取： ● 0b0：定时器单元 3 未启动 ● 0b1：定时器单元 3 已启动	RW
1	SSTA2	单元 2 软件启动	写 1 启动定时器单元 2，写 0 无效 读取： ● 0b0：定时器单元 2 未启动 ● 0b1：定时器单元 2 已启动	RW
0	SSTA1	单元 1 软件启动	写 1 启动定时器单元 1，写 0 无效 读取： ● 0b0：定时器单元 1 未启动 ● 0b1：定时器单元 1 已启动	RW

21.5.41 公共软件同步停止控制寄存器 (TMR6_SSTPR)

Offset	Bit Position																																																																																							
0x00F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																								
Reset	0x00000000																																																																																							
Name	Reserved																								SSTP8								SSTP7								SSTP6								SSTP5								SSTP4								SSTP3								SSTP2								SSTP1							
Access																																																	WO								WO								WO								WO								WO							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SSTP8	单元 8 软件停止	写 1 停止定时器单元 8，写 0 无效	WO
6	SSTP7	单元 7 软件停止	写 1 停止定时器单元 7，写 0 无效	WO
5	SSTP6	单元 6 软件停止	写 1 停止定时器单元 6，写 0 无效	WO
4	SSTP5	单元 5 软件停止	写 1 停止定时器单元 5，写 0 无效	WO
3	SSTP4	单元 4 软件停止	写 1 停止定时器单元 4，写 0 无效	WO
2	SSTP3	单元 3 软件停止	写 1 停止定时器单元 3，写 0 无效	WO
1	SSTP2	单元 2 软件停止	写 1 停止定时器单元 2，写 0 无效	WO
0	SSTP1	单元 1 软件停止	写 1 停止定时器单元 1，写 0 无效	WO

21.5.42 公共软件同步清零控制寄存器 (TMR6_SCLRR)

Offset	Bit Position																															
0x00F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								SCLE8							
Access																									WO							
																									SCLE7							
																									WO							
																									SCLE6							
																									WO							
																									SCLE5							
																									WO							
																									SCLE4							
																									WO							
																									SCLE3							
																									WO							
																									SCLE2							
																									WO							
																									SCLE1							
																									WO							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SCLE8	单元 8 软件清零	写 1 清零定时器单元 8，写 0 无效	WO
6	SCLE7	单元 7 软件清零	写 1 清零定时器单元 7，写 0 无效	WO
5	SCLE6	单元 6 软件清零	写 1 清零定时器单元 6，写 0 无效	WO
4	SCLE5	单元 5 软件清零	写 1 清零定时器单元 5，写 0 无效	WO
3	SCLE4	单元 4 软件清零	写 1 清零定时器单元 4，写 0 无效	WO
2	SCLE3	单元 3 软件清零	写 1 清零定时器单元 3，写 0 无效	WO
1	SCLE2	单元 2 软件清零	写 1 清零定时器单元 2，写 0 无效	WO
0	SCLE1	单元 1 软件清零	写 1 清零定时器单元 1，写 0 无效	WO

21.5.43 公共软件同步刷新控制寄存器（TMR6_SUPDR）

Offset	Bit Position																																							
0x00FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved																								WO SUPD8		WO SUPD7		WO SUPD6		WO SUPD5		WO SUPD4		WO SUPD3		WO SUPD2		WO SUPD1	
Access																									WO		WO		WO		WO		WO		WO		WO		WO	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SUPD8	单元 8 软件刷新	写 1 刷新定时器单元 8，写 0 无效	WO
6	SUPD7	单元 7 软件刷新	写 1 刷新定时器单元 7，写 0 无效	WO
5	SUPD6	单元 6 软件刷新	写 1 刷新定时器单元 6，写 0 无效	WO
4	SUPD5	单元 5 软件刷新	写 1 刷新定时器单元 5，写 0 无效	WO
3	SUPD4	单元 4 软件刷新	写 1 刷新定时器单元 4，写 0 无效	WO
2	SUPD3	单元 3 软件刷新	写 1 刷新定时器单元 3，写 0 无效	WO
1	SUPD2	单元 2 软件刷新	写 1 刷新定时器单元 2，写 0 无效	WO
0	SUPD1	单元 1 软件刷新	写 1 刷新定时器单元 1，写 0 无效	WO

21.6 注意事项

- 计数器在计数中时 CPU 不能对 CNTER 寄存器执行写入动作，只能在计数器停止时对 CNTER 寄存器执行写入动作。
- GCMCR~GCMFR 在对应的缓存功能有效时，其中断输出、事件输出无效。
- 在使用双边对称互补 PWM 输出功能时，用户需计算、设定好相关的基准值和端口输出状态，保证端口为预期的输出状态。
- 在使用 PTO 功能时，写入 TMR6_PTNUMR 为 0 时，TMR6_CNTER 继续计数，PTO 保持输出不变，再次下发 TMR6_PTNUMR 非 0 时，继续输出 ABZ 相。
- 在使用 PTO 功能时，下发 TMR6_PTNUMR/PTPERR 的时间点为 TMR6_CNTER 等于 TMR6_GCMAR，TMR6_GCMAR 应设置在 TMR6_PEAR（算法周期）的 20%~80%之间。
- 在一些特定场合，对计数器的控制事件有可能同时出现，此时，计数器（CNTER）的变化控制优先级如下表所示。

表 21-90 计数器（CNTER）控制优先级

计时器控制事件		计数器（CNTER）值	优先级
CPU 对 CNTER 执行写操作		CPU 的写入值	高
刷新事件（软件刷新或硬件刷新）		UPDAR 中的值	↓
清零事件（软件清零或硬件清零）	硬件计数或锯齿波模式时	根据计数方向变为 0 或 PERAR	↓
	三角波模式时	0	↓
硬件计数		正常计数值	↓
三角波模式计数		正常计数值	↓
锯齿波模式计数		正常计数值	低

- 在一些特定场合，对 PWM 端口输出的控制事件有可能同时出现，此时，两个端口（TMR6_<t>_PWMA 和 TMR6_<t>_PWMB）的输出变化控制优先级分别如下所示。

表 21-91 PWMA 端口输出控制优先级

PWMA 端口输出控制事件		端口状态	优先级
EMB 选择的事件发生		PCNAR.EMBCA 设定状态	高
设定强制端口输出		PCNAR.FORCA 设定状态	↓
计数开始时		PCNAR.STACA 设定状态	↓
锯齿波向上计数时	CNTER=PERAR	PCNAR.OVFCA 设定状态	↓
	CNTER=GCMAR	PCNAR.CMAUCA 设定状态	↓
	CNTER=GCMBR	PCNAR.CMBUCA 设定状态	↓

PWMA 端口输出控制事件		端口状态	优先级
锯齿波向下计数时	CNTER=0	PCNAR.UDFCA 设定状态	↓
	CNTER=GCMAR	PCNAR.CMADCA 设定状态	↓
	CNTER=GCMBR	PCNAR.CMBDCA 设定状态	↓
三角波向上计数时	CNTER=GCMAR	PCNAR.CMAUCA 设定状态	↓
	CNTER=GCMBR	PCNAR.CMBUCA 设定状态	↓
	CNTER=PERAR	PCNAR.OVFCA 设定状态	↓
三角波向下计数时	CNTER=GCMAR	PCNAR.CMADCA 设定状态	↓
	CNTER=GCMBR	PCNAR.CMBDCA 设定状态	↓
	CNTER=0	PCNAR.UDFCA 设定状态	↓
计数停止时		PCNAR.STPCA 设定状态	低

表 21-92 PWMB 端口输出控制优先级

PWMB 端口输出控制事件		端口状态	优先级
EMB 选择的事件发生		PCNBR.EMBCB 设定状态	高
设定强制端口输出		PCNBR.FORCB 设定状态	↓
计数开始时		PCNBR.STACB 设定状态	↓
锯齿波向上计数时	CNTER=PERAR	PCNBR.OVFCE 设定状态	↓
	CNTER=GCMBR	PCNBR.CMBUCB 设定状态	↓
	CNTER=GCMAR	PCNBR.CMAUCB 设定状态	↓
锯齿波向下计数时	CNTER=0	PCNBR.UDFCB 设定状态	↓
	CNTER=GCMBR	PCNBR.CMBDCB 设定状态	↓
	CNTER=GCMAR	PCNBR.CMADCB 设定状态	↓
三角波向上计数时	CNTER=GCMBR	PCNBR.CMBUCB 设定状态	↓
	CNTER=GCMAR	PCNBR.CMAUCB 设定状态	↓
	CNTER=PERAR	PCNBR.OVFCE 设定状态	↓
三角波向下计数时	CNTER=GCMBR	PCNBR.CMBDCB 设定状态	↓
	CNTER=GCMAR	PCNBR.CMADCB 设定状态	↓
	CNTER=0	PCNBR.UDFCB 设定状态	↓
计数停止时		PCNBR.STPCB 设定状态	低

22 通用控制定时器（Timer4）

22.1 简介

通用控制定时器 4（Timer4）是一个用于三相电机控制的定时器模块，提供各种不同应用的三相电机控制方案。该定时器支持三角波和锯齿波两种波形模式，可生成各种 PWM 波形；支持缓存功能；支持 EMB 控制。本系列产品中搭载 3 个单元的 Timer4。

22.2 主要特性

Timer4 基本的功能及特性如下表所示。

表 22-1 Timer4 的基本功能及特性

波形模式	锯齿波、三角波
基本功能	递加、递减计数方向
	缓存功能
	通用 PWM 输出
	通用比较匹配事件
	上溢、下溢、重载计数匹配事件
	专用比较匹配事件
	软件同步启动计数
	EMB 控制
中断类型	通用比较匹配中断
	计数周期匹配中断
	重载计数匹配中断

22.3 功能说明

22.3.1 Timer4 框图

下图描述了通用控制定时器 Timer4 的基本架构。框图中所示“<t>”表示单元编号，即“<t>”为 1~3，本章节后文提到“<t>”时均指单元编号，不再赘述。

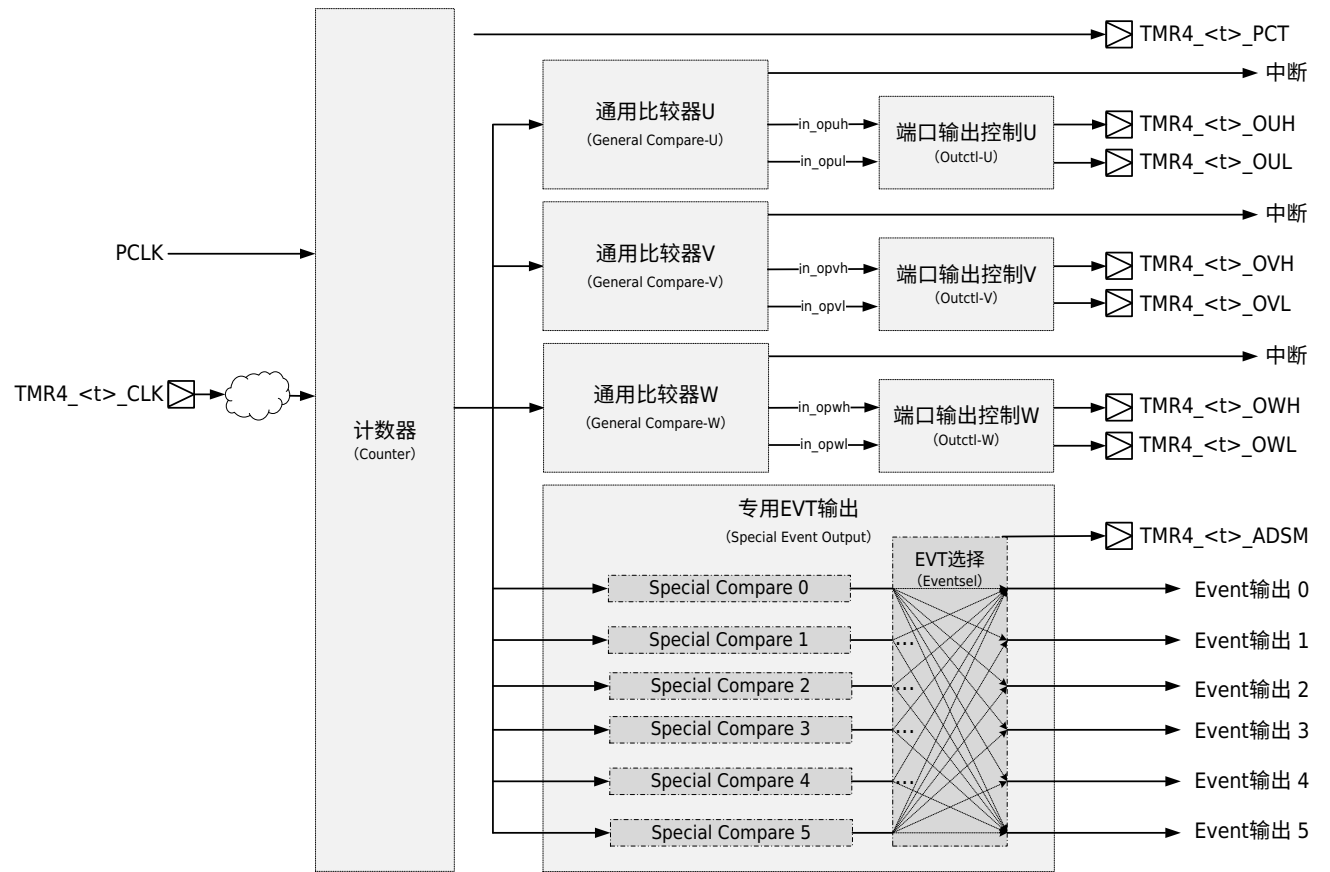


图 22-1 Timer4 基本框图

下表为 Timer4 的输入输出端口列表。

表 22-2 Timer4 端口列表

端口名	方向	功能
TMR4_<t>_CLK	输入	计数时钟输入端口
TMR4_<t>_PCT	输出	PWM 周期输出监测
TMR4_<t>_ADSM	输出	专用事件输出监测
TMR4_<t>_OUH	输出	PWM 输出端口
TMR4_<t>_OUL		
TMR4_<t>_OVH		
TMR4_<t>_OVL		
TMR4_<t>_OWH		
TMR4_<t>_OWL		

22.3.2 基本动作

22.3.2.1 波形模式

Timer4 有 2 种基本计数波形模式，锯齿波模式和三角波模式。根据 CCSR.MODE 的设定可分别实现两种波形模式。

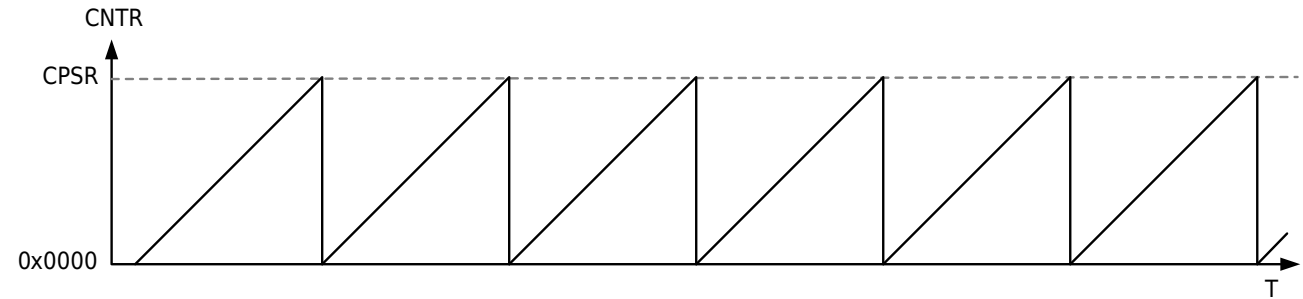


图 22-2 Timer4 锯齿波波形

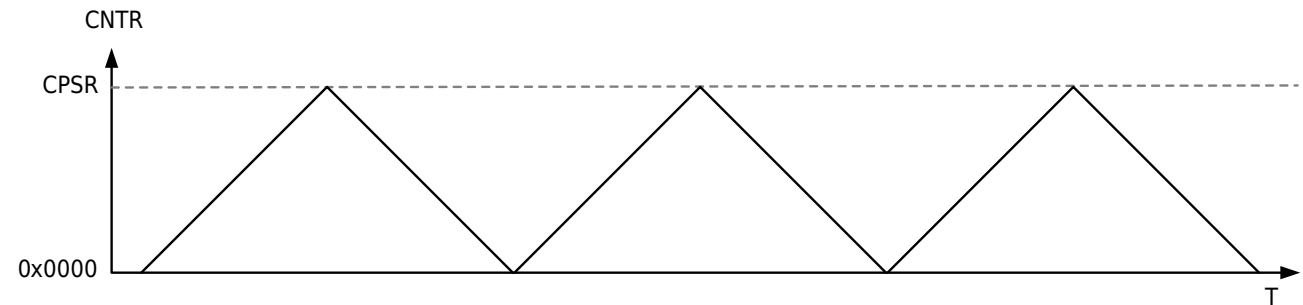


图 22-3 Timer4 三角波波形

22.3.2.2 计数动作

● 锯齿波计数操作与控制流程如下图所示：

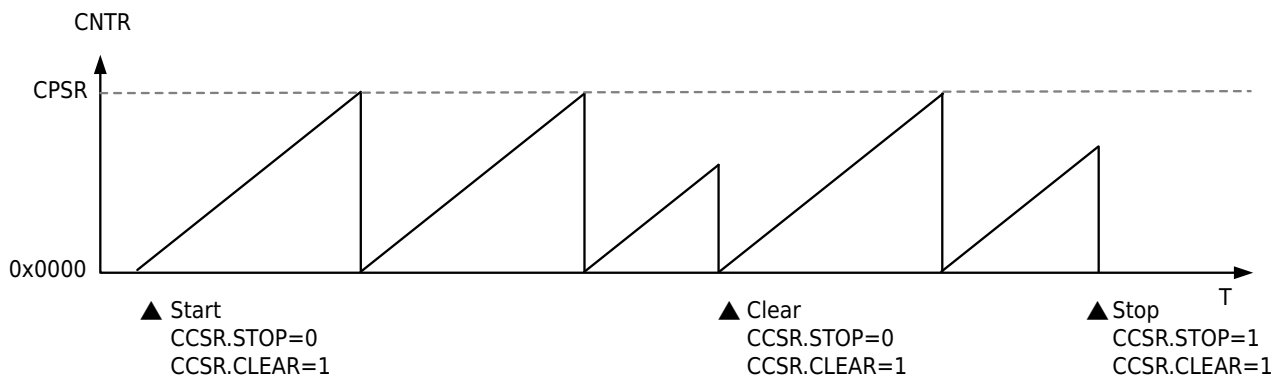


图 22-4 Timer4 锯齿波模式计数动作

1. 设置模式 CCSR.MODE=0;
2. 设置计数峰值 CPSR 寄存器;
3. 写入 CCSR.STOP=0 和 CCSR.CLEAR=1，计数器计数值 (CNTR) 被初始化到 0x0000 并启动计数操作。计数器值从 0x0000 开始递增计数，当达到峰值 (CPSR)，计数值归 0x0000，依次重复此操作;
4. 计数周期 = (CPSR + 1) × 计数时钟周期;
5. 计数过程中，写入 CCSR.STOP=0 和 CCSR.CLEAR=1 可以将计数值初始化到 0x0000 并继续进行计数操作;

6. 计数过程中，写入 `CCSR.STOP=1` 和 `CCSR.CLEAR=1` 可以将计数值初始化到 `0x0000` 并停止计数操作。

● 三角波计数操作与控制流程如下图所示：

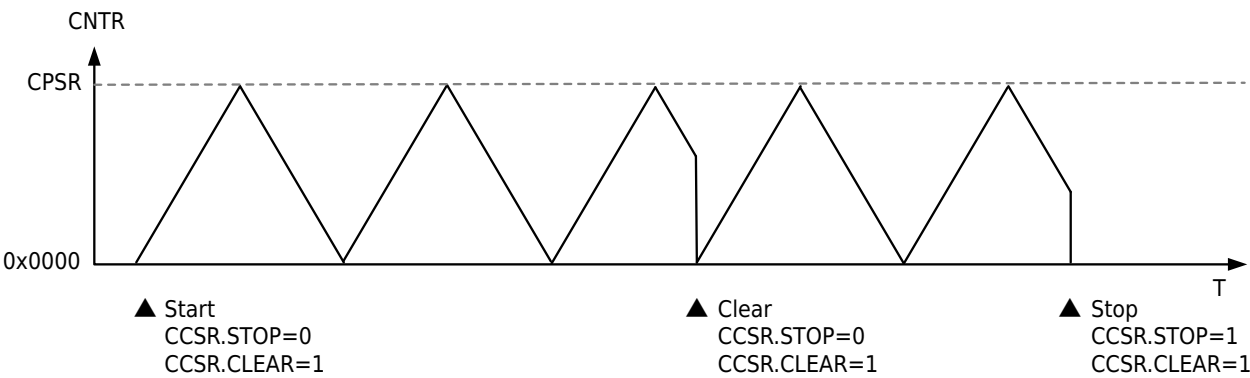


图 22-5 Timer4 三角波模式计数动作

1. 设置模式 `CCSR.MODE=1`;
2. 设置计数峰值 `CPSR` 寄存器;
3. 写入 `CCSR.STOP=0` 和 `CCSR.CLEAR=1`,计数器计数值（`CNTR`）被初始化到 `0x0000` 并启动计数操作。计数器值从 `0x0000` 开始做递增计数，在到达计数峰值之前做递增计数；当达到峰值（`CPSR`）时，计数器开始做递减计数，直到计数值归 `0x0000`；之后重新进行递增计数操作，依次重复此操作；
4. 计数周期 = $CPSR \times 2 \times$ 计数时钟周期；
5. 计数过程中，写入 `CCSR.STOP=0` 和 `CCSR.CLEAR=1` 可以将计数值初始化到 `0x0000` 并重新进行递增计数操作，之后重复上述操作；
6. 计数过程中，写入 `CCSR.STOP=1` 和 `CCSR.CLEAR=1` 可以将计数值初始化到 `0x0000` 并停止计数操作。

22.3.2.3 比较输出

比较输出功能由通用比较器 U、V 和 W 组成。通用比较器包含比较基准寄存器（`TMR4_OCCR`）、通用控制状态寄存器（`TMR4_OCSR`）、通用扩展控制寄存器（`TMR4_OCER`）、通用模式控制寄存器（`TMR4_OCMR`）。通用比较器输出内部信号，内部信号根据上述寄存器的配置，输出高低电平或 PWM。

表 22-3 通用比较器输出的内部信号

通用比较器	内部信号	描述
通用比较器 U	in_opuh	当 <code>OCSRu.OCEH=0</code> 时，由 <code>OCSRu.OCPh</code> 指定其输出高电平或低电平； 当 <code>OCSRu.OCEH=1</code> 时，由寄存器 <code>OCCRu</code> 、 <code>OCMRu</code> 以及 <code>OCERu</code> 相关位配置其 PWM 输出；
	in_opul	当 <code>OCSRu.OCEL=0</code> 时，由 <code>OCSRu.OCPL</code> 指定其输出高电平或低电平； 当 <code>OCSRu.OCEL=1</code> 时，由寄存器 <code>OCCRu</code> 、 <code>OCMRu</code> 以及 <code>OCERu</code> 相关位配置其 PWM 输出；

通用比较器	内部信号	描述
通用比较器 V	in_opvh	当 OCSRv.OCEH=0 时，由 OCSRv.OCPL 指定其输出高电平或低电平； 当 OCSRv.OCEH=1 时，由寄存器 OCCRvh、OCMRvh 以及 OCERv 相关位配置其 PWM 输出；
	in_opvl	当 OCSRv.OCEL=0 时，由 OCSRv.OCPL 指定其输出高电平或低电平； 当 OCSRv.OCEL=1 时，由寄存器 OCCRvl、OCMRvl 以及 OCERv 相关位配置其 PWM 输出；
通用比较器 W	in_opwh	当 OCSRw.OCEH=0 时，由 OCSRw.OCPL 指定其输出高电平或低电平； 当 OCSRw.OCEH=1 时，由寄存器 OCCRwh、OCMRwh 以及 OCERw 相关位配置其 PWM 输出；
	in_opwl	当 OCSRw.OCEL=0 时，由 OCSRw.OCPL 指定其输出高电平或低电平； 当 OCSRw.OCEL=1 时，由寄存器 OCCRwl、OCMRwl 以及 OCERw 相关位配置其 PWM 输出；

● 下图是锯齿波计数模式下的比较输出模块的波形输出示例。

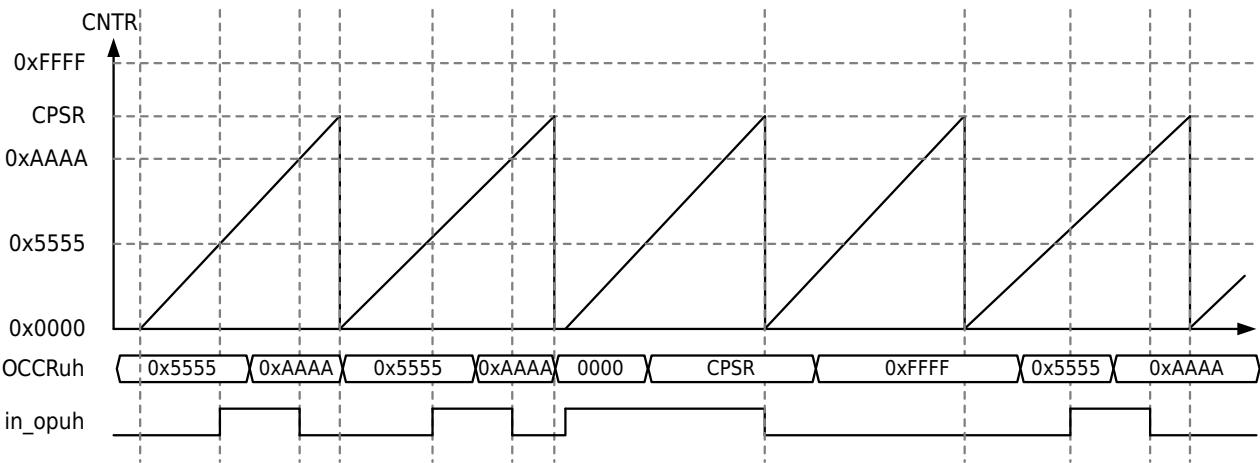


图 22-6 锯齿波模式波形输出例

● 下图是三角波计数模式下的比较输出模块的波形输出示例。

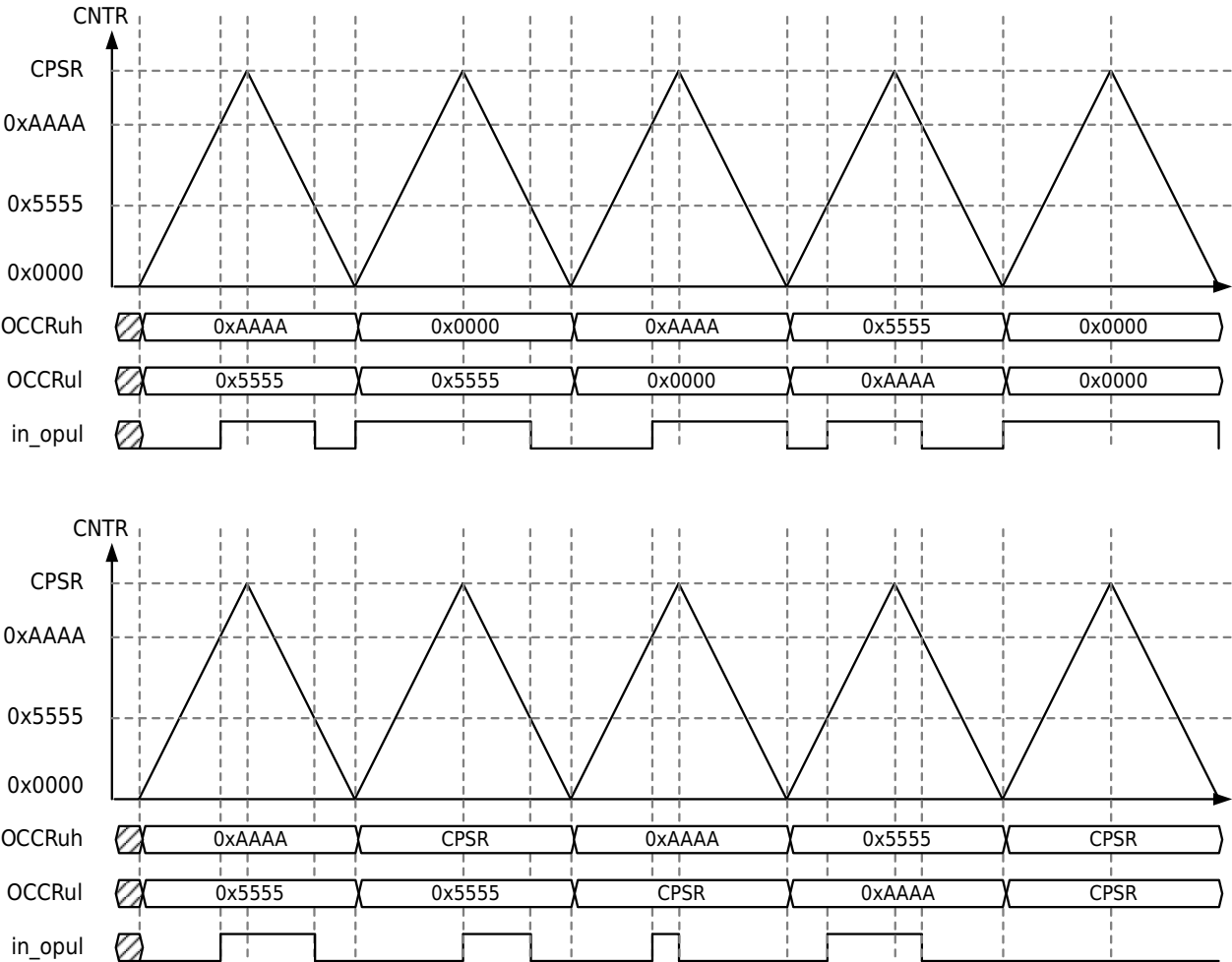


图 22-7 三角波模式波形输出例

22.3.3 缓存功能

Timer4 的周期基准寄存器 (CPSR)、通用比较基准寄存器 (OCCR)、通用模式控制寄存器 (OCMR)、专用比较基准寄存器 (SCCR) 和专用模式控制寄存器 (SCMR) 都具有缓存功能。

22.3.3.1 周期基准寄存器缓存功能

CPSR 具有缓存功能寄存器，写入的计数峰值数据先收纳到缓冲寄存器。在以下条件下，数据从缓冲器寄存器传输到 CPSR 寄存器中。

- 1. 缓冲功能禁止时 (CCSR.BUFEN=0)，写入数据立即从缓冲寄存器传输到 CPSR 寄存器。
- 2. 缓冲功能使能时 (CCSR.BUFEN=1)，在计数器停止时 (CCSR.STOP=1) 或者计数器的计数值为 "0x0000" 时，数据从缓冲寄存器传输到 CPSR 寄存器。

**注意**

从 CPSR 读取数据时，读取的并非 CPSR 缓冲寄存器的值，而是 CPSR 寄存器的值。当缓冲功能使能时，在传输完成之前所读取的值不是最近写入的值，而是上一次写入的 CPSR 值。

- 当锯齿波模式且禁止缓冲功能时修改计数峰值 CPSR 的操作：

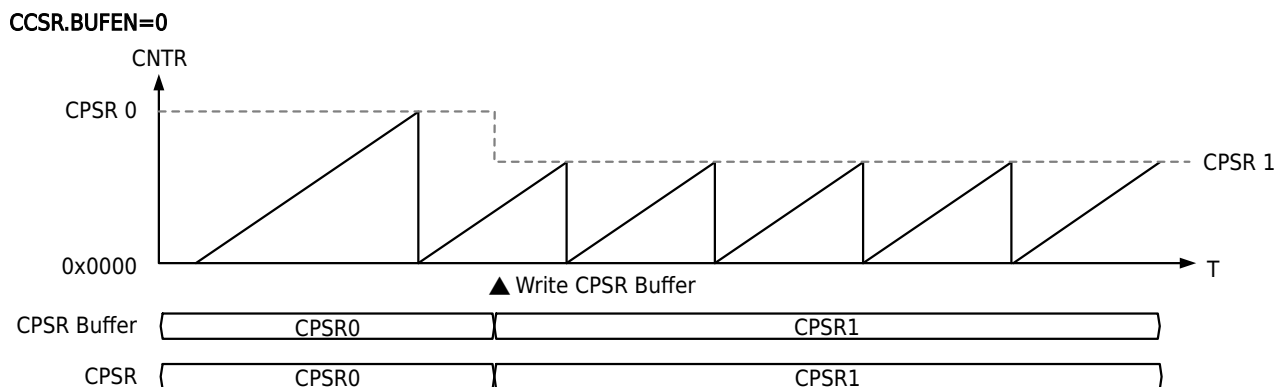


图 22-8 缓存无效时修改锯齿波计数周期

注意

当禁止缓冲功能时，写入数据立即从缓冲寄存器传输到 CPSR 计数器，计数周期在写入操作完成后立即改变。在此情况下，如果写入的值小于计数器当前计数值，计数器将持续进行递增计数，直到到达"0xFFFF"，因此，需特别注意此种状况。

- 当锯齿波模式且使能缓冲功能时修改计数峰值 CPSR 的操作:

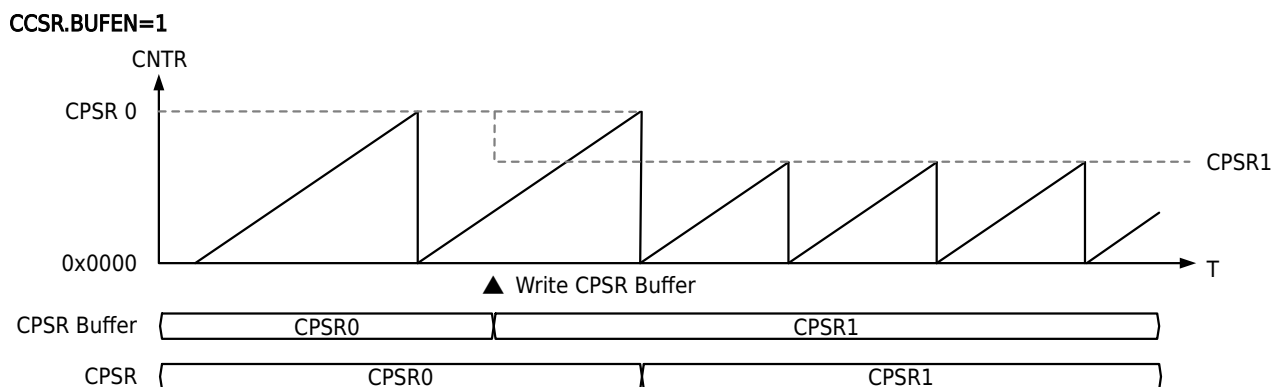


图 22-9 缓存使能时修改锯齿波计数周期

当使能缓冲功能时，在计数器停止时或者计数器的计数值为"0x0000"时，写入的数据从缓冲寄存器传输到 CPSR 寄存器。

- 当三角波模式且使能缓冲功能时修改计数峰值 CPSR 的操作:

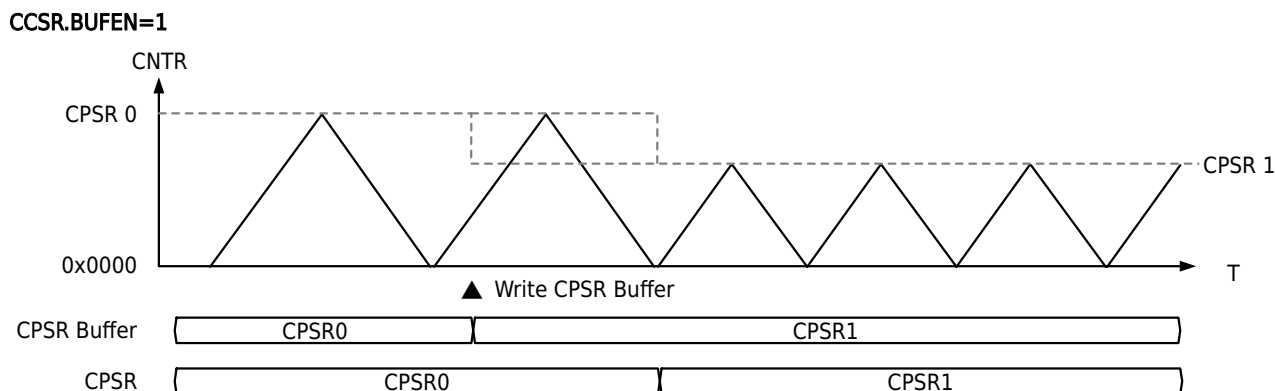


图 22-10 缓存使能时修改三角波计数周期

三角波计数模式下，当使能缓冲功能时，在计数器停止时或者计数器的下一次计数值为"0x0000"时，写入的数据从缓冲寄存器传输到 CPSR 寄存器。计数器周期的改变在写入操作完成后的下一个计数器周期开始。

22.3.3.2 通用比较寄存器缓存功能

通用比较基准值寄存器 (OCCR) 和通用模式控制寄存器 (OCMR) 均有缓冲寄存器功能，当使能缓冲功能时，在指定的传输时刻传输加载到 OCCR 和 OCMR 寄存器。OCCR 缓冲功能可以用于计数过程中同步改变比较值，OCMR 缓冲功能可以用于计数过程中计数上溢点（锯齿波）、计数谷点或峰点（三角波）同步改变内部 PWM 输出。

1. 当输出比较与计数器周期间隔响应功能的链接传输禁止时，缓冲值在设定的计数状态时加载到寄存器。此时加载情况与计数器周期间隔计数器无关。

下图所示为通用输出比较 OCCR 缓冲功能使能，计数器零值时加载 (OCER.CxBUFEN=0b01)，计数器周期间隔响应链接禁止 (OCER.LMCx=0) 时的波形 (x=L 或 H)。

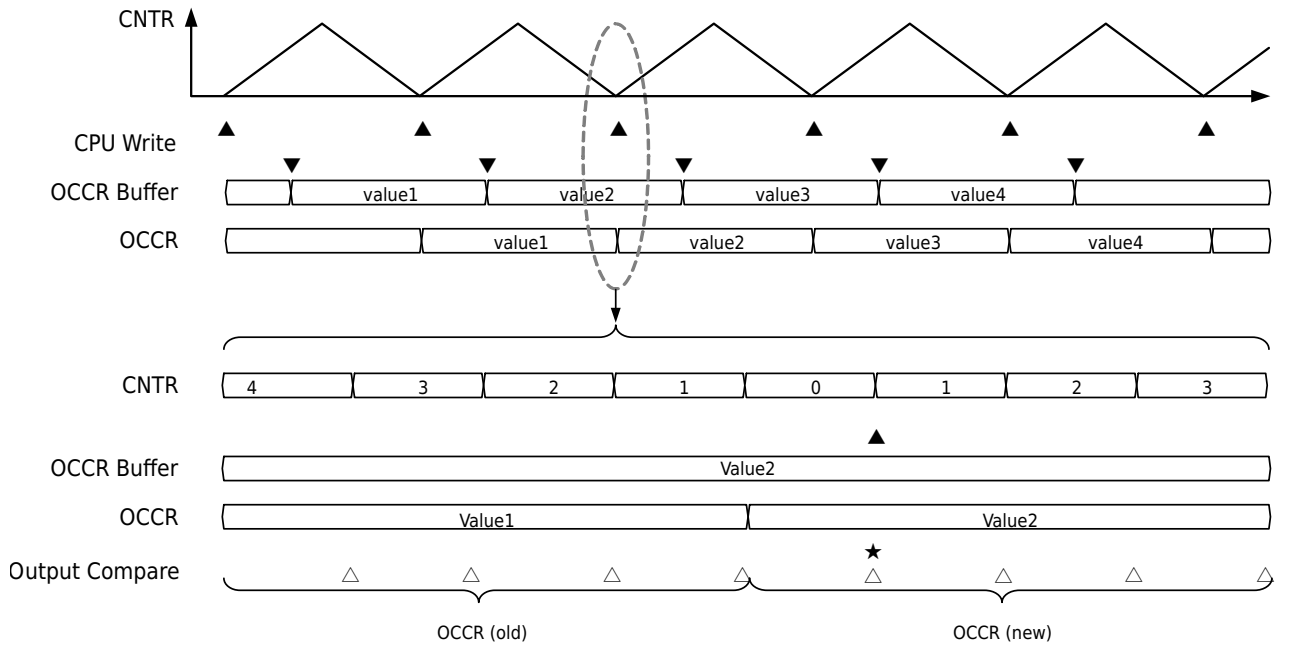


图 22-11 OCCR 缓冲数据传输 (周期间隔响应链接禁止时)

图中的上半部分是全局示意图，下半部分是传输操作时放大图。

计数器为三角波计数模式，下溢中断在标志▲（计数谷点）时刻产生。在▼时刻，CPU 改写 OCCR 寄存器，写入的数据存储在 OCCR 缓冲寄存器中。之后当计数器计数值为 0x0000 时，进行数据加载操作，数据从缓冲寄存器到 OCCR 寄存器，同时产生中断标志 IRQZF。

在△时刻，输出比较根据设定的 OCCR 寄存器值与计数值匹配事件，执行改变 PWM 输出和置位 OCSR.OCFx 位 (x=L 或 H)。在时刻★（计数值=0x0000）之后，端口输出控制根据新的 OCCR 数据执行操作。在时刻★之前根据原 OCCR 数据执行操作。

图中示意了 OCCR 缓冲寄存器在计数谷点的传输操作，OCMR 的缓冲寄存器的缓存传送操作与之类似；同理，在计数峰点的传输操作也类似。新的数据在传输时刻之后即刻生效（新写入数据将控制 PWM 输出和中断标志置位）。

2. 当计数器周期间隔响应链接使能时，缓冲值在设定的计数状态且周期间隔计数器计数值为 0 时执行缓冲寄存器传输操作。

下图所示为通用输出比较 OCCR 缓冲功能使能、计数器零值时加载 (OCER.CxBUFEN=0b01)，计数器周期间隔响应链接使能 (OCER.LMCx=1) 时的波形 (x=L 或 H)。

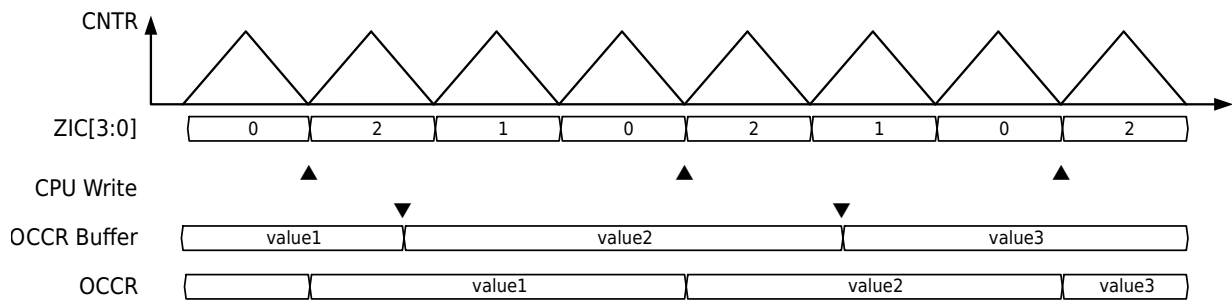


图 22-12 OCCR 缓冲数据传输（周期间隔响应链接使能）

计数器为三角波计数模式，周期间隔计数器（CVPR.ZIC）从 2 到 0 做递减计数，下溢中断在标志▲（计数谷点）时刻产生。在▼时刻，CPU 改写 OCCR 寄存器，写入的数据存储在 OCCR 缓冲寄存器中。之后当计数器计数值为 0x0000 且周期间隔计数器（CVPR.ZIC）为 0 时，进行数据加载操作，数据从缓冲寄存器到 OCCR 寄存器，同时产生中断标志 IRQZF。

图中示意了 OCCR 缓冲寄存器在计数谷点的传输操作，OCMR 的缓冲寄存器的缓存传送操作与之类似；同理，在计数峰点的传输操作也类似。新的数据在传输时刻之后即刻生效（新写入数据将控制 PWM 输出和中断标志置位）。

使用通道链接操作模式时，同时使能 OCCRuH 和 OCCRuL 缓冲功能可以产生各种不同的 PWM 输出波形。如下图所示，在输出比较寄存器 OCCRuH 和 OCCRuL 保持不变的情况下，改变 OCMR 寄存器值，使 in_opu1 输出不同的波形信号。

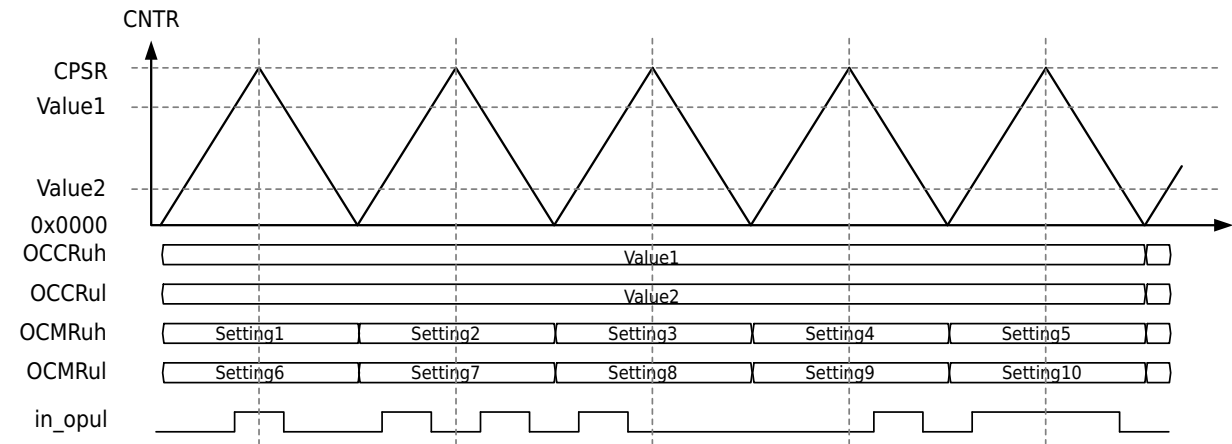


图 22-13 输出比较缓冲数据传输（OCMR 缓冲使能）

22.3.3.3 专用比较寄存器缓存功能

专用比较基准寄存器（SCCR）和在专用模式控制寄存器（SCMR）都带有缓冲寄存器。当使能缓冲功能时，CPU 写入 SCCR 和 SCMR 缓冲寄存器的值在设定的计数器状态下加载到 SCCR 和 SCMR 寄存器。

- 1. 当禁止计数器周期间隔响应链接传输时，缓冲传输操作仅与计数器状态有关，不受计数器周期间隔计数器影响。

下图所示为使能寄存器缓冲功能，禁止计数器周期间隔响应链接传输（SCSR.LMC=0），计数器零值（SCSR.BUFEN=0b01）时传送到 SCCR 和 SCMR 寄存器。

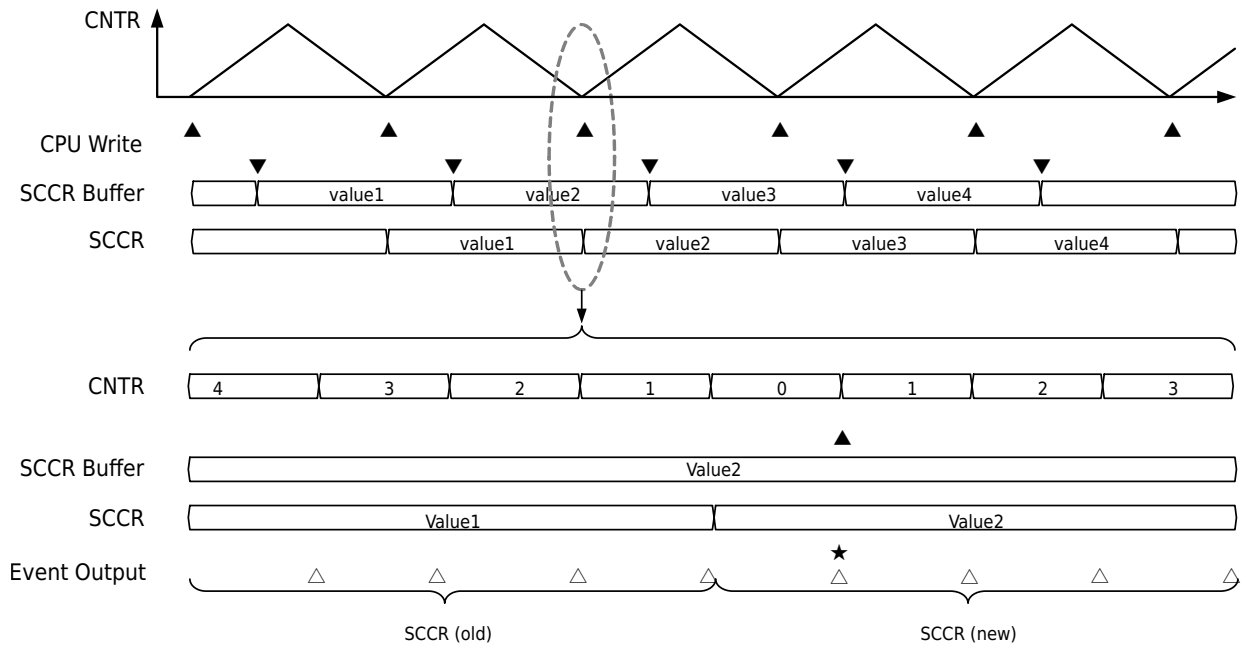


图 22-14 SCCR 缓冲传输操作（周期间隔响应链接传输禁止时）

图中的上半部分是全局示意图，下半部分缓冲寄存器传输操作的局部放大图。

计数器为三角波计数模式，下溢中断在标志▲（计数谷点）时刻产生。在▼时刻，CPU 改写 SCCR 寄存器，写入的数据存储在 SCCR 缓冲寄存器中。之后当计数器计数值为 0x0000 时，进行数据加载操作，数据从缓冲寄存器到 SCCR 寄存器，同时产生中断标志 IRQZF。

在△时刻，根据设定的 SCCR 寄存器值与计数值进行比较操作。在时刻★（计数值=0x0000）之后，专用事件输出信号根据新的 SCCR 数据执行操作。在时刻★之前根据 SCCR 数据执行操作。

图中示意了 SCCR 缓冲寄存器在计数谷点的传输操作，SCMR 的缓冲寄存器的缓存传送操作与之类似；同理，在计数峰点的传输操作也类似。新的数据在传输时刻之后即刻生效（新写入数据将专用事件输出信号和中断标志置位）。

2. 当计数器周期间隔响应链接使能时，缓冲值在设定的计数状态且周期间隔计数器计数值为 0 时执行缓冲寄存器传输操作。

下图所示为使能 SCCR 缓冲功能、计数器零值时加载（SCSR.BUFEN=0b01），计数器周期间隔响应链接传输使能时（SCSR.LMC=1）的示意图。

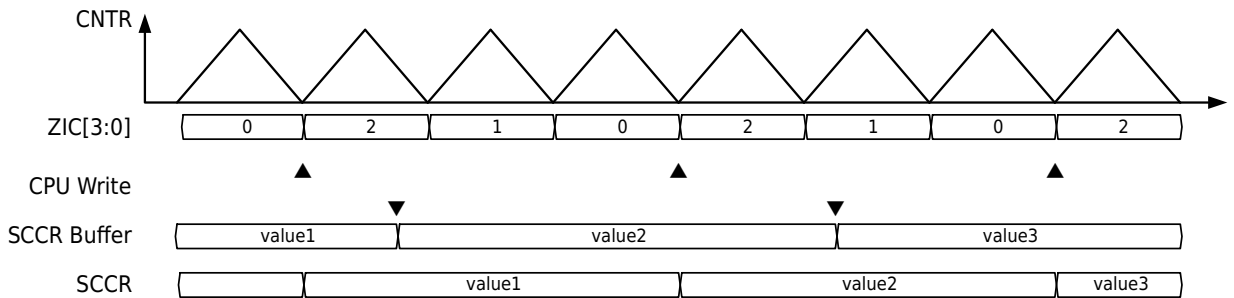


图 22-15 SCCR 缓冲传输操作（周期间隔响应链接传输使能时）

计数器为三角波计数模式，周期间隔计数器（CVPR.ZIC）从 2 到 0 做递减计数，下溢中断在标志▲（计数谷点）时刻产生。在▼时刻，CPU 改写 SCCR 寄存器，写入的数据存储在 SCCR 缓冲寄存器中。之后当计数器计数值为 0x0000 且周期间隔计数器（CVPR.ZIC）为 0 时，进行数据加载操作，数据从缓冲寄存器到 SCCR 寄存器，同时产生中断标志 IRQZF。

22.3.4 通用 PWM 输出

22.3.4.1 概述

本节后文提到的 OCCR*_H、OCCR*_L、OCMR*_H、OCMR*_L、PSCR.OE*_H、PSCR.OE*_L、PDAR*、PDBR*、PSCR.OS*_H、PSCR.OS*_L、OCER*_H时，通配符“*”代表 u、v、w；提到内部输出信号 in_op*_H、in_op*_L时，通配符“*”代表 u、v、w；提到端口 TMR4 <t> O*_H、TMR4 <t> O*_L时，通配符“*”代表 U、V、W。

端口输出控制（U、V、W）功能控制通用比较器内部信号到外部端口 TMR4_<t>_O*H、TMR4_<t>_O*L 的输出。端口输出控制包含 PWM 基本控制寄存器（TMR4_POCCR）、PWM 状态控制寄存器（TMR4_PSCR）、PWM 滤波控制寄存器（TMR4_PFSR）、PWM 死区控制寄存器（TMR4_PDAR）、重载控制状态寄存器（TMR4_RCSR）。

通用比较器输出的内部信号与端口输出控制的对应关系下表所示。

表 22-4 通用比较器输出的内部信号和端口输出控制的对应关系

通用比较器	通用比较器内部信号	端口输出控制
通用比较器 U	in_opuh	端口输出控制 U
	in_opul	
通用比较器 V	in_opvh	端口输出控制 V
	in_opvl	
通用比较器 W	in_opwh	端口输出控制 W
	in_opwl	

22.3.4.2 独立 PWM 输出

在直通模式（POCR.PWMMD=0b00）下，分别设定 OCCR*h、OCCR*l 的基准值和 OCMR*h、OCMR*l 的端口输出状态，可实现不同的 PWM 输出。此时，每个端口的 PWM 输出都独立控制。使用独立 PWM 输出模式前，需要设定 MOE、OE*H、OE*L 位（寄存器 PSCR）以使能 Timer4 的输出，详细描述请参考【PWM 状态控制寄存器（TMR4_PSCR）】。以下两张图分别是单元 1 的锯齿波和三角波下的独立 PWM 输出例。

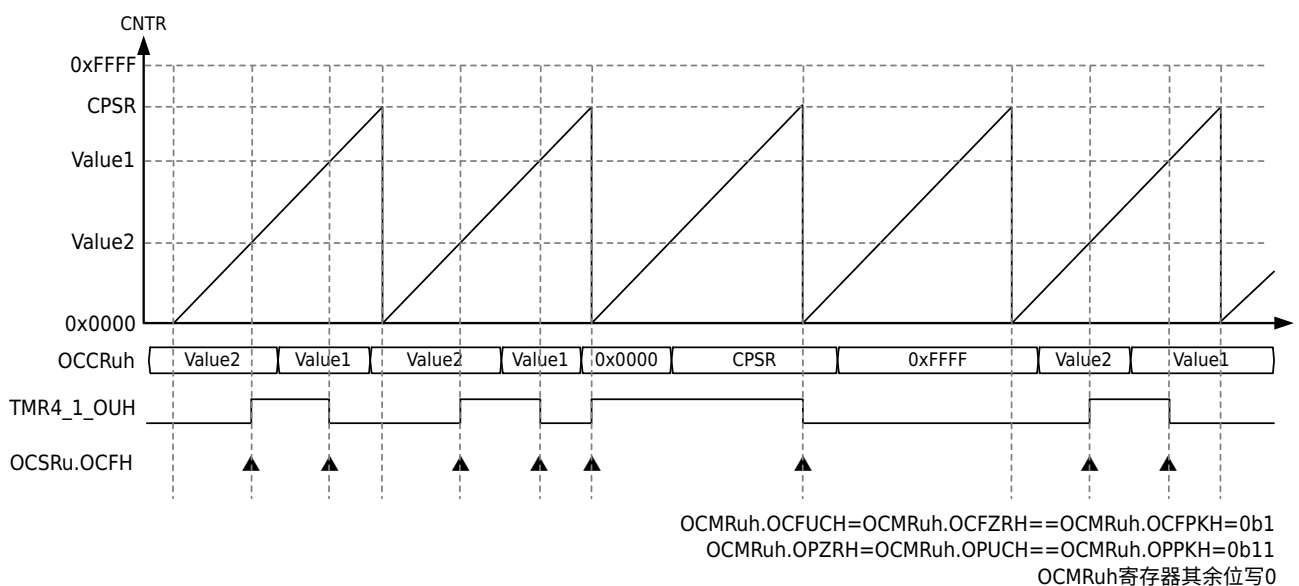
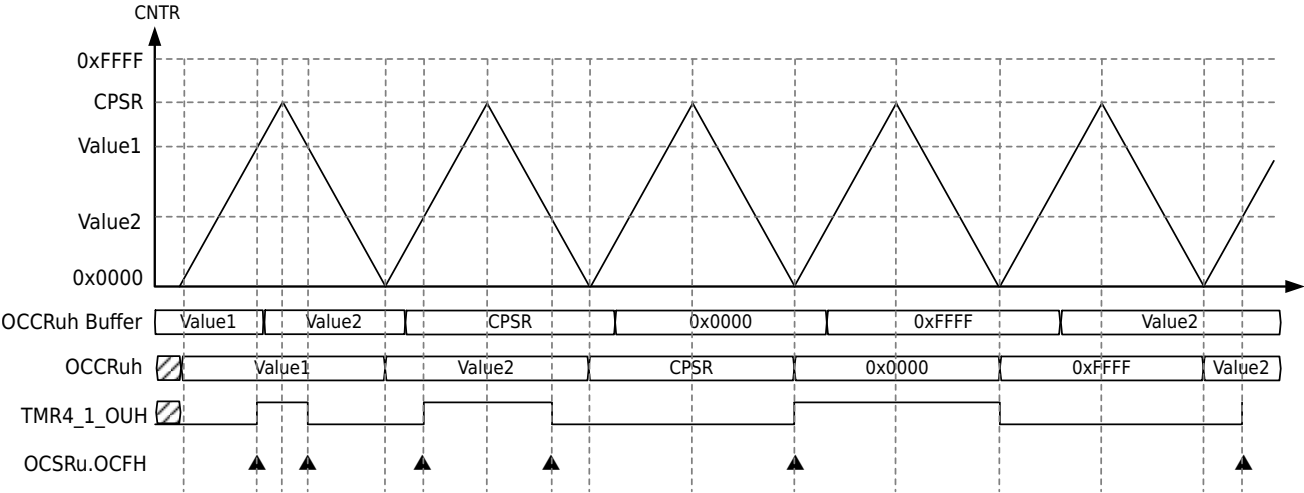


图 22-16 锯齿波独立 PWM 输出例



OCMRu.OCFUCH=OCMRu.OCFDCH=0b1 OCMRu.OPNPKH=OCMRu.OPPKH=0b00
OCMRu.OPZRH=OCMRu.OPUCH=0b01 OCMRu.OPNZRH=OCMRu.OPDCH=0b10

图 22-17 三角波独立 PWM 输出例

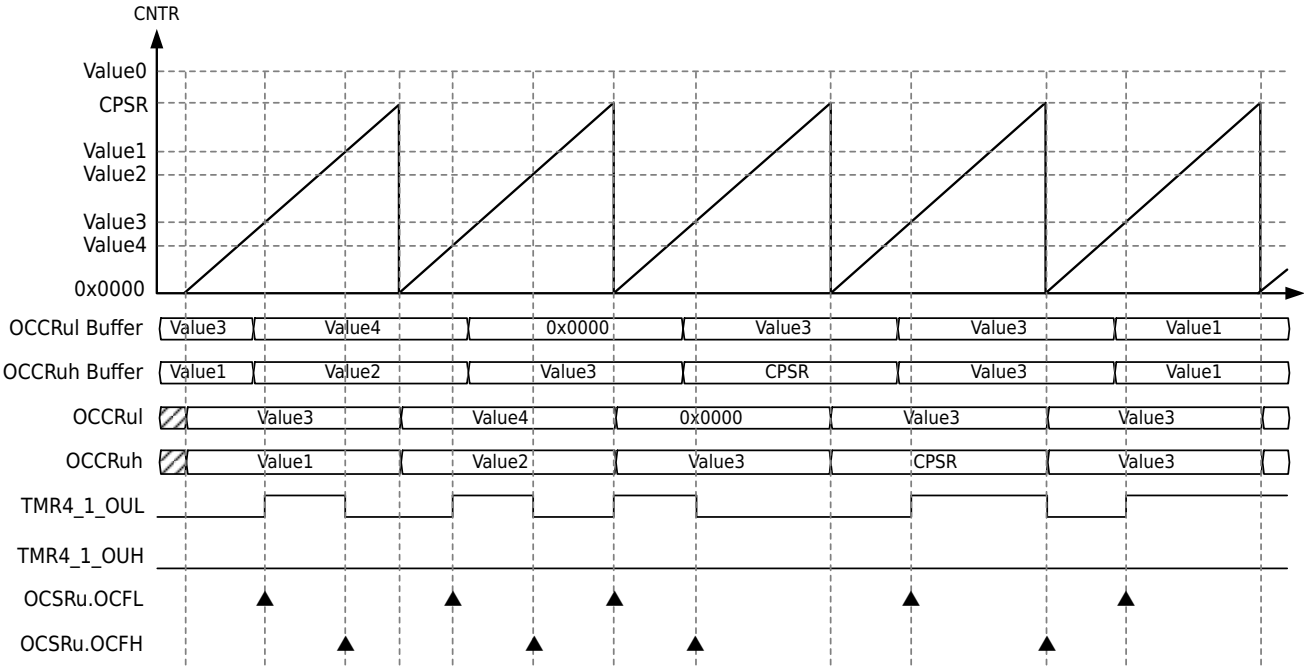


说明

直通模式是指通用比较基准寄存器（OCCR*h、OCCR*i）的值发生比较匹配所产生的内部输出信号（in_op*h、in_op*i）直接输出至对应的端口（TMR4_<t>_O*H、TMR4_<t>_O*L）上。

22.3.4.3 扩展 PWM 输出

在直通模式（POCR.PWMMD=0b00）下，TMR4_<t>_O*L 的端口输出状态也可由 OCMR*i 寄存器中的扩展位（Bit32~16）决定，该扩展位的设定与 OCCR*h 基准值相关，从而在 TMR4_<t>_O*L 端口上实现扩展 PWM 输出。使用扩展 PWM 输出模式前，需要设定 MOE、OE*H、OE*L 位（寄存器 PSCR）以使能 Timer4 的输出，详细描述请参考 PWM 状态控制寄存器（TMR4_PSCR）。下图所示是该方式下 TMR4_<t>_OUH 和 TMR4_<t>_OUL 端口的 PWM 输出。



OCMRu.OCFUCH=0b1 OCMRu.OPNPKH=OCMRu.OPUCH=OCMRu.OPPKH=0b00
OCMRu.OCFUCL=0b1 OCMRu.OPNPKL=0b00 OCMRu.OPUCL=OCMRu.OPPKL=0b11
OCMRu.EOPNPKL=OCMRu.EOPNUCL=0b11

图 22-18 锯齿波扩展 PWM 输出



说明

独立 PWM 输出方式下，TMR4_<t>_O*L 的端口状态由 OCMRxl 寄存器的 Bit15~Bit4 位决定，仅与 OCCR*I 基准值有关。

22.3.4.4 互补 PWM 输出

软件设定实现互补 PWM 输出

在直通模式 (POCR.PWMMD=0b00) 下，直接设定 OCCR*h 和 OCCR*I 基准值实现一对互补 PWM 波形输出至端口，3 组端口同样方式设定即可实现 3 相互补 PWM 输出。

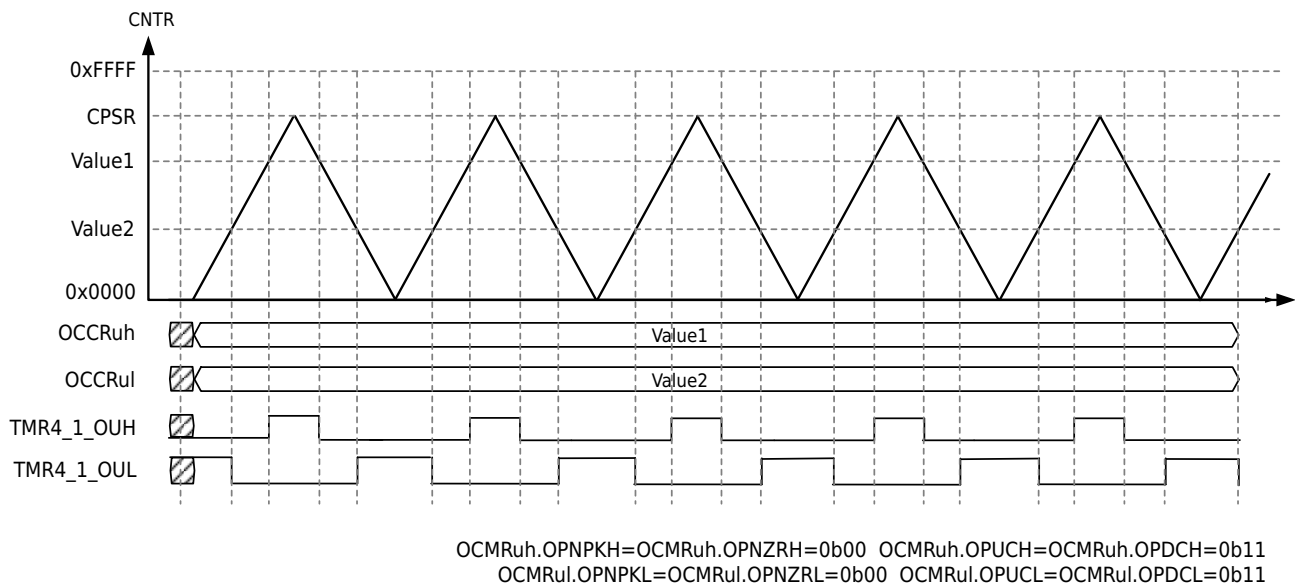


图 22-19 软件实现互补 PWM 输出

硬件设定实现互补 PWM 输出

在死区定时器模式 (POCR.PWMMD=0b01) 下，通用比较基准寄存器 (OCCR*I) 的值发生比较匹配产生的内部输出信号 (in_op*I) 和 PWM 死区控制寄存器 (PDAR/PDBR) 的设定值通过时序偏移，以硬件方式实现互补 PWM 输出。

在该模式下，TMR4_<t>_O*H 端口输出的极性与 in_op*I 相同，TMR4_<t>_O*L 端口输出的极性与 in_op*I 相反。下图是死区定时器模式下的互补 PWM 输出例。

如果检测到 in_op*I 上升沿，TMR4_<t>_O*L 输出变为低电平，死区计数器加载 PDBR*寄存器的设定值并启动递减计数，当计数值变为 0x0000 时，计数器停止并使 TMR4_<t>_O*H 输出高电平；如果检测到 in_op*I 下降沿，TMR4_<t>_O*H 输出变为低电平，死区计数器加载 PDAR*寄存器的设定值并启动递减计数，当计数值变为 0x0000 时，计数器停止并使 TMR4_<t>_O*L 输出高电平。

通过设置 PWM 死区控制寄存器 PDAR 和 PDBR，输出上升和下降变化的死区时间可以相应设定。

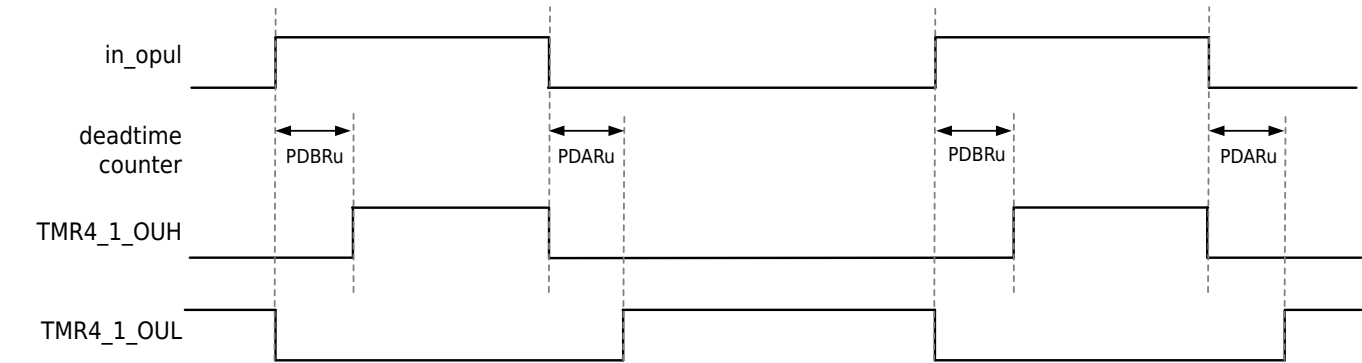


图 22-20 死区定时器模式下的互补 PWM 输出

in_opul 的高电平脉冲宽度小于 PDBR 设定的死区时间时，仅 TMR4_<t>_O*L 输出变为低电平。
TMR4_<t>_O*L 输出电平变为高的条件是当 in_opul 下降沿之后经过 PDAR 寄存器设定的死区时间之后。在此情况下，TMR4_<t>_O*H 输出将持续保持为低电平。
in_opul 的低电平脉冲宽度小于 PDAR 设定的死区时间时，仅 TMR4_<t>_O*H 输出变为低电平。
TMR4_<t>_O*H 输出电平变为高的条件是当 in_opul 上升沿之后经过 PDBR 寄存器设定的死区时间之后。在此情况下，TMR4_<t>_O*L 输出将持续保持为低电平。如下图 所示。

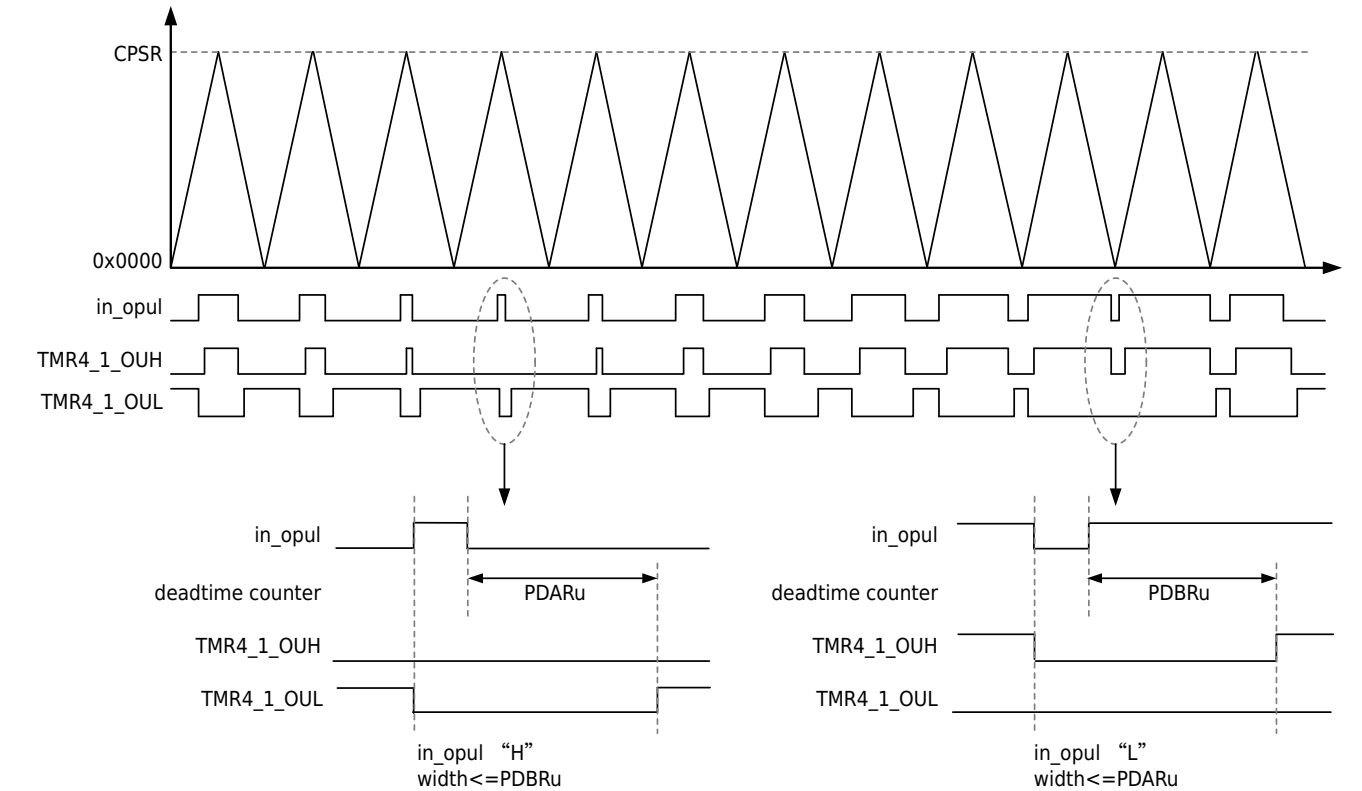


图 22-21 脉宽异常时的死区定时器模式下的波形输出

在上述硬件实现死区输出方式的基础上，还可以对内部的比较匹配信号（in_opul）进行脉冲宽度的监测，从而实现对 in_opul 信号的滤波控制。这种 in_opul 带脉宽滤波的死区输出实现方式叫死区计数器滤波模式（POCR.PWMMD=0b10）。

在死区计数器滤波模式下，滤波宽度由 PWM 滤波控制寄存器（PFSRn）的设定值决定。当 in_opul 的脉冲宽度大于寄存器 PFSRn 设定的时间时，滤波计数器将 in_opul 信号推迟 PFSR 设定的时间后输出，再经过死区定时器模式所描述的方式产生互补 PWM 输出。

如果检测到信号 in_opul 上升沿，滤波计数器加载 PFSR 寄存器值并启动计量 in_opul 的高电平宽度，当 in_opul 的高电平脉冲宽度大于寄存器 PFSR 设定的时间时，经过 PFSR 所设定时间之后，TMR4_<t>_O*L 输

出变为低电平，死区计数器加载 PDBR 寄存器的设定值并启动递减计数，当计数值变为 0x0000 时，计数器停止并使 TMR4_<t>_O*H 输出高电平；如果检测到信号 in_op**l* 下降沿，滤波计数器加载 PFSR 寄存器值并启动计量 in_op**l* 的低电平宽度，当 in_op**l* 的低电平脉冲宽度大于寄存器 PFSR 设定的时间时，经过 PFSR 所设定时间之后，TMR4_<t>_O*H 输出变为低电平，死区计数器加载 PDAR 寄存器的设定值并启动递减计数，当计数值变为 0x0000 时，计数器停止并使 TMR4_<t>_O*L 输出高电平。当 in_op**l* 的电平脉冲宽度小于寄存器 PFSR 设定的时间时，输出 TMR4_<t>_O*H 和 TMR4_<t>_O*L 将保持不变。

下图为死区计数器滤波模式下的互补 PWM 输出例。

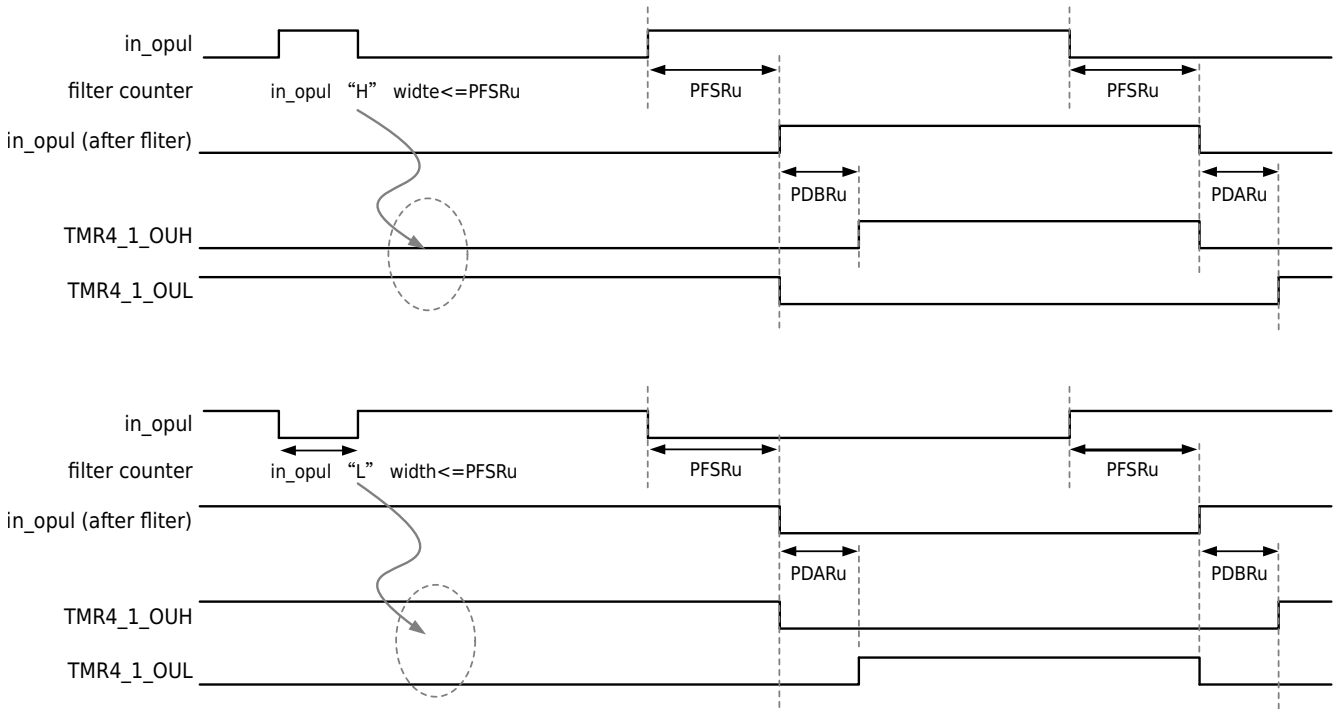


图 22-22 死区定时器滤波模式下的互补 PWM 输出

 说明

使用软件或硬件互补 PWM 输出模式前，需要设定 MOE、OE*H、OE*L 位（寄存器 PSCR）以使能 Timer4 的输出，详细描述请参考 [PWM 状态控制寄存器 \(TMR4_PSCR\)](#)。

22.3.5 周期间隔响应

下溢中断屏蔽计数器用于降低（掩盖）下溢标志位 (CCSR.IRQZF) 的被置位次数。下溢中断屏蔽计数器 (CVPR.ZIC[3:0]) 作为递减计数器操作，在开始时载入 CVPR.ZIM[3:0] 所设定的值，当 CVPR.ZIC[3:0] = 0 时，下溢标志位 (CCSR.IRQZF) 被置位为 1。

上溢中断屏蔽计数器用于降低（掩盖）上溢标志位 (CCSR.IRQPF) 的被置位次数。上溢中断屏蔽计数器 (CVPR.PIC[3:0]) 作为递减计数器操作，在开始时载入 CVPR.PIM[3:0] 所设定的值，当 CVPR.PIC[3:0] = 0 时，上溢标志位 (CCSR.IRQPF) 被置位为 1。

下图为周期间隔响应时的 IRQZF、IRQPF 的置位时序图。

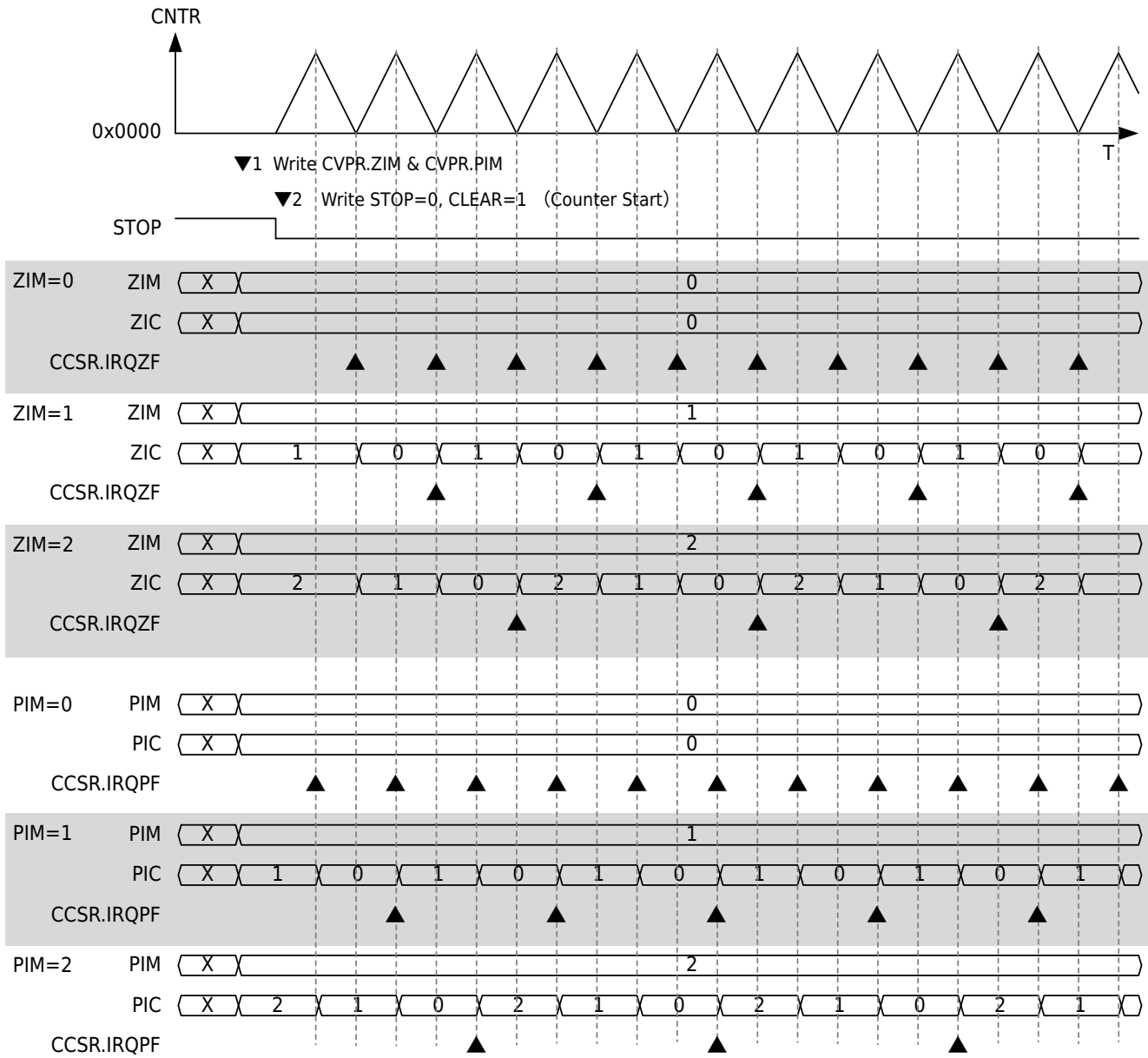


图 22-23 周期间隔响应时序图



说明

- 计数器停止时，写入 CVPR.ZIM 和 CVPR.PIM 初始化值，初始值立即反映在内部计数器（CVPR.ZIC、CVPR.PIC）。
- 初始化并启动计数器（STOP=0 且 CLEAR=1），计数器在总线复位或者从软件初始化 CLEAR=1 后从零开始进行递增计数，此刻 CCSR.IRQZF 标志不会立即置位，之后每当中断屏蔽计数器的计数值为 0x0000 且计数器的计数值为 0x0000 和 CPSR 时，标志▲为 CCSR.IRQZF 或 CCSR.IRQPF 置位的时刻。
- 在计数器运行中，写入 CVPR.ZIM 和 CVPR.PIM，该设置值不会立即反应到中断屏蔽计数器（CVPR.ZIC 和 CVPR.PIC）中。如果写入软复位（CLEAR=1），写入的 CVPR.ZIM 和 CVPR.PIM 值将立即加载为中断屏蔽计数器的初值。

专用比较基准寄存器（SCCRm）的比较匹配事件（专用事件输出）也具有周期间隔响应功能。下图为专用事件输出信号的周期间隔响应输出图。

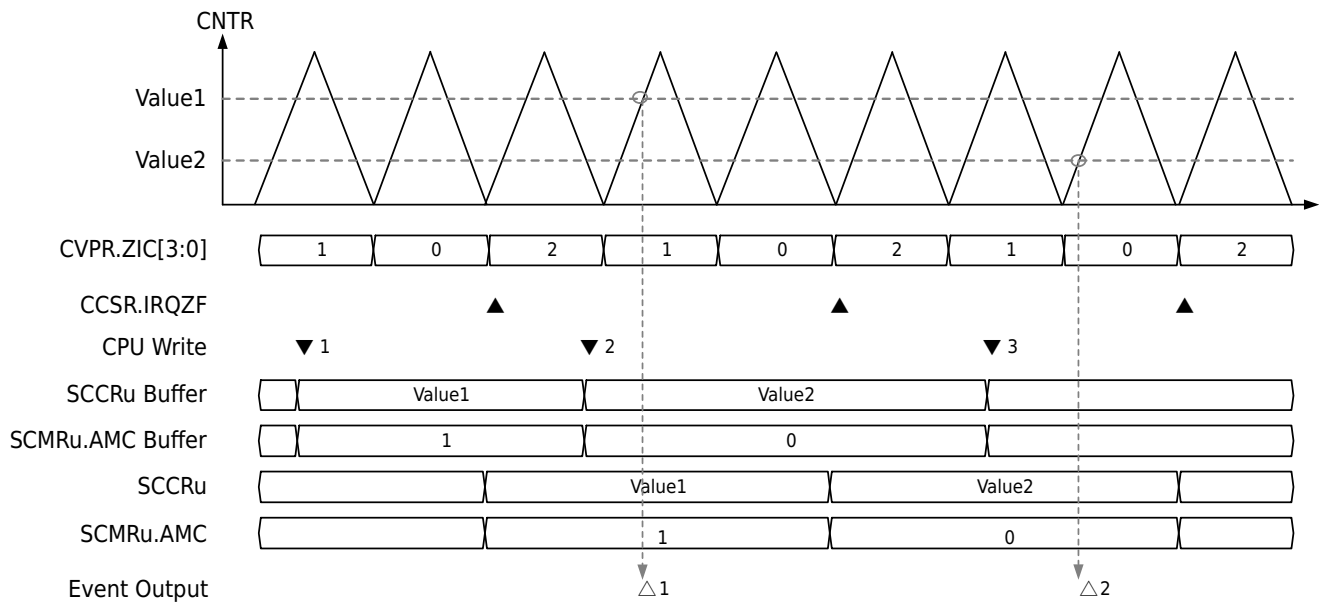


图 22-24 专用事件输出信号周期间隔响应输出

计数器是三角波计数模式，下溢中断屏蔽计数器（CVPR.ZIC）从 2-0 做递减计数。下溢中断在时刻▲产生。在时刻▼1，向 SCCRu 缓冲寄存器写入 value1。同时，MZCE=1、MPCE=0、AMC=0001 写入 SCMRu 缓冲寄存器。之后，缓冲寄存器 SCCRu 和 SCMRu 传送操作在▲时刻执行。因为 MZCE=1 且 AMC=1 且计数值=SCCRu=Value1，在时刻△1，专用事件输出信号置位。

在时刻▼2，向 SCCRu 缓冲寄存器写入 value2。同时，MZCE=1、MPCE=0、AMC=0000 写入 SCMRu 缓冲寄存器。之后，缓冲寄存器 SCCRu 和 SCMRu 传送操作在▲时刻执行。因为 MZCE=1 且 AMC=0 且计数值=SCCRu=Value2，在时刻△2，专用事件输出信号置位。

22.3.6 EMB 控制

每个 Timer4 单元都有一个输出无效事件接口，连接 EMB 模块输出的 EMB 事件。该接口上选通的异常状况事件可从 EMB 侧设定（参见《紧急刹车模块（EMB）》章节）。

各单元内 3 组 PWM 端口在正常输出期间，若监测到从 EMB 过来的 EMB 异常事件，PSCR.MOE 位由硬件异步清零，各 PWM 端口的输出状态变为预先设定好的状态。根据 PSCR.OS*H、PSCR.OS*L 的设定，该预设端口状态可以为输出高阻态、输出低电平或输出高电平。

例如，若设定 PSCR.OSUH=0b01 时，在 Timer4 的 TMR4_<t>_OUH 端口正常输出期间产生 EMB 事件，则 TMR4_<t>_OUH 端口上输出变为高阻态。

EMB 异常事件清除后，若此时 PSCR.AOE=1 则 PSCR.MOE 将由硬件自动置位，随后 PWM 端口恢复到正常输出；若此时 PSCR.AOE=0 则需要软件置位 PSCR.MOE，以恢复 PWM 端口到正常输出。

需要注意，硬件上控制 PWM 输出使能使用的是 MOE 的异步信号，当 MOE 的值发生变化后，需要等待几个时钟周期才能在总线上读取到实际的值（EMB 异常事件发生后需要等待 4 个 Timer4 总线时钟周期，EMB 异常事件解除后需要等待 6 个 Timer4 总线时钟周期）。

关于 EMB 的详细操作方法，请参见《紧急刹车模块（EMB）》章节相关描述。

22.3.7 监测输出

每个 Timer4 单元都有一个 PWM 周期输出监测端口 TMR4_<t>_PCT 和专用事件输出监测端口 TMR4_<t>_ADSM。这些监测端口输出到外部，用户可根据端口的变化信息灵活对应用系统进行控制。

PWM 周期输出监测端口 TMR4_<t>_PCT 用来监测当前计数器的计数状态。在三角波模式时，每次计数到波峰或波谷时，监测端口 TMR4_<t>_PCT 上就发生一次翻转。下图为计数方向信号的示例。

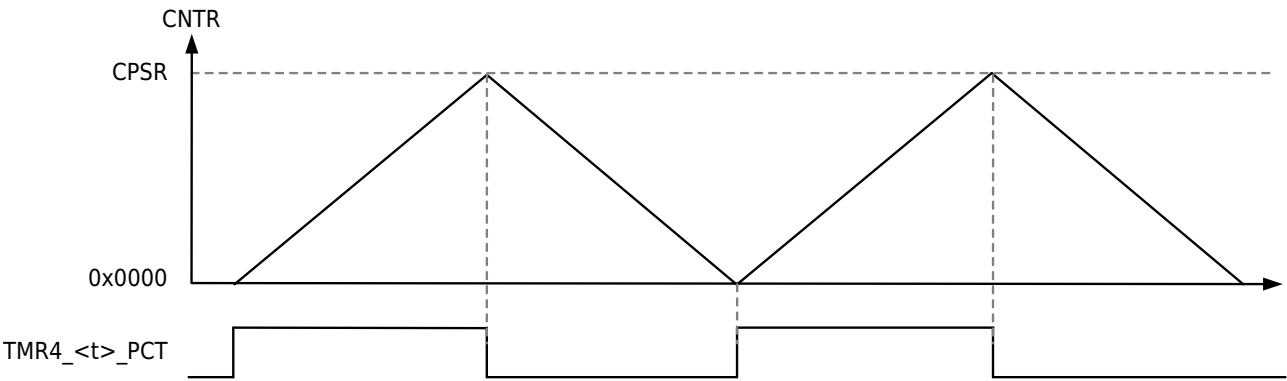


图 22-25 PWM 周期输出监测示例

专用事件输出监测端口 TMR4_<t>_ADSM 可以用来监测内部计数器的专用比较事件。通过专用扩展控制寄存器（SCER）的 SCER.EVTRS 位选择需要监测的专用比较输出通道，在该通道上的比较基准值和计数器产生比较匹配时，监测端口 TMR4_<t>_ADSM 上就会置位（输出为高）；在该通道上的计数器计数归零时（锯齿波溢出和三角波计数到波谷），监测端口 TMR4_<t>_ADSM 上就会复位（输出为低）。

需要注意，当配置寄存器 TMR4_SCSRm 同时使能向上计数 EVT 和向下计数 EVT 条件下（TMR4_SCSRm.UEN=1 且 TMR4_SCSRm.DEN=1），输出到外部端口的同步信号只对向上计数 EVT 有效。下图是比较启动模式下（SCSRm.EVTMS=0）输出到端口的同步信号示例，向上计数 EVT 有效后 TMR4_<t>_ADSM 置位，计数器下溢后 TMR4_<t>_ADSM 清零。

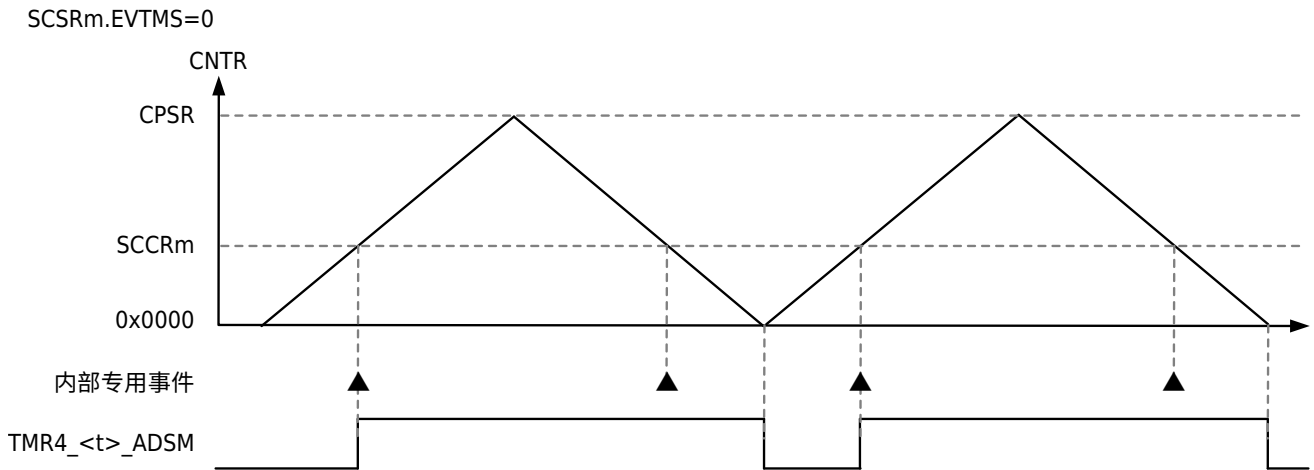


图 22-26 比较启动模式下专用事件输出 TMR4_<t>_ADSM 时序

22.3.8 同步启动

本产品所搭载的 3 个单元的 Timer4，可以实现软件同步启动。单元 1 可以同步启动单元 2 和单元 3。当被同步单元 m（m=2，3）中的 CCSR.SYNST 位设定为 1 时，该单元同步启动功能有效。此时，若软件设定单元 1 的 CCSR.STOP 位为 0，被同步单元 m 的计数器与单元 1 的计数器同步启动计数。另外，产品还配置了一个计数同步使能寄存器 TMR_SYNENRx（x=1~2），用于同步启动产品所搭载的 Timer（包括 Timer0、Timer2、Timer4、Timer6、TimerA）。使用方法和寄存器说明请参见《通用定时（Timer0）》章节描述。请不要在本定时器异步计数的情况下使用该功能。

22.3.9 中断和事件

通用计数比较匹配中断及事件

通用比较基准值寄存器（OCCRm）共计 6 个，可分别与计数值比较产生比较匹配有效信号。计数比较匹配时，通用控制状态寄存器（OCSRn）中的 OCSRn.OCFH、OCSRn.OCFL 位分别会被置为 1。此时若设定

OCSRn.OCIEH、OCSRn.OCIEL 使能中断，则对应的中断请求（TMR4_<t>_GCMmn，m=U、V、W；n=H、L）也会被触发。

通用计数比较匹配事件可以用于触发别的模块，事件编号请参见《中断控制器（INTC）》章节“中断事件表”。

计数周期匹配中断及事件

锯齿波递加计数至上溢点、三角波计数至谷点或三角波计数至峰点时，控制状态寄存器（CCSR）的 CCSR.IRQPF 或 CCSR.IRQZF 位会被置为 1。此时若设置 CCSR.IRQPEN 或 CCSR.IRQZEN 位使能中断，则在对应的时间点可触发计数周期匹配中断（TMR4_<t>_GOVF 和 TMR4_<t>_GUDF）。

计数周期匹配事件可以用于触发别的模块，事件编号请参见《中断控制器（INTC）》章节“中断事件表”。

重载计数匹配中断及事件

在重载功能有效时，TMR4_PFSRn 作为重载定时器的周期计数值，使能计数时从寄存器 TMR4_PFSRn 重载为计数器初始值，执行递减操作，完成一个周期后，产生重载计数中断请求，重载控制状态寄存器（RCSR）中的 RCSR.RTIFU、RCSR.RTIFV、RCSR.RTIFW 位分别会被置为 1。此时若设定 RCSR.RTIDU、RCSR.RTIDV、RCSR.RTIDW 中断屏蔽无效，则对应的重载计数匹配中断请求（TMR4_<t>_RLOm，m=U、V、W）也会被触发。

重载计数匹配事件可以用于触发别的模块，事件编号请参见《中断控制器（INTC）》章节“中断事件表”。

专用比较匹配事件

Timer4 的 6 个专用比较基准寄存器（SCCRm）对应产生 6 个专用事件输出信号可以用于选择触发别的模块，如启动 ADC 等。

时钟计数过程中，若专用比较基准值（SCCRm）发生计数比较匹配事件（TMR4_<t>_SCM0~5），会产生相应的有效请求信号，该请求信号可以被配置到任意的事件 EVT 输出信号上（由 SCSR.EVTOS 位设定）用于触发其他模块。

该事件请求信号的输出可以选择比较启动模式或延时启动模式。在比较启动模式时（SCSR.EVTMS=0），产生 SCCR 的计数比较匹配事件后，专用事件输出信号直接输出；在延时启动模式时（SCSR.EVTMS=1），产生 OCCR*h 或 OCCR*i（由 SCSR.EVTDS 位选择）的计数比较匹配事件后，经过 SCCR 设定的基准周期时间后，专用事件输出信号输出。下图是延时启动模式下的专用事件输出信号的请求输出例。

专用比较匹配事件可以用于触发别的模块，事件编号请参见《中断控制器（INTC）》章节“中断事件表”。

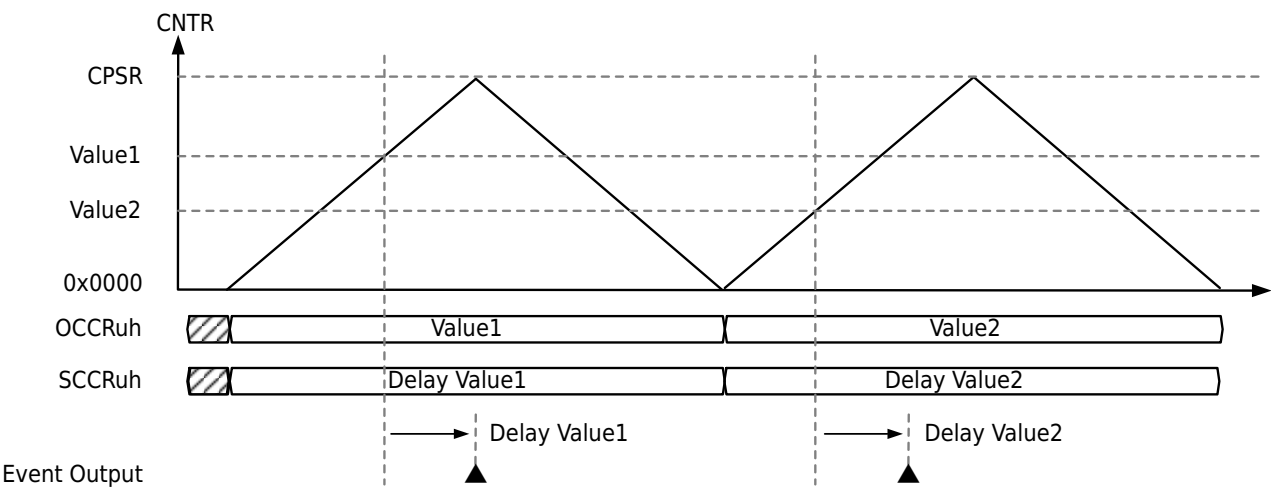


图 22-27 延时启动模式下专用事件输出信号的输出时序



说明

在延迟计数运行中，如果再次发生 OCCR 与计数器匹配事件，延迟计数器重新加载计数值并重新进行递减计数。因此，如果 OCCR 匹配事件时间间隔小于设定的延迟时间 SCCR 时，专用事件输出的请求信号可能一直不会产生。

22.4 寄存器

22.4.1 寄存器总表

- Timer4_1 基地址：0x40038000
- Timer4_2 基地址：0x40038400
- Timer4_3 基地址：0x40038800



说明

Timer4 为定时器名称，Timer4_<t>（<t>：1~3）为该定时器单元。

表 22-5 通用控制定时器（Timer4）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0046	TMR4_CNTR	计数值寄存器	16
0x0042	TMR4_CPSR	周期基准寄存器	16
0x0048	TMR4_CCSR	控制状态寄存器	16
0x004A	TMR4_CVPR	有效周期寄存器	16
0x0002	TMR4_OCCRuh	通用比较基准寄存器 UH	16
0x0006	TMR4_OCCRul	通用比较基准寄存器 UL	16
0x000A	TMR4_OCCRvh	通用比较基准寄存器 VH	16
0x000E	TMR4_OCCRvl	通用比较基准寄存器 VL	16
0x0012	TMR4_OCCRwh	通用比较基准寄存器 WH	16
0x0016	TMR4_OCCRwl	通用比较基准寄存器 WL	16
0x0018	TMR4_OCSRu	通用控制状态寄存器 U	16
0x001C	TMR4_OCSRv	通用控制状态寄存器 V	16
0x0020	TMR4_OCSRw	通用控制状态寄存器 W	16
0x001A	TMR4_OCERu	通用扩展控制寄存器 U	16
0x001E	TMR4_OCERv	通用扩展控制寄存器 V	16
0x0022	TMR4_OCERw	通用扩展控制寄存器 W	16
0x0024	TMR4_OCMRuh	通用模式控制寄存器 UH	16
0x0028	TMR4_OCMRul	通用模式控制寄存器 UL	32
0x002C	TMR4_OCMRvh	通用模式控制寄存器 VH	16
0x0030	TMR4_OCMRvl	通用模式控制寄存器 VL	32
0x0034	TMR4_OCMRwh	通用模式控制寄存器 WH	16
0x0038	TMR4_OCMRwl	通用模式控制寄存器 WL	32

偏移地址	寄存器	描述	位宽
0x00B2	TMR4_SCCRuh	专用比较基准寄存器 UH	16
0x00B6	TMR4_SCCRul	专用比较基准寄存器 UL	16
0x00BA	TMR4_SCCRvh	专用比较基准寄存器 VH	16
0x00BE	TMR4_SCCRvl	专用比较基准寄存器 VL	16
0x00C2	TMR4_SCCRwh	专用比较基准寄存器 WH	16
0x00C6	TMR4_SCCRwl	专用比较基准寄存器 WL	16
0x00C8	TMR4_SCSRuh	专用控制状态寄存器 UH	16
0x00CC	TMR4_SCSRul	专用控制状态寄存器 UL	16
0x00D0	TMR4_SCSRvh	专用控制状态寄存器 VH	16
0x00D4	TMR4_SCSRvl	专用控制状态寄存器 VL	16
0x00D8	TMR4_SCSRwh	专用控制状态寄存器 WH	16
0x00DC	TMR4_SCSRwl	专用控制状态寄存器 WL	16
0x00CA	TMR4_SCMRuh	专用模式控制寄存器 UH	16
0x00CE	TMR4_SCMRul	专用模式控制寄存器 UL	16
0x00D2	TMR4_SCMRvh	专用模式控制寄存器 VH	16
0x00D6	TMR4_SCMRvl	专用模式控制寄存器 VL	16
0x00DA	TMR4_SCMRwh	专用模式控制寄存器 WH	16
0x00DE	TMR4_SCMRwl	专用模式控制寄存器 WL	16
0x00E4	TMR4_SCER	专用扩展控制寄存器	16
0x0098	TMR4_POCRu	PWM 基本控制寄存器 U	16
0x009C	TMR4_POCRv	PWM 基本控制寄存器 V	16
0x00A0	TMR4_POCRw	PWM 基本控制寄存器 W	16
0x00E0	TMR4_PSCR	PWM 状态控制寄存器	32
0x0082	TMR4_PFSRu	PWM 滤波控制寄存器 U	16
0x008A	TMR4_PFSRv	PWM 滤波控制寄存器 V	16
0x0092	TMR4_PFSRw	PWM 滤波控制寄存器 W	16
0x0084	TMR4_PDARu	PWM 死区控制寄存器 AU	16
0x0086	TMR4_PDBRu	PWM 死区控制寄存器 BU	16
0x008C	TMR4_PDARv	PWM 死区控制寄存器 AV	16
0x008E	TMR4_PDBRv	PWM 死区控制寄存器 BV	16

偏移地址	寄存器	描述	位宽
0x0094	TMR4_PDARw	PWM 死区控制寄存器 AW	16
0x0096	TMR4_PDBRw	PWM 死区控制寄存器 BW	16
0x00A4	TMR4_RCSR	重载控制状态寄存器	16



说明

本节后续寄存器详细描述中，m=uh、ul、vh、vl、wh、wl，n=u、v、w。m 所指寄存器分别在功能实现时对应端口 TMR4_<t>_OUH、TMR4_<t>_OUL、TMR4_<t>_OVH、TMR4_<t>_OVL、TMR4_<t>_OWH、TMR4_<t>_OWL 的输出控制等；n 所指寄存器分别在功能实现时对应端口 TMR4_<t>_OUx、TMR4_<t>_OVx、TMR4_<t>_OWx 的输出控制等，其中 x=H 或 L，且 H 或 L 的具体控制在这些寄存器中有相对应的对称位。

22.4.2 计数值寄存器 (TMR4_CNTR)

Offset	Bit Position															
0x0046	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	CNTR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	CNTR	计数器当前值	在计数停止时，通过向该寄存器写入值，可初始化计数器计数值 在计数中时，该位指示当前计数器计数值  说明 在计数中时，不能向该寄存器写入值。	RW

22.4.3 周期基准寄存器 (TMR4_CPSR)

Offset	Bit Position															
0x0042	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFF															
Name	CPSR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	CPSR	通用周期基准值	计数器的计数周期值  说明 从本地地址区域读取数据时，读取的并非缓冲器寄存器的值，而是 CPSR 寄存器的值。	RW

22.4.4 控制状态寄存器 (TMR4_CCSR)

Offset	Bit Position															
0x0048	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0040															
Name	ECKEN	IRQZF	IRQZEN	SYNST	Reserved		IRQPF	IRQPEN	BUFEN	STOP	MODE	CLEAR			CKDIV	
Access	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW			RW	

位/位域	标记	位名	功能描述	读写
15	ECKEN	时钟源选择	0b0: 内部 PCLK0 时钟 0b1: 外部 TMR4_<t>_CLK 端口输入时钟  说明 - 该位在计数器停止时设定。 - 当使用外部输入时钟进行操作时，计数器操作开始后，最初的外部输入时钟的边沿，无论是上升沿还是下降沿都被忽略，计数操作从之后的边沿开始。	RW
14	IRQZF	下溢状态	0b0: 未发生计数下溢 0b1: 发生计数下溢  说明 - 当使用周期间隔响应功能时，该位的置位条件由 CVPR 所设定的周期间隔计数器设定。 - 当计数器被总线复位或者写入 CLEAR=1，IRQZF 位将不会被置位。	RW
13	IRQZEN	下溢中断使能	0b0: 禁止 IRQZF 产生中断到 CPU 0b1: 允许 IRQZF 产生中断到 CPU	RW
12	SYNST	同步启动使能	0b0: 该单元与单元 1 的同步启动功能无效 0b1: 该单元与单元 1 的同步启动功能有效  说明 仅单元 2 和单元 3 支持 SYNST 功能位，单元 1 的该位为保留位，请保持初始值 0。	RW
11:10	Reserved	保留	-	-
9	IRQPF	上溢状态	0b0: 未发生计数上溢 0b1: 发生计数上溢  说明 - 当使用周期间隔响应功能时，该位的置位条件由 CVPR 所设定的周期间隔计数器设定。 - 当计数器被总线复位或者写入 CLEAR=1，IRQPF 位将不会被置位。	RW
8	IRQPEN	上溢中断使能	0b0: 禁止 IRQPF 产生中断到 CPU 0b1: 允许 IRQPF 产生中断到 CPU	RW

位/位域	标记	位名	功能描述	读写
7	BUFEN	缓存使能	0b0: 禁止 CPSR 缓存功能 0b1: 使能 CPSR 缓存功能	RW
6	STOP	计数器使能	0b0: 计数器启动 0b1: 计数器停止	RW
5	MODE	波形模式	0b0: 锯齿波模式 (仅支持向上计数) 0b1: 三角波模式	RW
4	CLEAR	计数器清零	写 1 清零计数器, 写 0 无效; 读出恒为 0	RW
3:0	CKDIV	计数时钟分频	该位指示基本计数器的计数时钟分频 0b0000: 计数时钟为 PCLK0 0b0001: 计数时钟为 PCLK0/2 0b0010: 计数时钟为 PCLK0/4 0b0011: 计数时钟为 PCLK0/8 0b0100: 计数时钟为 PCLK0/16 0b0101: 计数时钟为 PCLK0/32 0b0110: 计数时钟为 PCLK0/64 0b0111: 计数时钟为 PCLK0/128 0b1000: 计数时钟为 PCLK0/256 0b1001: 计数时钟为 PCLK0/512 0b1010: 计数时钟为 PCLK0/1024 - 其他: 禁止设置  说明 时钟源为内部 PCLK0 时, 分频有效。	RW

22.4.5 有效周期寄存器 (TMR4_CVPR)

Offset	Bit Position															
0x004A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PIC				ZIC				PIM				ZIM			
Access	RW				RW				RW				RW			

位/位域	标记	位名	功能描述	读写
15:12	PIC	上溢中断屏蔽状态	PIC 是一个向下计数器，当 CNTR 计数上溢中断时 PIC 的值减 1，读取 PIC 可以获得当前剩余需要屏蔽的上溢中断个数。 写入条件：计数器停止状态 (CCSR.STOP=1) 满足以下之一，PIM 的值载入 PIC： 1. 计数器清零动作发生 (CCSR.CLEAR 写 1) 2. PIC=0 且 CNTR 计数发生上溢中断	RW
11:8	ZIC	下溢中断屏蔽状态	ZIC 是一个向下计数器，当 CNTR 计数下溢中断时 ZIC 的值减 1，读取 ZIC 可以获得当前剩余需要屏蔽的下溢中断个数。 写入条件：计数器停止状态 (CCSR.STOP=1) 满足以下之一，ZIM 的值载入 ZIC： 1. 计数器清零动作发生 (CCSR.CLEAR 写 1) 2. ZIC=0 且 CNTR 计数发生下溢中断	RW
7:4	PIM	上溢中断屏蔽设定	设置需要屏蔽的上溢中断个数 满足以下之一，PIM 的值载入 PIC： 1. 计数器清零动作发生 (CCSR.CLEAR 写 1) 2. PIC=0 且 CNTR 计数发生上溢中断	RW
3:0	ZIM	下溢中断屏蔽设定	设置需要屏蔽的下溢中断个数 满足以下之一，ZIM 的值载入 ZIC： 1. 计数器清零动作发生 (CCSR.CLEAR 写 1) 2. ZIC=0 且 CNTR 计数发生下溢中断	RW

22.4.6 通用比较基准寄存器（TMR4_OCCRm）

偏移地址（Offset）：0x0002（TMR4_OCCRu_h），0x0006（TMR4_OCCRu_l），0x000A（TMR4_OCCRv_h），0x000E（TMR4_OCCRv_l），0x0012（TMR4_OCCRw_h），0x0016（TMR4_OCCRw_l）

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	OCCR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	OCCR	通用比较基准值	通用比较基准值	RW
			说明 从本地地址区域读取数据时，读取的并非缓冲器寄存器的值，而是 OCCR 寄存器的值。	

22.4.7 通用控制状态寄存器 (TMR4_OCSRn)

偏移地址 (Offset) : 0x0018 (TMR4_OCSRu) , 0x001C (TMR4_OCSRv) , 0x0020 (TMR4_OCSRw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFF00															
Name	Reserved								OCFL	OCFH	OCIEL	OCIEH	OCPL	OPCH	OCEL	OCEH
Access									RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:8	Reserved	保留	所有位读出的 1, 写入时必须写 1	-
7	OCFL	计数匹配标志 L	0b0: 计数器计数值与 OCCR*l 设定值不相等 0b1: 计数器计数值与 OCCR*l 设定值相等  说明 该位在 OCEL=1 时有效。	RW
6	OCFH	计数匹配标志 H	0b0: 计数器计数值与 OCCR*h 设定值不相等 0b1: 计数器计数值与 OCCR*h 设定值相等  说明 该位在 OCEH=1 时有效。	RW
5	OCIEL	计数匹配中断使能 L	0b0: OCFL 置位时, 不发生中断 0b1: OCFL 置位时, 发生中断	RW
4	OCIEH	计数匹配中断使能 H	0b0: OCFH 置位时, 不发生中断 0b1: OCFH 置位时, 发生中断	RW
3	OCPL	比较输出无效时端口状态 L	写入: 0b0: OCEL=0 时, in_op*l 输出低电平 0b1: OCEL=0 时, in_op*l 输出高电平 读取: 0b0: in_op*l 当前状态为低电平 0b1: in_op*l 当前状态为高电平	RW
2	OPCH	比较输出无效时端口状态 H	写入: 0b0: OCEH=0 时, in_op*h 输出低电平 0b1: OCEH=0 时, in_op*h 输出高电平 读取: 0b0: in_op*h 当前状态为低电平 0b1: in_op*h 当前状态为高电平	RW
1	OCEL	端口输出选择 L	0b0: 比较输出无效, in_op*l 端口状态由 OCPL 决定 0b1: 比较输出有效, in_op*l 端口状态由 OCMR*l 的设定和 OCFL 状态决定	RW
0	OCEH	端口输出选择 H	0b0: 比较输出无效, in_op*h 端口状态由 OCPH 决定 0b1: 比较输出有效, in_op*h 端口状态由 OCMR*h 的设定和 OCFH 状态决定	RW

22.4.8 通用扩展控制寄存器 (TMR4_OCERn)

偏移地址 (Offset) : 0x001A (TMR4_OCERu) , 0x001E (TMR4_OCERv) , 0x0022 (TMR4_OCERw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved		MCECL	MCECH	LMML	LMMH	LMCL	LMCH	MLBUFEN		MHBUFEN		CLBUFEN		CHBUFEN	
Access			RW	RW	RW	RW	RW	RW	RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
15:14	Reserved	保留	-	-
13	MCECL	计数匹配增加条件使能 L	增加条件：计数上溢 (CNTR=CPSR) 且 OCCR*I>CPSR 0b0：禁止该条件作为匹配条件 0b1：使能该条件作为匹配条件，条件满足时 OCCR*I 计数匹配  说明 - 只能在输出比较功能无效 (OCSR.OCEL=0) 时修改该位。 - 具体请参考 TMR4_OCMRm 寄存器说明的注意事项。	RW
12	MCECH	计数匹配增加条件使能 H	增加条件：计数上溢 (CNTR=CPSR) 且 OCCR*h>CPSR 0b0：禁止该条件作为匹配条件 0b1：使能该条件作为匹配条件，条件满足时 OCCR*h 计数匹配  说明 - 只能在输出比较功能无效 (OCSR.OCEH=0) 时修改该位。 - 具体请参考 TMR4_OCMRm 寄存器说明的注意事项。	RW
11	LMML	OCMR*I 周期间隔响应功能链接 L	0b0：周期间隔响应功能链接无效，OCMR*I 的缓存传送由 MLBUFEN 设定决定 0b1：周期间隔响应功能链接有效，OCMR*I 的缓存传送在 MLBUFEN 设定的基础上，还必须满足 CVPR.PIC[3:0]=0b0000 (计数上溢时) 或 CVPR.ZIC[3:0]=0b0000 (计数下溢时)	RW
10	LMMH	OCMR*h 周期间隔响应功能链接 H	0b0：周期间隔响应功能链接无效，OCMR*h 的缓存传送由 MLBUFEN 设定决定 0b1：周期间隔响应功能链接有效，OCMR*h 的缓存传送在 MLBUFEN 设定的基础上，还必须满足 CVPR.PIC[3:0]=0b0000 (计数上溢时) 或 CVPR.ZIC[3:0]=0b0000 (计数下溢时)	RW

位/位域	标记	位名	功能描述	读写
9	LMCL	OCCR*I 周期间隔响应功能链接 L	0b0: 周期间隔响应功能链接无效, OCCR*I 的缓存传送由 MLBUFEN 设定决定 0b1: 周期间隔响应功能链接有效, OCCR*I 的缓存传送在 MLBUFEN 设定的基础上, 还必须满足 CVPR.PIC[3:0]=0b0000 (计数上溢时) 或 CVPR.ZIC[3:0]=0b0000 (计数下溢时)	RW
8	LMCH	OCCR*h 周期间隔响应功能链接 H	0b0: 周期间隔响应功能链接无效, OCCR*h 的缓存传送由 MLBUFEN 设定决定 0b1: 周期间隔响应功能链接有效, OCCR*h 的缓存传送在 MLBUFEN 设定的基础上, 还必须满足 CVPR.PIC[3:0]=0b0000 (计数上溢时) 或 CVPR.ZIC[3:0]=0b0000 (计数下溢时)	RW
7:6	MLBUFEN	OCMR*I 缓存传送	0b00: OCMR*I 缓存寄存器的值直接写入 OCMR*I 0b01: OCMR*I 缓存寄存器的值在计数下溢时写入 OCMR*I 0b10: OCMR*I 缓存寄存器的值在计数上溢时写入 OCMR*I 0b11: OCMR*I 缓存寄存器的值在计数下溢或上溢时写入 OCMR*I	RW
5:4	MHBUFEN	OCMR*h 缓存传送	0b00: OCMR*h 缓存寄存器的值直接写入 OCMR*h 0b01: OCMR*h 缓存寄存器的值在计数下溢时写入 OCMR*h 0b10: OCMR*h 缓存寄存器的值在计数上溢时写入 OCMR*h 0b11: OCMR*h 缓存寄存器的值在计数下溢或上溢时写入 OCMR*h	RW
3:2	CLBUFEN	OCCR*I 缓存传送	0b00: OCCR*I 缓存寄存器的值直接写入 OCCR*I 0b01: OCCR*I 缓存寄存器的值在计数下溢时写入 OCCR*I 0b10: OCCR*I 缓存寄存器的值在计数上溢时写入 OCCR*I 0b11: OCCR*I 缓存寄存器的值在计数下溢或上溢时写入 OCCR*I	RW
1:0	CHBUFEN	OCCR*h 缓存传送	0b00: OCCR*h 缓存寄存器的值直接写入 OCCR*h 0b01: OCCR*h 缓存寄存器的值在计数下溢时写入 OCCR*h 0b10: OCCR*h 缓存寄存器的值在计数上溢时写入 OCCR*h 0b11: OCCR*h 缓存寄存器的值在计数下溢或上溢时写入 OCCR*h	RW

22.4.9 通用模式控制寄存器 (TMR4_OCMRm)

偏移地址 (Offset) : 0x0024 (TMR4_OCMRuh) , 0x0028 (TMR4_OCMRul) , 0x002C (TMR4_OCMRvh) , 0x0030 (TMR4_OCMRvl) , 0x0034 (TMR4_OCMRwh) , 0x0038 (TMR4_OCMRwl)

TMR4_OCMRuh、TMR4_OCMRvh、TMR4_OCMRwh 寄存器:

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	OPNZRH		OPNPKH		OPZRH		OPUCH		OPPKH		OPDCH		OCFZRH		OCFUCH	
Access	RW		RW		RW		RW		RW		RW		RW		RW	

表 22-21 OCCR*h 计数匹配一览表

条件编号	OCCR*h 计数是否匹配
1	否
2	是

位/位域	标记	位名	功能描述	读写
15:14	OPNZRH	下溢条件 1 端口状态 H	下溢条件 1 (同时满足) : 计数下溢, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*h 保持不变 0b01: 满足条件时, in_op*h 输出高电平 0b10: 满足条件时, in_op*h 输出低电平 0b11: 满足条件时, in_op*h 输出反转	RW
13:12	OPNPKH	上溢条件 1 端口状态 H	上溢条件 1 (同时满足) : 计数上溢, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*h 保持不变 0b01: 满足条件时, in_op*h 输出高电平 0b10: 满足条件时, in_op*h 输出低电平 0b11: 满足条件时, in_op*h 输出反转	RW
11:10	OPZRH	下溢条件 2 端口状态 H	下溢条件 2 (同时满足) : 计数下溢, OCCR*h 计数匹配 0b00: 满足条件时, in_op*h 保持不变 0b01: 满足条件时, in_op*h 输出高电平 0b10: 满足条件时, in_op*h 输出低电平 0b11: 满足条件时, in_op*h 输出反转	RW
9:8	OPUCH	向上计数条件 2 端口状态 H	向上计数条件 2 (同时满足) : 计数器向上计数, OCCR*h 计数匹配 0b00: 满足条件时, in_op*h 保持不变 0b01: 满足条件时, in_op*h 输出高电平 0b10: 满足条件时, in_op*h 输出低电平 0b11: 满足条件时, in_op*h 输出反转	RW

位/位域	标记	位名	功能描述	读写
7:6	OPPKH	上溢条件 2 端口状态 H	上溢条件 2 (同时满足)：计数上溢，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*h 保持不变 ● 0b01：满足条件时，in_op*h 输出高电平 ● 0b10：满足条件时，in_op*h 输出低电平 ● 0b11：满足条件时，in_op*h 输出反转	RW
5:4	OPDCH	向下计数条件 2 端口状态 H	向下计数条件 2 (同时满足)：计数器向下计数，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*h 保持不变 ● 0b01：满足条件时，in_op*h 输出高电平 ● 0b10：满足条件时，in_op*h 输出低电平 ● 0b11：满足条件时，in_op*h 输出反转	RW
3	OCFZRH	下溢点 OCFH 状态 H	条件 (同时满足)：计数下溢，OCCR*h 计数匹配 ● 0b0：满足条件时，OCSR.OCFH 位保持不变 ● 0b1：满足条件时，OCSR.OCFH 位置位	RW
2	OCFUCH	向上计数 OCFH 状态 H	条件 (同时满足)：计数器向上计数，OCCR*h 计数匹配 ● 0b0：满足条件时，OCSR.OCFH 位保持不变 ● 0b1：满足条件时，OCSR.OCFH 位置位	RW
1	OCFPKH	上溢点 OCFH 状态 H	条件 (同时满足)：计数上溢，OCCR*h 计数匹配 ● 0b0：满足条件时，OCSR.OCFH 位保持不变 ● 0b1：满足条件时，OCSR.OCFH 位置位	RW
0	OCFDCH	向下计数 OCFH 状态 H	条件 (同时满足)：计数器向下计数，OCCR*h 计数匹配 ● 0b0：满足条件时，OCSR.OCFH 位保持不变 ● 0b1：满足条件时，OCSR.OCFH 位置位	RW

TMR4_OC MRul、TMR4_OC MRvl、TMR4_OC MRwl 寄存器：

Offset	Bit Position																																		
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	EOPNZRL	EOPNPKL		EOPZRL		EOPUCL		EOPPKL		EOPDCL		EOPNUCL		EOPNDCL		OPNZRL		OPNPKL		OPZRL		OPUCL		OPPKL		OPDCL		OCFZRL		OCFUCL		OCFPKL		OCFDCL	
Access	RW	RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW	

表 22-24 OCCR*h/I 计数匹配一览表

条件编号	OCCR*h 计数是否匹配	OCCR*I 计数是否匹配
1	否	否
2	否	是
3	是	否
4	是	是

位/位域	标记	位名	功能描述	读写
31:30	EOPNZRL	下溢条件 3 端口状态 L	下溢条件 3 (同时满足)：计数下溢，OCCR*I 计数不匹配，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
29:28	EOPNPKL	上溢条件 3 端口状态 L	上溢条件 3 (同时满足)：计数上溢，OCCR*I 计数不匹配，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
27:26	EOPZRL	下溢条件 4 端口状态 L	下溢条件 4 (同时满足)：计数下溢，OCCR*I 计数匹配，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
25:24	EOPUCL	向上计数条件 4 端口状态 L	向上计数条件 4 (同时满足)：计数器向上计数，OCCR*I 计数匹配，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
23:22	EOPPKL	上溢条件 4 端口状态 L	上溢条件 4 (同时满足)：计数上溢，OCCR*I 计数匹配，OCCR*h 计数匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平	RW

位/位域	标记	位名	功能描述	读写
			0b11: 满足条件时, in_op*I 输出反转	
21:20	EOPDCL	向下计数条件 4 端口状态 L	向下计数条件 4 (同时满足): 计数器向下计数, OCCR*I 计数匹配, OCCR*h 计数匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
19:18	EOPNUCL	向上计数条件 3 端口状态 L	向上计数条件 3 (同时满足): 计数器向上计数, OCCR*I 计数不匹配, OCCR*h 计数匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
17:16	EOPNDCL	向下计数条件 3 端口状态 L	向下计数条件 3 (同时满足): 计数器向下计数, OCCR*I 计数不匹配, OCCR*h 计数匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
15:14	OPNZRL	下溢条件 1 端口状态 L	下溢条件 1 (同时满足): 计数下溢, OCCR*I 计数不匹配, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
13:12	OPNPKL	上溢条件 1 端口状态 L	上溢条件 1 (同时满足): 计数上溢, OCCR*I 计数不匹配, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
11:10	OPZRL	下溢条件 2 端口状态 L	下溢条件 2 (同时满足): 计数下溢, OCCR*I 计数匹配, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW
9:8	OPUCL	向上计数条件 2 端口状态 L	向上计数条件 2 (同时满足): 计数器向上计数, OCCR*I 计数匹配, OCCR*h 计数不匹配 0b00: 满足条件时, in_op*I 保持不变 0b01: 满足条件时, in_op*I 输出高电平 0b10: 满足条件时, in_op*I 输出低电平 0b11: 满足条件时, in_op*I 输出反转	RW

位/位域	标记	位名	功能描述	读写
7:6	OPPKL	上溢条件 2 端口状态 L	上溢条件 2 (同时满足)：计数上溢，OCCR*I 计数匹配，OCCR*h 计数不匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
5:4	OPDCL	向下计数条件 2 端口状态 L	向下计数条件 2 (同时满足)：计数器向下计数，OCCR*I 计数匹配，OCCR*h 计数不匹配 ● 0b00：满足条件时，in_op*I 保持不变 ● 0b01：满足条件时，in_op*I 输出高电平 ● 0b10：满足条件时，in_op*I 输出低电平 ● 0b11：满足条件时，in_op*I 输出反转	RW
3	OCFZRL	下溢点 OCFL 状态 L	条件 (同时满足)：计数下溢，OCCR*I 计数匹配 ● 0b0：满足条件时，OCSR.OCFL 位保持不变 ● 0b1：满足条件时，OCSR.OCFL 位置位	RW
2	OCFUCL	向上计数 OCFL 状态 L	条件 (同时满足)：计数器向上计数，OCCR*I 计数匹配 ● 0b0：满足条件时，OCSR.OCFL 位保持不变 ● 0b1：满足条件时，OCSR.OCFL 位置位	RW
1	OCFPKL	上溢点 OCFL 状态 L	条件 (同时满足)：计数上溢，OCCR*I 计数匹配 ● 0b0：满足条件时，OCSR.OCFL 位保持不变 ● 0b1：满足条件时，OCSR.OCFL 位置位	RW
0	OCFDCL	向下计数 OCFL 状态 L	条件 (同时满足)：计数器向下计数，OCCR*I 计数匹配 ● 0b0：满足条件时，OCSR.OCFL 位保持不变 ● 0b1：满足条件时，OCSR.OCFL 位置位	RW



注意

- 从本地地址区域读取数据时，读取的并非缓冲器寄存器的值，而是 OCMR 寄存器的值。
- TMR4_<t>_O*I 可以由 OCCR*I 与计数器的计数值（独立操作模式），或者 OCCR*h 与计数器的计数值且 OCCR*I 与计数器的计数值（链接操作模式）来决定。向寄存器 OCMR*I 的 Bit[31:20]和 Bit[15:4]写入相同的 12 位值，同时将 OCMR*I[19:16]写入 0b0000，此时，in_op*I 输出将不受 OCCR*h 的影响，而仅由 OCCR*I 决定。独立操作模式：通道*H 由 OCCR*h 决定，通道*L 由 OCCR*I 决定配置。如果不满足上述条件的独立操作模式，就是链接操作模式：通道*L 输出同时受 OCCR*h 和 OCCR*I 影响。
- 下面 2 种情况满足时，也认为是符合 OCMR*h.OPPKH[7:6]设定的条件：
 - CCSR.MODE=1 && 计数上溢 && OCCR*h=0xFFFF
 - OCER*.MCECH=1 && 计数上溢 && OCCR*h≥CNTR
- 下面 2 种情况满足时，也认为是符合 OCMR*I.OPPKL[7:6]设定的条件：
 - CCSR.MODE=1 && 计数上溢 && OCCR*I=0xFFFF
 - OCER*.MCECL=1 && 计数上溢 && OCCR*I≥CNTR

22.4.10 专用比较基准寄存器（TMR4_SCCRm）

偏移地址（Offset）：0x00B2（TMR4_SCCRuh），0x00B6（TMR4_SCCRul），0x00BA（TMR4_SCCRvh），0x00BE（TMR4_SCCRvl），0x00C2（TMR4_SCCRwh），0x00C6（TMR4_SCCRwl）

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	SCCR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	SCCR	专用比较基准值	专用比较基准值（比较启动模式）或延迟基准值（延迟启动模式）  说明 从本地地址区域读取数据时，读取的并非缓冲器寄存器的值，而是 SCCR 寄存器的值。	RW

22.4.11 专用控制状态寄存器 (TMR4_SCSRm)

偏移地址 (Offset) : 0x00C8 (TMR4_SCSRuh) , 0x00CC (TMR4_SCSRul) , 0x00D0 (TMR4_SCSRvh) , 0x00D4 (TMR4_SCSRvl) , 0x00D8 (TMR4_SCSRwh) , 0x00DC (TMR4_SCSRwl)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	ZEN	UEN	PEN	DEN	Reserved	EVTDS	EVTMS	Reserved	LMC	EVTOS			BUFEN			
Access	RW	RW	RW	RW		RW	RW		RW	RW			RW			

位/位域	标记	位名	功能描述	读写
15	ZEN	下溢点 EVT 使能	0b0: 计数下溢时, EVT 无操作 0b1: 计数下溢时: - EVTMS=0&SCCR 比较匹配&SCMR 设定匹配时, EVT 启动输出 - EVTMS=1&OCCR 比较匹配&SCMR 设定匹配时, EVT 延时模式启动	RW
14	UEN	向上计数 EVT 使能	0b0: 向上计数时, EVT 无操作 0b1: 向上计数时: - EVTMS=0&SCCR 比较匹配&SCMR 设定匹配时, EVT 启动输出 - EVTMS=1&OCCR 比较匹配&SCMR 设定匹配时, EVT 延时模式启动	RW
13	PEN	上溢点 EVT 使能	0b0: 计数上溢时, EVT 无操作 0b1: 计数上溢时: - EVTMS=0&SCCR 比较匹配&SCMR 设定匹配时, EVT 启动输出 - EVTMS=1&OCCR 比较匹配&SCMR 设定匹配时, EVT 延时模式启动	RW
12	DEN	向下计数 EVT 使能	0b0: 向下计数时, EVT 无操作 0b1: 向下计数时: - EVTMS=0&SCCR 比较匹配&SCMR 设定匹配时, EVT 启动输出 - EVTMS=1&OCCR 比较匹配&SCMR 设定匹配时, EVT 延时模式启动	RW
11:10	Reserved	保留	-	-
9	EVTDS	EVT 延时对象选择	0b0: 在延时启动模式时, OCCR*h 作为延时比较匹配对象 0b1: 在延时启动模式时, OCCR*I 作为延时比较匹配对象  说明 该位在 EVTMS=0 时无效。	RW

位/位域	标记	位名	功能描述	读写
8	EVTMS	EVT 模式选择	0b0: 比较启动模式 (CNTR 和 SCCR 的比较结果触发) 0b1: 延时启动模式 (比较匹配事件经 SCCR 延迟后触发)	RW
7:6	Reserved	保留	-	-
5	LMC	周期间隔响应功能链接	0b0: 周期间隔响应功能链接无效, SCCR 的缓存传送由 BUFEN 设定决定 0b1: 周期间隔响应功能链接有效, SCCR 的缓存传送在 BUFEN 设定的基础上, 还必须满足 CVPR.PIC[3:0]=0b0000 (计数上溢时) 或 CVPR.ZIC[3:0]=0b0000 (计数下溢时)	RW
4:2	EVTOS	EVT 输出选择	0b000: Special Event 0 (TMR4_<t>_SCM0) 的 EVT 输出有效 0b001: Special Evnet 1 (TMR4_<t>_SCM1) 的 EVT 输出有效 0b010: Special Evnet 2 (TMR4_<t>_SCM2) 的 EVT 输出有效 0b011: Special Evnet 3 (TMR4_<t>_SCM3) 的 EVT 输出有效 0b100: Special Evnet 4 (TMR4_<t>_SCM4) 的 EVT 输出有效 0b101: Special Evnet 5 (TMR4_<t>_SCM5) 的 EVT 输出有效 - 其他: 禁止设置	RW
1:0	BUFEN	SCCR&SCMR 缓存传送	0b00: SCCR、SCMR 缓存寄存器的值直接写入 SCCR、SCMR 0b01: SCCR、SCMR 缓存寄存器的值在计数下溢时写入 SCCR、SCMR 0b10: SCCR、SCMR 缓存寄存器的值在计数上溢时写入 SCCR、SCMR 0b11: SCCR、SCMR 缓存寄存器的值在计数下溢或上溢时写入 SCCR、SCMR	RW

22.4.12 专用扩展控制寄存器 (TMR4_SCER)

Offset	Bit Position															
0x00E4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFF00															
Name	Reserved												PCTS	EVTRS		
Access													RW	RW		

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	所有位读出的 1，写入时必须写 1	-
3	PCTS	计数方向输出使能	计数器工作在三角波模式时，控制计数方向输出到端口 0b0：禁止计数方向信号输出到 TMR4_<t>_PCT 端口 0b1：允许计数方向信号输出到 TMR4_<t>_PCT 端口	RW
2:0	EVTRS	专用事件输出选择	0b000：禁止专用事件输出到端口 0b001：Special Event 0 (TMR4_<t>_SCM0) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b010：Special Event1 (TMR4_<t>_SCM1) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b011：Special Event 2 (TMR4_<t>_SCM2) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b100：Special Event 3 (TMR4_<t>_SCM3) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b101：Special Event 4 (TMR4_<t>_SCM4) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b110：Special Event 5 (TMR4_<t>_SCM5) 的 EVT 输出至 TMR4_<t>_ADSM 端口 0b111：禁止设置	RW

22.4.13 专用模式控制寄存器 (TMR4_SCMRm)

偏移地址 (Offset) : 0x00CA (TMR4_SCMRuh) , 0x00CE (TMR4_SCMRul) , 0x00D2 (TMR4_SCMRvh) , 0x00D6 (TMR4_SCMRvl) , 0x00DA (TMR4_SCMRwh) , 0x00DE (TMR4_SCMRwl)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFF00															
Name	Reserved								MPCE	MZCE	Reserved			AMC		
Access									RW	RW				RW		

位/位域	标记	位名	功能描述	读写
15:8	Reserved	保留	所有位读出的 1，写入时必须写 1	-
7	MPCE	周期间隔响应使能	0b0: 禁止 AMC 与 CVPR.PIC 比较 0b1: 使能 AMC 与 CVPR.PIC 比较	RW
6	MZCE	周期间隔响应使能	0b0: 禁止 AMC 与 CVPR.ZIC 比较 0b1: 使能 AMC 与 CVPR.ZIC 比较	RW
5:4	Reserved	保留	-	-
3:0	AMC	专用事件输出周期间隔值	该位设定专用事件输出功能时的周期间隔值，在 AMC 和 CVPR.PIC 或 CVPR.ZIC 相等时，专用事件输出功能有效	RW

说明

从本地址区域读取数据时，读取的并非缓冲器寄存器的值，而是 SCMR 寄存器的值。

22.4.14 PWM 基本控制寄存器 (TMR4_POCRn)

偏移地址 (Offset) : 0x0098 (TMR4_POCRu) , 0x009C (TMR4_POCRv) , 0x00A0 (TMR4_POCRw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFF00															
Name	Reserved								LVLS		PWMMD		Reserved	DIVCK		
Access									RW		RW			RW		

位/位域	标记	位名	功能描述	读写
15:8	Reserved	保留	所有位读出于 1, 写入时必须写 1	-
7:6	LVLS	PWM 输出极性控制	0b00: TMR4_<t>_O*H 和 TMR4_<t>_O*L 的输出都不反转 0b01: TMR4_<t>_O*H 和 TMR4_<t>_O*L 的输出都反转 0b10: TMR4_<t>_O*H 的输出反转, TMR4_<t>_O*L 的输出不反转 0b11: TMR4_<t>_O*H 的输出不反转, TMR4_<t>_O*L 的输出反转	RW
5:4	PWMMD	PWM 输出模式	0b00: 直通模式 0b01: 死区定时器模式 0b10: 死区定时器滤波模式 0b11: 禁止设置	RW
3	Reserved	保留	-	-
2:0	DIVCK	计数时钟分频	该位指示滤波计数器、死区计数器和重载计数器的计数时钟分频 0b000: 计数时钟为 PCLK0 0b001: 计数时钟为 PCLK0/2 0b010: 计数时钟为 PCLK0/4 0b011: 计数时钟为 PCLK0/8 0b100: 计数时钟为 PCLK0/16 0b101: 计数时钟为 PCLK0/32 0b110: 计数时钟为 PCLK0/64 0b111: 计数时钟为 PCLK0/128	RW

22.4.15 PWM 状态控制寄存器 (TMR4_PSCR)

Offset	Bit Position																																					
0x00E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x05550000																																					
Name	Reserved				OSWL		OSWH		OSVL		OSVH		OSUL		OSUH		Reserved				AOE		MOE		ODT		OEWL		OEWH		OEVL		OEVH		OEUL		OEUH	
Access					RW		RW		RW		RW		RW		RW						RW		RW		RW		RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27:26	OSWL	TMR4_<t>_O WL 输出禁止 状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEVL=0 0b00: 满足以上条件之一时, TMR4_<t>_OWL 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OWL 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OWL 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OWL 端口输出固定为高电平  说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
25:24	OSWH	TMR4_<t>_O WH 输出禁止 状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEWH=0 0b00: 满足以上条件之一时, TMR4_<t>_OWH 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OWH 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OWH 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OWH 端口输出固定为高电平  说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
23:22	OSVL	TMR4_<t>_O VL 输出禁止 状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEVL=0 0b00: 满足以上条件之一时, TMR4_<t>_OVL 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OVL 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OVL 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OVL 端口输出固定为高电平	RW

位/位域	标记	位名	功能描述	读写
23:22	OSVL	TMR4_<t>_OVL 输出禁止状态	 说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
21:20	OSVH	TMR4_<t>_OVH 输出禁止状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEVL=0 0b00: 满足以上条件之一时, TMR4_<t>_OVH 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OVH 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OVH 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OVH 端口输出固定为高电平  说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
19:18	OSUL	TMR4_<t>_OUL 输出禁止状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEVL=0 0b00: 满足以上条件之一时, TMR4_<t>_OUL 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OUL 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OUL 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OUL 端口输出固定为高电平  说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
17:16	OSUH	TMR4_<t>_OUH 输出禁止状态	条件: 1.发生 EMB 事件; 2.软件写 MOE=0; 3. MOE=1&OEVL=0 0b00: 满足以上条件之一时, TMR4_<t>_OUH 端口正常输出 0b01: 满足以上条件之一时, TMR4_<t>_OUH 端口输出为 Hi-Z 0b10: 满足以上条件之一时, TMR4_<t>_OUH 端口输出固定为低电平 0b11: 满足以上条件之一时, TMR4_<t>_OUH 端口输出固定为高电平  说明 只能在 Timer4 停止 (CCSR.STOP=1) 时修改该位。	RW
15:10	Reserved	保留	-	-
9	AOE	自动输出使能	0b0: EMB 事件清除后 MOE 位只能软件置 1 以恢复 PWM 正常输出 0b1: EMB 事件清除后 MOE 位硬件自动置 1, PWM 输出恢复正常  说明 置位 EMB_STATCLR 或清零 EMB_SOE 可以清除 EMB 事件。	RW

位/位域	标记	位名	功能描述	读写
8	MOE	主输出使能	0b0: TMR4_<t>_O*H/L 端口输出由 OS*H/L 位指定的状态 0b1: OE*H、OE*L 位设定为使能时 TMR4_<t>_O*H、TMR4_<t>_O*L 端口输出使能  说明 EMB 事件发生后此位由硬件立即清零。EMB 事件清除后，依据 AOE 位的设置此位可以软件置 1 或硬件自动置 1。	RW
7:6	ODT	端口使能位生效时间	条件：端口使能位 OE*L、OE*H 的值改变后 0b0x: 满足条件时，立即生效 0b10: 满足条件时，计数器下次下溢时生效 0b11: 满足条件时，计数器下次上溢时生效	RW
5	OEWL	TMR4_<t>_OVL 输出使能	0b0: TMR4_<t>_OWL 端口输出状态由 OSWL 位设定 0b1: MOE=1 时，TMR4_<t>_OWL 端口输出正常 PWM 波形	RW
4	OEWH	TMR4_<t>_OWH 输出使能	0b0: TMR4_<t>_OWH 端口输出状态由 OSWH 位设定 0b1: MOE=1 时，TMR4_<t>_OWH 端口输出正常 PWM 波形	RW
3	OEVL	TMR4_<t>_OVL 输出使能	0b0: TMR4_<t>_OVL 端口输出状态由 OSVL 位设定 0b1: MOE=1 时，TMR4_<t>_OVL 端口输出正常 PWM 波形	RW
2	OEVL	TMR4_<t>_OVH 输出使能	0b0: TMR4_<t>_OVH 端口输出状态由 OSVH 位设定 0b1: MOE=1 时，TMR4_<t>_OVH 端口输出正常 PWM 波形	RW
1	OEUL	TMR4_<t>_OUL 输出使能	0b0: TMR4_<t>_OUL 端口输出状态由 OSUL 位设定 0b1: MOE=1 时，TMR4_<t>_OUL 端口输出正常 PWM 波形	RW
0	OEUL	TMR4_<t>_OUH 输出使能	0b0: TMR4_<t>_OUH 端口输出状态由 OSUH 位设定 0b1: MOE=1 时，TMR4_<t>_OUH 端口输出正常 PWM 波形	RW

下表为 PWM 端口输出状态与寄存器设定值的关系说明：

MOE 位	OE*L/H 位	OS*L/H 位	TMR4_<t>_O*L/H 输出状态
0b1	0b0	0b00	正常输出 (Timer4 驱动)
		0b01	Hi-Z
		0b10	低电平
		0b11	高电平
	0b1	0bXX	正常输出 (Timer4 驱动)
0b0	0bX	0b00	正常输出 (Timer4 驱动)
		0b01	Hi-Z
		0b10	低电平
		0b11	高电平

22.4.16 PWM 滤波控制寄存器 (TMR4_PFSRn)

偏移地址 (Offset) : 0x0082 (TMR4_PFSRu) , 0x008A (TMR4_PFSRv) , 0x0092 (TMR4_PFSRw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PFSR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PFSR	滤波初始值	滤波计数初始值  说明 当 PWM 波形输出模式不选死区定时器滤波模式时, 16 位滤波计数器用作 16 位重载计数器, 可以周期性产生中断输出。此功能与 PWM 波形生成器功能无关。	RW

22.4.17 PWM 死区控制寄存器 A (TMR4_PDARn)

偏移地址 (Offset) : 0x0084 (TMR4_PDARu) , 0x008C (TMR4_PDARv) , 0x0094 (TMR4_PDARw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PDAR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PDAR	死区初始值	死区计数初始值	RW

22.4.18 PWM 死区控制寄存器 B (TMR4_PDBRn)

偏移地址 (Offset) : 0x0086 (TMR4_PDBRu) , 0x008E (TMR4_PDBRv) , 0x0096 (TMR4_PDBRw)

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	PDBR															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PDBR	死区初始值	死区计数初始值	RW

22.4.19 重载控制状态寄存器 (TMR4_RCSR)

Offset	Bit Position															
0x00A4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RTSW	RTEW	RTICW	RTIFW	RTSV	RTEV	RTICV	RTIFV	RTSU	RTEU	RTICU	RTIFU	Reserved	RTIDW	RTIDV	RTIDU
Access	RW	RW	RW	RO	RW	RW	RW	RO	RW	RW	RW	RO		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	RTSW	重载计数器停止 W	写 1 停止重载计数器 W 并清除 RTIFW，写 0 无效；读出恒为 0	RW
14	RTEW	重载计数器启动 W	写 1 启动重载计数器 W，写 0 无效 读取： ● 0b0：重载计数器 W 已停止 ● 0b1：重载计数器 W 已启动	RW
13	RTICW	清零计数匹配状态 W	写 1 清零 RTIFW 标志位，写 0 无效；读出恒为 0	RW
12	RTIFW	计数匹配状态 W	● 0b0：重载计数器计数值与 0 发生未比较匹配 ● 0b1：重载计数器计数值与 0 发生比较匹配	RO
11	RTSV	重载计数器停止 V	写 1 停止重载计数器 V 并清除 RTIFV，写 0 无效；读出恒为 0	RW
10	RTEV	重载计数器启动 V	写 1 启动重载计数器 V，写 0 无效 读取： ● 0b0：重载计数器 V 已停止 ● 0b1：重载计数器 V 已启动	RW
9	RTICV	清零计数匹配状态 V	写 1 清零 RTIFV 标志位，写 0 无效；读出恒为 0	RW
8	RTIFV	计数匹配状态 V	● 0b0：重载计数器计数值与 0 发生未比较匹配 ● 0b1：重载计数器计数值与 0 发生比较匹配	RO
7	RTSU	重载计数器停止 U	写 1 停止重载计数器 U 并清除 RTIFU，写 0 无效；读出恒为 0	RW
6	RTEU	重载计数器启动 U	写 1 启动重载计数器 U，写 0 无效 读取： ● 0b0：重载计数器 U 已停止 ● 0b1：重载计数器 U 已启动	RW
5	RTICU	清零计数匹配状态 U	写 1 清零 RTIFU 标志位，写 0 无效；读出恒为 0	RW
4	RTIFU	计数匹配状态 U	● 0b0：重载计数器计数值与 0 发生未比较匹配 ● 0b1：重载计数器计数值与 0 发生比较匹配	RO
3	Reserved	保留	-	-
2	RTIDW	重载中断屏蔽 W	● 0b0：重载功能有效时，重载中断 W 输出有效 ● 0b1：重载功能有效时，重载中断 W 输出无效	RW

位/位域	标记	位名	功能描述	读写
1	RTIDV	重载中断屏蔽 V	● 0b0: 重载功能有效时, 重载中断 V 输出有效 ● 0b1: 重载功能有效时, 重载中断 V 输出无效	RW
0	RTIDU	重载中断屏蔽 U	● 0b0: 重载功能有效时, 重载中断 U 输出有效 ● 0b1: 重载功能有效时, 重载中断 U 输出无效	RW

23 紧急刹车模块 (EMB)

23.1 简介

紧急刹车模块是在满足一定条件时产生控制事件输出给定时器，以控制定时器停止或更改向外部电机输出 PWM 信号的功能模块，下列要因用于产生控制事件：

- 外部端口输入电平变化
- PWM 输出端口电平发生同相（同高或同低）
- 电压比较器比较结果
- 系统错误发生
- 写寄存器软件控制

23.2 功能说明

23.2.1 EMB 通道功能框图

EMB 用于向具有 PWM 功能的定时器模块（Timer4，Timer6）在满足一定条件时输出一个控制事件信号，通知定时器模块关闭当前 PWM 输出。EMB 模块具有 7 个 Group，其中 Group0~Group3 用于控制 Timer6，由 Timer6 寄存器设定选择使用；Group4~Group6 用于控制 Timer4，分别对应 Timer4 的 3 个单元。

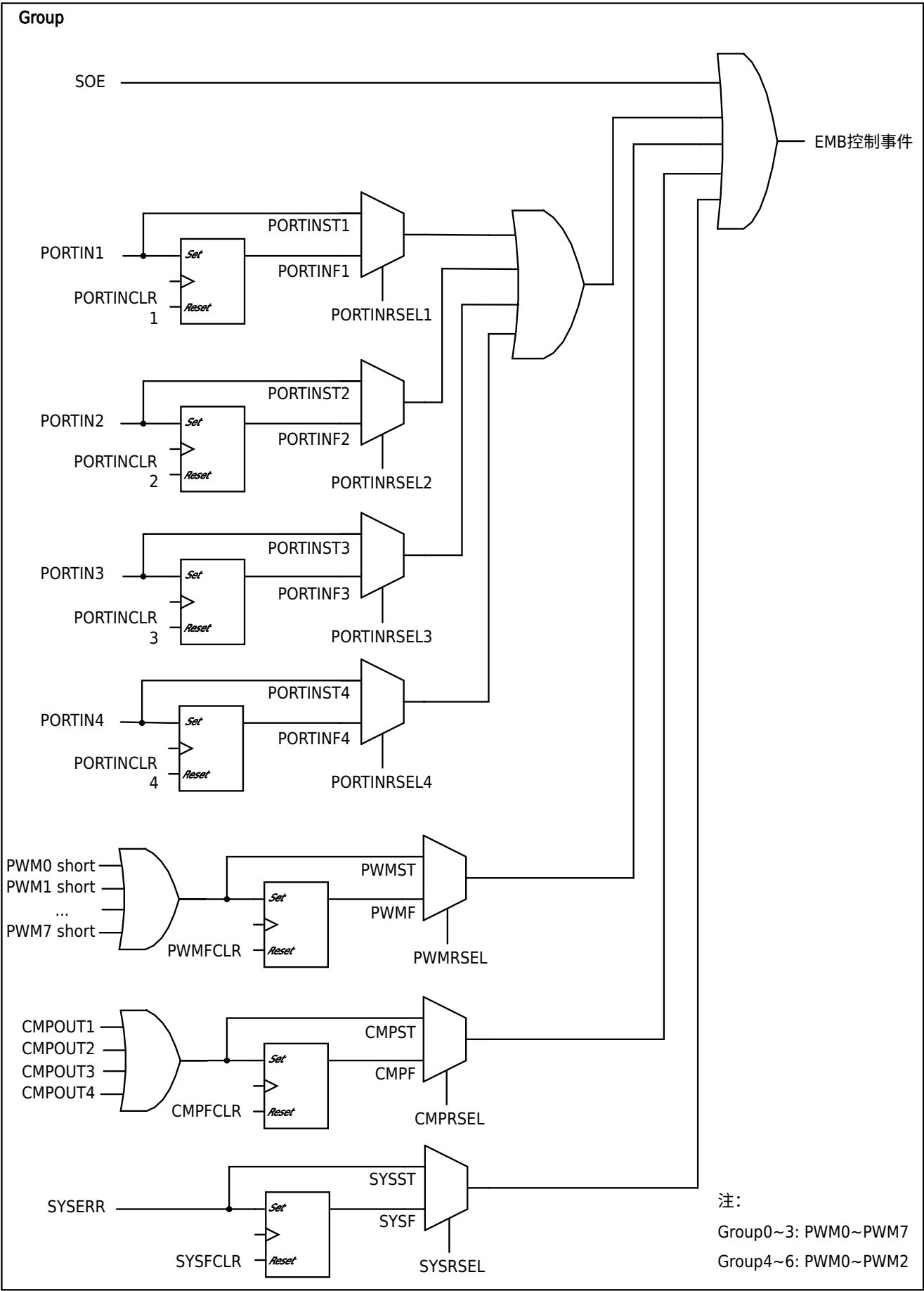


图 23-1 EMB 通道功能框图

23.2.2 EMB 结构框图

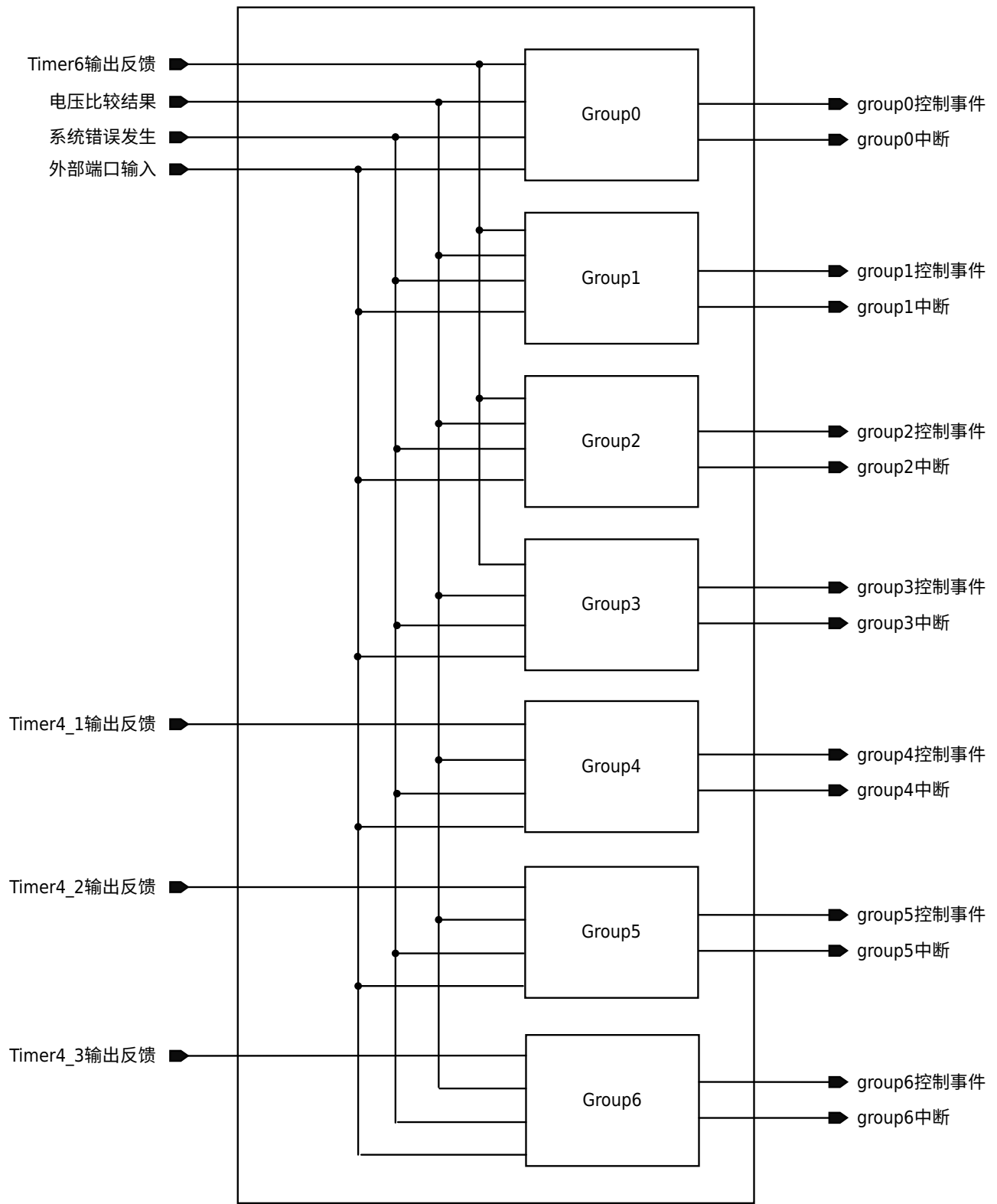


图 23-2 EMB 结构框图

23.2.3 外部端口输入电平变化时控制 PWM 信号输出

EMB 共有 4 个外部端口用于实现当输入电平变化时控制 PWM 信号输出。每个 Group 能够独立设置 4 个外部端口中的 1 个或多个有效，端口分配如下表所示。

表 23-1 EMB 端口分配

端口	PIN 名
EMB_IN1	PA11, PB2, PC5, PD7
EMB_IN2	PA6, PB11, PB12, PE15
EMB_IN3	PA7, PC2, PD9
EMB_IN4	PB15, PD4, PH2, PI4

使用外部端口输入电平变化时控制 PWM 信号输出时，首先将使能位，即 EMB 控制寄存器 EMB_CTL1.PORTINENy (y=1~4) 置为有效，同时通过 EMB_CTL1.INVSELy (y=1~4) 设置在端口电平为高 (INVSEL=0) 时或端口电平为低 (INVSEL=1) 时产生控制信号。根据需要通过 EMB_CTL2.NFENy (y=1~4) 和 EMB_CTL2.NFSELy (y=1~4) 使能滤波功能和设置滤波时钟。

滤波器根据滤波时钟对输入信号进行采样后，采用三次比较一致的方式进行滤波，即当滤波时钟采样到端口上 3 次一致的电平时，该电平被当作有效电平传送到模块内部；小于 3 次一致的电平会被当作外部干扰滤掉，不传送到模块内部。

当端口上产生符合条件的有效电平时，EMB 状态寄存器的端口输入状态 EMB_STAT.PORTINSTy (y=1~4) 被置位，同时端口输入控制标志位 EMB_STAT.PORTINFy (y=1~4) 被置位。当中断许可寄存器 EMB_INTEN.PORTINEN=1 时将产生中断。

EMB 的控制事件通过 EMB 释放方式选择寄存器 EMB_RLSSEL.PORTINRSELy (y=1~4) 选择由 EMB_STAT.PORTINSTy (y=1~4) 或者 EMB_STAT.PORTINFy (y=1~4) 向 Timer6 和 Timer4 输出。Timer6 和 Timer4 在接到控制事件后，可根据寄存器设定将输出端口置为高电平，低电平或高阻态。释放控制时，如果 EMB_RLSSEL.PORTINRSELy (y=1~4) =1，当端口输入电平转为无效时，EMB_STAT.PORTINSTy (y=1~4) 将自动清零，同时控制事件将立即释放。如果 EMB_RLSSEL.PORTINRSELy (y=1~4) =0，当端口输入电平转为无效后，需要写 EMB 状态复位寄存器 EMB_STATCLR.PORTINFCLR 使 EMB_STAT.PORTINF 清 0，将控制事件释放。

23.2.4 PWM 输出端口电平发生同相（同高或同低）时停止 PWM 信号输出

使用时首先将使能位，即 EMB 控制寄存器 EMB_CTL1.PWMSENy (Group=0~3, y=0~7 或 Group=4~6, y=0~2) 置为有效，同时通过 EMB 控制寄存器 EMB_CTL2.PWMLVy (Group=0~3, y=0~7 或 Group=4~6, y=0~2) 选择需要监控的有效电平。

完成设置后，EMB 监控 Timer6 和 Timer4 的互补 PWM 输出信号，当输出信号出现同高或同低情况时，EMB 状态寄存器的 PWM 输出状态 EMB_STAT.PWMST 被置位，同时 PWM 输出同相位控制标志位 EMB_STAT.PWMSF 被置位。当中断许可寄存器 EMB_INTEN.PWMEN=1 时将产生中断。

EMB 的控制事件通过 EMB 释放方式选择寄存器 EMB_RLSSEL.PWMRSEL 选择由 EMB_STAT.PWMST 或者 EMB_STAT.PWMSF 向 Timer6 和 Timer4 输出。Timer6 和 Timer4 在接到控制事件后可根据寄存器设定将输出端口置为高电平、低电平或高阻态。

释放控制时，如果 EMB_RLSSEL.PWMRSEL=1，当 PWM 输出同高或同低状态解除，EMB_STAT.PWMST 将自动清零，同时控制事件将立即释放。如果 EMB_RLSSEL.PWMST=0，当 PWM 输出同高或同低状态解除后，需要写 EMB 状态复位寄存器 EMB_STATCLR.PWMSFCLR 使 EMB_STAT.PWMSF 清 0 后，将控制事件释放。Group0~3 可用于监控 Timer6 的互补 PWM 输出信号，Group4 用于监控 Timer4_1 的互补 PWM 输出信号，Group5 用于监控 Timer4_2 的互补 PWM 输出信号，Group6 用于监控 Timer4_3 的互补 PWM 输出信号。

表 23-2 EMB Group 对照表

端口名	功能	对应 Group	EMB_CTL1 控制位	EMB_CTL2 控制位
TMR6_<t>_PWMA (<t>为 1~8)	Timer6 的互补 PWM 输出信号	Group0	PWMSEN7~PWMSE N0	PWMLV7~PWMLV0
TMR6_<t>_PWMB (<t>为 1~8)		Group1		
		Group2		
		Group3		
TMR4_1_OUH	Timer4_1 的互补 PWM 输出信号	Group4	PWMSEN2	PWMLV2
TMR4_1_OUL				
TMR4_1_OVH			PWMSEN1	PWMLV1
TMR4_1_OVL				
TMR4_1_OWH			PWMSEN0	PWMLV0
TMR4_1_OWL				
TMR4_2_OUH	Timer4_2 的互补 PWM 输出信号	Group5	PWMSEN2	PWMLV2
TMR4_2_OUL				
TMR4_2_OVH			PWMSEN1	PWMLV1
TMR4_2_OVL				
TMR4_2_OWH			PWMSEN0	PWMLV0
TMR4_2_OWL				
TMR4_3_OUH	Timer4_3 的互补 PWM 输出信号	Group6	PWMSEN2	PWMLV2
TMR4_3_OUL				
TMR4_3_OVH			PWMSEN1	PWMLV1
TMR4_3_OVL				
TMR4_3_OWH			PWMSEN0	PWMLV0
TMR4_3_OWL				

23.2.5 根据电压比较器比较结果停止 PWM 信号输出

EMB 的每个 Group 能够根据电压比较器的 4 组比较结果向 Timer6 和 Timer4 发送控制事件信号。电压比较器输出结果的设定请参见《电压比较器（CMP）》章节相关描述。

使用时首先将使能位，即 EMB 控制寄存器 EMB_CTL1.CMPENy（y=1~4）置为有效。当电压比较器比较结果标志位被置起时，EMB 状态寄存器的电压比较器状态 EMB_STAT.CMPST 被置位，同时 EMB 电压比较器控制标志位 EMB_STAT.CMPF 被置位。当中断许可寄存器 EMB_INTEN.CMPINTEN=1 时将产生中断。

EMB 的控制事件通过 EMB 释放方式选择寄存器 EMB_RLSSEL.CMPRSEL 选择由 EMB_STAT.CMPST 或者 EMB_STAT.CMPF 向 Timer6 和 Timer4 输出。Timer6 和 Timer4 在接到控制事件信号后可根据寄存器设定将输出端口置为高电平、低电平或高阻态。

释放控制时, 如果 EMB_RLSSEL.CMPRSEL=1, 当电压比较器结果转为无效时, EMB_STAT.CMPST 将自动清零, 同时控制事件将立即释放。如果 EMB_RLSSEL.CMPRSEL=0, 当电压比较器结果转为无效后, 需要写 EMB 状态复位寄存器 EMB_STATCLR.CMPFCLR 使 EMB_STAT.CMPF 清 0 后, 将控制事件释放。

23.2.6 系统错误发生时停止 PWM 信号输出

EMB 的每个 Group 能够在系统错误发生时向 Timer6 和 Timer4 发送通知信号。系统错误信号来自 PVD 检测、Cortex-M4 锁定信号、SRAM ECC 校验错误、外部振荡器停止、Flash ECC 校验错误和 FCM 频率测量异常。

使用时首先将使能位, 即 EMB 控制寄存器 EMB_CTL1.SYSEN 置为有效, 设置 EMB_CTL1.PVDEN、LOCKUPEN、SRAMECCERREN、OSCSTPEN、FLASHECCERREN、FCMERREN 位, 使能需要检测系统错误源。当选择的系统错误源标志被置起时, EMB 状态寄存器的系统错误状态 EMB_STAT.SYSST 被置位, 同时 EMB 系统错误控制标志位 EMB_STAT.SYSF 被置位。当中断许可寄存器 EMB_INTEN.SYSINTEN=1 时还将产生中断。

EMB 的控制事件通过 EMB 释放方式选择寄存器 EMB_RLSSEL.SYSRSEL 选择由 EMB_STAT.SYSST 或者 EMB_STAT.SYSF 向 Timer6 和 Timer4 输出。Timer6 和 Timer4 在接到控制事件信号后可根据寄存器设定将输出端口置为高电平, 低电平或高阻态。

释放控制时, 如果 EMB_RLSSEL.SYSRSEL=1, 当系统错误信号转为无效时, EMB_STAT.SYSST 将自动清零, 同时控制事件将立即释放。如果 EMB_RLSSEL.SYSRSEL=0, 当系统错误信号转为无效后, 需要写 EMB 状态复位寄存器 EMB_STATCLR.SYSFCLR 使 EMB_STAT.SYSF 清 0 后, 将控制事件释放。

23.2.7 写寄存器软件控制 PWM 信号输出

EMB 的软件输出使能控制寄存器 (EMB_SOE) 能够允许用户通过软件直接置位和复位的方式向 Timer6 和 Timer4 发送控制信号, 软件控制 PWM 输出时不会产生中断请求。

23.3 寄存器

23.3.1 寄存器总表

- EMB Group0 基地址：0x40017C00
- EMB Group1 基地址：0x40017C20
- EMB Group2 基地址：0x40017C40
- EMB Group3 基地址：0x40017C60
- EMB Group4 基地址：0x40017C80
- EMB Group5 基地址：0x40017CA0
- EMB Group6 基地址：0x40017CC0



说明

EMB 包含 7 个 Group，分别为 Group0~6。

表 23-3 紧急刹车模块（EMB）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	EMB_CTL1	EMB 控制寄存器 1（Group0~3）	32
0x00	EMB_CTL1	EMB 控制寄存器 1（Group4~6）	32
0x04	EMB_CTL2	EMB 控制寄存器 2（Group0~3）	32
0x04	EMB_CTL2	EMB 控制寄存器 2（Group4~6）	32
0x08	EMB_SOE	EMB 软件输出使能控制寄存器	32
0x0C	EMB_STAT	EMB 状态寄存器	32
0x10	EMB_STATCLR	EMB 状态复位寄存器	32
0x14	EMB_INTEN	EMB 中断许可寄存器	32
0x18	EMB_RLSSEL	EMB 控制 PWM 输出释放方式选择寄存器	32

23.3.2 EMB 控制寄存器 1 (EMB_CTL1) (Group0~3)

该寄存器为单次写入寄存器，即复位之后仅能够写入一次。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PVDEN	LOCKUPEN	Reserved	SRAMECCERREN	OSCSTPEN	FLASHECCERREN	INVSEL4	INVSEL3	INVSEL2	INVSEL1	Reserved	PORTINEN4	PORTINEN3	PORTINEN2	PORTINEN1	FCMERREN	Reserved	PWMSEN7	PWMSEN6	PWMSEN5	PWMSEN4	PWMSEN3	PWMSEN2	PWMSEN1	PWMSEN0	SYSEN	CMPEN4	CMPEN3	CMPEN2	CMPEN1		
Access	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	PVDEN	PVD 检测控制使能	0b0: PVD 检测结果输出控制无效 0b1: PVD 检测结果输出控制有效	RW
30	LOCKUPEN	Cortex-M4 Lockup 控制使能	0b0: Lockup 发生时输出控制无效 0b1: Lockup 发生时输出控制有效	RW
29	Reserved	保留	-	-
28	SRAMECCERREN	SRAM ECC 校验错误控制使能	0b0: SRAM ECC 校验错误发生时输出控制无效 0b1: SRAM ECC 校验错误发生时输出控制有效	RW
27	OSCSTPEN	振荡器停止控制使能	0b0: 振荡器停止发生时输出控制无效 0b1: 振荡器停止发生时输出控制有效	RW
26	FLASHECCERREN	FLASH ECC 校验错误控制使能	0b0: FLASH ECC 校验错误发生时输出控制无效 0b1: FLASH ECC 校验错误发生时输出控制有效	RW
25	INVSEL4	端口输入 4 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
24	INVSEL3	端口输入 4 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
23	INVSEL2	端口输入 2 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
22	INVSEL1	端口输入 1 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
21:20	Reserved	保留	-	-
19	PORTINEN4	端口输入 4 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
18	PORTINEN3	端口输入 3 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW

位/位域	标记	位名	功能描述	读写
17	PORTINEN2	端口输入 2 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
16	PORTINEN1	端口输入 1 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
15	FCMERREN	FCM 频率测量异常控制使能	0b0: FCM 频率测量异常时输出控制无效 0b1: FCM 频率测量异常时输出控制有效	RW
14:13	Reserved	保留	-	-
12	PWMSEN7	TMR6_8_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
11	PWMSEN6	TMR6_7_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
10	PWMSEN5	TMR6_6_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
9	PWMSEN4	TMR6_5_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
8	PWMSEN3	TMR6_4_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
7	PWMSEN2	TMR6_3_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
6	PWMSEN1	TMR6_2_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
5	PWMSEN0	TMR6_1_PWMA/B 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
4	SYSEN	系统错误控制使能	0b0: 系统错误发生时输出控制无效 0b1: 系统错误发生时输出控制有效	RW
3	CM PEN4	CMP4 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
2	CM PEN3	CMP3 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
1	CM PEN2	CMP2 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
0	CM PEN1	CMP1 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW

23.3.3 EMB 控制寄存器 1 (EMB_CTL1) (Group4~6)

该寄存器为单次写入寄存器，即复位之后仅能够写入一次。

Offset	Bit Position																																
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	PVDEN	LOCKUPEN	Reserved	SRAMECCERREN	OSCSTPEN	FLASHECCERREN	INVSEL4	INVSEL3	INVSEL2	INVSEL1	Reserved	PORTINEN4	PORTINEN3	PORTINEN2	PORTINEN1	FCMERREN	Reserved								PWMSEN2	PWMSEN1	PWMSEN0	SYSEN	CMPEN4	CMPEN3	CMPEN2	CMPEN1	
Access	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW																	

位/位域	标记	位名	功能描述	读写
31	PVDEN	PVD 检测控制使能	0b0: PVD 检测结果输出控制无效 0b1: PVD 检测结果输出控制有效	RW
30	LOCKUPEN	Cortex-M4 Lockup 控制使能	0b0: Lockup 发生时输出控制无效 0b1: Lockup 发生时输出控制有效	RW
29	Reserved	保留	-	-
28	SRAMECCERREN	SRAM ECC 校验错误控制使能	0b0: SRAM ECC 校验错误发生时输出控制无效 0b1: SRAM ECC 校验错误发生时输出控制有效	RW
27	OSCSTPEN	振荡器停止控制使能	0b0: 振荡器停止发生时输出控制无效 0b1: 振荡器停止发生时输出控制有效	RW
26	FLASHECCERREN	FLASH ECC 校验错误控制使能	0b0: FLASH ECC 校验错误发生时输出控制无效 0b1: FLASH ECC 校验错误发生时输出控制有效	RW
25	INVSEL4	端口输入 4 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
24	INVSEL3	端口输入 3 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
23	INVSEL2	端口输入 2 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
22	INVSEL1	端口输入 1 有效电平选择	0b0: 高电平有效 0b1: 低电平有效	RW
21:20	Reserved	保留	-	-
19	PORTINEN4	端口输入 4 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
18	PORTINEN3	端口输入 3 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW

位/位域	标记	位名	功能描述	读写
17	PORTINEN2	端口输入 2 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
16	PORTINEN1	端口输入 1 控制使能	0b0: 端口输入控制无效 0b1: 端口输入控制有效	RW
15	FCMERREN	FCM 频率测量异常控制使能	0b0: FCM 频率测量异常时输出控制无效 0b1: FCM 频率测量异常时输出控制有效	RW
14:8	Reserved	保留	-	-
7	PWMSEN2	TMR4 <t>_OUH/L 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
6	PWMSEN1	TMR4 <t>_OVH/L 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
5	PWMSEN0	TMR4 <t>_OWH/L 短路输出控制使能	0b0: 短路时输出控制无效 0b1: 短路时输出控制有效	RW
4	SYSEN	系统错误控制使能	0b0: 系统错误发生时输出控制无效 0b1: 系统错误发生时输出控制有效	RW
3	CMPEN4	CMP4 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
2	CMPEN3	CMP3 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
1	CMPEN2	CMP2 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW
0	CMPEN1	CMP1 电压比较器比较结果控制使能	0b0: 电压比较器比较结果输出控制无效 0b1: 电压比较器比较结果输出控制有效	RW



说明

- Group4: <t>为 1
- Group5: <t>为 2
- Group6: <t>为 3

23.3.4 EMB 控制寄存器 2 (EMB_CTL2) (Group0~3)

该寄存器为单次写入寄存器，即复位之后仅能够写入一次。

Offset	Bit Position																																									
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
Reset	0x00000000																																									
Name	Reserved		NFEN4		NFSEL4		NFEN3		NFSEL3		NFEN2		NFSEL2		NFEN1		NFSEL1		Reserved								PWMLV7		PWMLV6		PWMLV5		PWMLV4		PWMLV3		PWMLV2		PWMLV1		PWMLV0	
Access					RW		RW		RW		RW		RW		RW		RW												RW		RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27	NFEN4	端口输入 4 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
26:25	NFSEL4	端口输入 4 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
24	NFEN3	端口输入 3 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
23:22	NFSEL3	端口输入 3 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
21	NFEN2	端口输入 2 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
20:19	NFSEL2	端口输入 2 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
18	NFEN1	端口输入 1 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
17:16	NFSEL1	端口输入 1 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
15:8	Reserved	保留	-	-
7	PWMLV7	TMR6_8_PWMA/B 输出有效电平选择	0b0: 低电平为有效电平 0b1: 高电平为有效电平	RW
6	PWMLV6	TMR6_7_PWMA/B 输出有效电平选择	0b0: 低电平为有效电平 0b1: 高电平为有效电平	RW

位/位域	标记	位名	功能描述	读写
5	PWMLV5	TMR6_6_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW
4	PWMLV4	TMR6_5_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW
3	PWMLV3	TMR6_4_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW
2	PWMLV2	TMR6_3_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW
1	PWMLV1	TMR6_2_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW
0	PWMLV0	TMR6_1_PWMA/B 输出有效电平选择	● 0b0: 低电平为有效电平 ● 0b1: 高电平为有效电平	RW

23.3.5 EMB 控制寄存器 2 (EMB_CTL2) (Group4~6)

该寄存器为单次写入寄存器，即复位之后仅能够写入一次。

Offset	Bit Position																																			
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved		NFEN4	NFSEL4		NFEN3	NFSEL3		NFEN2	NFSEL2		NFEN1	NFSEL1		Reserved																PWMLV2		PWMLV1		PWMLV0	
Access			RW	RW		RW	RW		RW	RW		RW	RW																		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27	NFEN4	端口输入 4 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
26:25	NFSEL4	端口输入 4 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
24	NFEN3	端口输入 3 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
23:22	NFSEL3	端口输入 3 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
21	NFEN2	端口输入 2 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
20:19	NFSEL2	端口输入 2 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
18	NFEN1	端口输入 1 数字滤波器使能	0b0: 滤波器无效 0b1: 滤波器有效	RW
17:16	NFSEL1	端口输入 1 数字滤波器滤波时钟选择	0b00: 使用 PCLK1 滤波 0b01: 使用 PCLK1 的 8 分频滤波 0b10: 使用 PCLK1 的 32 分频滤波 0b11: 使用 PCLK1 的 128 分频滤波	RW
15:3	Reserved	保留	-	-
2	PWMLV2	TMR4_<t>_OUH/L 输出有效电平选择	0b0: 低电平为有效电平 0b1: 高电平为有效电平	RW
1	PWMLV1	TMR4_<t>_OVH/L 输出有效电平选择	0b0: 低电平为有效电平 0b1: 高电平为有效电平	RW

位/位域	标记	位名	功能描述	读写
0	PWMLV0	TMR4_<t>_OWH/L 输出有效电平选择	0b0: 低电平为有效电平 0b1: 高电平为有效电平	RW

说明

- Group4: <t>为 1
- Group5: <t>为 2
- Group6: <t>为 3

23.3.6 EMB 软件输出使能控制寄存器 (EMB_SOE)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															SOE
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	SOE	软件控制输出	0b0: PWM 正常输出 0b1: PWM 停止输出	RW

23.3.7 EMB 状态寄存器 (EMB_STAT)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														PORTINST4	PORTINST3	PORTINST2	PORTINST1	Reserved	PORTINF4	PORTINF3	PORTINF2	PORTINF1	SYSST	CMPST	PWMST	Reserved	SYSF	CMPF	PWMSF	Reserved	
Access															RO	RO	RO	RO		RO	RO	RO	RO	RO	RO	RO		RO	RO	RO		

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	PORTINST4	端口输入 4 控制状态	0b0: 端口输入控制处于无效电平状态 0b1: 端口输入控制处于有效电平状态	RO
16	PORTINST3	端口输入 3 控制状态	0b0: 端口输入控制处于无效电平状态 0b1: 端口输入控制处于有效电平状态	RO
15	PORTINST2	端口输入 2 控制状态	0b0: 端口输入控制处于无效电平状态 0b1: 端口输入控制处于有效电平状态	RO
14	PORTINST1	端口输入 1 控制状态	0b0: 端口输入控制处于无效电平状态 0b1: 端口输入控制处于有效电平状态	RO
13:12	Reserved	保留	-	-
11	PORTINF4	端口输入 4 控制标志位	EMB_STAT.PORTINST4 为 1 时置位 EMB_STATCLR.PORTINFCLR4 写 1 时清 0	RO
10	PORTINF3	端口输入 3 控制标志位	EMB_STAT.PORTINST3 为 1 时置位 EMB_STATCLR.PORTINFCLR3 写 1 时清 0	RO
9	PORTINF2	端口输入 2 控制标志位	EMB_STAT.PORTINST2 为 1 时置位 EMB_STATCLR.PORTINFCLR2 写 1 时清 0	RO
8	PORTINF1	端口输入 1 控制标志位	EMB_STAT.PORTINST1 为 1 时置位 EMB_STATCLR.PORTINFCLR1 写 1 时清 0	RO
7	SYSST	系统错误状态	0b0: 系统错误未发生 0b1: 系统错误发生	RO
6	CMPST	电压比较器状态	0b0: 电压比较器比较结果标志位未被置位 0b1: 电压比较器比较结果标志位被置位	RO
5	PWMST	PWM 输出状态	0b0: 没有发生 PWM 输出同相 0b1: 发生 PWM 输出同相	RO
4	Reserved	保留	-	-
3	SYSF	系统错误标志位	EMB_STAT.SYSST 为 1 时置位 EMB_STATCLR.SYSFCLR 写 1 时清 0	RO
2	CMPF	电压比较器控制标志位	EMB_STAT.CMPST 为 1 时置位 EMB_STATCLR.CMPFCLR 写 1 时清 0	RO
1	PWMSF	PWM 输出同相位控制标志位	EMB_STAT.PWMST 为 1 时置位 EMB_STATCLR.PWMSCLR 写 1 时清 0	RO

位/位域	标记	位名	功能描述	读写
0	Reserved	保留	-	-

23.3.8 EMB 状态复位寄存器 (EMB_STATCLR)

Offset	Bit Position																																																											
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																												
Reset	0x00000000																																																											
Name	Reserved																				PORTINFCLR4		PORTINFCLR3		PORTINFCLR2		PORTINFCLR1		Reserved				SYSFCLR		CMPFCLR		PWMSFCLR		Reserved																					
WO																					WO		WO		WO		WO						WO		WO																									
Access																																									WO		WO		WO		WO						WO		WO		WO			
																																									WO		WO		WO		WO						WO		WO		WO			

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	PORTINFCLR4	复位 EMB_STAT.PORTINF4	写 1 将 EMB_STAT.PORTINF4 清 0 (当 EMB_STAT.PORTINST4=0 时) ; 写 0 无效	WO
10	PORTINFCLR3	复位 EMB_STAT.PORTINF3	写 1 将 EMB_STAT.PORTINF3 清 0 (当 EMB_STAT.PORTINST3=0 时) ; 写 0 无效	WO
9	PORTINFCLR2	复位 EMB_STAT.PORTINF2	写 1 将 EMB_STAT.PORTINF2 清 0 (当 EMB_STAT.PORTINST2=0 时) ; 写 0 无效	WO
8	PORTINFCLR1	复位 EMB_STAT.PORTINF1	写 1 将 EMB_STAT.PORTINF1 清 0 (当 EMB_STAT.PORTINST1=0 时) ; 写 0 无效	WO
7:4	Reserved	保留	-	-
3	SYSFCLR	复位 EMB_STAT.SYSF	写 1 将 EMB_STAT.SYSF 清 0 (当 EMB_STAT.SYSST=0 时) ; 写 0 无效	WO
2	CMPFCLR	复位 EMB_STAT.CMPF	写 1 将 EMB_STAT.CMPF 清 0 (当 EMB_STAT.CMPST=0 时) ; 写 0 无效	WO
1	PWMSFCLR	复位 EMB_STAT.PWMSF	写 1 将 EMB_STAT.PWMSF 清 0 (当 EMB_STAT.PWMST=0 时) ; 写 0 无效	WO
0	Reserved	保留	-	-

23.3.9 EMB 中断许可寄存器 (EMB_INTEN)

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				PORTININTEN4	PORTININTEN3	PORTININTEN2	PORTININTEN1	Reserved				SYSINTEN	CMPINTEN	PWMSINTEN	Reserved
Access																					RW	RW	RW	RW					RW	RW	RW	

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	PORTININTEN4	端口输入 4 控制中断使能	● 0b0: 端口输入控制不产生中断 ● 0b1: 端口输入控制产生中断	RW
10	PORTININTEN3	端口输入 3 控制中断使能	● 0b0: 端口输入控制不产生中断 ● 0b1: 端口输入控制产生中断	RW
9	PORTININTEN2	端口输入 2 控制中断使能	● 0b0: 端口输入控制不产生中断 ● 0b1: 端口输入控制产生中断	RW
8	PORTININTEN1	端口输入 1 控制中断使能	● 0b0: 端口输入控制不产生中断 ● 0b1: 端口输入控制产生中断	RW
7:4	Reserved	保留	-	-
3	SYSINTEN	系统错误控制中断使能	● 0b0: 系统错误控制不产生中断 ● 0b1: 系统错误控制产生中断	RW
2	CMPINTEN	电压比较器比较结果控制中断使能	● 0b0: 电压比较器比较结果控制不产生中断 ● 0b1: 电压比较器比较结果控制产生中断	RW
1	PWMSINTEN	PWM 输出同相位控制中断使能	● 0b0: PWM 输出同相位控制不产生中断 ● 0b1: PWM 输出同相位控制产生中断	RW
0	Reserved	保留	-	-

23.3.10 EMB 控制 PWM 输出释放方式选择寄存器 (EMB_RLSSEL)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				PORTINRSEL4	PORTINRSEL3	PORTINRSEL2	PORTINRSEL1	Reserved				SYSRSEL	CMRSEL	PWMRSEL	Reserved
Access																					RW	RW	RW	RW						RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	PORTINRSEL4	端口输入 4 控制释放方式选择	0b0: EMB_STAT.PORTINF4=0 时释放 PWM 输出控制 0b1: EMB_STAT.PORTINST4=0 时释放 PWM 输出控制	RW
10	PORTINRSEL3	端口输入 3 控制释放方式选择	0b0: EMB_STAT.PORTINF3=0 时释放 PWM 输出控制 0b1: EMB_STAT.PORTINST3=0 时释放 PWM 输出控制	RW
9	PORTINRSEL2	端口输入 2 控制释放方式选择	0b0: EMB_STAT.PORTINF2=0 时释放 PWM 输出控制 0b1: EMB_STAT.PORTINST2=0 时释放 PWM 输出控制	RW
8	PORTINRSEL1	端口输入 1 控制释放方式选择	0b0: EMB_STAT.PORTINF1=0 时释放 PWM 输出控制 0b1: EMB_STAT.PORTINST1=0 时释放 PWM 输出控制	RW
7:4	Reserved	保留	-	-
3	SYSRSEL	系统错误控制释放方式选择	0b0: EMB_STAT.SYSF=0 时释放 PWM 输出控制 0b1: EMB_STAT.SYSST=0 时释放 PWM 输出控制	RW
2	CMRSEL	电压比较器比较结果控制释放方式选择	0b0: EMB_STAT.CMPF=0 时释放 PWM 输出控制 0b1: EMB_STAT.CMPST=0 时释放 PWM 输出控制	RW
1	PWMRSEL	PWM 输出控制释放方式选择	0b0: EMB_STAT.PWMSF=0 时释放 PWM 输出控制 0b1: EMB_STAT.PWMST=0 时释放 PWM 输出控制	RW
0	Reserved	保留	-	-

24 通用定时器 (TimerA)

24.1 简介

通用定时器 A (TimerA) 是一个具有 16 位计数宽度、4 路 PWM 输出的定时器。该定时器支持三角波和锯齿波两种波形模式，可生成各种 PWM 波形（单边对齐 PWM、双边对称 PWM）；支持计数器同步启动；比较基准值寄存器支持缓存功能；支持单元间级联实现 32 位计数；支持 2 相正交编码计数和 3 相正交编码计数。本系列产品搭载 12 个单元 TimerA，最大可实现 48 路 PWM 输出。

24.2 主要特性

TimerA 基本的功能及特性如下表所示。

表 24-1 TimerA 的基本功能及特性

波形模式	锯齿波、三角波
基本功能	递加、递减计数方向
	同步启动计数器
	基准值缓存功能
	32 位级联计数
	正交编码计数
	4 路 PWM 输出
	比较匹配事件输出
中断类型	比较匹配中断
	周期匹配中断

24.3 功能说明

24.3.1 TimerA 框图

TimerA 的基本框图如所示。图中“<t>”为单元编号，即“<t>”为 1~12，本章节后文提到“<t>”时均指单元编号，不再赘述。

本系列产品搭载的 12 个单元的 TimerA，其中单元 1~单元 4 的总线时钟和计数时钟均为 PCLK0（此时，框图中所示 PCLK 是指 PCLK0）；单元 5~单元 12 的总线时钟和计数时钟为 PCLK1（此时，框图中所示 PCLK 是指 PCLK1）。各单元的计数分频时钟源及端口数字滤波的采样基准时钟源与其总线时钟和计数时钟相同。

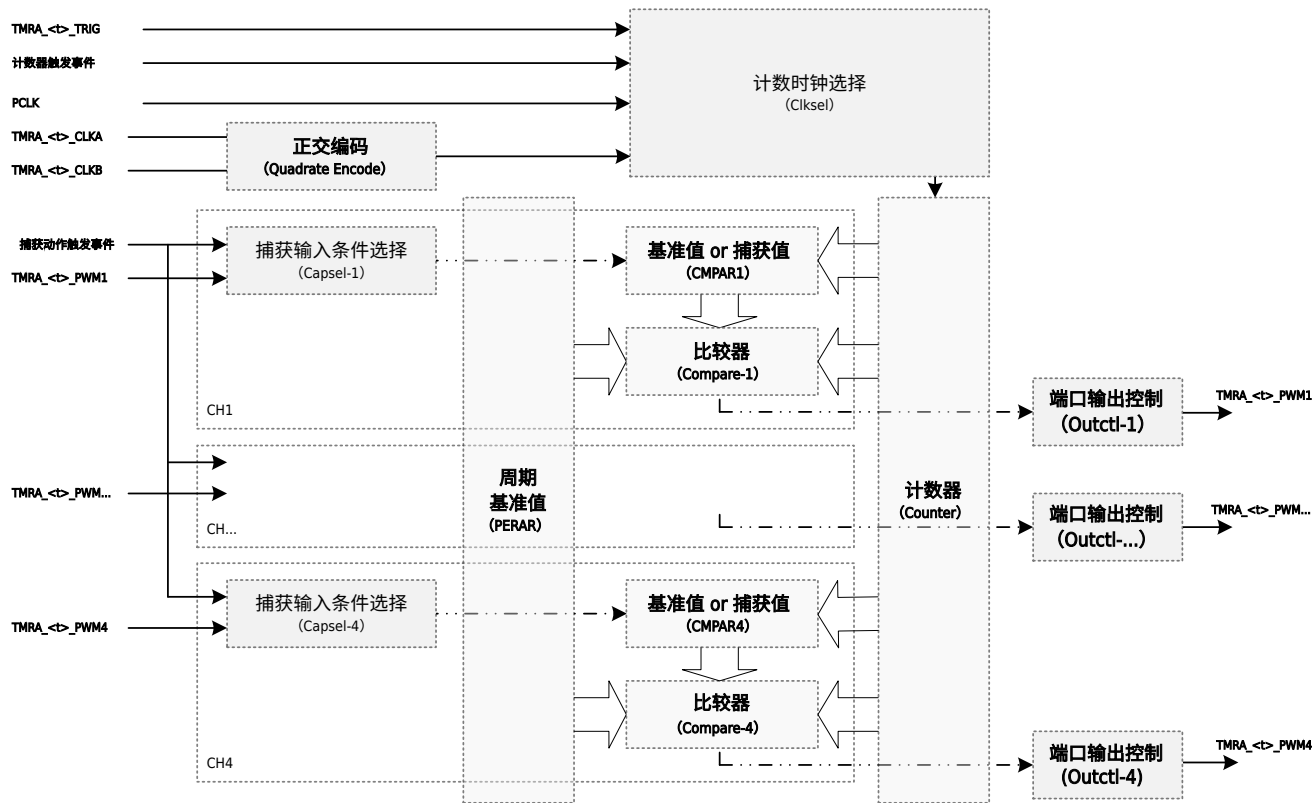


图 24-1 TimerA 基本框图

下表为 TimerA 的输入输出端口列表。

表 24-2 TimerA 端口列表

端口名	方向	功能
TMRA_<t>_PWMm	输入/输出	捕获输入事件端口或 PWM 输出端口（m=1~4）
TMRA_<t>_CLKA	输入	正交编码计数事件输入端口
TMRA_<t>_CLKB		
TMRA_<t>_TRIG	输入	硬件触发启动、停止、清零事件输入端口

24.3.2 对称单元

TimerA 的单元 1 和单元 2 互为对称单元，单元 3 和单元 4 互为对称单元，以此类推。对称单元间可联动，实现级联计数、同步启动等，详见以下各章节描述。

24.3.3 基本动作

24.3.3.1 波形模式

TimerA 有 2 种基本计数波形模式，锯齿波模式和三角波模式。两种波形模式的基本波形如下：

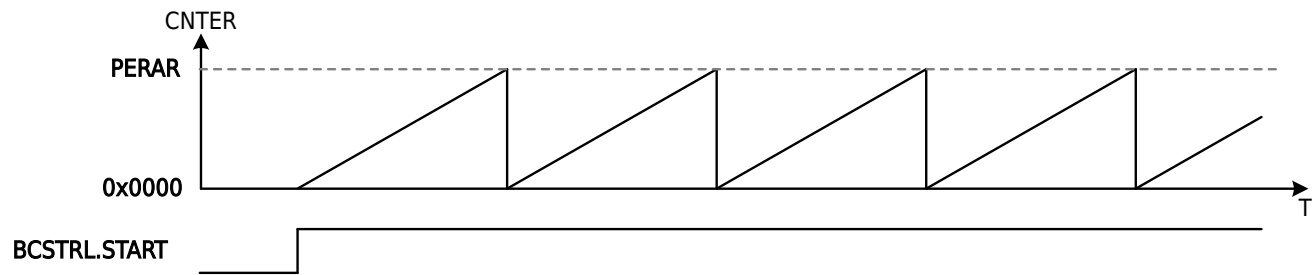


图 24-2 锯齿波波形 (递加计数)

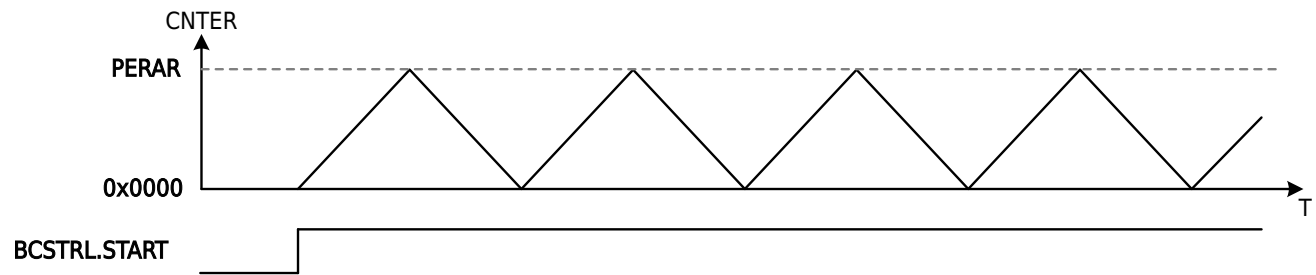


图 24-3 三角波波形

24.3.3.2 比较输出

各个 TimerA 单元内部均含有 4 个通道的比较输出 (TMRA_<t>_PWMm)，可在计数值与比较基准值比较匹配时输出指定的电平。TMRA_CMPARm 寄存器分别对应了 TMRA_<t>_PWMm 输出端口的计数比较基准值。当定时器的计数值和 TMRA_CMPARm 相等时，TMRA_<t>_PWMm 端口输出指定的电平 (m=1~4)。

TMRA_<t>_PWMm 端口的计数开始时的电平、计数停止时的电平、计数比较匹配时的电平、计数周期匹配时的电平等，可通过端口控制寄存器 (PCONRm) 的 STAC、STPC、CMPC、PERC、FORC 位设定控制 (m=1~4)。下图为单元 1 的比较输出动作例。

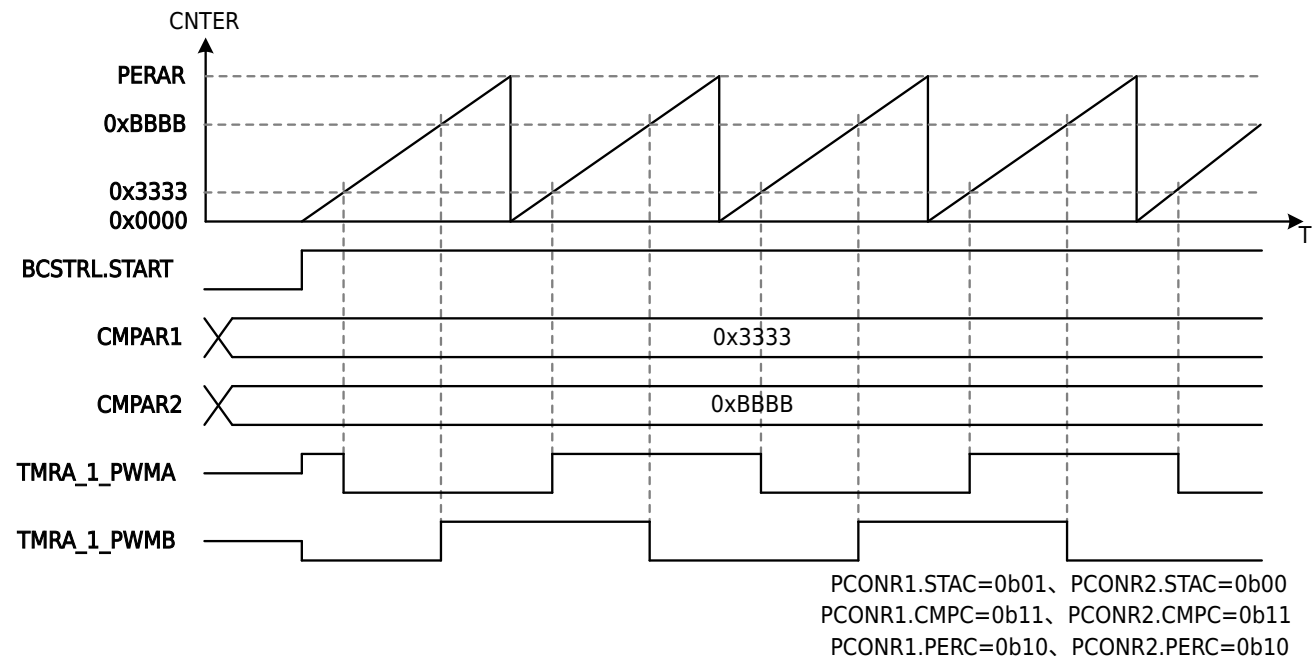


图 24-4 比较输出动作

24.3.3.3 捕获输入

各个 TimerA 单元的每个 PWM 输出通道都具有捕获输入功能，用于保存捕获到的计数值。设定捕获控制寄存器 (CCONRm) 的 CCONR.CAPMD 位为 1，捕获输入功能变为有效。此时选择对应的捕获输入条件且该条件有效时，当前的计数值就被保存到相应的寄存器 (CMPARm) 中 (m=1~4)。

捕获输入条件可以选择内部捕获动作触发事件（通过 TMRA_TRGSELx 寄存器选择，具体的寄存器内容详见《自动运行系统 (AOS)》章节“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3)”）、TMRA_<t>_PWMm 端口输入等，具体的条件选择可通过捕获控制寄存器 (CCONRm) 的 HICP 位来设定 (m=1~4)。

内部触发事件选择寄存器为 12 个单元的 TimerA 所共有，具体分配如下表所示。

表 24-3 内部触发事件 TRGSEL 选择关系对应表

TimerA 单元	事件分类	所对应的选择寄存器
TimerA_1	计数器触发事件：HCONR、HCUPR、HCDOR 所指的 TRGSEL 选择事件	TMRA_TRGSEL0
TimerA_2		TMRA_TRGSEL1
TimerA_3		TMRA_TRGSEL2
TimerA_4		TMRA_TRGSEL3
TimerA_5		TMRA_TRGSEL0
TimerA_6		TMRA_TRGSEL1
TimerA_7		TMRA_TRGSEL2
TimerA_8		TMRA_TRGSEL3
TimerA_9		TMRA_TRGSEL0
TimerA_10		TMRA_TRGSEL1
TimerA_11		TMRA_TRGSEL2
TimerA_12		TMRA_TRGSEL3
TimerA_1	捕获动作触发事件：CCONR 中所指的 TRGSEL 选择事件	TMRA_TRGSEL1
TimerA_2		TMRA_TRGSEL0
TimerA_3		TMRA_TRGSEL3
TimerA_4		TMRA_TRGSEL2
TimerA_5		TMRA_TRGSEL1
TimerA_6		TMRA_TRGSEL0
TimerA_7		TMRA_TRGSEL3
TimerA_8		TMRA_TRGSEL2
TimerA_9		TMRA_TRGSEL1
TimerA_10		TMRA_TRGSEL0

TimerA 单元	事件分类	所对应的选择寄存器
TimerA_11	捕获动作触发事件：CCONR 中所指的 TRGSEL 选择事件	TMRA_TRGSEL3
TimerA_12		TMRA_TRGSEL2

下图为捕获输入动作例。

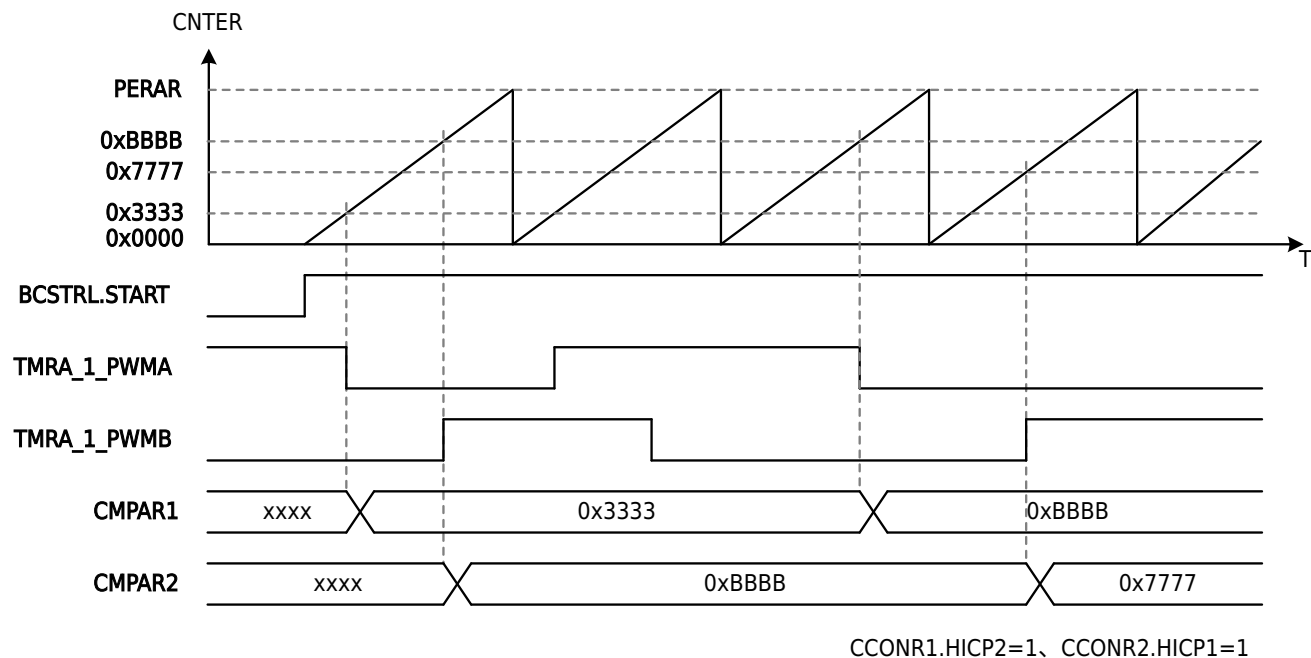


图 24-5 捕获输入动作

24.3.4 时钟源选择

TimerA 的计数时钟可以有以下几种选择：

- 1. PCLK 的 1、2、4、8、16、32、64、128、256、512、1024 分频（BCSTR.CKDIV[3:0]设定）
- 2. TMRA_<t>_TRIG 端口事件输入（HCUPR[9:8]或 HCDOR[9:8]设定）
- 3. 内部计数器触发事件输入（HCUPR[10]或 HCDOR[10]设定）
- 4. 对称单元的计数上溢或计数下溢事件输入（HCUPR[12:11]或 HCDOR[12:11]设定）
- 5. TMRA_<t>_CLKA、TMRA_<t>_CLKB 的端口正交编码输入（HCUPR[7:0]或 HCDOR[7:0]设定）

计数时钟源选择 1 时为软件计数模式，计数时钟源选择 2、3、4、5 时为硬件计数模式，硬件计数模式需确保 BCSTR.CKDIV 为 0。计数时钟选择 4 时多用于三相正交编码计数的公转计数模式（参见“[位置溢出计数](#)”和“[混合计数](#)”章节），也可用于级联计数。上述描述可以看到，2、3、4、5 时钟互相独立，可分别设定有效或无效，并且当选择 2、3、4、5 时钟时，1 时钟自动无效。

24.3.5 计数方向

TimerA 的定时器计数方向可通过软件方式改变。不同波形模式，改变计数方向的方法略有不同。

24.3.5.1 锯齿波计数方向

锯齿波计数模式时，计数方向可在定时器计数中或计数停止时设定，设定之后方向立即改变。

24.3.5.2 三角波计数方向

三角波计数模式时，计数方向在计数停止时设定，正在计数时设定方向无效。

24.3.6 数字滤波

各个单元的 TMRA_<t>_CLKA、TMRA_<t>_CLKB、TMRA_<t>_TRIG、TMRA_<t>_PWMm（捕获输入功能时）端口输入都有数字滤波功能。各端口的滤波功能的使能和滤波时钟的选择可通过设定滤波控制寄存器（FCONR）和捕获控制寄存器（CCONRm）的对应位来实现（m=1~4）。
在滤波采样基准时钟采样到端口上 3 次一致的电平时，该电平被当作有效电平传送到模块内部；小于 3 次一致的电平会被当作外部干扰滤掉，不传送到模块内部。下图为 TMRA_1_CLKA 端口滤波动作例。

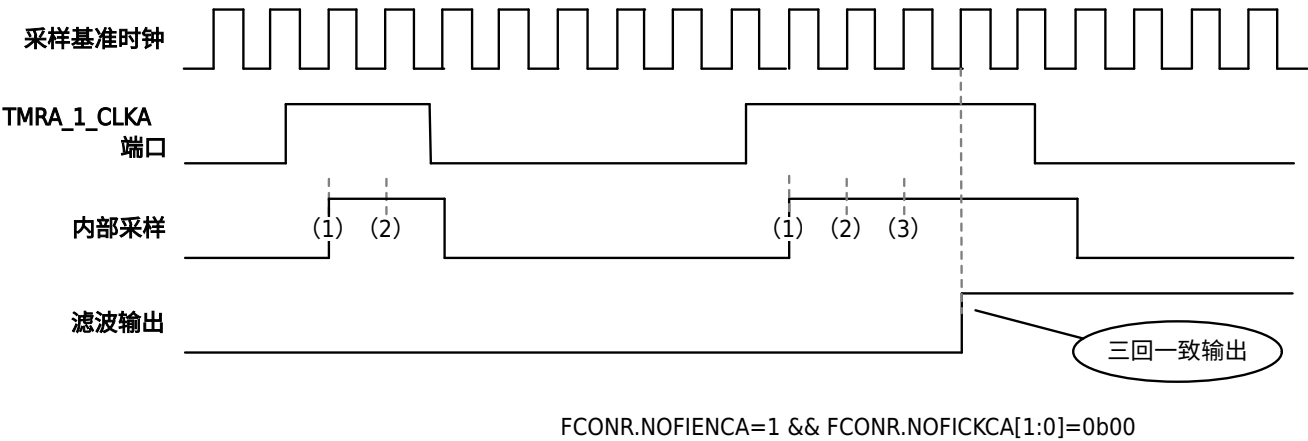


图 24-6 时钟输入端口的滤波功能

24.3.7 同步启动

TimerA 的对称单元，可以实现软件同步启动或硬件同步启动，由奇数单元同步启动偶数单元。
当对称单元中偶数单元的 BCSTRL.SYNST 位设定为 1 时，偶数单元与奇数单元的同步启动功能有效。此时，若软件设定奇数单元的 BCSTRL.START 位为 1，被同步的偶数单元的计数器开始软件同步计数；若硬件设定奇数单元的 HCONR.HSTA1~0 中任意位为 1，且其对应硬件事件发生时，被同步的偶数单元的计数器开始硬件同步计数。在选择硬件同步计数启动功能时，被同步的偶数单元的 HCONR.HSTA1~0 的对应位也必须设定为有效。
下图为设定单元 2 的 BCSTRL.SYNST=1 时的软件同步启动例。

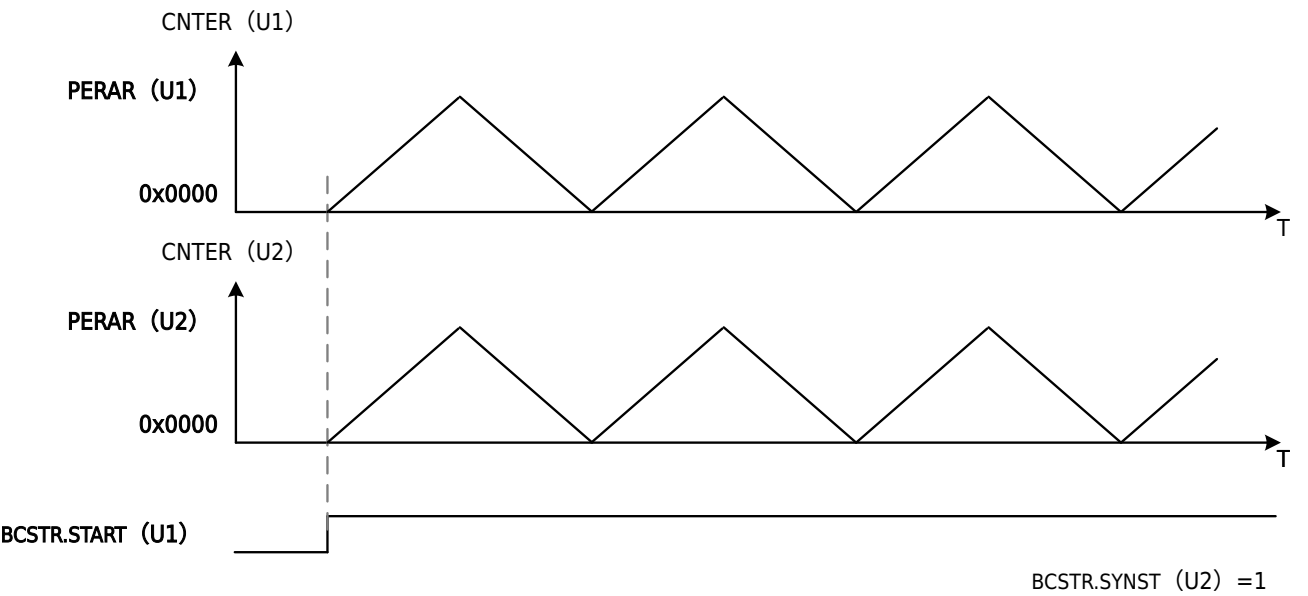


图 24-7 软件同步动作

另外，产品配置了计数同步使能寄存器 TMR_SYNENRx (x=1~2)，用于同步启动产品所搭载的所有 Timer (包括 Timer0、Timer2、Timer4、Timer6、TimerA)。使用方法和寄存器说明参考《通用定时器 (Timer0)》章节。

24.3.8 缓存功能

TimerA 的共计 4 个比较基准寄存器 (CMPARm) (m=1~4) 可以成对实现缓存功能。即 CMPAR2 作为 CMPAR1 的缓存基准值、CMPAR4 作为 CMPAR3 的缓存基准值。缓存控制寄存器 (BCONRm) (m=1~2) 分别实现对两组缓存功能的控制。

当缓存控制寄存器 (BCONRm) (m=1~2) 的 BEN 位被置位时，缓存功能变为有效。计数器计数到特定时间点时就发生一次缓存传送 (CMPAR2/4->CMPAR1/3)。该“特定时间点”有以下几种情况：

- 1. 硬件计数模式时，计数到上溢点 (HCUPR[12:0] != 0 时) 或下溢点 (HCDOR[12:0] != 0 时) 或发生硬件清零计数器动作
- 2. 锯齿波计数模式 (BCSTR.MODE=0) 时，计数器计数到上溢点 (BCSTR.DIR=1 时) 或下溢点 (BCSTR.DIR=0 时) 或发生硬件清零计数器动作
- 3. 三角波计数模式 (BCSTR.MODE=1) 时，计数到峰点 (BCSTR.DIR=1 && BCONRm.BSE0=1 时) (m=1~2)
- 4. 三角波计数模式 (BCSTR.MODE=1) 时，计数到谷点 (BCSTR.DIR=0 && BCONRm.BSE1=1 时) (m=1~2)

当缓存控制寄存器 (BCONRm) 的 BSEN 位被置位时，缓存特殊功能变为有效 (m=1~2)。对称单元的 CNTER 被读时，本单元的 CMPAR 发生一次缓存传送 (CMPAR2/4->CMPAR1/3)。

下图是锯齿波模式时的缓存传送示意图。

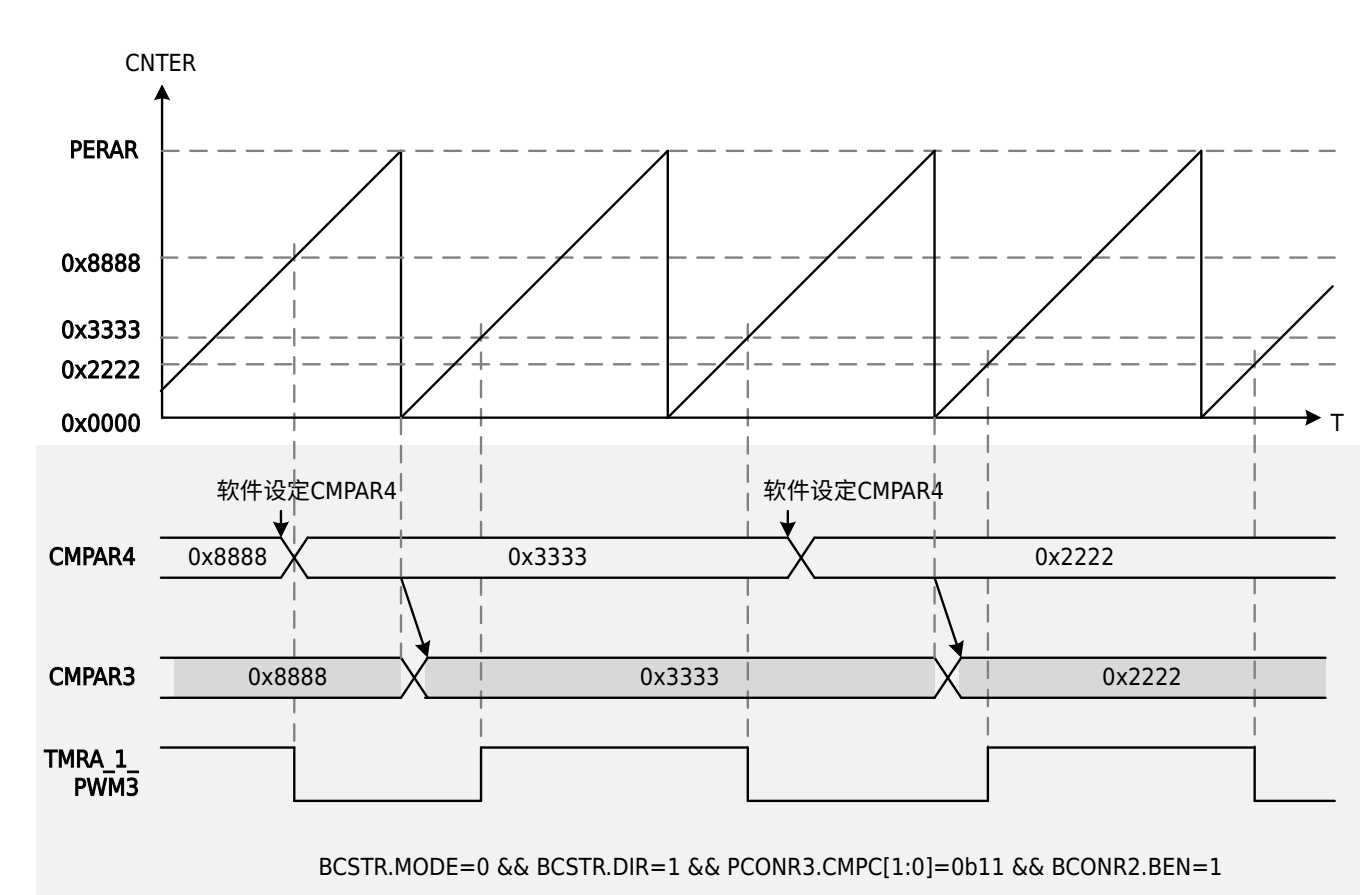


图 24-8 锯齿波模式时缓存动作

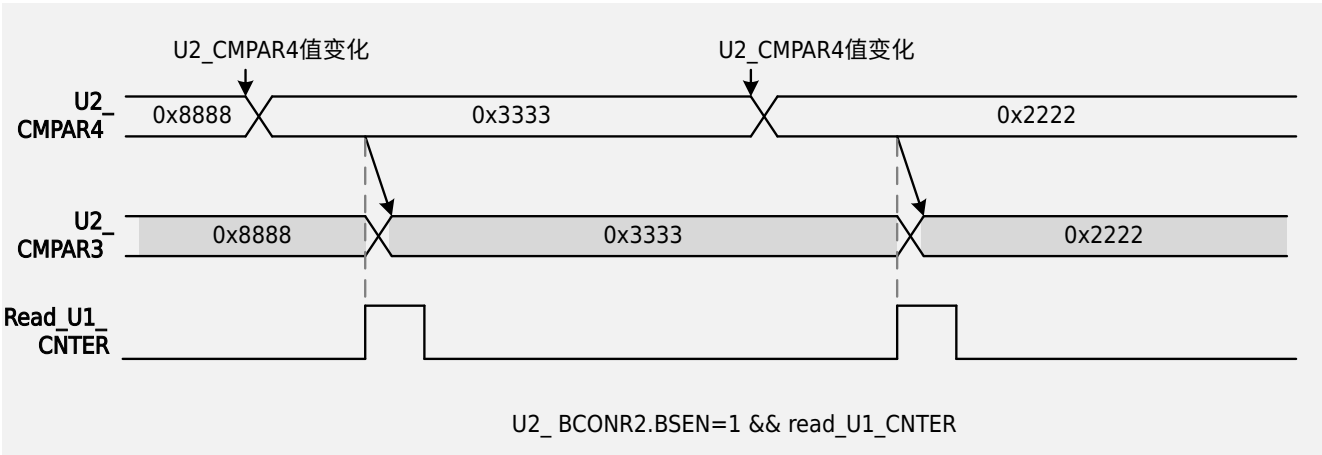


图 24-9 锯齿波模式时缓存与特俗缓存动作

24.3.9 级联计数

在“计数时钟源选择”章节中，当时钟源选择 4 时，本单元的计数时钟就选择为对称单元的计数溢出（计数上溢或计数下溢）事件，此时两个单元级联合并可实现 32 位计数器。在级联计数中，对称单元的 CNTER 为低 16 位计数器，本单元的 CNTER 为高 16 位计数器。

例如，在三角波向上计数模式（BCSTR.MODE=0、BCSTR.DIR=1）时，设定单元 1 的计数时钟为 PCLK，设定单元 2 的计数时钟源为单元 1 的计数上溢事件（单元 2 的 TMRA_HCUPR.HCUP11=1），启动单元 2、1 计数（先启动单元 2，再启动单元 1）就实现级联计数。单元 1 的 CNTER 位低 16 位计数器，单元 2 的 CNTER 为高 16 位计数器。下图是单元 1、2 级联计数的示意图。

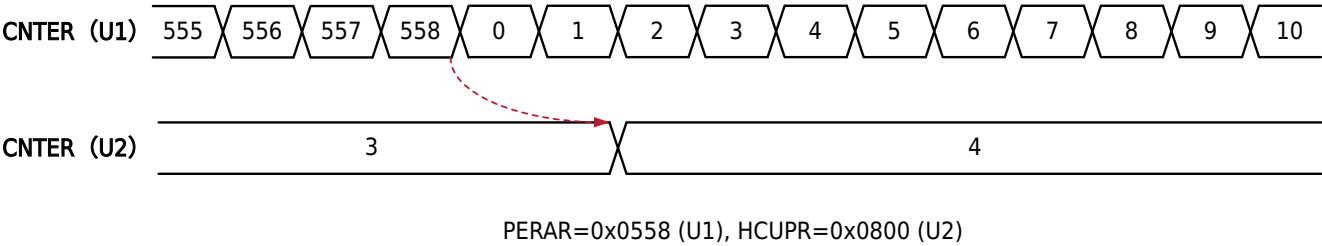


图 24-10 32 位级联计数动作

24.3.10 通用 PWM 输出

24.3.10.1 单边对齐 PWM 输出

锯齿波计数模式时，一个单元内的各个通道通过各种端口控制设定，可以实现单边对齐 PWM 输出（在计数周期点对齐）。例如，设定在比较基准值比较匹配时翻转（PCONR.CMPC=0b11）、在周期基准值比较匹配时翻转（PCONR.PERC=0b11），就可实现在一个周期内产生单边对齐的 PWM 输出。

下图是锯齿波模式下，单边对齐 PWM 输出波形例。

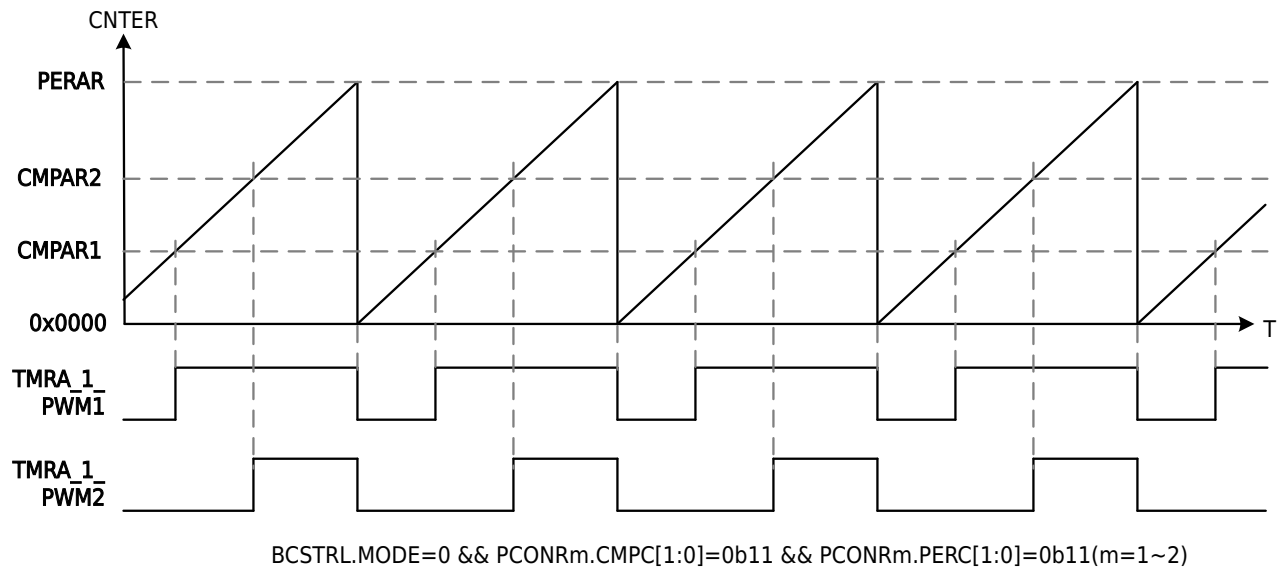


图 24-11 单边对齐 PWM 输出例

24.3.10.2 双边对称 PWM 输出

三角波计数模式时，一个单元内的各个通道通过各种端口控制设定，可以实现双边对称 PWM 输出（以计数峰点对称）。根据通道间的输出关系，可以实现独立的 PWM 输出或互补的 PWM 输出。

下图是三角波模式下，通道 1、2、3、4 的双边对称 PWM 输出波形例。其中，通道 1、2 可以作为一对互补 PWM 输出。

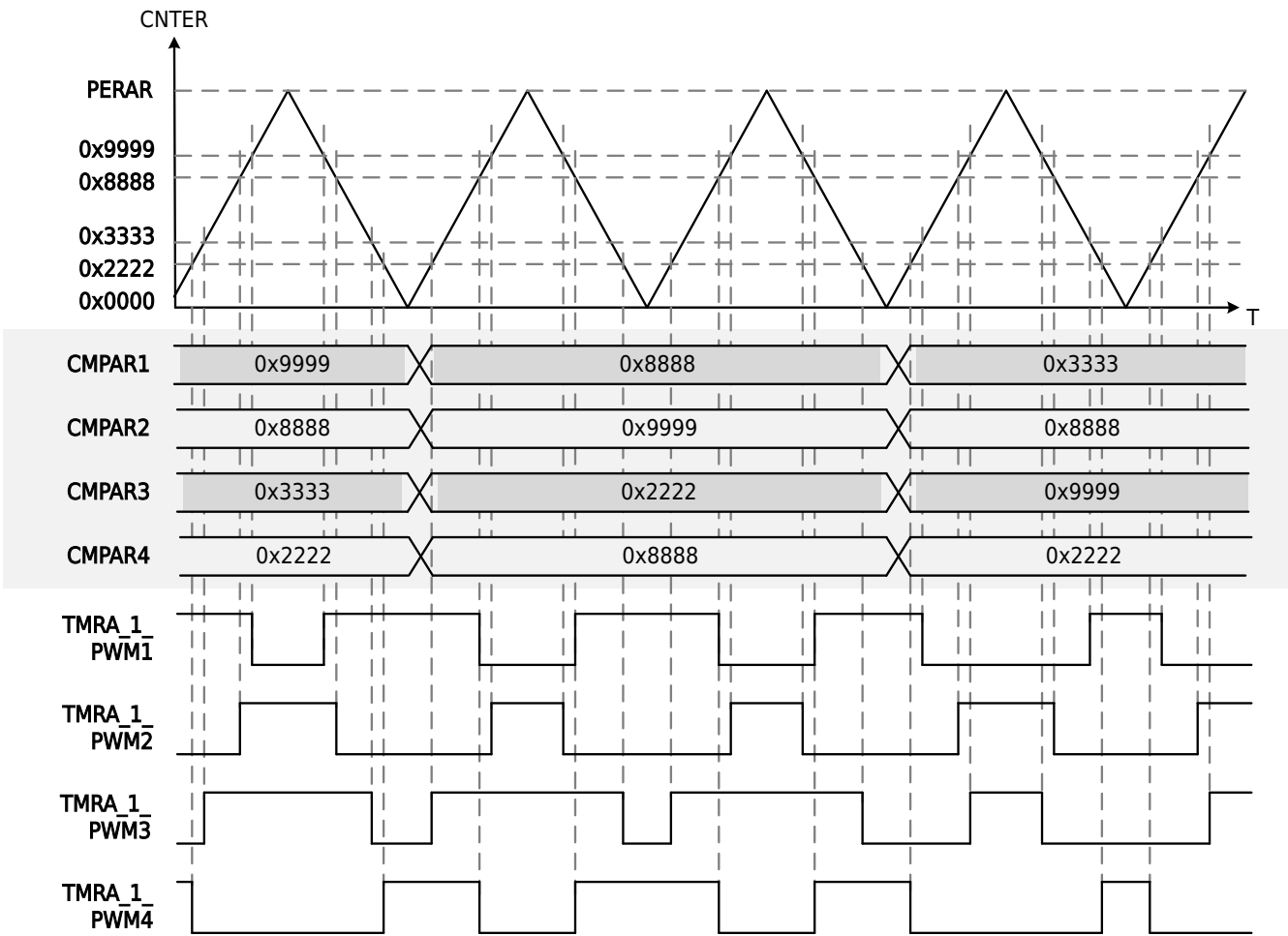


图 24-12 双边对称 PWM 输出例

24.3.11 正交编码计数

将 TMRA_<t>_CLKA 输入看作 AIN 输入、TMRA_<t>_CLKB 输入看作 BIN 输入、TMRA_<t>_TRIG 输入看作 ZIN 输入，TimerA 就可以实现三路输入的正交编码计数。

每个单元的 AIN、BIN 单独动作可以实现位置计数模式；两个单元的 AIN、BIN、ZIN 组合动作可以实现公转计数模式，其中用于位置计数的单元称之为位置计数单元、用于公转计数的单元称之为公转计数单元。公转计数模式时，每两个单元间互相组合（单元 1、2 组合；单元 3、4 组合；单元 5、6 组合；单元 7、8 组合；单元 9、10 组合；单元 11、12 组合），组合内位置计数单元和公转计数单元可以任意指定。

AIN 和 BIN 的计数条件使能通过设定硬件递加事件选择寄存器（HCUPR）和硬件递减事件选择寄存器（HCDOR）中 TMRA_<t>_CLKA 和 TMRA_<t>_CLKB 的正交关系来实现；ZIN 的输入动作通过设定位置计数单元的硬件触发事件选择寄存器（HCONR）的清零使能位实现位置定时器清零、通过设定公转单元的硬件递加事件选择寄存器（HCUPR）实现公转定时器计数。

24.3.11.1 位置计数模式

正交编码位置计数模式，是指根据 AIN、BIN 的输入实现基本计数功能、相位差计数功能和方向计数功能。

基本计数

基本计数动作是根据 AIN 或 BIN 端口的输入时钟进行计数。

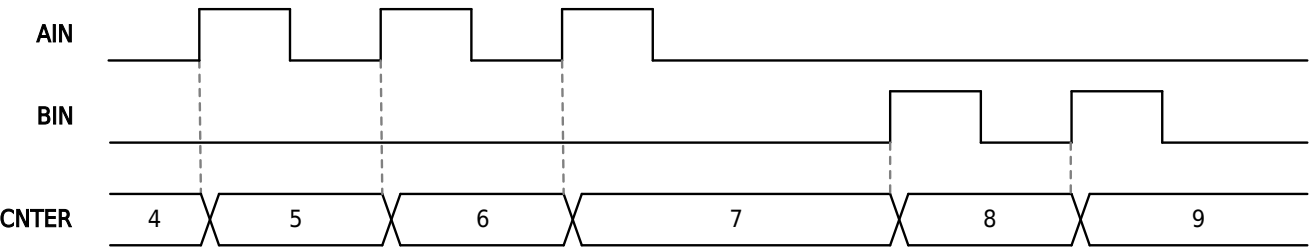
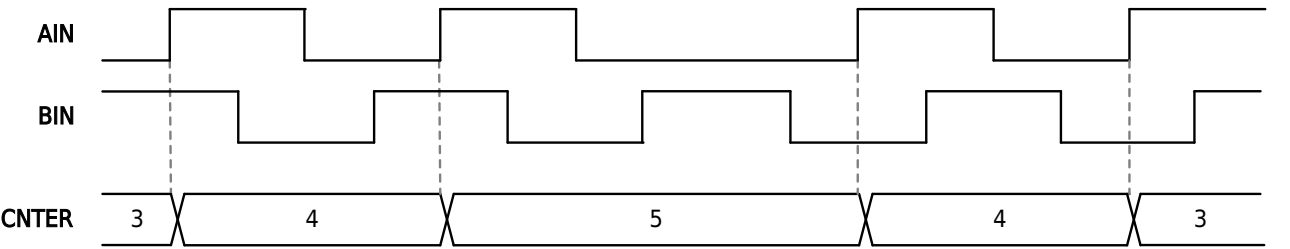


图 24-13 位置模式-基本计数

相位差计数

相位差计数是指根据 AIN 和 BIN 的相位关系进行计数。根据设定的不同，可以实现 1 倍计数、2 倍计数、4 倍计数等。



HCUPR.HCUP6=0b1 && HCDOR.HCDO4=0b1

图 24-14 位置计数模式-相位差计数（1 倍计数）

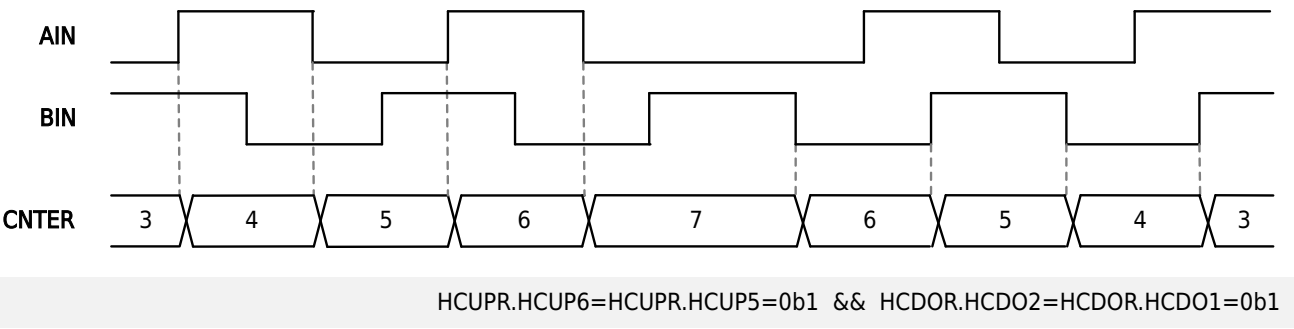


图 24-15 位置计数模式-相位差计数（2 倍计数）

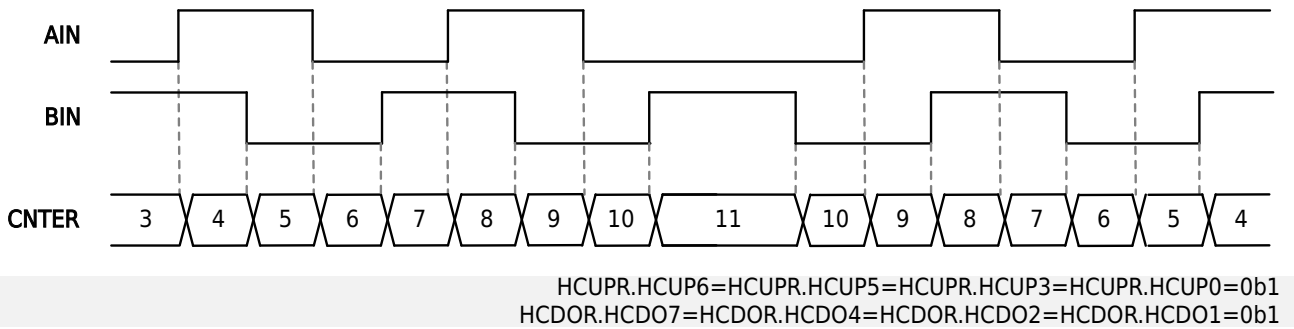


图 24-16 位置计数模式-相位差计数（4 倍计数）

方向计数

方向计数是指将 AIN 的输入状态设定为方向控制，将 BIN 的输入作为时钟计数。

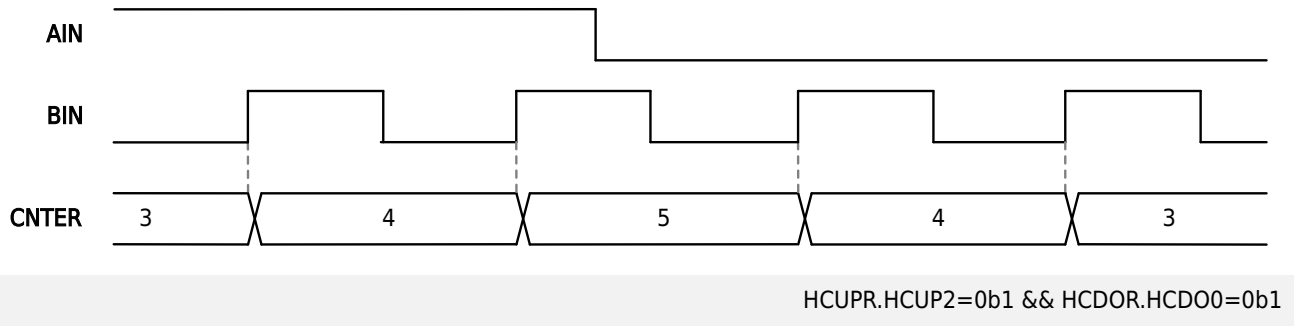


图 24-17 位置计数模式-方向计数

24.3.11.2 公转计数模式

正交编码公转模式，是指在 AIN、BIN 计数的基础上，加入 ZIN 的输入事件以实现对公转圈数等的判断。公转计数模式时根据公转定时器的计数方式，可实现 Z 相计数功能、位置溢出计数功能和混合计数功能。

Z 相计数

Z 相计数是指根据 ZIN 的输入，公转计数单元进行计数，同时将位置计数单元清零的计数动作。

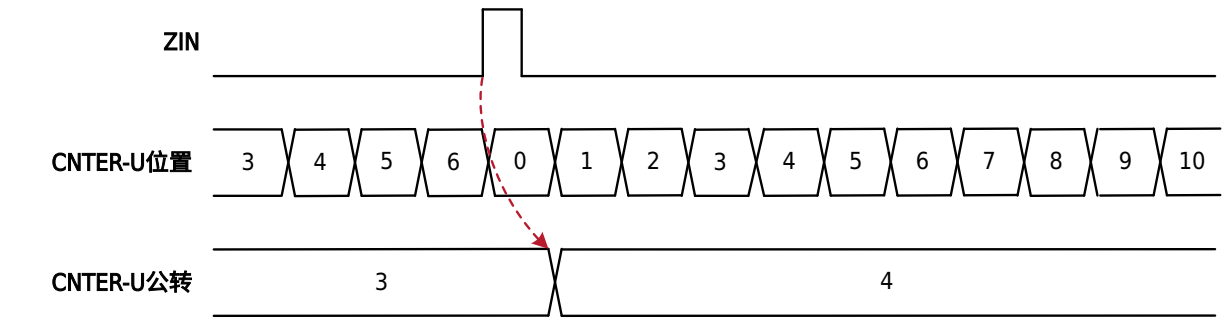


图 24-18 公转计数模式-Z 相计数

位置溢出计数

位置溢出计数是指位置计数单元计数发生上溢或下溢时，产生一个溢出事件，从而触发公转计数单元的定时器进行一次计数（在该计数方式时 ZIN 的输入不进行公转计数单元的计数动作和位置计数单元的清零动作）。

公转计数单元的硬件递加（递减）事件选择寄存器（HCUPR 或 HCDOR）的递加（递减）事件 Bit12~11 位使能，位置计数单元的溢出事件就可以触发公转计数单元实现一次计数。

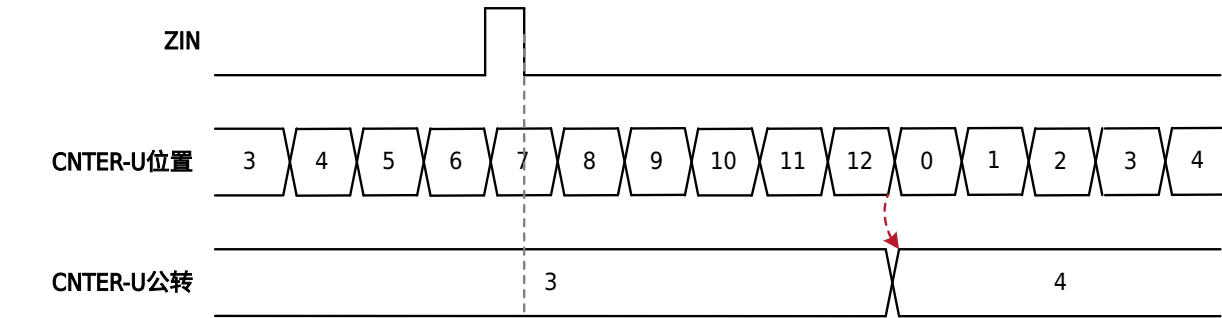


图 24-19 公转计数模式-位置溢出计数

混合计数

混合计数是指上述 Z 相计数和位置溢出计数两种计数方式合并起来的计数动作，其实现方式也是上述两种计数方式的组合。

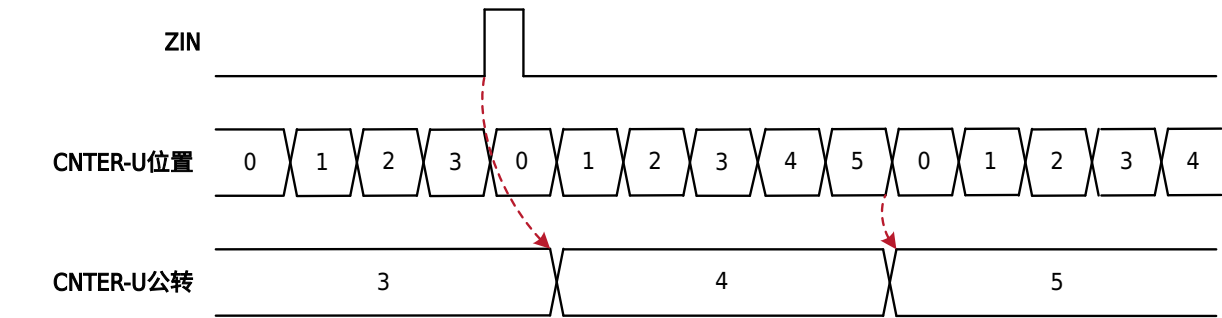


图 24-20 公转计数模式-混合计数

24.3.12 中断和事件

TimerA 含有 3 个中断输出和 3 个事件输出，分别是 1 个比较匹配中断和事件、2 个周期匹配中断和事件。

24.3.12.1 比较匹配中断及事件

比较基准值寄存器（CMPARm）与计数值比较发生比较匹配时，状态标志寄存器（STFLR）中的对应位（STFLR.CMPFm）会被置为 1。此时，若中断控制寄存器（ICONR）的对应位（ICONR.ITENm）设定为 1，则对应的中断请求（TMRA_<t>_CMP）会被触发；若事件控制寄存器（ECONR）的对应位（ECONR.ETENm）设定为 1，则对应的事件请求（TMRA_<t>_CMP）会被触发（m=1~4）。

捕获控制寄存器 (CCONRm) 选择的捕获输入有效条件产生时, 捕获输入动作发生。此时, 若中断控制寄存器 (ICONR) 的对应位 (ICONR.ITENm) 设定为 1, 则对应的中断请求 (TMRA_<t>_CMP) 被触发; 若事件控制寄存器 (ECONR) 的对应位 (ECONR.ETENm) 设定为 1, 则对应的事件请求 (TMRA_<t>_CMP) 会被触发 (m=1~4)。

每个单元内部的 4 个基准值的比较匹配中断和比较匹配事件并非独立输出, 比较匹配中断通过“或逻辑”汇总成一个中断输出至中断模块 (参见《中断控制器 (INTC)》章节), 比较匹配事件通过“或逻辑”汇总成一个事件输出用于选择触发别的模块。

24.3.12.2 周期匹配中断及事件

锯齿波模式递加计数至上溢点、锯齿波模式递减计数至下溢点、三角波模式计数至谷点或峰点, 控制状态寄存器 (BCSTRH) 的 OVFF 或 UDF 位会被置为 1。此时, 若 BCSTRH.ITENOVF 或 BCSTRH.ITENUDF 位设定为 1 使能中断, 则在对应的周期点可触发周期匹配中断 (TMRA_<t>_OVF 和 TMRA_<t>_UDF) 输出至中断模块 (INTC); 周期匹配事件则无对应的使能位控制, 在对应的计数周期点就触发周期匹配事件 (TMRA_<t>_OVF 和 TMRA_<t>_UDF) 输出用于选择触发别的模块。

24.4 寄存器

24.4.1 寄存器总表

- TimerA_1 基地址: 0x4003A000
- TimerA_2 基地址: 0x4003A400
- TimerA_3 基地址: 0x4003A800
- TimerA_4 基地址: 0x4003AC00
- TimerA_5 基地址: 0x40026000
- TimerA_6 基地址: 0x40026400
- TimerA_7 基地址: 0x40026800
- TimerA_8 基地址: 0x40026C00
- TimerA_9 基地址: 0x40027000
- TimerA_10 基地址: 0x40027400
- TimerA_11 基地址: 0x40027800
- TimerA_12 基地址: 0x40027C00



说明

TimerA 为定时器名称, TimerA_<t> (<t>: 1~12) 为该定时器单元。

表 24-4 通用定时器 (TimerA) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	TMRA_CNTER	通用计数值寄存器	16
0x0004	TMRA_PERAR	周期基准值寄存器	16
0x003C+0x4*m	TMRA_CMPARm	比较基准值寄存器 m (m=1~4)	16
0x0080	TMRA_BCSTRL	控制状态寄存器 L	8
0x0081	TMRA_BCSTRH	控制状态寄存器 H	8
0x0090	TMRA_ICONR	中断控制寄存器	16
0x0094	TMRA_ECONR	事件控制寄存器	16
0x0098	TMRA_FCONR	滤波控制寄存器	16
0x009C	TMRA_STFLR	状态标志寄存器	16
0x00B8+0x8*m	TMRA_BCONRm	缓存控制寄存器 m (m=1~2)	16
0x00FC+0x4*m	TMRA_CCONRm	捕获控制寄存器 m (m=1~4)	16
0x013C+0x4*m	TMRA_PCONRm	端口控制寄存器 m (m=1~4)	16
0x0084	TMRA_HCONR	硬件触发事件选择寄存器	16
0x0088	TMRA_HCUPR	硬件递加事件选择寄存器	16
0x008C	TMRA_HCDOR	硬件递减事件选择寄存器	16

24.4.2 通用计数值寄存器 (TMRA_CNTER)

Offset	Bit Position															
0x0000	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	CNT															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	CNT	计数值	当前定时器的计数值  说明 锯齿波向上计数模式，CNT 初始值设置为 0；锯齿波向下计数模式，CNT 初始值设置为周期值；三角波模式计数 CNT 初始值设置为 1。	RW

24.4.3 周期基准值寄存器 (TMRA_PERAR)

Offset	Bit Position															
0x0004	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFF															
Name	PER															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	PER	计数周期值	设定每轮计数的计数周期值	RW

24.4.4 比较基准值寄存器 (TMRA_CMPARm) (m=1~4)

偏移地址 (Offset) : 0x003C+0x4*m

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFF															
Name	CMP															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	CMP	计数比较基准值	设定比较基准值	RW

24.4.5 控制状态寄存器 L (TMRA_BCSTRL)

Offset	Bit Position							
0x0080	7	6	5	4	3	2	1	0
Reset	0x02							
Name	CKDIV				SYNST	MODE	DIR	START
Access	RW				RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:4	CKDIV	计数时钟选择	● 0b0000: PCLK ● 0b0001: PCLK/2 ● 0b0010: PCLK/4 ● 0b0011: PCLK/8 ● 0b0100: PCLK/16 ● 0b0101: PCLK/32 ● 0b0110: PCLK/64 ● 0b0111: PCLK/128 ● 0b1000: PCLK/256 ● 0b1001: PCLK/512 ● 0b1010: PCLK/1024 ● 其他: 禁止设置	RW
3	SYNST	同步启动使能	● 0b0: 对称单元中偶数单元与计数单元的同步启动无效 ● 0b1: 对称单元中偶数单元与计数单元的同步启动有效  说明 对称单元中奇数单元 TimerA 或无对称单元的 TimerA 的该位设定无效，读出时为 0。	RW
2	MODE	计数模式	● 0b0: 锯齿波模式 ● 0b1: 三角波模式	RW
1	DIR	计数方向	● 0b0: 计数器向下计数 ● 0b1: 计数器向上计数  说明 三角波模式下，该位表示计数器方向状态。	RW
0	START	定时器启动	● 0b0: 定时器关闭 ● 0b1: 定时器启动  说明 ● 该位在硬件停止条件有效时，会自动变为 0。 ● 对称单元中偶数单元的同步启动功能有效时，奇数单元软件启动后，偶数单元的该位也会被置位。	RW

**注意**

1. 对于 TimerA 单元 1~单元 4，PCLK 为 PCLK0；对于 TimerA 单元 5~单元 12，PCLK 为 PCLK1。

2. 三角波模式下，定时器动作过程中请勿对该寄存器进行写操作。

24.4.6 控制状态寄存器 H (TMRA_BCSTRH)

Offset	Bit Position							
0x0081	7	6	5	4	3	2	1	0
Reset	0x00							
Name	UDFF	OVFF	ITENUDF	ITENOVF	Reserved			OVSTP
Access	RW	RW	RW	RW				RW

位/位域	标记	位名	功能描述	读写
7	UDFF	下溢标志	● 0b0：向下计数时，未发生计数下溢 ● 0b1：向下计数时，发生计数下溢	RW
6	OVFF	上溢标志	● 0b0：向上计数时，未发生计数上溢 ● 0b1：向上计数时，发生计数上溢	RW
5	ITENUDF	下溢中断使能	● 0b0：禁止 ● 0b1：使能	RW
4	ITENOVF	上溢中断使能	● 0b0：禁止 ● 0b1：使能	RW
3:1	Reserved	保留	-	-
0	OVSTP	计数溢出停止控制	● 0b0：计数器在计数上溢或下溢后，继续计数 ● 0b1：计数器在计数上溢或下溢后，停止计数  说明 溢出停止后再次启动定时器需初始化 TMRA_CNTER 值。	RW

24.4.7 中断控制寄存器 (TMRA_ICONR)

Offset	Bit Position															
0x0090	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												ITEN4	ITEN3	ITEN2	ITEN1
Access													RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3	ITEN4	计数匹配中断使能 4	中断条件：CMPAR4 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止中断输出 ● 0b1：使能中断输出	RW
2	ITEN3	计数匹配中断使能 3	中断条件：CMPAR3 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止中断输出 ● 0b1：使能中断输出	RW
1	ITEN2	计数匹配中断使能 2	中断条件：CMPAR2 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止中断输出 ● 0b1：使能中断输出	RW
0	ITEN1	计数匹配中断使能 1	中断条件：CMPAR1 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止中断输出 ● 0b1：使能中断输出	RW

24.4.8 事件控制寄存器 (TMRA_ECONR)

Offset	Bit Position															
0x0094	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												ETEN4	ETEN3	ETEN2	ETEN1
Access													RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3	ETEN4	计数匹配事件使能 4	事件条件：CMPAR4 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止事件输出 ● 0b1：使能事件输出	RW
2	ETEN3	计数匹配事件使能 3	事件条件：CMPAR3 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止事件输出 ● 0b1：使能事件输出	RW
1	ETEN2	计数匹配事件使能 2	事件条件：CMPAR2 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止事件输出 ● 0b1：使能事件输出	RW
0	ETEN1	计数匹配事件使能 1	事件条件：CMPAR1 寄存器与计数器值 (COUNTER) 相等，或发生输入捕获 ● 0b0：禁止事件输出 ● 0b1：使能事件输出	RW

24.4.9 滤波控制寄存器 (TMRA_FCONR)

Offset	Bit Position															
0x0098	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved	NOFICKCB		NOFIENCB		Reserved	NOFICKCA		NOFIENCA	Reserved				NOFICKTG		NOFIENTG
Access		RW		RW			RW		RW					RW		RW

位/位域	标记	位名	功能描述	读写
15	Reserved	保留	-	-
14:13	NOFICKCB	滤波采样基准时钟选择 CB	0b00: PCLK 0b01: PCLK/4 0b10: PCLK/16 0b11: PCLK/64	RW
12	NOFIENCB	捕获输入端口滤波 CB	0b0: TMRA_<t>_CLKB 端口输入滤波功能无效 0b1: TMRA_<t>_CLKB 端口输入滤波功能使能	RW
11	Reserved	保留	-	-
10:9	NOFICKCA	滤波采样基准时钟选择 CA	0b00: PCLK 0b01: PCLK/4 0b10: PCLK/16 0b11: PCLK/64	RW
8	NOFIENCA	捕获输入端口滤波 CA	0b0: TMRA_<t>_CLKA 端口输入滤波功能无效 0b1: TMRA_<t>_CLKA 端口输入滤波功能使能	RW
7:3	Reserved	保留	-	-
2:1	NOFICKTG	滤波采样基准时钟选择 TG	0b00: PCLK 0b01: PCLK/4 0b10: PCLK/16 0b11: PCLK/64	RW
0	NOFIENTG	捕获输入端口滤波 TG	0b0: TMRA_<t>_TRIG 输入端口滤波功能无效 0b1: TMRA_<t>_TRIG 输入端口滤波功能使能	RW



说明

对于 TimerA 单元 1~单元 4, PCLK 为 PCLK0; 对于 TimerA 单元 5~单元 12, PCLK 为 PCLK1。

24.4.10 状态标志寄存器 (TMRA_STFLR)

Offset	Bit Position															
0x009C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved				ICPF4	ICPF3	ICPF2	ICPF1	Reserved				CMPF4	CMPF3	CMPF2	CMPF1
Access					RW	RW	RW	RW					RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:12	Reserved	保留	-	-
11	ICPF4	重复捕获标志 4	写 0 清除标志, 写 1 无效 读取: 0b0: 计数匹配标志位 CMPF4 为 1 后未发生捕获完成动作 0b1: 计数匹配标志位 CMPF4 为 1 后再次发生捕获完成动作	RW
10	ICPF3	重复捕获标志 3	写 0 清除标志, 写 1 无效 读取: 0b0: 计数匹配标志位 CMPF3 为 1 后未发生捕获完成动作 0b1: 计数匹配标志位 CMPF3 为 1 后再次发生捕获完成动作	RW
9	ICPF2	重复捕获标志 2	写 0 清除标志, 写 1 无效 读取: 0b0: 计数匹配标志位 CMPF2 为 1 后未发生捕获完成动作 0b1: 计数匹配标志位 CMPF2 为 1 后再次发生捕获完成动作	RW
8	ICPF1	重复捕获标志 1	写 0 清除标志, 写 1 无效 读取: 0b0: 计数匹配标志位 CMPF1 为 1 后未发生捕获完成动作 0b1: 计数匹配标志位 CMPF1 为 1 后再次发生捕获完成动作	RW
7:4	Reserved	保留	-	-
3	CMPF4	计数匹配标志 4	写 0 清除标志, 写 1 无效 读取: 0b0: CMPAR4 寄存器的值与计数值不相等, 且未发生捕获完成动作 0b1: CMPAR4 寄存器的值与计数值相等, 或发生捕获完成动作	RW

位/位域	标记	位名	功能描述	读写
2	CMPF3	计数匹配标志 3	写 0 清除标志，写 1 无效 读取： ● 0b0：CMPAR3 寄存器的值与计数值不相等，且未发生捕获完成动作 ● 0b1：CMPAR3 寄存器的值与计数值相等，或发生捕获完成动作	RW
1	CMPF2	计数匹配标志 2	写 0 清除标志，写 1 无效 读取： ● 0b0：CMPAR2 寄存器的值与计数值不相等，且未发生捕获完成动作 ● 0b1：CMPAR2 寄存器的值与计数值相等，或发生捕获完成动作	RW
0	CMPF1	计数匹配标志 1	写 0 清除标志，写 1 无效 读取： ● 0b0：CMPAR1 寄存器的值与计数值不相等，且未发生捕获完成动作 ● 0b1：CMPAR1 寄存器的值与计数值相等，或发生捕获完成动作	RW

24.4.11 缓存控制寄存器 (TMRA_BCONRm) (m=1~2)

偏移地址 (Offset) : 0x00B8+0x8*m

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved												BSEN	BSE1	BSE0	BEN
Access													RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:4	Reserved	保留	-	-
3	BSEN	缓存特殊使能	0b0: 当对称单元的 COUNTER 寄存器进行读操作时, 本单元不进行缓存值传送 0b1: 当对称单元的 COUNTER 寄存器进行读操作时, 本单元进行缓存值传送, 即: CMPARm->CMPARn (m=2、4, n=1、3)	RW
2	BSE1	三角波缓存传送选择 1	0b0: 三角波计数模式计数到谷点时, 缓存值不传送 0b1: 三角波计数模式计数到谷点时, 缓存值传送, 即: CMMARm -> CMPARn (m=2、4, n=1、3)  说明 仅当 BEN 为 1 时, 该位的配置有效。	RW
1	BSE0	三角波缓存传送选择 0	0b0: 三角波计数模式计数到峰点时, 缓存值不传送 0b1: 三角波计数模式计数到峰点时, 缓存值传送, 即: CMMARm -> CMPARn (m=2、4, n=1、3)  说明 仅当 BEN 为 1 时, 该位的配置有效。	RW
0	BEN	缓存使能	0b0: CMPARn 基准值的缓存功能无效 (n=1、3) 0b1: CMPARn 基准值的缓存功能有效 (n=1、3)	RW



说明

- 请不要将 BEN 与 BSEN 同时设定为 1。
- BEN 为 1 且 BSE1 和 BSE0 都为 0 时, 硬件计数模式或软件计数的锯齿波模式, 计数到上溢点或下溢点或发生硬件清零计数器动作时, 缓存值传送。

24.4.12 捕获控制寄存器 (TMRA_CCONRm) (m=1~4)

偏移地址 (Offset) : 0x00FC+0x4*m

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved	NOFICKCP		NOFIENCP	HICP6	HICP5	HICP4	HICP3	Reserved	HICP2	HICP1	HICP0	Reserved			CAPMD
Access		RW		RW	RW	RW	RW	RW		RW	RW	RW				RW

位/位域	标记	位名	功能描述	读写
15	Reserved	保留	-	-
14:13	NOFICKCP	滤波采样基准时钟选择 CP	0b00: PCLK 0b01: PCLK/4 0b10: PCLK/16 0b11: PCLK/64	RW
12	NOFIENCP	捕获输入端口滤波 CP	0b0: TMRA_<t>_PWMm 端口输入滤波功能无效 0b1: TMRA_<t>_PWMm 端口输入滤波功能使能	RW
11	HICP6	捕获输入条件使能 6	0b0: TMRA_<t>_XOR 端口输入采样到下降沿时, 不发生捕获输入动作 0b1: TMRA_<t>_XOR 端口输入采样到下降沿时, 产生捕获输入动作  说明 该位只有 CCONR4 寄存器有效。即, 该位有效后且对应事件发生时, 在 CCONR4.CAPMD=1 的条件下、当前计数器值被捕获保存到 CMPAR4, 且 STFLR.CMPF4 置位。	RW
10	HICP5	捕获输入条件使能 5	0b0: TMRA_<t>_XOR 端口输入采样到上升沿时, 不发生捕获输入动作 0b1: TMRA_<t>_XOR 端口输入采样到上升沿时, 产生捕获输入动作  说明 该位只有 CCONR4 寄存器有效。即, 该位有效后且对应事件发生时, 在 CCONR4.CAPMD=1 的条件下、当前计数器值被捕获保存到 CMPAR4, 且 STFLR.CMPF4 置位。	RW

位/位域	标记	位名	功能描述	读写
9	HICP4	捕获输入条件使能 4	0b0: TMRA_<t>_TRIG 端口输入采样到下降沿时, 不发生捕获输入动作 0b1: TMRA_<t>_TRIG 端口输入采样到下降沿时, 产生捕获输入动作  说明 该位只有 CCONR3 寄存器有效。即, 该位有效后且对应事件发生时, 在 CCONR3.CAPMD=1 的条件下、当前计数器值被捕获保存到 CMPAR3, 且 STFLR.CMPF3 置位。	RW
8	HICP3	捕获输入条件使能 3	0b0: TMRA_<t>_TRIG 端口输入采样到上升沿时, 不发生捕获输入动作 0b1: TMRA_<t>_TRIG 端口输入采样到上升沿时, 产生捕获输入动作  说明 该位只有 CCONR3 寄存器有效。即, 该位有效后且对应事件发生时, 在 CCONR3.CAPMD=1 的条件下、当前计数器值被捕获保存到 CMPAR3, 且 STFLR.CMPF3 置位。	RW
7	Reserved	保留	-	-
6	HICP2	捕获输入条件使能 2	0b0: TMRA_TRGSEL 寄存器中指定的事件发生时, 不发生捕获输入动作 0b1: TMRA_TRGSEL 寄存器中指定的事件发生时, 产生捕获输入动作  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
5	HICP1	捕获输入条件使能 1	0b0: TMRA_<t>_PWMm 端口输入采样到下降沿时, 不发生捕获输入动作 0b1: TMRA_<t>_PWMm 端口输入采样到下降沿时, 产生捕获输入动作	RW
4	HICP0	捕获输入条件使能 0	0b0: TMRA_<t>_PWMm 端口输入采样到上升沿时, 不发生捕获输入动作 0b1: TMRA_<t>_PWMm 端口输入采样到上升沿时, 产生捕获输入动作	RW
3:1	Reserved	保留	-	-
0	CAPMD	功能模式选择	0b0: 比较输出功能 0b1: 捕获输入功能	RW

**说明**

- 对于 TimerA 单元 1~单元 4, PCLK 为 PCLK0; 对于 TimerA 单元 5~单元 12, PCLK 为 PCLK1。
- TMRA_<t>_XOR 信号为各个单元的正交输入 AIN 与 BIN 异或后得到的信号。

24.4.13 端口控制寄存器 (TMRA_PCONRm) (m=1~4)

偏移地址 (Offset) : 0x013C+0x4*m

Offset	Bit Position															
-	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved			OUTEN	Reserved		FORC	PERC		CMPC		STPC		STAC		
Access				RW			RW	RW		RW		RW		RW		

位/位域	标记	位名	功能描述	读写
15:13	Reserved	保留	-	-
12	OUTEN	输出使能	0b0: PWM 输出功能时的 TMRA_<t>_PWMm 端口输出无效 0b1: PWM 输出功能时的 TMRA_<t>_PWMm 端口输出有效	RW
11:10	Reserved	保留	-	-
9:8	FORC	强制端口状态设定	0b0x: 禁止设置 0b10: 下周期开始, TMRA_<t>_PWMm 端口输出设定为低电平 0b11: 下周期开始, TMRA_<t>_PWMm 端口输出设定为高电平  说明 下周期是指硬件计数模式或锯齿波计数到上溢点或下溢点、三角波计数到谷点。 该位可用于实现 PWM 输出占空比 0%或 100%的控制。	RW
7:6	PERC	周期值匹配时端口状态设定	0b00: 计数值与周期值相等时, TMRA_<t>_PWMm 端口输出设定为低电平 0b01: 计数值与周期值相等时, TMRA_<t>_PWMm 端口输出设定为高电平 0b10: 计数值与周期值相等时, TMRA_<t>_PWMm 端口输出保持先前状态 0b11: 计数值与周期值相等时, TMRA_<t>_PWMm 端口输出设定为反转电平  说明 锯齿波模式向上计数周期点为 PERAR, 锯齿波模式向下计数和三角波模式周期点为零点。	RW
5:4	CMPC	比较值匹配时端口状态设定	0b00: 计数值与 CMPARn 相等时, TMRA_<t>_PWMm 端口输出设定为低电平 0b01: 计数值与 CMPARn 相等时, TMRA_<t>_PWMm 端口输出设定为高电平 0b10: 计数值与 CMPARn 相等时, TMRA_<t>_PWMm 端口输出保持先前状态 0b11: 计数值与 CMPARn 相等时, TMRA_<t>_PWMm 端口输出设定为反转电平	RW

位/位域	标记	位名	功能描述	读写
3:2	STPC	计数停止时端口状态设定	● 0b00: 计数停止时, TMRA_<t>_PWMm 端口输出设定为低电平 ● 0b01: 计数停止时, TMRA_<t>_PWMm 端口输出设定为高电平 ● 0b10: 计数停止时, TMRA_<t>_PWMm 端口输出保持先前状态 ● 0b11: 计数停止时, TMRA_<t>_PWMm 端口输出保持先前状态	RW
1:0	STAC	计数开始时端口状态设定	● 0b00: 计数开始时, TMRA_<t>_PWMm 端口输出设定为低电平 ● 0b01: 计数开始时, TMRA_<t>_PWMm 端口输出设定为高电平 ● 0b10: 计数开始时, TMRA_<t>_PWMm 端口输出保持先前状态 ● 0b11: 计数开始时, TMRA_<t>_PWMm 端口输出保持先前状态	RW

24.4.14 硬件触发事件选择寄存器 (TMRA_HCONR)

Offset	Bit Position															
0x0084	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	HCLE6	HCLE5	HCLE4	HCLE3	Reserved	HCLE2	HCLE1	HCLE0	Reserved	HSTP2	HSTP1	HSTP0	Reserved	HSTA2	HSTA1	HSTA0
Access	RW	RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	HCLE6	硬件清零条件 6	条件: TMRA_<t>_PWM3 端口输入采样到下降沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
14	HCLE5	硬件清零条件 5	条件: TMRA_<t>_PWM3 端口输入采样到上升沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
13	HCLE4	硬件清零条件 4	条件: 对称单元的 TRIG 端口输入采样到下降沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
12	HCLE3	硬件清零条件 3	条件: 对称单元的 TRIG 端口输入采样到上升沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
11	Reserved	保留	-	-
10	HCLE2	硬件清零条件 2	条件: TMRA_TRGSEL 寄存器中指定的事件发生 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
9	HCLE1	硬件清零条件 1	条件: TMRA_<t>_TRIG 端口输入采样到下降沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
8	HCLE0	硬件清零条件 0	条件: TMRA_<t>_TRIG 端口输入采样到上升沿 ● 0b0: 条件匹配时, 硬件清零无效 ● 0b1: 条件匹配时, 硬件清零有效	RW
7	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
6	HSTP2	硬件停止条件 2	条件: TMRA_TRGSEL 寄存器中指定的事件发生 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
5	HSTP1	硬件停止条件 1	条件: TMRA_<t>_TRIG 端口输入采样到下降沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
4	HSTP0	硬件停止条件 0	条件: TMRA_<t>_TRIG 端口输入采样到上升沿 ● 0b0: 条件匹配时, 硬件停止无效 ● 0b1: 条件匹配时, 硬件停止有效	RW
3	Reserved	保留	-	-
2	HSTA2	硬件启动条件 2	条件: TMRA_TRGSEL 寄存器中指定的事件发生 ● 0b0: 条件匹配时, 硬件启动无效 ● 0b1: 条件匹配时, 硬件启动有效  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
1	HSTA1	硬件启动条件 1	条件: 1. 同步启动功能无效时, TMRA_<t>_TRIG 端口输入采样到下降沿 2. 同步启动功能有效时, 对称单元的 TMRA_<t>_TRIG 端口输入采样到下降沿 (仅对称单元中的偶数单元支持该功能) ● 0b0: 条件匹配时, 硬件启动无效 ● 0b1: 条件匹配时, 硬件启动有效	RW
0	HSTA0	硬件启动条件 0	条件: 1. 同步启动功能无效时, TMRA_<t>_TRIG 端口输入采样到上升沿 2. 同步启动功能有效时, 对称单元的 TMRA_<t>_TRIG 端口输入采样到上升沿 (仅对称单元中的偶数单元支持该功能) ● 0b0: 条件匹配时, 硬件启动无效 ● 0b1: 条件匹配时, 硬件启动有效	RW

24.4.15 硬件递加事件选择寄存器 (TMRA_HCUPR)

Offset	Bit Position															
0x0088	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved			HCUP12	HCUP11	HCUP10	HCUP9	HCUP8	HCUP7	HCUP6	HCUP5	HCUP4	HCUP3	HCUP2	HCUP1	HCUP0
Access				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:13	Reserved	保留	-	-
12	HCUP12	硬件递加条件 12	条件：对称单元发生计数下溢 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
11	HCUP11	硬件递加条件 11	条件：对称单元发生计数上溢 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
10	HCUP10	硬件递加条件 10	条件：TMRA_TRGSEL 寄存器中指定的事件发生 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
9	HCUP9	硬件递加条件 9	条件：TMRA_<t>_TRIG 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
8	HCUP8	硬件递加条件 8	条件：TMRA_<t>_TRIG 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
7	HCUP7	硬件递加条件 7	条件：TMRA_<t>_CLKB 端口为高电平时，TMRA_<t>_CLKA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
6	HCUP6	硬件递加条件 6	条件：TMRA_<t>_CLKB 端口为高电平时，TMRA_<t>_CLKA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW
5	HCUP5	硬件递加条件 5	条件：TMRA_<t>_CLKB 端口为低电平时，TMRA_<t>_CLKA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递加无效 ● 0b1：条件匹配时，硬件递加有效	RW

位/位域	标记	位名	功能描述	读写
4	HCUP4	硬件递加条件 4	条件: TMRA_<t>_CLKB 端口为低电平时, TMRA_<t>_CLKA 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
3	HCUP3	硬件递加条件 3	条件: TMRA_<t>_CLKA 端口为高电平时, TMRA_<t>_CLKB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
2	HCUP2	硬件递加条件 2	条件: TMRA_<t>_CLKA 端口为高电平时, TMRA_<t>_CLKB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
1	HCUP1	硬件递加条件 1	条件: TMRA_<t>_CLKA 端口为低电平时, TMRA_<t>_CLKB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW
0	HCUP0	硬件递加条件 0	条件: TMRA_<t>_CLKA 端口为低电平时, TMRA_<t>_CLKB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递加无效 ● 0b1: 条件匹配时, 硬件递加有效	RW

24.4.16 硬件递减事件选择寄存器 (TMRA_HCDOR)

Offset	Bit Position															
0x008C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved			HCDO12	HCDO11	HCDO10	HCDO9	HCDO8	HCDO7	HCDO6	HCDO5	HCDO4	HCDO3	HCDO2	HCDO1	HCDO0
Access				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:13	Reserved	保留	-	-
12	HCDO12	硬件递减条件 12	条件：对称单元发生计数下溢 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
11	HCDO11	硬件递减条件 11	条件：对称单元发生计数上溢 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
10	HCDO10	硬件递减条件 10	条件：TMRA_TRGSEL 寄存器中指定的事件发生 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效  说明 具体信息请参见《自动运行系统 (AOS) 》的“TimerA 硬件触发事件选择寄存器 (TMRA_TRGSELx) (x=0~3) ”。	RW
9	HCDO9	硬件递减条件 9	条件：TMRA_<t>_TRIG 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
8	HCDO8	硬件递减条件 8	条件：TMRA_<t>_TRIG 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
7	HCDO7	硬件递减条件 7	条件：TMRA_<t>_CLKB 端口为高电平时，TMRA_<t>_CLKA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
6	HCDO6	硬件递减条件 6	条件：TMRA_<t>_CLKB 端口为高电平时，TMRA_<t>_CLKA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW
5	HCDO5	硬件递减条件 5	条件：TMRA_<t>_CLKB 端口为低电平时，TMRA_<t>_CLKA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件递减无效 ● 0b1：条件匹配时，硬件递减有效	RW

位/位域	标记	位名	功能描述	读写
4	HCDO4	硬件递减条件 4	条件: TMRA_<t>_CLKB 端口为低电平时, TMRA_<t>_CLKA 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递减无效 ● 0b1: 条件匹配时, 硬件递减有效	RW
3	HCDO3	硬件递减条件 3	条件: TMRA_<t>_CLKA 端口为高电平时, TMRA_<t>_CLKB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递减无效 ● 0b1: 条件匹配时, 硬件递减有效	RW
2	HCDO2	硬件递减条件 2	条件: TMRA_<t>_CLKA 端口为高电平时, TMRA_<t>_CLKB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递减无效 ● 0b1: 条件匹配时, 硬件递减有效	RW
1	HCDO1	硬件递减条件 1	条件: TMRA_<t>_CLKA 端口为低电平时, TMRA_<t>_CLKB 端口上采样到下降沿 ● 0b0: 条件匹配时, 硬件递减无效 ● 0b1: 条件匹配时, 硬件递减有效	RW
0	HCDO0	硬件递减条件 0	条件: TMRA_<t>_CLKA 端口为低电平时, TMRA_<t>_CLKB 端口上采样到上升沿 ● 0b0: 条件匹配时, 硬件递减无效 ● 0b1: 条件匹配时, 硬件递减有效	RW

25 通用定时器（Timer2）

25.1 简介

通用定时器 2（Timer2）是一个可以实现同步计数、异步计数方式的基本定时器。该定时器内含 2 个通道（CH-A 和 CH-B）。每个通道均有一个输出端口，可实现基本的方波输出；每个通道均有 2 个输入端口，一个是时钟输入端口，可实现端口异步计数；一个是触发输入端口，可实现定时器启动、停止、清零、计数动作及计数值捕获输入；支持脉宽测量和周期测量。本系列产品中搭载 4 个单元的 Timer2。

25.2 主要特性

Timer2 基本的功能及特性如下：

表 25-1 Timer2 的基本功能及特性

基本动作	硬件触发计数器启动、停止、清零、计数及捕获输入
	同步计数、异步计数
	脉宽测量
	周期测量
	方波输出
	事件信号输出
中断类型	计数比较匹配中断
	计数溢出中断

25.3 功能说明

25.3.1 基本框图

框图中所示“<t>”表示单元编号，即“<t>”为 1~4，本章节后文提到“<t>”时均指单元编号，不再赘述。框图中 TMR2_<t>_CLKA/B 端口输入上的 ANF 是一个模拟滤波单元，滤波宽度为 40ns（典型条件）。

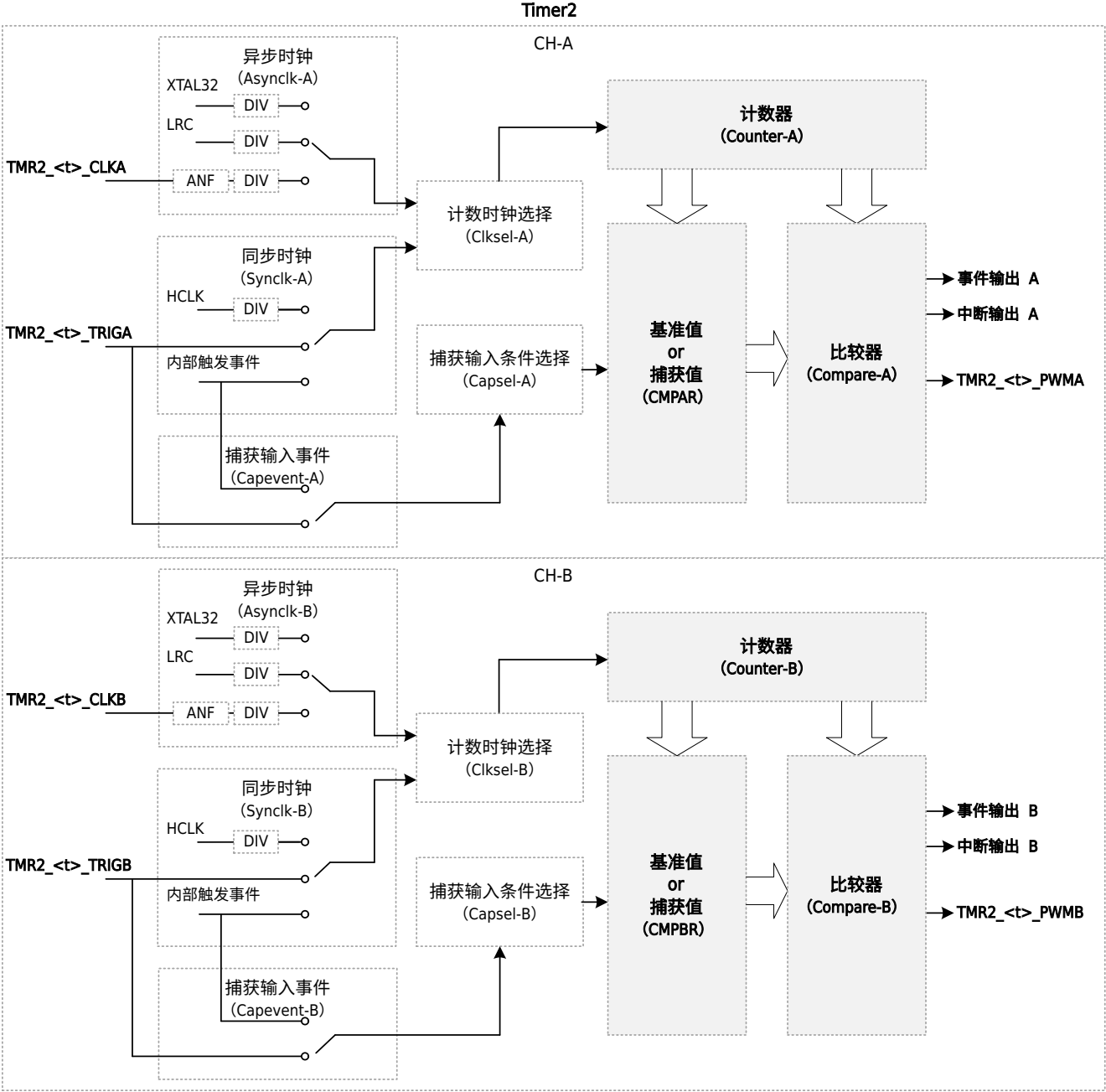


图 25-1 Timer2 基本框图

下表为 Timer2 的输入输出端口列表。

表 25-2 Timer2 端口列表

端口名	方向	功能
TMR2_<t>_CLKA/B	输入	异步时钟输入端口
TMR2_<t>_TRIGA/B	输入	1. 硬件启动、停止、清零、计数输入端口 2. 捕获输入端口
TMR2_<t>_PWMA/B	输出	方波输出端口

25.3.2 时钟源选择

Timer2 的计数方式可以选择同步计数方式或异步计数方式。

同步计数方式是指定时器的计数时钟和总线访问时钟（寄存器读写操作时钟）有同步时序关系；异步计数方式是指定时器的计数时钟和总线访问时钟（寄存器读写操作时钟）是非同步的时序关系。异步计数方式时对寄存器读操作时，定时器等的状态可能正在发生变化、读出不可预期的状态。因此，在异步计数方式时，寄存器读操作须在计数停止状态下实现。

同步计数时钟源

同步计数方式时（BCONR.SYNSA=0），时钟源可以有以下几种选择（BCONR.SYNCLKA[1:0]或BCONR.SYNCLKAT[1:0]设定）：

1. PCLK1 及 PCLK1 的 2、4、8、16、32、64、128、256、512、1024 分频作为同步计数时钟（BCONR.SYNCLKA[1:0]=0b00 & BCONR.CKDIVA[3:0]设定）；
2. TMR2_<t>_TRIGA、TMR2_<t>_TRIGB 端口的输入有效选择沿被内部 PCLK1 采样后作为同步计数时钟（BCONR.SYNCLKA[1:0]=0b01 或 0b10）；
3. 内部硬件触发事件输入作为同步计数时钟（BCONR.SYNCLKA[1:0]=0b11）；
4. Timer6 的计数上溢事件（BCONR.SYNCLKAT[0]=1）；
5. Timer6 的计数下溢事件（BCONR.SYNCLKAT[1]=1）。



说明

时钟源选择 2、3、4、5 时需确保 PCLK1 分频为 1。

异步计数时钟源

异步计数方式时（BCONR.SYNSA=1），时钟源可以有以下几种选择（BCONR.ASYNCLKA[1:0]设定）：

1. LRC 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为异步计数时钟（BCONR.ASYNCLKA=0 & BCONR.CKDIVA[3:0]设定）；
2. XTAL32 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为异步计数时钟（BCONR.ASYNCLKA=1 & BCONR.CKDIVA[3:0]设定）；
3. TMR2_<t>_CLKA、TMR2_<t>_CLKB 时钟输入端口及其 2、4、8、16、32、64、128、256、512、1024 分频作为异步计数时钟（BCONR.ASYNCLKA=0b10 & BCONR.CKDIVA[3:0]设定）。

25.3.3 比较输出

Timer2 有 2 个输出端口（TMR2_<t>_PWMA、TMR2_<t>_PWMB），可以实现基本的方波输出。

CMPAR、CMPBR 寄存器对应了 TMR2_<t>_PWMA、TMR2_<t>_PWMB 的计数比较基准值。当定时器的计数值 CNTAR 和 CMPAR 相等时，TMR2_<t>_PWMA 端口输出指定的电平；当定时器的计数值 CNTBR 和 CMPBR 相等时，TMR2_<t>_PWMB 端口输出指定的电平。

TMR2_<t>_PWMA/B 端口的计数起始电平、停止电平、计数比较匹配时的电平等，可由端口控制寄存器（PCONR）的 PCONR.STACA[1:0]、PCONR.STPCA[1:0]、PCONR.CMPCA[1:0]位设定。下图为比较输出动作例。

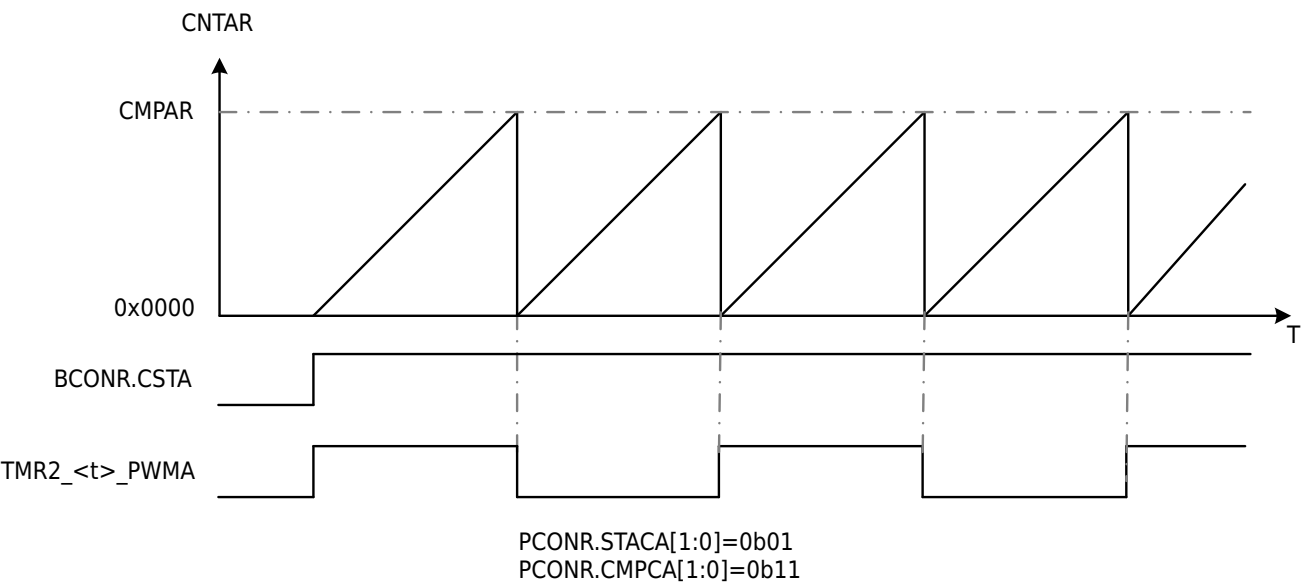


图 25-2 比较输出动作

25.3.4 硬件触发

25.3.4.1 功能概览

Timer2 的硬件触发动作有两类，一个是 TMR2_<t>_TRIGA/B 端口的输入事件，另一个是内部硬件触发事件。这两类事件均可以实现计数器的启动、停止、清零及硬件捕获输入功能。各种功能组合设定也可实现脉宽测量、周期测量等。

TMR2_<t>_TRIGA/B 端口输入类型选择（上升沿或下降沿）及功能选择（启动、停止、清零、捕获输入）、内部硬件触发事件的功能选择（启动、停止、清零、捕获输入）可通过硬件控制寄存器（HCONR）的相应位设定。

Timer2 有一个内部硬件触发源，事件源可通过触发选择寄存器（TMR2_TRGSEL）中对应的编号设定来选择，具体的事件对应关系请参见《中断控制器（INTC）》章节相关描述。使用内部硬件触发功能时，需要先将功能时钟控制寄存器（PWC_FCG0）的外围电路触发功能使能位置 1。

25.3.4.2 启动停止清零

每个通道的硬件启动、停止、清零条件由硬件控制寄存器（HCONR）的相关设定来决定。下图是硬件启动、清零的动作例。

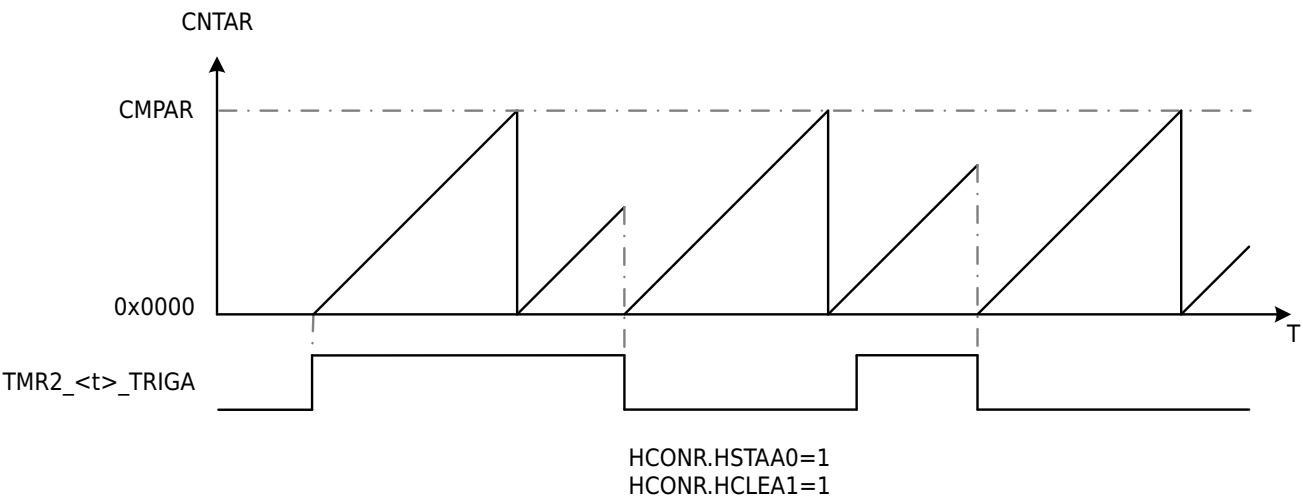


图 25-3 硬件启动、清零动作

25.3.4.3 捕获输入动作

TMR2_<t>_TRIGA/B 端口输入或内部触发事件作为捕获输入条件时，可实现捕获输入功能。当硬件控制寄存器 (HCONR) 选择的捕获输入条件有效时，当前的计数值就被保存到相应的寄存器 (CMPAR、CMPBR) 中。下图为捕获输入的动作例。

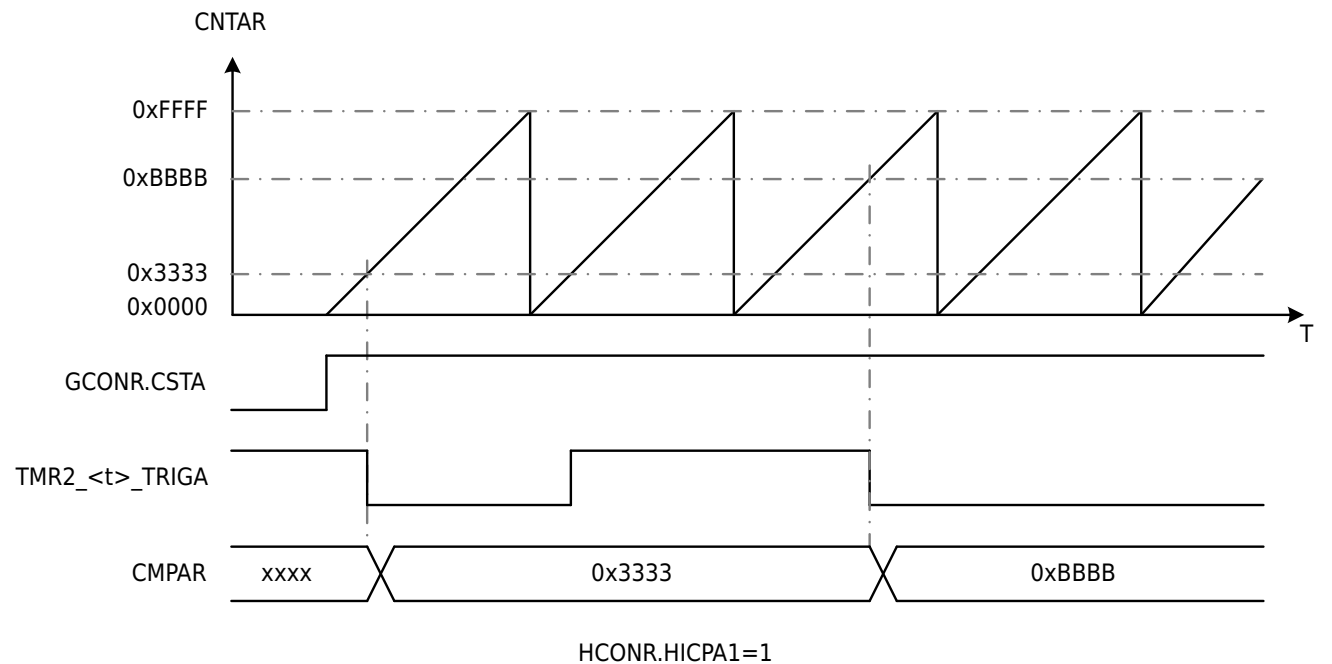


图 25-4 捕获输入动作

25.3.4.4 脉宽测量

将 A 通道的硬件启动条件设为 TMR2_<t>_TRIGA 的上升沿，硬件清零条件、停止条件和捕获输入条件均设为 TMR2_<t>_TRIGA 的下降沿，就可以实现连续的脉冲宽度测量。B 通道可实现同样功能。对应动作如下图所示。

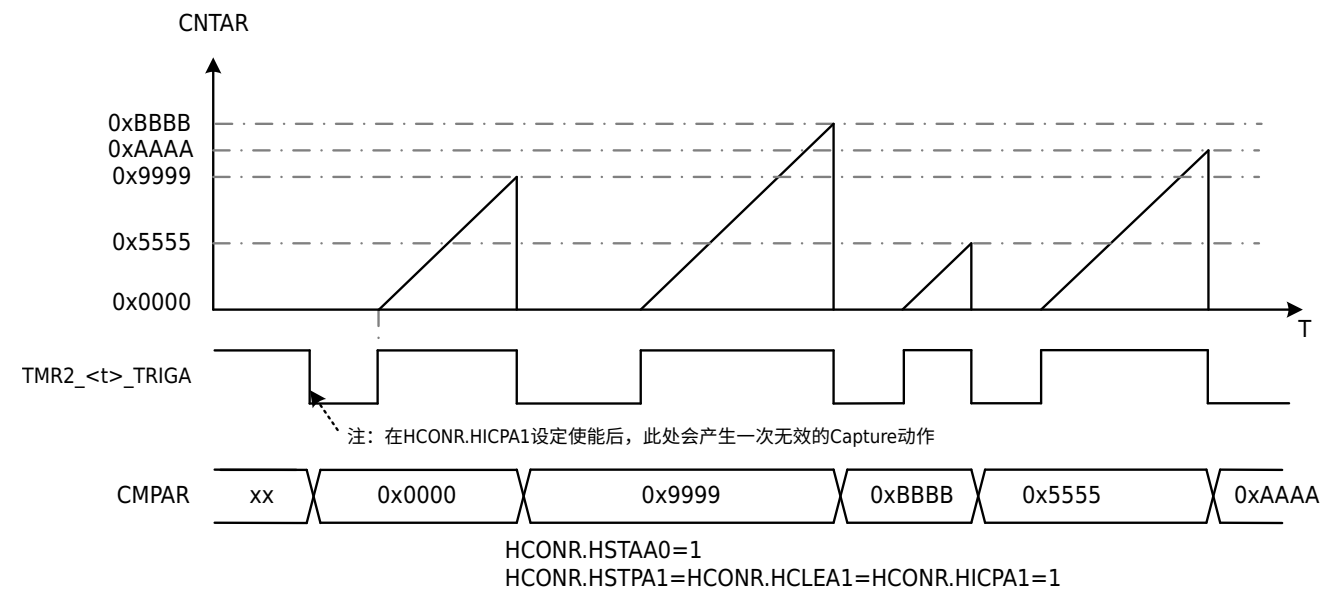


图 25-5 脉宽测量

25.3.4.5 周期测量

将 A 通道的硬件启动条件、硬件清零条件和捕获输入条件设为 TMR2_<t>_TRIGA 的相同沿（上升沿或下降沿），就可以实现连续的周期宽度测量。B 通道可实现同样功能。对应动作如下图所示。

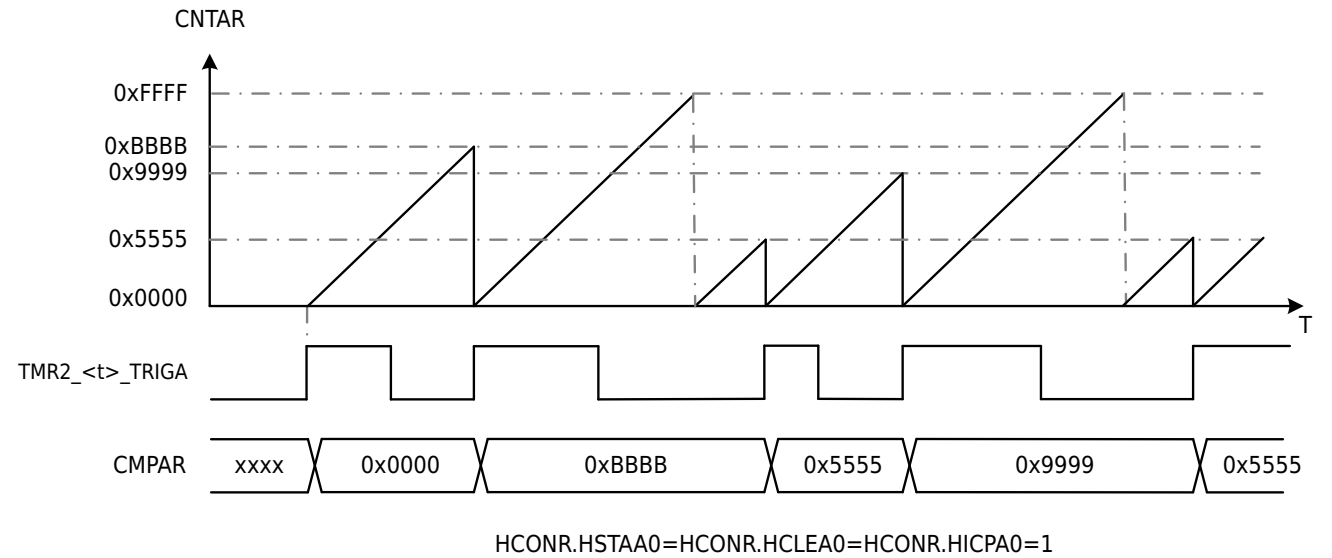


图 25-6 周期测量

25.3.5 数字滤波

Timer2 的 TRIG 输入端口（TMR2_<t>_TRIGA/B）具有数字滤波功能。可通过设定端口控制寄存器（PCONR.NOFIENA）的使能位开启对应端口的滤波功能。滤波用的基准时钟也通过端口控制寄存器（PCONR.NOFICKA[1:0]）设定。

在滤波采样基准时钟采样到端口上 3 次一致的电平时，该电平被当作有效电平传送到模块内部；小于 3 次一致的电平会被当作外部干扰滤掉，不传送到模块内部。其动作例如下图所示。

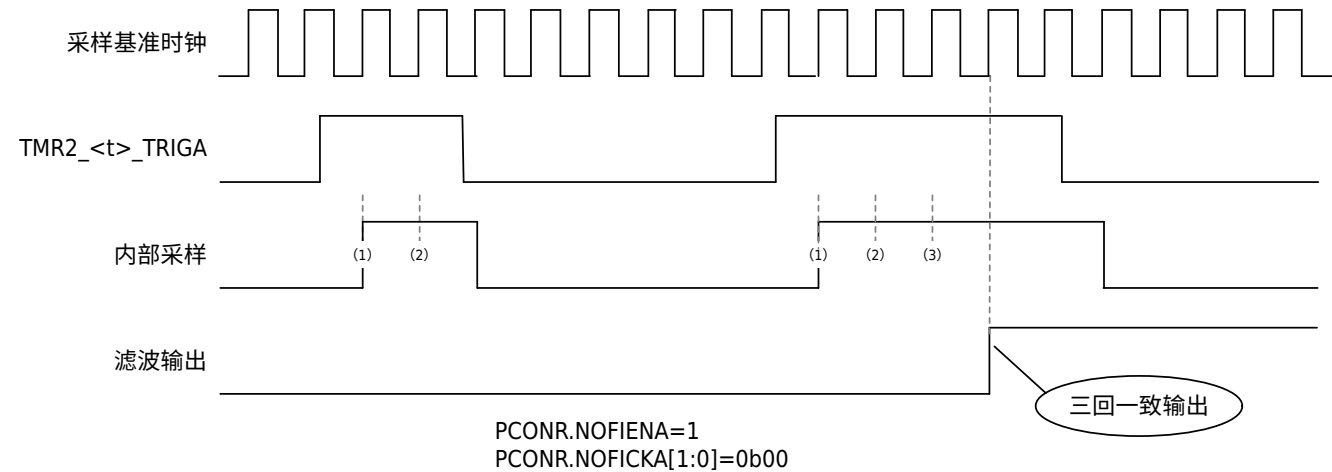


图 25-7 TRIG 输入端口的数字滤波

25.3.6 软件同步启动

产品配置了一个计数同步使能寄存器（TMR_SYNENR、TMR_SYNENR2），用于同步启动产品所搭载的 Timer（包括 Timer0，Timer2，Timer4，Timer6，TimerA）。使用方法和寄存器说明请参见《通用定时（Timer0）》章节相关描述。请不要在本定时器异步计数的情况下使用该功能。

25.3.7 中断和事件

25.3.7.1 中断输出

一个 Timer2 含有 4 个中断，分别是通道 A 和通道 B 的计数比较匹配中断或捕获输入中断、通道 A 和通道 B 的计数溢出中断。

基准值寄存器 (CMPAR、CMPBR) 共计 2 个，可分别与计数值寄存器 (CNTAR、CNTBR) 比较产生比较匹配有效信号。计数比较匹配时，状态标志寄存器 (STFLR) 中的 STFLR.CMFA位分别会被置为 1。此时若设定中断控制寄存器 (ICONR) 的 ICONR.CMENA位使能中断，则对应的中断请求 (TMR2_<t>_CMPm, m=A、B) 也会被触发。

在硬件控制寄存器 (HCONR) 选择的捕获输入有效条件产生时，可以产生相应的捕获输入动作。此时若设定中断控制寄存器 (ICONR) 的 ICONR.CMENA位使能中断，则对应的中断请求 (TMR2_<t>_CMPm, m=A、B) 被触发。

计数值寄存器 (CNTAR、CNTBR) 计数到 0xFFFF 时发生计数溢出事件，此时状态标志寄存器 (STFLR) 中的 STFLR.OVFA位会被置为 1。若设定中断控制寄存器 (ICONR) 的 ICONR.OVENA位使能中断，则对应的中断请求 (TMR2_<t>_OVFm, m=A、B) 也会被触发。

在选择异步计数模式 (BCONR.ASYNCLKA[1:0]=0b10 除外) 时，单元 1 的基准值寄存器 (CMPAR) 产生的比较匹配中断可用于在低功耗模式时对系统进行唤醒，请参见《中断控制器 (INTC)》章节相关描述。

25.3.7.2 事件输出

一个 Timer2 含有 4 个事件，分别是通道 A 和通道 B 的计数比较匹配事件或捕获输入事件、通道 A 和通道 B 的计数溢出事件。

在计数过程中发生计数比较匹配或捕获输入动作或计数溢出时，会产生相应的事件请求 (TMR2_<t>_CMPm 或 TMR2_<t>_OVFm, m=A、B) 输出信号，可以用于选择触发其它模块。

25.4 寄存器

25.4.1 寄存器总表

- Timer2_1 基地址：0x40024800
- Timer2_2 基地址：0x40024C00
- Timer2_3 基地址：0x40025000
- Timer2_4 基地址：0x40025400



说明

Timer2 为定时器名称，Timer2_<t> (<t>:1~4) 为该定时器单元。

表 25-3 通用定时器 (Timer2) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	TMR2_CNTAR	计数值寄存器	32
0x0004	TMR2_CNTBR	计数值寄存器	32
0x0008	TMR2_CMPAR	基准值寄存器	32
0x000C	TMR2_CMPBR	基准值寄存器	32
0x0010	TMR2_BCONR	基本控制寄存器	32
0x0014	TMR2_ICONR	中断控制寄存器	32
0x0018	TMR2_PCONR	端口控制寄存器	32
0x001C	TMR2_HCONR	硬件控制寄存器	32
0x0020	TMR2_STFLR	状态标志寄存器	32

25.4.2 计数值寄存器 m (TMR2_CNTmR) (m=A/B)

偏移地址 (Offset) : 0x0000 (CNTAR) , 0x0004 (CNTBR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNT	计数值	当前定时器的计数值  说明 在计数中时，不能向该寄存器中写入值。 计数停止后再次启动定时器需初始化该寄存器。	RW

25.4.3 基准值寄存器 m (TMR2_CMPmR) (m=A/B)

偏移地址 (Offset) : 0x0008 (CMPAR) , 0x000C (CMPBR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																CMP															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CMP	基准值	设定计数基准值，产生比较匹配事件	RW

25.4.4 基本控制寄存器 (TMR2_BCONR)

Offset	Bit Position																																				
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved	SYNCLKBT		ASYNCLKB		SYNCLKB		CKDIVB				SYNSB	Reserved	CAPMDB		CSTB		Reserved		SYNCLKAT		ASYNCLKA		SYNCLKA		CKDIVA				SYNSA		Reserved	CAPMDA		CSTA		
Access		RW		RW		RW		RW				RW		RW		RW				RW		RW		RW		RW		RW				RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	SYNCLKBT[1]	同步计数同时钟源选择 BT	条件：Timer6 的单元 n 发生计数下溢 0b0：条件匹配时，Timer2 单元 m 同步计数无效 0b1：条件匹配时，Timer2 单元 m 进行一次同步计数 (当 m=1、2、3、4 时，n=2、4、6、8)	RW
28	SYNCLKBT[0]	同步计数同时钟源选择 BT	条件：Timer6 的单元 n 发生计数上溢 0b0：条件匹配时，Timer2 单元 m 同步计数无效 0b1：条件匹配时，Timer2 单元 m 进行一次同步计数 (当 m=1、2、3、4 时，n=2、4、6、8)	RW
27:26	ASYNCLKB	异步计数同时钟源选择 B	0b00：LRC 0b01：XTAL32 0b10：TMR2_<t>_CLKB 时钟输入 0b11：禁止设置	RW
25:24	SYNCLKB	同步计数同时钟源选择 B	0b00：PCLK1 0b01：TMR2_<t>_TRIGB 上升沿 (内部 PCLK1 同步) 0b10：TMR2_<t>_TRIGB 下降沿 (内部 PCLK1 同步) 0b11：内部触发事件输入	RW
23:20	CKDIVB	计数时钟分频选择 B	0b0000：时钟源 0b0001：时钟源/2 0b0010：时钟源/4 0b0011：时钟源/8 0b0100：时钟源/16 0b0101：时钟源/32 0b0110：时钟源/64 0b0111：时钟源/128 0b1000：时钟源/256 0b1001：时钟源/512 0b1010：时钟源/1024 - 其他：禁止设置	RW

位/位域	标记	位名	功能描述	读写
23:20	CKDIVB	计数时钟分频选择 B	 说明 被分频的时钟源可以是异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
19	SYNSB	计数方式选择 B	0b0: 同步计数方式 0b1: 异步计数方式	RW
18	Reserved	保留	-	-
17	CAPMDB	功能模式选择 B	0b0: 比较输出功能 0b1: 捕获输入功能	RW
16	CSTB	定时器启动 B	0b0: 通道 B 定时器关闭 0b1: 通道 B 定时器启动  说明 该位在硬件触发停止条件有效时, 会自动变为 0。	RW
15:14	Reserved	保留	-	-
13	SYNCLKAT[1]	同步计数同时钟源选择 AT	条件: Timer6 的单元 n 发生计数下溢 0b0: 条件匹配时, Timer2 单元 m 同步计数无效 0b1: 条件匹配时, Timer2 单元 m 进行一次同步计数 (当 m=1、2、3、4 时, n=1、3、5、7)	RW
12	SYNCLKAT[0]	同步计数同时钟源选择 AT	条件: Timer6 的单元 n 发生计数上溢 0b0: 条件匹配时, Timer2 单元 m 同步计数无效 0b1: 条件匹配时, Timer2 单元 m 进行一次同步计数 (当 m=1、2、3、4 时, n=1、3、5、7)	
11:10	ASYNCLKA	异步计数同时钟源选择 A	0b00: LRC 0b01: XTAL32 0b10: TMR2_<t>_CLKB 时钟输入 0b11: 禁止设置	RW
9:8	SYNCLKA	同步计数同时钟源选择 A	0b00: PCLK1 0b01: TMR2_<t>_TRIGA 上升沿 (内部 PCLK1 同步) 0b10: TMR2_<t>_TRIGA 下降沿 (内部 PCLK1 同步) 0b11: 内部触发事件输入	RW
7:4	CKDIVA	计数时钟分频选择 A	0b0000: 时钟源 0b0001: 时钟源/2 0b0010: 时钟源/4 0b0011: 时钟源/8 0b0100: 时钟源/16 0b0101: 时钟源/32 0b0110: 时钟源/64 0b0111: 时钟源/128 0b1000: 时钟源/256 0b1001: 时钟源/512 0b1010: 时钟源/1024	RW

位/位域	标记	位名	功能描述	读写
7:4	CKDIVA	计数时钟分频选择 A	● 其他：禁止设置  说明 被分频的时钟源可以是异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
3	SYNSA	计数方式选择 A	● 0b0：同步计数方式 ● 0b1：异步计数方式	RW
2	<i>Reserved</i>	<i>保留</i>	-	-
1	CAPMDA	功能模式选择 A	● 0b0：比较输出功能 ● 0b1：捕获输入功能	RW
0	CSTA	定时器启动 A	● 0b0：通道 A 定时器关闭 ● 0b1：通道 A 定时器启动  说明 该位在硬件触发停止条件有效时，会自动变为 0。	RW

25.4.5 中断控制寄存器 (TMR2_ICONR)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														OVENB	CMENB	Reserved														OVENA	CMENA
Access															RW	RW															RW	RW

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	OVENB	计数溢出中断使能 B	中断产生条件：计数值 (CNTBR) =0xFFFF 时 ● 0b0：禁止中断 ● 0b1：使能中断	RW
16	CMENB	计数匹配中断使能 B	中断产生条件：CMPBR 寄存器与计数值 (CNTBR) 相等时，或者发生捕获输入事件时 ● 0b0：禁止中断 ● 0b1：使能中断	RW
15:2	Reserved	保留	-	-
1	OVENA	计数溢出中断使能 A	中断产生条件：计数值 (CNTAR) =0xFFFF 时 ● 0b0：禁止中断 ● 0b1：使能中断	RW
0	CMENA	计数匹配中断使能 A	中断产生条件：CMPAR 寄存器与计数值 (CNTAR) 相等时，或者发生捕获输入事件时 ● 0b0：禁止中断 ● 0b1：使能中断	RW

25.4.6 端口控制寄存器 (TMR2_PCONR)

Offset	Bit Position																															
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	NOFICKB		NOFIENB		Reserved		OUTENB	Reserved		CMPCB		STPCB		STACB		Reserved	NOFICKA		NOFIENA		Reserved		OUTENA	Reserved		CMPCA		STPCA		STACA	
Access		RW		RW				RW			RW		RW		RW			RW		RW				RW			RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	NOFICKB	滤波采样基准时钟选择 B	0b00: 时钟源 0b01: 时钟源/4 0b10: 时钟源/16 0b11: 时钟源/64  说明 该时钟源为异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
28	NOFIENB	触发端口滤波 B	0b0: TMR2_<t>_TRIGB 输入端口滤波功能无效 0b1: TMR2_<t>_TRIGB 输入端口滤波功能使能	RW
27:25	Reserved	保留	-	-
24	OUTENB	端口输出使能 B	0b0: TMR2_<t>_PWMB 端口输出无效 0b1: TMR2_<t>_PWMB 端口输出有效	RW
23:22	Reserved	保留	-	-
21:20	CMPCB	比较值匹配时端口状态设定 B	0b00: 计数值 (CNTBR) 与 CMPBR 相等时, TMR2_<t>_PWMB 端口输出设定为低电平 0b01: 计数值 (CNTBR) 与 CMPBR 相等时, TMR2_<t>_PWMB 端口输出设定为高电平 0b10: 计数值 (CNTBR) 与 CMPBR 相等时, TMR2_<t>_PWMB 端口输出保持先前状态 0b11: 计数值 (CNTBR) 与 CMPBR 相等时, TMR2_<t>_PWMB 端口输出设定为反转电平	RW
19:18	STPCB	计数停止时端口状态设定 B	0b00: 计数停止时, TMR2_<t>_PWMB 端口输出设定为低电平 0b01: 计数停止时, TMR2_<t>_PWMB 端口输出设定为高电平 0b10: 计数停止时, TMR2_<t>_PWMB 端口输出保持先前状态 0b11: 计数停止时, TMR2_<t>_PWMB 端口输出保持先前状态	RW

位/位域	标记	位名	功能描述	读写
17:16	STACB	计数启动时端口状态设定 B	0b00: 计数启动时, TMR2_<t>_PWMB 端口输出设定为低电平 0b01: 计数启动时, TMR2_<t>_PWMB 端口输出设定为高电平 0b10: 计数启动时, TMR2_<t>_PWMB 端口输出保持先前状态 0b11: 计数启动时, TMR2_<t>_PWMB 端口输出保持先前状态	RW
15	Reserved	保留	-	-
14:13	NOFICKA	滤波采样基准时钟选择 A	0b00: 时钟源 0b01: 时钟源/4 0b10: 时钟源/16 0b11: 时钟源/64  说明 该时钟源为异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
12	NOFIENA	触发端口滤波 A	0b0: TMR2_<t>_TRIGA 输入端口滤波功能无效 0b1: TMR2_<t>_TRIGA 输入端口滤波功能使能	RW
11:9	Reserved	保留	-	-
8	OUTENA	端口输出使能 A	0b0: TMR2_<t>_PWMA 端口输出无效 0b1: TMR2_<t>_PWMA 端口输出有效	RW
7:6	Reserved	保留	-	-
5:4	CMPCA	比较值匹配时端口状态设定 A	0b00: 计数值 (CNTAR) 与 CMPAR 相等时, TMR2_<t>_PWMA 端口输出设定为低电平 0b01: 计数值 (CNTAR) 与 CMPAR 相等时, TMR2_<t>_PWMA 端口输出设定为高电平 0b10: 计数值 (CNTAR) 与 CMPAR 相等时, TMR2_<t>_PWMA 端口输出保持先前状态 0b11: 计数值 (CNTAR) 与 CMPAR 相等时, TMR2_<t>_PWMA 端口输出设定为反转电平	RW
3:2	STPCA	计数停止时端口状态设定 A	0b00: 计数停止时, TMR2_<t>_PWMA 端口输出设定为低电平 0b01: 计数停止时, TMR2_<t>_PWMA 端口输出设定为高电平 0b10: 计数停止时, TMR2_<t>_PWMA 端口输出保持先前状态 0b11: 计数停止时, TMR2_<t>_PWMA 端口输出保持先前状态	RW
1:0	STACA	计数启动时端口状态设定 A	0b00: 计数启动时, TMR2_<t>_PWMA 端口输出设定为低电平 0b01: 计数启动时, TMR2_<t>_PWMA 端口输出设定为高电平 0b10: 计数启动时, TMR2_<t>_PWMA 端口输出保持先前状态 0b11: 计数启动时, TMR2_<t>_PWMA 端口输出保持先前状态	RW

25.4.7 硬件控制寄存器 (TMR2_HCONR)

Offset	Bit Position																															
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	HICPB2	HICPB1	HICPB0	Reserved	HCLEB2	HCLEB1	HCLEB0	Reserved	HSTPB2	HSTPB1	HSTPB0	Reserved	HSTAB2	HSTAB1	HSTAB0	Reserved	HICPA2	HICPA1	HICPA0	Reserved	HCLEA2	HCLEA1	HCLEA0	Reserved	HSTPA2	HSTPA1	HSTPA0	Reserved	HSTAA2	HSTAA1	HSTAA0
Access		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30	HICPB2	硬件捕获输入条件 B2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
29	HICPB1	硬件捕获输入条件 B1	条件：TMR2_<t>_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
28	HICPB0	硬件捕获输入条件 B0	条件：TMR2_<t>_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
27	Reserved	保留	-	-
26	HCLEB2	硬件清零条件 B2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
25	HCLEB1	硬件清零条件 B1	条件：TMR2_<t>_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
24	HCLEB0	硬件清零条件 B0	条件：TMR2_<t>_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
23	Reserved	保留	-	-
22	HSTPB2	硬件停止条件 B2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
21	HSTPB1	硬件停止条件 B1	条件：TMR2_<t>_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
20	HSTPB0	硬件停止条件 B0	条件：TMR2_<t>_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
19	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
18	HSTAB2	硬件启动条件 B2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
17	HSTAB1	硬件启动条件 B1	条件：TMR2_<t>_TRIGB 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
16	HSTAB0	硬件启动条件 B0	条件：TMR2_<t>_TRIGB 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
15	Reserved	保留	-	-
14	HICPA2	硬件捕获输入条件 A2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
13	HICPA1	硬件捕获输入条件 A1	条件：TMR2_<t>_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
12	HICPA0	硬件捕获输入条件 A0	条件：TMR2_<t>_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件捕获输入无效 ● 0b1：条件匹配时，硬件捕获输入有效	RW
11	Reserved	保留	-	-
10	HCLEA2	硬件清零条件 A2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
9	HCLEA1	硬件清零条件 A1	条件：TMR2_<t>_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
8	HCLEA0	硬件清零条件 A0	条件：TMR2_<t>_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件清零无效 ● 0b1：条件匹配时，硬件清零有效	RW
7	Reserved	保留	-	-
6	HSTPA2	硬件停止条件 A2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
5	HSTPA1	硬件停止条件 A1	条件：TMR2_<t>_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
4	HSTPA0	硬件停止条件 A0	条件：TMR2_<t>_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件停止无效 ● 0b1：条件匹配时，硬件停止有效	RW
3	Reserved	保留	-	-
2	HSTAA2	硬件启动条件 A2	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW

位/位域	标记	位名	功能描述	读写
1	HSTAA1	硬件启动条件 A1	条件：TMR2_<t>_TRIGA 端口上采样到下降沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW
0	HSTAA0	硬件启动条件 A0	条件：TMR2_<t>_TRIGA 端口上采样到上升沿 ● 0b0：条件匹配时，硬件启动无效 ● 0b1：条件匹配时，硬件启动有效	RW

25.4.8 状态标志寄存器 (TMR2_STFLR)

Offset	Bit Position																																																							
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																								
Reset	0x01000100																																																							
Name	Reserved								Reserved								Reserved								Reserved								Reserved								Reserved								Reserved							
Access								RO								RW	RW	RW								RO								RW	RW	RW																				

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24	SYDB	同步完成标志 B	0b0: 异步计数时 B 通道寄存器正在同步 0b1: 异步计数时 B 通道寄存器同步完成	RO
23:19	Reserved	保留	-	-
18	ICPB	重复捕获标志 B	0b0: 计数匹配标志位 CMFB 为 1 后未发生捕获输入动作 0b1: 计数匹配标志位 CMFB 为 1 后再次发生捕获输入动作	RW
17	OVFB	计数溢出标志 B	0b0: 计数值 (CNTBR) 未计数到 0xFFFF 0b1: 计数值 (CNTBR) 计数到 0xFFFF	RW
16	CMFB	计数匹配标志 B	0b0: CMPBR 寄存器的值与计数值 (CNTBR) 不相等且未发生捕获输入动作 0b1: CMPBR 寄存器的值与计数值 (CNTBR) 相等或发生捕获输入动作	RW
15:9	Reserved	保留	-	-
8	SYDA	同步完成标志 A	0b0: 异步计数时 A 通道寄存器正在同步 0b1: 异步计数时 A 通道寄存器同步完成	RO
7:3	Reserved	保留	-	-
2	ICPA	重复捕获标志 A	0b0: 计数匹配标志位 CMFA 为 1 后未发生捕获输入动作 0b1: 计数匹配标志位 CMFA 为 1 后再次发生捕获输入动作	RW
1	OVFA	计数溢出标志 A	0b0: 计数值 (CNTAR) 未计数到 0xFFFF 0b1: 计数值 (CNTAR) 计数到 0xFFFF	RW
0	CMFA	计数匹配标志 A	0b0: CMPAR 寄存器的值与计数值 (CNTAR) 不相等且未发生捕获输入动作 0b1: CMPAR 寄存器的值与计数值 (CNTAR) 相等或发生捕获输入动作	RW

25.5 注意事项

1. 在异步计数动作时，需先设定 BCONR.ASYNCLKA位选择异步时钟源,再设定 BCONR.SYNSA位选择异步计数方式，然后再启动 Timer2。
2. 在选择异步计数的情况下，修改计数值(CNTAR)、基准值 (CMPAR)、启动位 (BCONR.CSTA)、状态位 (STFLR.CMFA、STFLR.OVFA、STFLR.ICPA) 时，Timer2 从接收到写动作后经过 3 个异步计数时钟才将修改值写入对应的寄存器中。
3. 在选择异步计数 (BCONR.SYNSA=0b1) 的情况下，连续对以下任一寄存器:计数值(CNTAR)、基准值 (CMPAR)、启动位 (BCONR.CSTA)、状态位 (STFLR.CMFA、STFLR.OVFA、STFLR.ICPA) 进行写动作时，需等待状态位(STFLR.SYDA)为 1 后再进行下一个上述任一寄存器的写动作。
4. 在选择异步计数的情况下，请将 BCONR.SYNCLKA[1:0]设定为 0。
5. Timer2 作为事件输出时基准值设置需要大于 5。

26 通用定时器 (Timer0)

26.1 简介

通用定时器 0 (Timer0) 是一个可以实现同步计数和异步计数方式的基本定时器。该定时器内含 2 个通道 (CH-A 和 CH-B)，可以在计数期间产生比较匹配事件与计数溢出事件。该事件可以触发中断，也可以用于触发其他模块动作。本系列产品中搭载 5 个单元的 Timer0。

26.2 功能说明

26.2.1 Timer0 框图

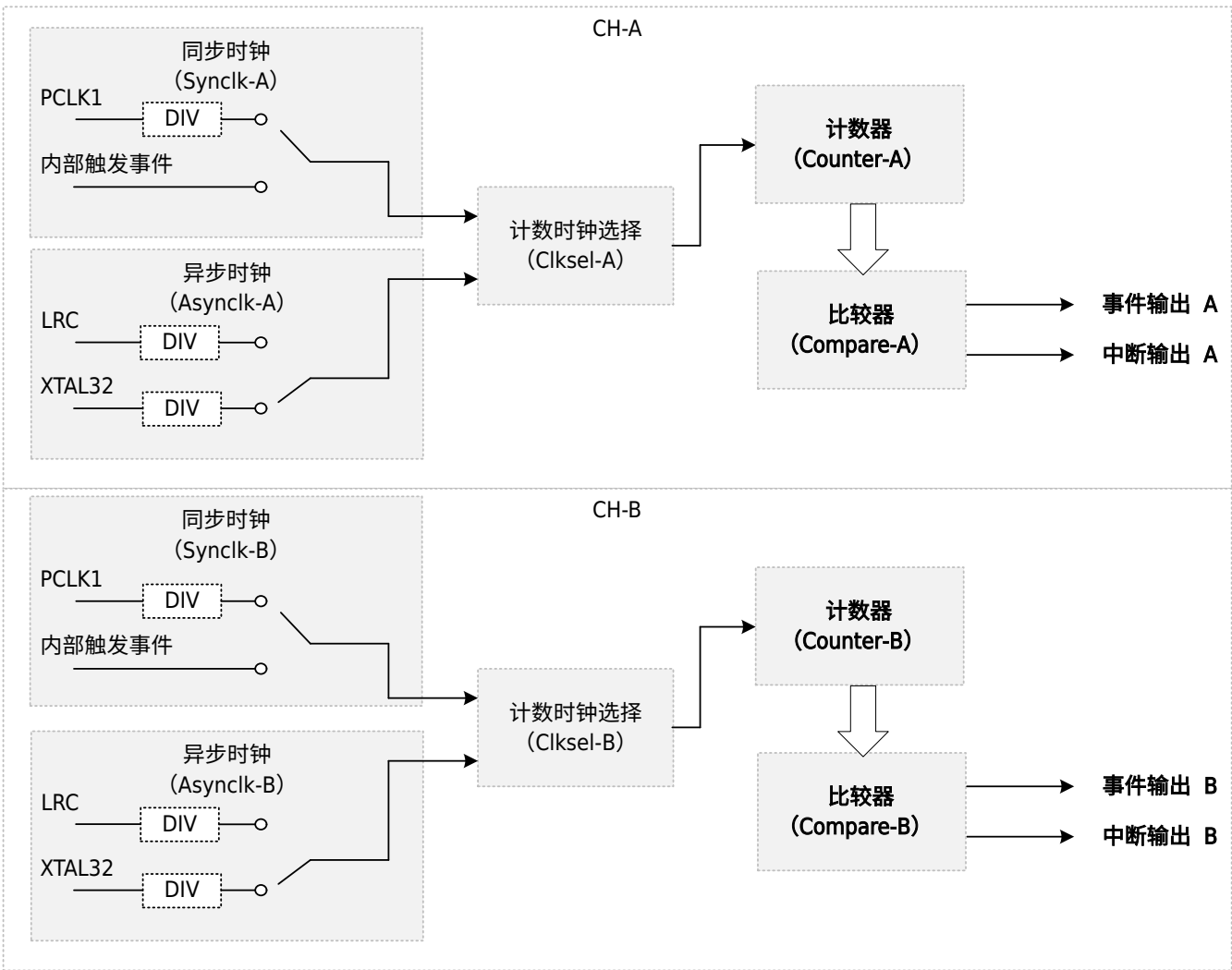


图 26-1 Timer0 基本框图

26.2.2 时钟源选择

Timer0 的计数方式可以选择同步计数方式或异步计数方式。

同步计数方式是指定时器的计数时钟和总线访问时钟（寄存器读写操作时钟）有同步时序关系；异步计数方式是指定时器的计数时钟和总线访问时钟（寄存器读写操作时钟）是非同步的时序关系。异步计数方式时对寄存器读操作时，定时器等的状态可能正在发生变化、读出不可预期的状态。因此，在异步计数方式时，寄存器读操作须在计数停止状态下实现。

同步计数时钟源

同步计数方式时 (BCONR.SYNSA=0)，时钟源可以有以下几种选择 (BCONR.SYNCLKA设定)：

- 1. PCLK1 及 PCLK1 的 2、4、8、16、32、64、128、256、512、1024 分频作为同步计数时钟 (BCONR.SYNCLKA=0 & BCONR.CKDIVA[3:0]设定)；
- 2. 内部硬件触发事件输入作为同步计数时钟 (BCONR.SYNCLKA=1)。

异步计数时钟源

异步计数方式时 (BCONR.SYNSA=1)，时钟源可以有以下几种选择 (BCONR.ASYNCLKA设定选择)：

- 1. LRC 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为异步计数时钟 (BCONR.ASYNCLKA=0 & BCONR.CKDIVA [3:0]设定)；
- 2. XTAL32 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为异步计数时钟 (BCONR.ASYNCLKA=1 & BCONR.CKDIVA [3:0]设定)。

26.2.3 基本计数

Timer0 的每个通道可设定基准计数值，在计数值和基准值相等时产生计数比较匹配事件。

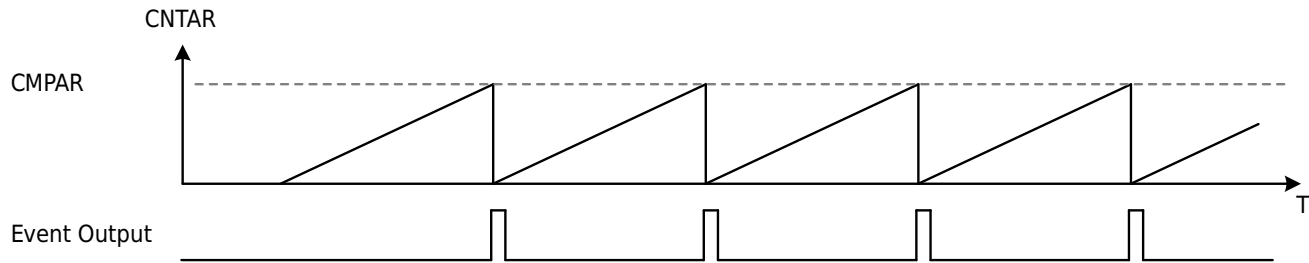


图 26-2 Timer0 计数时序图

26.2.4 硬件触发

Timer0 的 2 个通道有一个共用的内部硬件触发源，可以通过基本控制寄存器 (BCONR) 的相关设定来控制定时器的状态 (启动、停止、清零) 以及捕获输入动作等。

该硬件触发源的源选择通过向硬件触发选择寄存器 (TMR0_TRGSEL) 中输入对应的编号来实现，寄存器内容请参见《自动运行系统 (AOS)》章节相关描述，具体的事件对应关系请参见《中断控制器 (INTC)》章节相关描述。使用内部硬件触发功能时，需要先将功能时钟控制寄存器 0 (PWC_FCG0) 的 AOS 功能控制位使能。

26.2.5 软件同步启动

产品配置了两个计数同步使能寄存器 (TMR_SYNNR1 和 TMR_SYNNR2)，用于同步启动产品所搭载的定时器，如下表所示 (表中 * 表示各类型 TMR 任意单元，TMR* 表示各类型 TMR)。对 TMR_SYNNR1 和 TMR_SYNNR2 寄存器的多个启动位进行置 1，可以实现多个定时器同时开始计数。启动位的具体分配请参考计数同步使能寄存器 (TMR_SYNNR1 和 TMR_SYNNR2)，先配置 TMR_SYNNR1 寄存器再配置 TMR_SYNNR2 寄存器。



注意

- 使用同步启动功能时，请确保需要同步启动定时器的 PCLK 时钟频率相同且同时打开需要同步启动的定时器时钟 (PWC_FCG2)。
- 使用同步启动功能时，请确保需要同步启动定时器均是同步计数方式。
- TMR_SYNNR1 寄存器必须在 TMR_SYNNR2 寄存器写入的情况下被同步写入，即单独设置 TMR_SYNNR1 寄存器无效。

表 26-1 计数器与 PWM 同步启动状态

同步启动	TMR 单元	计数器同步	PWM 同步
同类型 TMR	TMR0_1~3	✓	×
	TMR0_4~5	✓	×
	TMR2_*	✓	✓
	TMR4_*	✓	✓
	TMR6_*	✓	✓
	TMRA_*	✓	✓
不同类型 TMR	TMR0_*+TMR*	×	×
	TMR4_*+TMR*	×	×
	TMR2_*+TMR6_*	✓	✓
	TMR2_*+TMRA_*	✓	✓
	TMR6_*+TMRA_*	✓	✓
	TMR2_*+TMR6_*+TMRA_*	✓	✓

26.2.6 中断和事件

26.2.6.1 中断

一个 Timer0 含有 4 个中断输出，分别是通道 A 和通道 B 的计数比较匹配中断或捕获输入中断、通道 A 和通道 B 的计数溢出中断。

基准值寄存器（CMPAR、CMPBR）共计 2 个，可分别与计数值寄存器（CNTAR、CNTBR）比较产生比较匹配有效信号。计数比较匹配时，状态标志寄存器（STFLR）中的 STFLR.CMFA位分别会被置为 1。此时若设定基本控制寄存器（BCONR）的 BCONR.CMENA位使能中断，则对应的中断请求（TMR0_<t>_CMPn，<t>=1~5；n=A、B）也会被触发。

在内部硬件触发输入作为捕获输入条件时，可以产生相应的捕获输入动作。此时若设定基本控制寄存器（BCONR）的 BCONR.CMENA位使能中断，则对应的中断请求（TMR0_<t>_CMPn，<t>=1~5；n=A、B）被触发。

计数值寄存器（CNTAR、CNTBR）计数到 0xFFFF 时发生计数溢出事件，此时状态标志寄存器（STFLR）中的 STFLR.OVFA位会被置为 1。若设定基本控制寄存器（BCONR）的 BCONR.OVENA位使能中断，则对应的中断请求（TMR0_<t>_OVFn，<t>=1~5；n=A、B）也会被触发。

在选择异步计数模式时，单元 1 的基准值寄存器（CMPAR）产生的比较匹配中断可用于在停止模式时对系统进行唤醒，具体请参见《中断控制器（INTC）》章节相关描述。

26.2.6.2 事件

一个 Timer0 含有 4 个事件输出，分别是通道 A 和通道 B 的计数比较匹配事件或捕获输入事件、通道 A 和通道 B 的计数溢出事件。

在计数过程中发生计数比较匹配或捕获输入动作或计数溢出时，会分别产生相应的事件请求（TMR0_<t>_CMPn 或 TMR0_<t>_OVFn，<t>=1~5；n=A、B）输出信号，可以用于选择触发其他模块。

26.3 寄存器

26.3.1 寄存器总表

- Timer0_1 基地址：0x40024000
- Timer0_2 基地址：0x40024400
- Timer0_3 基地址：0x40029000
- Timer0_4 基地址：0x40029400
- Timer0_5 基地址：0x40029800



说明

Timer0 为定时器名称，Timer0_<t>（<t>：1~5）为该定时器单元。

表 26-2 通用定时器（Timer0）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	TMRO_CNTAR	计数值寄存器 A	32
0x0004	TMRO_CNTBR	计数值寄存器 B	32
0x0008	TMRO_CMPAR	基准值寄存器 A	32
0x000C	TMRO_CMPBR	基准值寄存器 B	32
0x0010	TMRO_BCONR	基本控制寄存器	32
0x0014	TMRO_STFLR	状态标志寄存器	32

TMR_SYNENR 基地址：0x40055400

表 26-3 TMR_SYNENR 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x14	TMR_SYNENR1	计数同步使能寄存器 1	32
0x40	TMR_SYNENR2	计数同步使能寄存器 2	32

26.3.2 计数值寄存器 m (TMR0_CNTmR) (m=A/B)

偏移地址 (Offset) : 0x0000 (CNTAR) , 0x0004 (CNTBR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNT	计数值	当前定时器的计数值  说明 在计数中时，不能向该寄存器中写入值。	RW

26.3.3 基准值寄存器 m (TMR0_CMPmR) (m=A/B)

偏移地址 (Offset) : 0x0008 (CMPAR) , 0x000C (CMPBR)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																CMP															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CMP	基准值	设定计数基准值，产生比较匹配事件	RW

26.3.4 基本控制寄存器 (TMR0_BCONR)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name																																
	HICPB	HCLEB	HSTPB	HSTAB	Reserved	ASYNCLKB	SYNCLKB	SYNSB		CKDIVB			OVENB	CMENB	CAPMDB	CSTB	HICPA	HCLEA	HSTPA	HSTAA	Reserved	ASYNCLKA	SYNCLKA	SYNSA		CKDIVA			OVENA	CMENA	CAPMDA	CSTA
Access	RW	RW	RW	RW		RW	RW	RW		RW			RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW		RW			RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	HICPB	硬件触发捕获输入 B	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，捕获输入无效 ● 0b1：条件匹配时，捕获输入有效	RW
30	HCLEB	硬件触发清零 B	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，定时器清零无效 ● 0b1：条件匹配时，定时器清零有效	RW
29	HSTPB	硬件触发停止 B	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，定时器停止无效 ● 0b1：条件匹配时，定时器停止有效	RW
28	HSTAB	硬件触发启动 B	条件：内部硬件触发事件有效 ● 0b0：条件匹配时，定时器启动无效 ● 0b1：条件匹配时，定时器启动有效	RW
27	Reserved	保留	-	-
26	ASYNCLKB	通道 B 异步计数时钟源选择	● 0b0：LRC ● 0b1：XTAL32  说明 当使能 UART 对应通道的接收超时功能时，时钟源不再是 XTAL32（硬件上可以不存在 XTAL32 晶振），而是 UART 波特率生成器产生的时钟 tick。	RW
25	SYNCLKB	通道 B 同步计数时钟源选择	● 0b0：PCLK1 ● 0b1：内部硬件触发事件	RW
24	SYNSB	通道 B 计数方式选择	● 0b0：同步计数方式 ● 0b1：异步计数方式	RW
23:20	CKDIVB	通道 B 计数时钟分频选择	● 0b0000：时钟源 ● 0b0001：时钟源/2 ● 0b0010：时钟源/4 ● 0b0011：时钟源/8 ● 0b0100：时钟源/16 ● 0b0101：时钟源/32 ● 0b0110：时钟源/64 ● 0b0111：时钟源/128 ● 0b1000：时钟源/256 ● 0b1001：时钟源/512	RW

位/位域	标记	位名	功能描述	读写
23:20	CKDIVB	通道 B 计数时钟分频选择	● 0b1010: 时钟源/1024 ● 其他: 禁止设置  说明 被分频的时钟源可以是异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
19	OVENB	计数溢出中断使能 B	中断条件: 计数溢出, 即计数值 (CNTBR) = 0xFFFF 时 ● 0b0: 禁止中断 ● 0b1: 使能中断	RW
18	CMENB	计数匹配中断使能 B	中断条件: CMPBR 寄存器与计数值 (CNTBR) 相等时, 或者发生捕获输入事件 ● 0b0: 禁止中断 ● 0b1: 使能中断	RW
17	CAPMDB	功能模式选择 B	● 0b0: 比较输出功能 ● 0b1: 捕获输入功能	RW
16	CSTB	通道 B 定时器启动	● 0b0: 通道 B 定时器关闭 ● 0b1: 通道 B 定时器启动  说明 该位在硬件触发停止条件有效时, 会自动变为 0。	RW
15	HICPA	硬件触发捕获输入 A	条件: 内部硬件触发事件有效 ● 0b0: 条件匹配时, 捕获输入无效 ● 0b1: 条件匹配时, 捕获输入有效	RW
14	HCLEA	硬件触发清零 A	条件: 内部硬件触发事件有效 ● 0b0: 条件匹配时, 定时器清零无效 ● 0b1: 条件匹配时, 定时器清零有效	RW
13	HSTPA	硬件触发停止 A	条件: 内部硬件触发事件有效 ● 0b0: 条件匹配时, 定时器停止无效 ● 0b1: 条件匹配时, 定时器停止有效	RW
12	HSTAA	硬件触发启动 A	条件: 内部硬件触发事件有效 ● 0b0: 条件匹配时, 定时器启动无效 ● 0b1: 条件匹配时, 定时器启动有效	RW
11	Reserved	保留	-	-
10	ASYNCLKA	通道 A 异步计数时钟源选择	● 0b0: LRC ● 0b1: XTAL32  说明 当使能 UART 对应通道的接收超时功能时, 时钟源不再是 XTAL32 (硬件上可以不存在 XTAL32 晶振), 而是 UART 波特率生成器产生的时钟 tick。	RW
9	SYNCLKA	通道 A 同步计数时钟源选择	● 0b0: PCLK1 ● 0b1: 内部硬件触发事件	RW
8	SYNSA	通道 A 计数方式选择	● 0b0: 同步计数方式 ● 0b1: 异步计数方式	RW

位/位域	标记	位名	功能描述	读写
7:4	CKDIVA	通道 A 计数时钟分频选择	● 0b0000: 时钟源 ● 0b0001: 时钟源/2 ● 0b0010: 时钟源/4 ● 0b0011: 时钟源/8 ● 0b0100: 时钟源/16 ● 0b0101: 时钟源/32 ● 0b0110: 时钟源/64 ● 0b0111: 时钟源/128 ● 0b1000: 时钟源/256 ● 0b1001: 时钟源/512 ● 0b1010: 时钟源/1024 ● 其他: 禁止设置  说明 被分频的时钟源可以是异步计数时的各种时钟源、同步计数时的 PCLK1。	RW
3	OVENA	计数溢出中断使能 A	中断条件: 计数溢出, 即计数值 (CNTAR) = 0xFFFF 时 ● 0b0: 禁止中断 ● 0b1: 使能中断	RW
2	CMENA	计数匹配中断使能 A	中断条件: CMPAR 寄存器与计数值 (CNTAR) 相等时, 或者发生捕获输入事件 ● 0b0: 禁止中断 ● 0b1: 使能中断	RW
1	CAPMDA	功能模式选择 A	● 0b0: 比较输出功能 ● 0b1: 捕获输入功能	RW
0	CSTA	通道 A 定时器启动	● 0b0: 通道 A 定时器关闭 ● 0b1: 通道 A 定时器启动  说明 该位在硬件触发停止条件有效时, 会自动变为 0。	RW



说明

该寄存器中提到的内部硬件触发事件 (Bit31~Bit28 和 Bit15~Bit12) 及异步计数时的 XTAL32 时钟源 (Bit26 和 Bit10), 在 USART 模块的 TIMEOUT 功能有效时均由 USART 模块提供输入, 具体请参见《通用同步异步收发器 (USART)》章节相关描述。

26.3.5 状态标志寄存器 (TMR0_STFLR)

Offset	Bit Position																																			
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x01000100																																			
Name	Reserved							SYDB	Reserved							ICPB	OVFB	CMFB	Reserved							SYDA	Reserved							ICPA	OVFA	CMFA
Access								RO								RW	RW	RW								RO								RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24	SYDB	同步完成标志 B	0b0: 异步计数时 B 通道寄存器正在同步 0b1: 异步计数时 B 通道寄存器同步完成	RO
23:19	Reserved	保留	-	-
18	ICPB	重复捕获标志 B	0b0: 计数匹配标志位 CMFB 为 1 后未发生捕获输入动作 0b1: 计数匹配标志位 CMFB 为 1 后再次发生捕获输入动作	RW
17	OVFB	计数溢出标志 B	0b0: 计数值 (CNTBR) 未计数到 0xFFFF 0b1: 计数值 (CNTBR) 计数到 0xFFFF	RW
16	CMFB	计数匹配标志 B	0b0: CMPBR 寄存器的值与计数值 (CNTBR) 不相等且未发生捕获输入动作 0b1: CMPBR 寄存器的值与计数值 (CNTBR) 相等或发生捕获输入动作	RW
15:9	Reserved	保留	-	-
8	SYDA	同步完成标志 A	0b0: 异步计数时 A 通道寄存器正在同步 0b1: 异步计数时 A 通道寄存器同步完成	RO
7:3	Reserved	保留	-	-
2	ICPA	重复捕获标志 A	0b0: 计数匹配标志位 CMFA 为 1 后未发生捕获输入动作 0b1: 计数匹配标志位 CMFA 为 1 后再次发生捕获输入动作	RW
1	OVFA	计数溢出标志 A	0b0: 计数值 (CNTAR) 未计数到 0xFFFF 0b1: 计数值 (CNTAR) 计数到 0xFFFF	RW
0	CMFA	计数匹配标志 A	0b0: CMPAR 寄存器的值与计数值 (CNTAR) 不相等且未发生捕获输入动作 0b1: CMPAR 寄存器的值与计数值 (CNTAR) 相等或发生捕获输入动作	RW

26.3.6 计数同步使能寄存器 1 (TMR_SYNENR1)

Offset	Bit Position																																
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	TMR6_8	TMR6_7	TMR6_6	TMR6_5	TMR6_4	TMR6_3	TMR6_2	TMR6_1	Reserved						TMR4_3	TMR4_2	TMR4_1	Reserved						TMR0_5_B	TMR0_5_A	TMR0_4_B	TMR0_4_A	TMR0_3_B	TMR0_3_A	TMR0_2_B	TMR0_2_A	TMR0_1_B	TMR0_1_A
Access	RW	RW	RW	RW	RW	RW	RW	RW																									

位/位域	标记	位名	功能描述	读写
31	TMR6_8	Timer6_8 软件启动使能位	0b0: Timer6 单元 8 软件启动无效 0b1: Timer6 单元 8 软件启动有效且定时器开始计数	RW
30	TMR6_7	Timer6_7 软件启动使能位	0b0: Timer6 单元 7 软件启动无效 0b1: Timer6 单元 7 软件启动有效且定时器开始计数	RW
29	TMR6_6	Timer6_6 软件启动使能位	0b0: Timer6 单元 6 软件启动无效 0b1: Timer6 单元 6 软件启动有效且定时器开始计数	RW
28	TMR6_5	Timer6_5 软件启动使能位	0b0: Timer6 单元 5 软件启动无效 0b1: Timer6 单元 5 软件启动有效且定时器开始计数	RW
27	TMR6_4	Timer6_4 软件启动使能位	0b0: Timer6 单元 4 软件启动无效 0b1: Timer6 单元 4 软件启动有效且定时器开始计数	RW
26	TMR6_3	Timer6_3 软件启动使能位	0b0: Timer6 单元 3 软件启动无效 0b1: Timer6 单元 3 软件启动有效且定时器开始计数	RW
25	TMR6_2	Timer6_2 软件启动使能位	0b0: Timer6 单元 2 软件启动无效 0b1: Timer6 单元 2 软件启动有效且定时器开始计数	RW
24	TMR6_1	Timer6_1 软件启动使能位	0b0: Timer6 单元 1 软件启动无效 0b1: Timer6 单元 1 软件启动有效且定时器开始计数	RW
23:19	Reserved	保留	-	-
18	TMR4_3	Timer4_3 软件启动使能位	0b0: Timer4 单元 3 软件启动无效 0b1: Timer4 单元 3 软件启动有效且定时器开始计数	RW
17	TMR4_2	Timer4_2 软件启动使能位	0b0: Timer4 单元 2 软件启动无效 0b1: Timer4 单元 2 软件启动有效且定时器开始计数	RW
16	TMR4_1	Timer4_1 软件启动使能位	0b0: Timer4 单元 1 软件启动无效 0b1: Timer4 单元 1 软件启动有效且定时器开始计数	RW

位/位域	标记	位名	功能描述	读写
15:10	Reserved	保留	-	-
9	TMR0_5_B	Timer0_5 通道 B 软件启动使能位	0b0: Timer0 单元 5 通道 B 软件启动无效 0b1: Timer0 单元 5 通道 B 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
8	TMR0_5_A	Timer0_5 通道 A 软件启动使能位	0b0: Timer0 单元 5 通道 A 软件启动无效 0b1: Timer0 单元 5 通道 A 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
7	TMR0_4_B	Timer0_4 通道 B 软件启动使能位	0b0: Timer0 单元 4 通道 B 软件启动无效 0b1: Timer0 单元 4 通道 B 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
6	TMR0_4_A	Timer0_4 通道 A 软件启动使能位	0b0: Timer0 单元 4 通道 A 软件启动无效 0b1: Timer0 单元 4 通道 A 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
5	TMR0_3_B	Timer0_3 通道 B 软件启动使能位	0b0: Timer0 单元 3 通道 B 软件启动无效 0b1: Timer0 单元 3 通道 B 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
4	TMR0_3_A	Timer0_3 通道 A 软件启动使能位	0b0: Timer0 单元 3 通道 A 软件启动无效 0b1: Timer0 单元 3 通道 A 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
3	TMR0_2_B	Timer0_2 通道 B 软件启动使能位	0b0: Timer0 单元 2 通道 B 软件启动无效 0b1: Timer0 单元 2 通道 B 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW

位/位域	标记	位名	功能描述	读写
2	TMR0_2_A	Timer0_2 通道 A 软件启动使能位	● 0b0: Timer0 单元 2 通道 A 软件启动无效 ● 0b1: Timer0 单元 2 通道 A 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
1	TMR0_1_B	Timer0_1 通道 B 软件启动使能位	● 0b0: Timer0 单元 1 通道 B 软件启动无效 ● 0b1: Timer0 单元 1 通道 B 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
0	TMR0_1_A	Timer0_1 通道 A 软件启动使能位	● 0b0: Timer0 单元 1 通道 A 软件启动无效 ● 0b1: Timer0 单元 1 通道 A 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW



说明

- 请不要在本定时器异步计数的情况下使用计数同步功能。
- TMR_SYNENR1 不受 RMU_FRST 寄存器的控制。

26.3.7 计数同步使能寄存器 2 (TMR_SYNENR2)

Offset	Bit Position																																				
0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved								TMR2_4_B	TMR2_4_A	TMR2_3_B	TMR2_3_A	TMR2_2_B	TMR2_2_A	TMR2_1_B	TMR2_1_A	Reserved								TMRA_12	TMRA_11	TMRA_10	TMRA_9	TMRA_8	TMRA_7	TMRA_6	TMRA_5	TMRA_4	TMRA_3	TMRA_2	TMRA_1	
Access									RW	RW	RW	RW	RW	RW	RW	RW									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	TMR2_4_B	Timer2_4 通道 B 启动使能位	0b0: Timer2 单元 4 通道 B 启动无效 0b1: Timer2 单元 4 通道 B 启动有效且定时器开始计数	RW
22	TMR2_4_A	Timer2_4 通道 A 启动使能位	0b0: Timer2 单元 4 通道 A 启动无效 0b1: Timer2 单元 4 通道 A 启动有效且定时器开始计数	RW
21	TMR2_3_B	Timer2_3 通道 B 启动使能位	0b0: Timer2 单元 3 通道 B 启动无效 0b1: Timer2 单元 3 通道 B 启动有效且定时器开始计数	RW
20	TMR2_3_A	Timer2_3 通道 A 启动使能位	0b0: Timer2 单元 3 通道 A 启动无效 0b1: Timer2 单元 3 通道 A 启动有效且定时器开始计数	RW
19	TMR2_2_B	Timer2_2 通道 B 启动使能位	0b0: Timer2 单元 2 通道 B 启动无效 0b1: Timer2 单元 2 通道 B 启动有效且定时器开始计数	RW
18	TMR2_2_A	Timer2_2 通道 A 启动使能位	0b0: Timer2 单元 2 通道 A 启动无效 0b1: Timer2 单元 2 通道 A 启动有效且定时器开始计数	RW
17	TMR2_1_B	Timer2_1 通道 B 启动使能位	0b0: Timer2 单元 1 通道 B 启动无效 0b1: Timer2 单元 1 通道 B 启动有效且定时器开始计数	RW
16	TMR2_1_A	Timer2_1 通道 A 启动使能位	0b0: Timer2 单元 1 通道 A 启动无效 0b1: Timer2 单元 1 通道 A 启动有效且定时器开始计数	RW
15:12	Reserved	保留	-	-
11	TMRA_12	TimerA_12 软件启动使能位	0b0: TimerA 单元 12 软件启动无效 0b1: TimerA 单元 12 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW

位/位域	标记	位名	功能描述	读写
10	TMRA_11	TimerA_11 软件启动使能位	0b0: TimerA 单元 11 软件启动无效 0b1: TimerA 单元 11 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
9	TMRA_10	TimerA_10 软件启动使能位	0b0: TimerA 单元 10 软件启动无效 0b1: TimerA 单元 10 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
8	TMRA_9	TimerA_9 软件启动使能位	0b0: TimerA 单元 9 软件启动无效 0b1: TimerA 单元 9 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
7	TMRA_8	TimerA_8 软件启动使能位	0b0: TimerA 单元 8 软件启动无效 0b1: TimerA 单元 8 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
6	TMRA_7	TimerA_7 软件启动使能位	0b0: TimerA 单元 7 软件启动无效 0b1: TimerA 单元 7 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
5	TMRA_6	TimerA_6 软件启动使能位	0b0: TimerA 单元 6 软件启动无效 0b1: TimerA 单元 6 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
4	TMRA_5	TimerA_5 软件启动使能位	0b0: TimerA 单元 5 软件启动无效 0b1: TimerA 单元 5 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
3	TMRA_4	TimerA_4 软件启动使能位	0b0: TimerA 单元 4 软件启动无效 0b1: TimerA 单元 4 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW

位/位域	标记	位名	功能描述	读写
2	TMRA_3	TimerA_3 软件启动使能位	● 0b0: TimerA 单元 3 软件启动无效 ● 0b1: TimerA 单元 3 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
1	TMRA_2	TimerA_2 软件启动使能位	● 0b0: TimerA 单元 2 软件启动无效 ● 0b1: TimerA 单元 2 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW
0	TMRA_1	TimerA_1 软件启动使能位	● 0b0: TimerA 单元 1 软件启动无效 ● 0b1: TimerA 单元 1 软件启动有效且定时器开始计数  说明 请在同步计数的情况下使用该功能。	RW



说明

1. 请不要在本定时器异步计数的情况下使用计数同步功能。
2. TMR_SYNENR2 不受 RMU_FRST 寄存器的控制。

26.4 注意事项

1. 在异步计数动作时，需先设定 BCONR.ASYNCLKA位选择异步时钟源，再设定 BCONR.SYNSA位选择异步计数方式，然后再启动 Timer0。
2. 在选择异步计数的情况下，修改计数值 (CNTAR)、基准值 (CMPAR)、启动位 (BCONR.CSTA)、状态位 (STFLR.CMFA、STFLR.OVFA、STFLR.ICPA) 时，Timer0 从接收到写动作后经过 3 个异步计数时钟才将修改值写入对应的寄存器中。
3. 在选择异步计数的情况下，连续对计数值 (CNTAR)、基准值 (CMPAR)、启动位 (BCONR.CSTA)、状态位 (STFLR.CMFA、STFLR.OVFA、STFLR.ICPA) 进行写动作时，需等待状态位(STFLR.SYDA)为 1 后再进行下一个上述任一寄存器的写动作。
4. 在选择异步计数的情况下，请将 BCONR.SYNCLKA设定为 0。

27 实时时钟（RTC）

27.1 简介

实时时钟 (RTC) 是一个以 BCD 码格式保存时间信息的计数器。记录从 00 年到 99 年间的具体日历时间。支持 12/24 小时两种时制，根据月份和年份自动计算日数 28、29（闰年）、30 和 31 日。

27.2 主要特性

表 27-1 RTC 主要特性

计数时钟源	外部低速振荡器（32.768kHz） RTC 专用内部低速振荡器（32.768kHz） XTAL 小数分频时钟
基本功能	BCD 码表示秒、分、时、日、周、月、年时间
	软件启动或停止
	12/24 时制可选、闰年自动识别
	可编程闹钟
	分布式/均匀式补偿 1Hz 时钟输出
	时钟误差补偿功能
	时间戳功能
	备份寄存器复位功能
中断	周期中断
	闹钟中断
	入侵事件中断

27.3 功能说明

27.3.1 RTC 框图

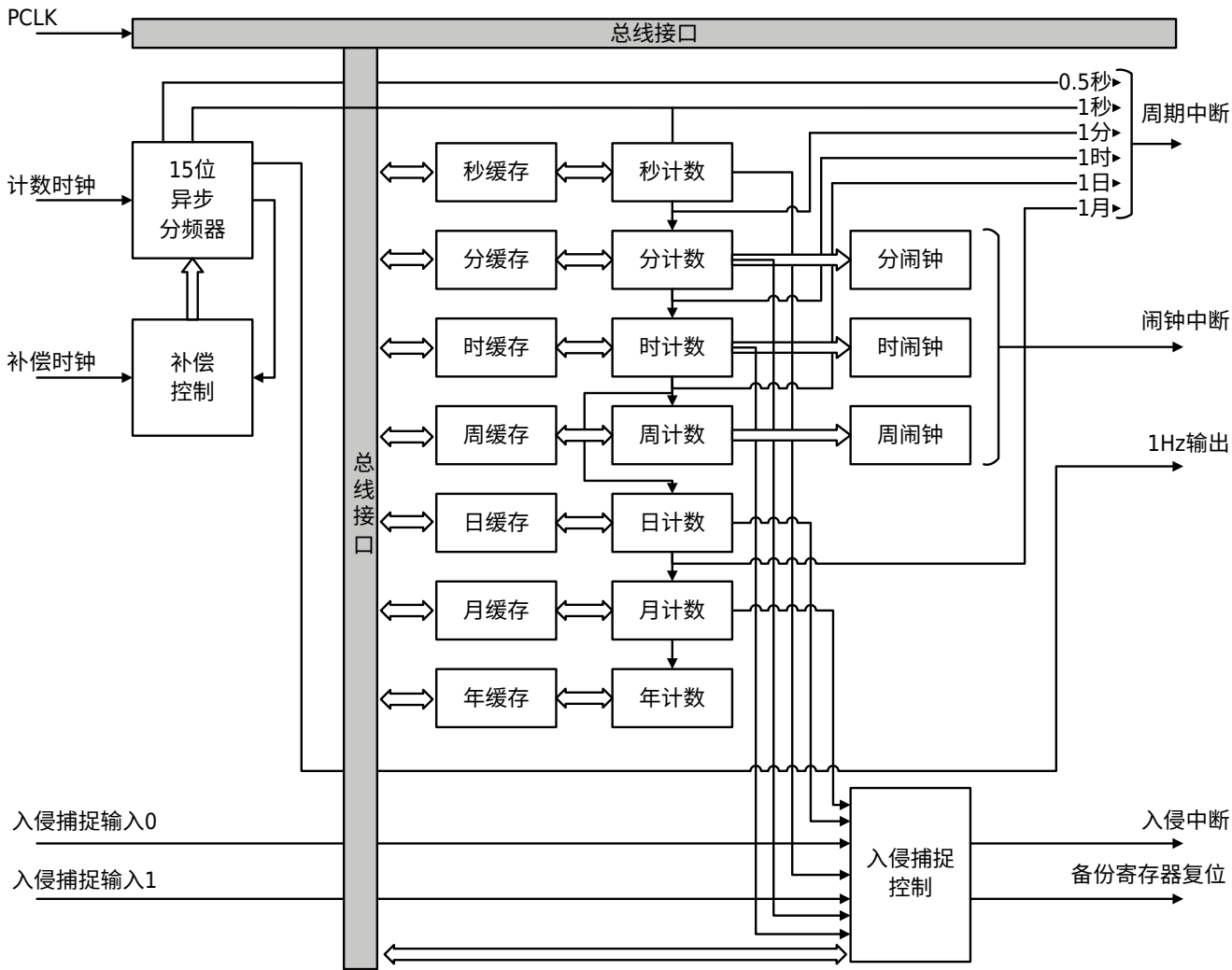


图 27-1 RTC 基本框图

27.3.2 上电设定

RTC 寄存器值上电复位后不定，上电复位后，必须先设设备份域复位寄存器 PWC_VBATRSTR 后，再设定控制寄存器 RTC_CR0.RESET 位复位所有寄存器。然后设置控制寄存器，日历初始值、闹钟设置，启动 RTC。RTC 启动之后，其他外部各种复位请求都不能复位 RTC，RTC 会一直处于工作状态。可以设定控制寄存器的 RTC_CR1.START 位为 0 停止 RTC 工作。RTC 在进行设定时，必须保证时钟源已稳定。

27.3.3 系统低功耗模式切换

在 RTC 计数开始后，系统如果立即切换为低功耗模式时，请执行下列任意一种确认后再进行模式切换。

- 在 RTC_CR1.START=1 设定后，经过 2 个以上的 RTC 计数时钟后再进行模式切换。
- 在 RTC_CR1.START=1 设定后，设定 RTC_CR2.RWREQ=1，查询 RTC_CR2.RWEN=1。设定日历计数寄存器，再设定 RTC_CR2.RWREQ=0，查询 RTC_CR2.RWEN=0，进行模式切换。

27.3.4 时钟误差补偿

由于外部低速晶振在各种温度条件下存在偏差，在需要得到高精度的计数结果时，需要对误差进行补偿。补偿方法请参见“[时钟误差补偿寄存器 \(RTC_ERRCRH、RTC_ERRCRL\)](#)”。

27.3.5 入侵检测

当入侵检测 RTC_TPCRn.TPEN(n=0~1) 有效时, RTC 的外部管脚检测到有效的入侵事件后可分别触发入侵事件检测功能, 时间戳功能和备份寄存器复位功能。

入侵事件检测功能: 入侵事件发生时, 对应引脚的入侵标志会被置起, 当入侵事件连续发生时, 上溢标志会被置起。入侵事件 0 和 1 同时发生时, 入侵事件 0 标志置起后经过 1 个 RTC 计数时钟后入侵事件 1 标志再被置起, 同时上溢标志也被置起。RTC_TPCRn.TPIE (n=0~1) 入侵事件中断使能时, 将发生入侵检测中断。

时间戳功能: RTC_TPCRn.TSTPE (n=0~1) 时间戳功能有效时, 时间戳寄存器将记录入侵事件发生时的秒, 分, 时, 日, 月, 日历时间。

备份寄存器复位: RTC_TPCRn.TPRSTE (n=0~1) 备份寄存器复位使能时, 发生入侵事件后将复位备份寄存器。备份寄存器包括 32 个 32 位寄存器, 用于存储用户 128 字节的备份数据。



注意

在入侵检测有效时, 不管时间戳功能, 入侵事件中断和备份寄存器复位功能是否有效, 事件发生后都将会置起入侵事件标志 RTC_TPSRn.TPFn (n=0~1)。

27.3.6 时间戳功能

当入侵检测 RTC_TPCRn.TPEN(n=0~1)有效时, 且时间戳记录功能 RTC_TPCRn.TSTPE (n=0~1) 有效时, RTC 的外部管脚检测到有效的入侵事件后将触发时间戳功能。时间戳寄存器将记录秒, 分, 时, 日, 月, 日历时间。一组时间戳寄存器分别对应两个入侵事件输入管脚。当管脚在第一次发生入侵事件时, 入侵标志置起, 时间戳被记录, 再次发生入侵事件时, 则上溢标志位被置起, 但是时间戳寄存器仍然保持第一次的记录值。入侵标志寄存器被清除后, 入侵检测发生时, 时间戳才会再次被记录。

27.3.7 中断

RTC 支持 3 种中断类型。计时闹钟中断、定周期中断和入侵检测中断。

闹钟中断

闹钟中断 RTC_ALM, 在控制寄存器 2 (RTC_CR2) 的 ALMIE=1 并且控制寄存器 2 (RTC_CR2) 的 ALME=1 时, 若当前日历时间与分闹钟寄存器 (RTC_ALMMIN)、时闹钟寄存器 (RTC_ALMHOUR)、周闹钟寄存器 (RTC_ALMWEK) 相等时, 触发闹钟中断。闹钟配置独立的标志寄存器位 RTC_CR2.ALMF, 对 RTC_CR2.ALMF 位写 0 清除闹钟标志。

定周期中断

定周期中断 RTC_PRD, 控制寄存器 2 (RTC_CR2) 的 PRDIE=1 时, 选择的周期发生后, 触发定周期唤醒中断。定周期中断配置独立的标志寄存器 RTC_CR2.PRDF, 可通过对 RTC_CR2.PRDF 位写 0 清除定周期标志。

入侵检测中断

当入侵检测有效 RTC_TPCRn.TPEN(n=0~1), 且 RTC_TPCRn.TPIE (n=0~1) 入侵事件中断使能时, RTC 的外部管脚检测到有效的入侵事件后, 将发生入侵事件中断。由于两组入侵检测共用一个中断, 可通过入侵事件检测中断标志位 RTC_TPSR.TPFn (n=0~1) 来区分。

27.4 操作示例

27.4.1 RTC 计数开始设定

- 步骤 1. 上电后, 先设定备份域复位寄存器 PWC_VBATRSTR 后, 再设定 RTC_CR0.RESET=0, 确认 RESET 位为 0 后, 设定 RTC_CR0.RESET=1, 复位所有寄存器;
- 步骤 2. 设定 RTC_CR1.START=0, 停止计数;
- 步骤 3. 设定系统时钟寄存器, 打开外部低速振荡器, 再设定 RTC_CR3, 选择 RTC 计数时钟源;
- 步骤 4. 设定 RTC_CR1, 设定分频、周期、1Hz 时钟输出;

- 步骤 5. 设定秒, 分, 时, 周, 日, 月, 年的日历计数寄存器;
- 步骤 6. 需要进行时钟误差补偿时, 设定计数时钟误差补偿寄存器 RTC_ERRCRL, RTC_ERRCRH;
- 步骤 7. 清除寄存器 RTC_CR2, RTC_TPSR 中的标志寄存器位, 并使能中断;
- 步骤 8. 设定 RTC_CR1.START=1, 计数开始。

27.4.2 读出计数寄存器

- 步骤 1. 在 RTC_CR1.START=1 设定后, 经过 2 个以上的 RTC 计数时钟后, 设定 RTC_CR2.RWREQ=1, 进行日历寄存器读请求;
- 步骤 2. 查询直到 RTC_CR2.RWEN=1;
- 步骤 3. 读出全部或者部分秒, 分, 时, 周, 日, 月, 年计数寄存器值;
- 步骤 4. 设定 RTC_CR2.RWREQ=0;
- 步骤 5. 查询直到 RTC_CR2.RWEN=0。

27.4.3 写入计数寄存器

- 步骤 1. 在 RTC_CR1.START=1 设定后, 经过 2 个以上的 RTC 计数时钟后, 设定 RTC_CR2.RWREQ=1, 进行日历寄存器写请求;
- 步骤 2. 查询直到 RTC_CR2.RWEN=1;
- 步骤 3. 写入全部或者部分秒, 分, 时, 周, 日, 月, 年计数寄存器值;
- 步骤 4. 设定 RTC_CR2.RWREQ=0。注意, 须在 1 秒内完成所有写操作;
- 步骤 5. 查询直到 RTC_CR2.RWEN=0。

27.4.4 闹钟设定

- 步骤 1. 设定 RTC_CR2.ALME=0, 闹钟禁止;
- 步骤 2. 设定 RTC_CR2.ALMIE=1, 闹钟中断许可;
- 步骤 3. 分闹钟 RTC_ALMMIN, 时闹钟 RTC_ALMHOUR, 周闹钟 RTC_ALMWEED 设定;
- 步骤 4. 设定 RTC_CR2.ALME=1, 闹钟许可;
- 步骤 5. 等待闹钟中断;
- 步骤 6. 闹钟发生, RTC_CR2.ALMF=1, 进入闹钟中断处理。

27.4.5 1Hz 输出

RTC 可输出 1Hz 时钟, 提供三种精度输出方式, 第一种, 无时钟补偿的普通精度 1Hz 输出; 第二种, 每 32 秒内平均补偿的分布式补偿 1Hz 输出和第三种每秒补偿的均匀式补偿 1Hz 输出。当时钟误差补偿功能有效 RTC_ERRCRH.COMPEN=1 时可选择分布式补偿 1Hz 输出和均匀式补偿 1Hz 输出。

普通精度的 1Hz 输出

普通精度的 1Hz 输出设定如下:

- 步骤 1. 设定 RTC_CR0.RESET=0, 确认 RESET 位为 0 后, 设定 RTC_CR0.RESET=1, 复位日历计数寄存器;
- 步骤 2. 设定 RTC_CR1.START=0, 计数停止;
- 步骤 3. 1Hz 输出引脚设定;
- 步骤 4. RTC_CR1.ONEHZOE=1, 时钟输出许可;
- 步骤 5. 设定 RTC_CR1.START=1, 计数开始;
- 步骤 6. 等待 2 个计数周期以上;
- 步骤 7. 1Hz 输出开始。

分布式补偿 1Hz 输出

分布式补偿 1Hz 输出设定如下:

- 步骤 1. 设定 RTC_CR0.RESET=0, 确认 RESET 位为 0 后, 设定 RTC_CR0.RESET=1, 复位日历计数寄存器;
- 步骤 2. 设定 RTC_CR1.START=0, 计数停止;
- 步骤 3. 1Hz 输出引脚设定;
- 步骤 4. RTC_CR1.ONEHZOE=1, 时钟输出许可;
- 步骤 5. 时钟误差补偿寄存器 RTC_ERRCRL.COMP[7:0] 与 RTC_ERRCRH.COMP[8] 补偿数设定;
- 步骤 6. 时钟误差补偿寄存器 RTC_ERRCRH.COMPEN=1, 误差补偿有效;
- 步骤 7. 设定 RTC_CR1.START=1, 计数开始;
- 步骤 8. 等待 2 个计数周期以上;
- 步骤 9. 1Hz 输出开始。

均匀式补偿 1Hz 输出

均匀式补偿 1Hz 输出设定如下:

- 步骤 1. 设定 RTC_CR0.RESET=0, 确认 RESET 位为 0 后, 设定 RTC_CR0.RESET=1, 复位日历计数寄存器;
- 步骤 2. 设定 RTC_CR1.START=0, 计数停止;
- 步骤 3. RTC 输出引脚设定;
- 步骤 4. RTC_CR1.ONEHZOE=1, 时钟输出许可;
- 步骤 5. RTC_CR1.ONEHZSEL=1, 选择输出均匀式补偿 1Hz 时钟;
- 步骤 6. 时钟误差补偿寄存器 RTC_ERRCRL.COMP[7:0] 与 RTC_ERRCRH.COMP[8] 补偿数设定;
- 步骤 7. 时钟误差补偿寄存器 RTC_ERRCRH.COMPEN=1, 精度补偿有效;
- 步骤 8. 设定 RTC_CR1.START=1, 计数开始;
- 步骤 9. 等待 2 个计数周期以上;
- 步骤 10. 1Hz 输出开始。

27.5 寄存器

27.5.1 寄存器总表

RTC 基地址：0x4004C000

表 27-2 实时时钟 (RTC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	RTC_CR0	控制寄存器 0	8
0x0004	RTC_CR1	控制寄存器 1	8
0x0008	RTC_CR2	控制寄存器 2	8
0x000C	RTC_CR3	控制寄存器 3	8
0x0010	RTC_SEC	秒计数寄存器	8
0x0014	RTC_MIN	分计数寄存器	8
0x0018	RTC_HOUR	时计数寄存器	8
0x001C	RTC_WEEK	周计数寄存器	8
0x0020	RTC_DAY	日计数寄存器	8
0x0024	RTC_MON	月计数寄存器	8
0x0028	RTC_YEAR	年计数寄存器	8
0x002C	RTC_ALMMIN	分闹钟寄存器	8
0x0030	RTC_ALMHOUR	时闹钟寄存器	8
0x0034	RTC_ALM WEEK	周闹钟寄存器	8
0x0038	RTC_ERRCRH	时钟误差补偿寄存器 H	8
0x003C	RTC_ERRCRL	时钟误差补偿寄存器 L	8
0x0040	RTC_TPCR0	入侵控制寄存器 0	8
0x0044	RTC_TPCR1	入侵控制寄存器 1	8
0x0048	RTC_TPSR	入侵状态寄存器	8
0x004C	RTC_SECTP	秒时间戳寄存器	8
0x0050	RTC_MINTP	分时间戳寄存器	8
0x0054	RTC_HOURTP	时时间戳寄存器	8
0x0058	RTC_DAYTP	日时间戳寄存器	8
0x005C	RTC_MONTTP	月时间戳寄存器	8

27.5.2 控制寄存器 0（RTC_CR0）

Offset	Bit Position							
0x0000	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved							RESET
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	RESET	RTC 日历计数器 复位	写 1 初始化所有 RTC 寄存器，写 0 无效 读取： ● 0b0：正常计数状态或 RTC 软件复位结束 ● 0b1：RTC 处于复位状态	RW

27.5.3 控制寄存器 1 (RTC_CR1)

Offset	Bit Position							
0x0004	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	START	ONEHZSEL	ONEHZOE	Reserved	AMPM	PRDS		
Access	RW	RW	RW		RW	RW		

位/位域	标记	位名	功能描述	读写																																
7	START	RTC 计数开始	0b0: RTC 计数停止 0b1: RTC 计数开始	RW																																
6	ONEHZSEL	1Hz 输出选择	0b0: 分布式补偿 1Hz 输出 0b1: 均匀式补偿 1Hz 输出  说明 RTC_ERRCRH.COMPEN=1 时, 该位设定有效。	RW																																
5	ONEHZOE	1Hz 输出许可	0b0: 1Hz 输出禁止 0b1: 1Hz 输出许可	RW																																
4	Reserved	保留	-	-																																
3	AMPM	时制选择	0b0: 12 小时时制 0b1: 24 小时时制	RW																																
2:0	PRDS	周期中断选择	周期选择设定:<table><tr><th>PRDS[2]</th><th>PRDS[1]</th><th>PRDS[0]</th><th>周期选择</th></tr><tr><td>0b0</td><td>0b0</td><td>0b0</td><td>不选择</td></tr><tr><td>0b0</td><td>0b0</td><td>0b1</td><td>每 0.5 秒周期</td></tr><tr><td>0b0</td><td>0b1</td><td>0b0</td><td>每 1 秒周期</td></tr><tr><td>0b0</td><td>0b1</td><td>0b1</td><td>每 1 分周期</td></tr><tr><td>0b1</td><td>0b0</td><td>0b0</td><td>每 1 时周期</td></tr><tr><td>0b1</td><td>0b0</td><td>0b1</td><td>每 1 日周期（每日 00 时 00 分 00 秒）</td></tr><tr><td>0b1</td><td>0b1</td><td>-</td><td>每 1 月周期（每月 1 日 00 时 00 分 00 秒）</td></tr></table>  注意 在 START=1 计数过程中写入周期选择时, 为防止误动作请将周期中断许可关闭。并且在写入后应将相关标志位清除。	PRDS[2]	PRDS[1]	PRDS[0]	周期选择	0b0	0b0	0b0	不选择	0b0	0b0	0b1	每 0.5 秒周期	0b0	0b1	0b0	每 1 秒周期	0b0	0b1	0b1	每 1 分周期	0b1	0b0	0b0	每 1 时周期	0b1	0b0	0b1	每 1 日周期（每日 00 时 00 分 00 秒）	0b1	0b1	-	每 1 月周期（每月 1 日 00 时 00 分 00 秒）	RW
PRDS[2]	PRDS[1]	PRDS[0]	周期选择																																	
0b0	0b0	0b0	不选择																																	
0b0	0b0	0b1	每 0.5 秒周期																																	
0b0	0b1	0b0	每 1 秒周期																																	
0b0	0b1	0b1	每 1 分周期																																	
0b1	0b0	0b0	每 1 时周期																																	
0b1	0b0	0b1	每 1 日周期（每日 00 时 00 分 00 秒）																																	
0b1	0b1	-	每 1 月周期（每月 1 日 00 时 00 分 00 秒）																																	

27.5.4 控制寄存器 2 (RTC_CR2)

Offset	Bit Position							
0x0008	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	ALME	ALMIE	PRDIE	Reserved	ALMF	PRDF	RWEN	RWREQ
Access	RW	RW	RW		RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	ALME	闹钟功能使能	0b0: 闹钟功能禁止 0b1: 闹钟功能许可	RW
6	ALMIE	闹钟中断使能	0b0: 闹钟中断禁止 0b1: 闹钟中断许可	RW
5	PRDIE	周期中断使能	0b0: 周期中断禁止 0b1: 周期中断许可	RW
4	Reserved	保留	-	-
3	ALMF	闹钟标志	写 0 清除标志, 写 1 无效 读取: 0b0: 闹钟不匹配 0b1: 闹钟匹配  说明 在 ALME=1 时有效, 闹钟匹配时, 一个计数时钟后置 1。	RW
2	PRDF	周期标志	写 0 清除标志, 写 1 无效 读取: 0b0: 周期不发生 0b1: 周期发生  说明 设定周期发生后, 该位置 1。	RW
1	RWEN	读出/写入允许	0b0: 读出/写入禁止 0b1: 读出/写入允许  说明 日历寄存器读写允许标志。在读出/写入前请确认该位是否为 1。日历寄存器包括秒, 分, 时, 周, 日, 月, 年计数寄存器。	RW

位/位域	标记	位名	功能描述	读写
0	RWREQ	读出/写入请求	0b0: 正常计数模式 0b1: 读出/写入请求  说明 在读出/写入日历寄存器时请将该位置 1，请求读写，由于计数器在连续计数，请在 1 秒的时间内完成读出/写入操作并将该位清 0。 如果 RWREQ 置位后，没有在 1 秒内计时清零，将会导致计数器停止计数，使实时时钟 RTC 不能精确计时；如果 RWREQ 位一致维持置位状态，则实时时钟 RTC 会停止计时。	RW

27.5.5 控制寄存器 3 (RTC_CR3)

Offset	Bit Position							
0x000C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	RCKSEL		Reserved	LRCEN	Reserved			
Access	RW			RW				

位/位域	标记	位名	功能描述	读写
7:6	RCKSEL	RTC 计数时钟选择	0b00: 外部低速振荡器 (XTAL32) 0b01: XTAL 小数分频时钟 0b10: RTC 专用内部低速振荡器 (RTCLRC) 0b11: 禁止设置	RW
5	Reserved	保留	-	-
4	LRCEN	内部低速发振器使能	0b0: RTC 专用内部低速振荡器 RTCLRC 停止 0b1: RTC 专用内部低速振荡器 RTCLRC 工作  说明 RTC 专用内部低速振荡器作为 RTC 时钟源时，请设定 LRCEN 位使能。	RW
3:0	Reserved	保留	-	-

27.5.6 秒计数寄存器 (RTC_SEC)

Offset	Bit Position							
0x0010	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	SECD			SECU			
Access		RW			RW			

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:4	SECD	秒十位	秒十位计数值	RW
3:0	SECU	秒个位	秒个位计数值	RW

表示 0~59 秒，采用十进制计数。请写入十进制 0~59 的 BCD 码，写入超出范围的值将被忽略。

27.5.7 分计数寄存器 (RTC_MIN)

Offset	Bit Position							
0x0014	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	MIND			MINU			
Access		RW			RW			

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:4	MIND	分十位	分十位计数值	RW
3:0	MINU	分个位	分个位计数值	RW

表示 0~59 分，采用十进制计数。请写入十进制 0~59 的 BCD 码，写入超出范围的值将被忽略。

27.5.8 时计数寄存器 (RTC_HOUR)

Offset	Bit Position							
0x0018	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved		HOURD		HOURU			
Access			RW		RW			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5:4	HOURD	时十位	时十位计数值	RW
3:0	HOURU	时个位	时个位计数值	RW

24 小时制时，表示 0~23 小时。

12 小时制时，表示 0~12 小时。Bit5=0 表示 AM，则 01~12 表示上午；Bit5=1 表示 PM，则 01~12 表示下午。

请根据控制位 RTC_CR1.AMPM 的值，设定正确的十进制 0~23 或者 01~12，01~12 的 BCD 码，写入超出范围的值将被忽略。

具体时间取值可参考下表：

24 小时制 (AMPM=1)		12 小时制 (AMPM=0)	
时间	寄存器值	时间	寄存器值
00 时	0x00	AM 12 时	0x12
01 时	0x01	AM 01 时	0x01
02 时	0x02	AM 02 时	0x02
03 时	0x03	AM 03 时	0x03
04 时	0x04	AM 04 时	0x04
05 时	0x05	AM 05 时	0x05
06 时	0x06	AM 06 时	0x06
07 时	0x07	AM 07 时	0x07
08 时	0x08	AM 08 时	0x08
09 时	0x09	AM 09 时	0x09
10 时	0x10	AM 10 时	0x10
11 时	0x11	AM 11 时	0x11
12 时	0x12	PM 12 时	0x32
13 时	0x13	PM 01 时	0x21
14 时	0x14	PM 02 时	0x22
15 时	0x15	PM 03 时	0x23
16 时	0x16	PM 04 时	0x24
17 时	0x17	PM 05 时	0x25
18 时	0x18	PM 06 时	0x26

24 小时制 (AMPM=1)		12 小时制 (AMPM=0)	
时间	寄存器值	时间	寄存器值
19 时	0x19	PM 07 时	0x27
20 时	0x20	PM 08 时	0x28
21 时	0x21	PM 09 时	0x29
22 时	0x22	PM 10 时	0x30
23 时	0x23	PM 11 时	0x31

27.5.9 日计数寄存器 (RTC_DAY)

Offset	Bit Position							
0x0020	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved		DAYD		DAYU			
Access			RW		RW			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5:4	DAYD	日十位	日十位计数值	RW
3:0	DAYU	日个位	日个位计数值	RW

表示 1~31 日，采用十进制计数，自动计算闰年和月份。请写入十进制 1~28、1~29、1~30 或 1~31 的 BCD 码，写入超出范围的值将被忽略。

具体日计数取值可参考下表：

月份	日计数值
2 月（普通年）	01~28
2 月（闰年）	01~29
4、6、9、11 月	01~30
1、3、5、7、8、10、12 月	01~31

27.5.10 周计数寄存器（RTC_WEEK）

Offset	Bit Position							
0x001C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved					WEEK		
Access						RW		

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2:0	WEEK	周	周计数值	RW

十进制 0~6 表示周日~周六。请写入十进制 0~6 的 BCD 码，写入超出范围的值将被忽略。

具体周计数取值可参考下表：

周	周计数值
周日	0x00
周一	0x01
周二	0x02
周三	0x03
周四	0x04
周五	0x05
周六	0x06

27.5.11 月计数寄存器（RTC_MON）

Offset	Bit Position							
0x0024	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved			MON				
Access				RW				

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4:0	MON	月	月计数值	RW

表示 1~12 月，采用十进制计数。请写入十进制 1~12 的 BCD 码，写入超出范围的值将被忽略。

27.5.12 年计数寄存器 (RTC_YEAR)

Offset	Bit Position							
0x0028	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	YEAR				YEARU			
Access	RW				RW			

位/位域	标记	位名	功能描述	读写
7:4	YEAR	年十位	年十位计数值	RW
3:0	YEARU	年个位	年个位计数值	RW

表示 0~99 年，采用十进制计数，根据月进位计数。自动计算闰年如：00、04、08、...、92、96 等。请写入十进制 0~99 的 BCD 码，写入超出范围的值将被忽略。

27.5.13 分闹钟寄存器 (RTC_ALMMIN)

Offset	Bit Position							
0x002C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	ALMMIND				ALMMINU		
Access		RW				RW		

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:4	ALMMIND	分闹钟十位	分闹钟十位计数值	RW
3:0	ALMMINU	分闹钟个位	分闹钟个位计数值	RW

表示 0~59 分，采用十进制计数。请写入十进制 0~59 的 BCD 码，写入超出范围的值将不会发生闹钟匹配。

27.5.14 时闹钟寄存器 (RTC_ALMHOUR)

Offset	Bit Position							
0x0030	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved		ALMHOURD		ALMHOURU			
Access			RW		RW			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5:4	ALMHOURD	时闹钟十位	时闹钟十位计数值	RW
3:0	ALMHOURU	时闹钟个位	时闹钟个位计数值	RW

请根据控制位 RTC_CR1.AMPM 的值，设定正确的十进制 0~23 或者 01~12，21~32 的 BCD 码，写入超出范围的值将不会发生闹钟匹配。

27.5.15 周闹钟寄存器 (RTC_ALMWEEK)

Offset	Bit Position							
0x0034	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	ALMWEEK						
Access		RW						

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:0	ALMWEEK	周闹钟	周闹钟匹配值 Bit0~Bit6 分别对应周日~周六，对应位置 1 时，代表每周该日闹钟有效。 例如：设置 Bit0=1，Bit5=1 表示周日和周五闹钟设定有效。	RW

十进制 0~6 表示周日~周六。请写入十进制 0~6 的 BCD 码，写入超出范围的值将不会发生闹钟匹配。

27.5.16 时钟误差补偿寄存器 (RTC_ERRCRH、RTC_ERRCRL)

RTC_ERRCRH

Offset	Bit Position							
0x0038	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	COMPEN	Reserved						COMP[8]
Access	RW							RW

位/位域	标记	位名	功能描述	读写
7	COMPEN	补偿使能	0b0: 时钟误差补偿无效 0b1: 时钟误差补偿有效	RW
6:1	Reserved	保留	-	-
0	COMP[8]	补偿值	与 COMP[7:0]共同设定补偿值	RW

RTC_ERRCRL

Offset	Bit Position							
0x003C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	COMP[7:0]							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	COMP[7:0]	补偿值	通过补偿值设定，可针对每秒进行±0.96ppm 的精度补偿。补偿值为 9 位带小数点的 2 的补码，后 5 位为小数部分。可补偿范围-275.5ppm~+212.9ppm。最小分辨率 0.96ppm。	RW

具体补偿精度请参考下表：

补偿值设定										补偿值
COMPEN	COMP[8:0]									
1	1	0	0	0	0	0	0	0	0	-275.5ppm
	1	0	0	0	0	0	0	0	1	-274.6ppm
	...									...
	0	0	0	0	0	0	0	0	0	-30.5ppm
	...									...
	0	0	0	0	1	1	1	1	1	-0.96ppm
	0	0	0	1	0	0	0	0	0	0ppm
	0	0	0	1	0	0	0	0	1	+0.96ppm
	...									...
	0	0	1	0	0	0	0	0	0	+30.5ppm
	...									...
	0	1	1	1	1	1	1	1	0	+212.0ppm
	0	1	1	1	1	1	1	1	1	+212.9ppm
0	X	X	X	X	X	X	X	X	X	无补偿

补偿计算说明

当默认状态下直接输出 1Hz 时钟，通过测定该时钟的精度，计算补偿目标值。
假设实际测定值为 0.9999888Hz，则：

实际发振频率 = 32768 × 0.9999888 ≈ 32767.63

补偿目标值 = (实际发振频率 - 目标频率) / 目标频率 × 10⁶

= (32767.63 - 32768) / 32768 × 10⁶

= -11.29ppm

根据公式：

COMP[8:0] = $\left(\frac{\text{补偿目标值 [ppm]} \times 2^{15}}{10^6}\right)_{\text{取2的补码}} + 0b0001.00000$

如果补偿目标值为+20.3ppm，计算相应的寄存器值如下：

COMP[8:0] = (20.3 × 2¹⁵ / 10⁶)_{取2的补码} + 0b0001.00000

= (0.6651904)_{取2的补码} + 0b0001.00000

= 0b0000.10101 + 0b0001.00000

= 0b0001.10101

如果补偿目标值为-20.3ppm，计算相应的寄存器值如下：

COMP[8:0] = (-20.3 × 2¹⁵ / 10⁶)_{取2的补码} + 0b0001.00000

= (-0.6651904)_{取2的补码} + 0b0001.00000

= 0b1111.01011 + 0b0001.00000

= 0b0000.01011

27.5.17 入侵控制寄存器 0 (RTC_TPCR0)

Offset	Bit Position							
0x0040	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	TPEN0	TSTPE0	TPIE0	TPRSTE0	TPNF0		TPCT0	
Access	RW	RW	RW	RW	RW		RW	

位/位域	标记	位名	功能描述	读写
7	TPEN0	入侵检测使能	● 0b0: RTC_IC0 输入引脚上入侵功能无效 ● 0b1: RTC_IC0 输入引脚上入侵功能有效	RW
6	TSTPE0	时间戳使能	● 0b0: 时间戳功能禁止 ● 0b1: 时间戳功能允许	RW
5	TPIE0	入侵事件中断使能	● 0b0: 入侵事件中断禁止 ● 0b1: 入侵事件中断许可	RW
4	TPRSTE0	入侵事件复位使能	● 0b0: 备份寄存器复位禁止 ● 0b1: 备份寄存器复位许可	RW
3:2	TPNF0	滤波功能	● 0b0X: 滤波功能无效 ● 0b10: RTC_IC0 输入引脚检测按照计时时钟进行 3 次一致性检测滤波 ● 0b11: RTC_IC0 输入引脚检测按照计时时钟 32 分频进行 3 次一致性检测滤波	RW
1:0	TPCT0	有效边沿选择	● 0b00: RTC_IC0 输入引脚不检测 ● 0b01: RTC_IC0 输入引脚检测到上升沿时有效 ● 0b10: RTC_IC0 输入引脚检测到下降沿时有效 ● 0b11: RTC_IC0 输入引脚检测到上升沿或下降沿时有效	RW

27.5.18 入侵控制寄存器 1 (RTC_TPCR1)

Offset	Bit Position							
0x0044	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	TPEN1	TSTPE1	TPIE1	TPRSTE1	TPNF1		TPCT1	
Access	RW	RW	RW	RW	RW		RW	

位/位域	标记	位名	功能描述	读写
7	TPEN1	入侵检测使能	● 0b0: RTC_IC1 输入引脚上入侵功能无效 ● 0b1: RTC_IC1 输入引脚上入侵功能有效	RW
6	TSTPE1	时间戳使能	● 0b0: 时间戳功能禁止 ● 0b1: 时间戳功能允许	RW
5	TPIE1	入侵事件中断使能	● 0b0: 入侵事件中断禁止 ● 0b1: 入侵事件中断许可	RW
4	TPRSTE1	入侵事件复位使能	● 0b0: 备份寄存器复位禁止 ● 0b1: 备份寄存器复位许可	RW
3:2	TPNF1	滤波功能	● 0b0X: 滤波功能无效 ● 0b10: RTC_IC1 输入引脚检测按照计时时钟进行 3 次一致性检测滤波 ● 0b11: RTC_IC1 输入引脚检测按照计时时钟 32 分频进行 3 次一致性检测滤波	RW
1:0	TPCT1	有效边沿选择	● 0b00: RTC_IC1 输入引脚不检测 ● 0b01: RTC_IC1 输入引脚检测到上升沿时有效 ● 0b10: RTC_IC1 输入引脚检测到下降沿时有效 ● 0b11: RTC_IC1 输入引脚检测到上升沿或下降沿时有效	RW

27.5.19 入侵状态寄存器 (RTC_TPSR)

Offset	Bit Position							
0x0048	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved					TPOVF	TPF1	TPF0
Access						RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2	TPOVF	入侵状态上溢标志	写 0 清除标志，写 1 无效 读取： ● 0b0：复位状态或未发生入侵状态上溢 ● 0b1：发生入侵上溢	RW
1	TPF1	入侵状态标志 1	写 0 清除标志，写 1 无效 读取： ● 0b0：RTC_IC1 输入引脚上未发生入侵事件 ● 0b1：RTC_IC1 输入引脚上发生入侵事件  说明 为防止误动作，必须在 RTC_TPCR1.TPCT1[1:0]=0b00，入侵检测边沿无效时 才能清除该位。请在设置入侵功能有效前，确认该标志 位的状态为 0。	RW
0	TPF0	入侵状态标志 0	写 0 清除标志，写 1 无效 读取： ● 0b0：RTC_IC0 输入引脚上未发生入侵事件 ● 0b1：RTC_IC0 输入引脚上发生入侵事件  说明 为防止误动作，必须在 RTC_TPCR0.TPCT0[1:0]=0b00，入侵检测边沿无效时 才能清除该位。请在设置入侵功能有效前，确认该标志 位的状态为 0。	RW

27.5.20 秒时间戳寄存器 (RTC_SECTP)

Offset	Bit Position							
0x004C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	RO SECTPD			RO SECTPU			
Access		RO			RO			

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:4	SECTPD	秒十位	秒时间戳十位计数值	RO
3:0	SECTPU	秒个位	秒时间戳个位计数值	RO

27.5.21 分时间戳寄存器 (RTC_MINTP)

Offset	Bit Position							
0x0050	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved	RO MINTPD			RO MINTPU			
Access		RO			RO			

位/位域	标记	位名	功能描述	读写
7	Reserved	保留	-	-
6:4	MINTPD	分十位	分时间戳十位计数值	RO
3:0	MINTPU	分个位	分时间戳个位计数值	RO

27.5.22 时时间戳寄存器 (RTC_HOURTP)

Offset	Bit Position							
0x0054	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved		HOURTPD		HOURTPU			
Access			RO		RO			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5:4	HOURTPD	时十位	时时间戳十位计数值	RO
3:0	HOURTPU	时个位	时时间戳个位计数值	RO

27.5.23 日时间戳寄存器 (RTC_DAYTP)

Offset	Bit Position							
0x0058	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved		DAYTPD		DAYTPU			
Access			RO		RO			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5:4	DAYTPD	日十位	日时间戳十位计数值	RO
3:0	DAYTPU	日个位	日时间戳个位计数值	RO

27.5.24 月时间戳寄存器 (RTC_MONTP)

Offset	Bit Position							
0x005C	7	6	5	4	3	2	1	0
Reset	0xXX							
Name	Reserved			MONTP				
Access				RO				

位/位域	标记	位名	功能描述	读写
7:5	Reserved	保留	-	-
4:0	MONTP	月	月时间戳计数值	RO

28 看门狗计数器 (WDT/SWDT)

28.1 简介

本产品有两个看门狗计数器，一种是计数时钟源为专用内部 RC (SWDTLRC: 10kHz) 的专用看门狗计数器 (SWDT)，另一种是计数时钟源为 PCLK3 的通用看门狗计数器 (WDT)。专用看门狗和通用看门狗是 16 位递减计数器，用来监测由于外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行而产生的软件故障。

28.2 主要特性

两个看门狗都支持窗口功能。在计数开始前可预设窗口区间，计数值位于窗口区间时，可刷新计数器，计数重新开始。

表 28-1 看门狗计数器的基本特性

计数时钟	● SWDT: SWDTLRC 的 1/16/32/64/128/256/2048 分频 ● WDT: PCLK3 的 4/64/128/256/512/1024/2048/8192 分频
最长溢出时间	● SWDT: 3.72 小时 (SWDTLRC=10kHz) ● WDT: 10.7 秒 (PCLK3=50MHz)
计数模式	递减计数
窗口功能	可设定窗口区间，定义刷新动作的允许区间
启动方式	1. 硬件启动 2. 软件启动
停止条件	1. 复位中 2. 下溢，刷新错误发生时再开始： ● 硬件启动模式下，复位或中断请求输出后自动开始 ● 软件启动模式下，再次设定刷新寄存器
中断/复位条件	1. 计数下溢 2. 刷新错误

28.3 功能说明

28.3.1 启动看门狗

看门狗计数器的启动方式有两种：硬件启动方式和软件启动方式。

硬件启动方式是指启动时从主闪存区域读取看门狗计数器的设定信息 (ICG0 寄存器)，计数器自动开始计数；软件启动方式是指设定控制寄存器后，写刷新寄存器完成刷新动作，计数器开始计数。

28.3.2 硬件启动方式

ICG0 寄存器的位 16 (WDTAUS)、位 0 (SWDTAUS) 为 0 时，为硬件启动方式。选择硬件启动方式时，WDT_CR 和 SWDT_CR 寄存器的相关设定信息无效。

硬件启动方式时，在复位期间将 ICG0 寄存器里的 WDT/SWDT 相关设定（计数时钟、窗口设定值、计数周期等）载入到 WDT/SWDT 的模块中，复位之后、计数器按照设定自动开始计数。下图为硬件启动方式的动作例。

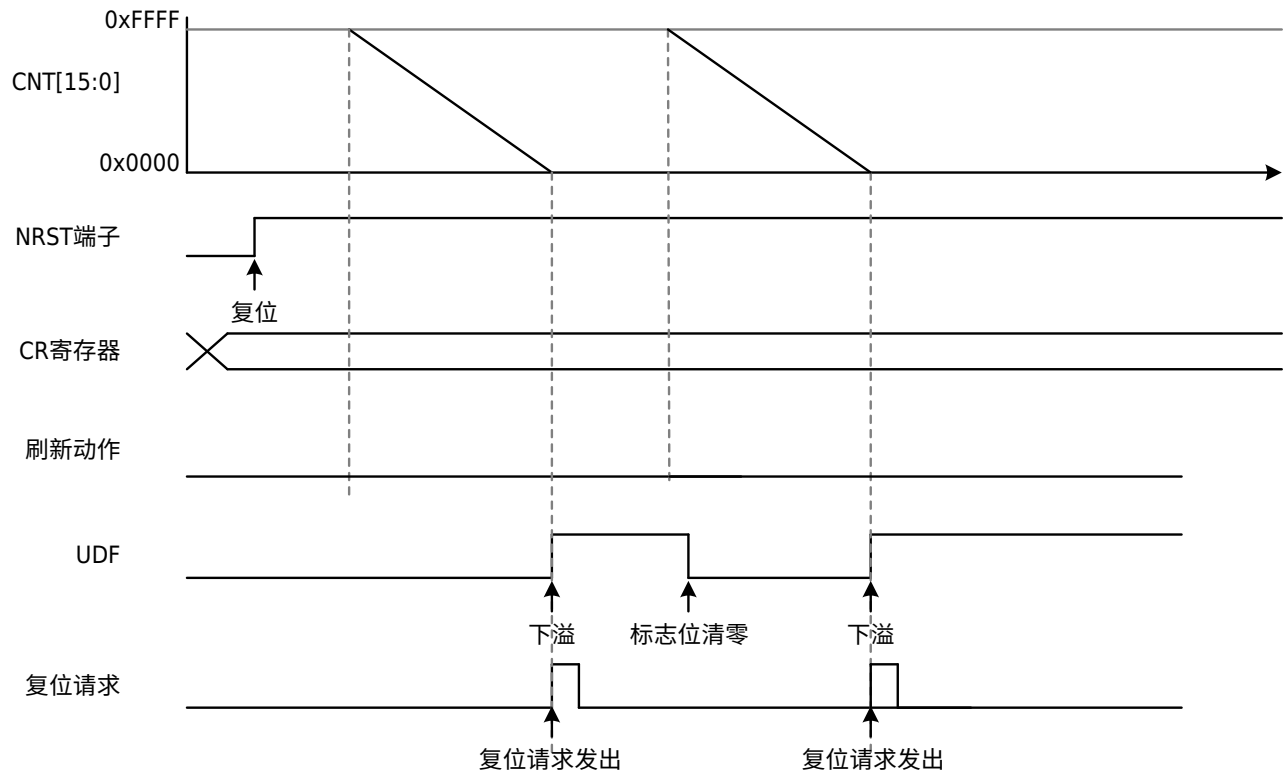


图 28-1 硬件启动例

28.3.3 软件启动方式

ICG0 寄存器的位 16 (WDTAUTS)，位 0 (SWDTAUTS) 为 1 时，通过设定刷新寄存器的方式启动 WDT/SWDT 为软件启动方式。复位后，设定 WDT_CR/SWDT_CR 寄存器中的计数时钟、窗口设定值、计数周期等，然后执行刷新动作，计数器就开始计数。WDT_CR/SWDT_CR 设定只能 1 次，再次设定写入值无效。下图为软件启动方式的动作例。

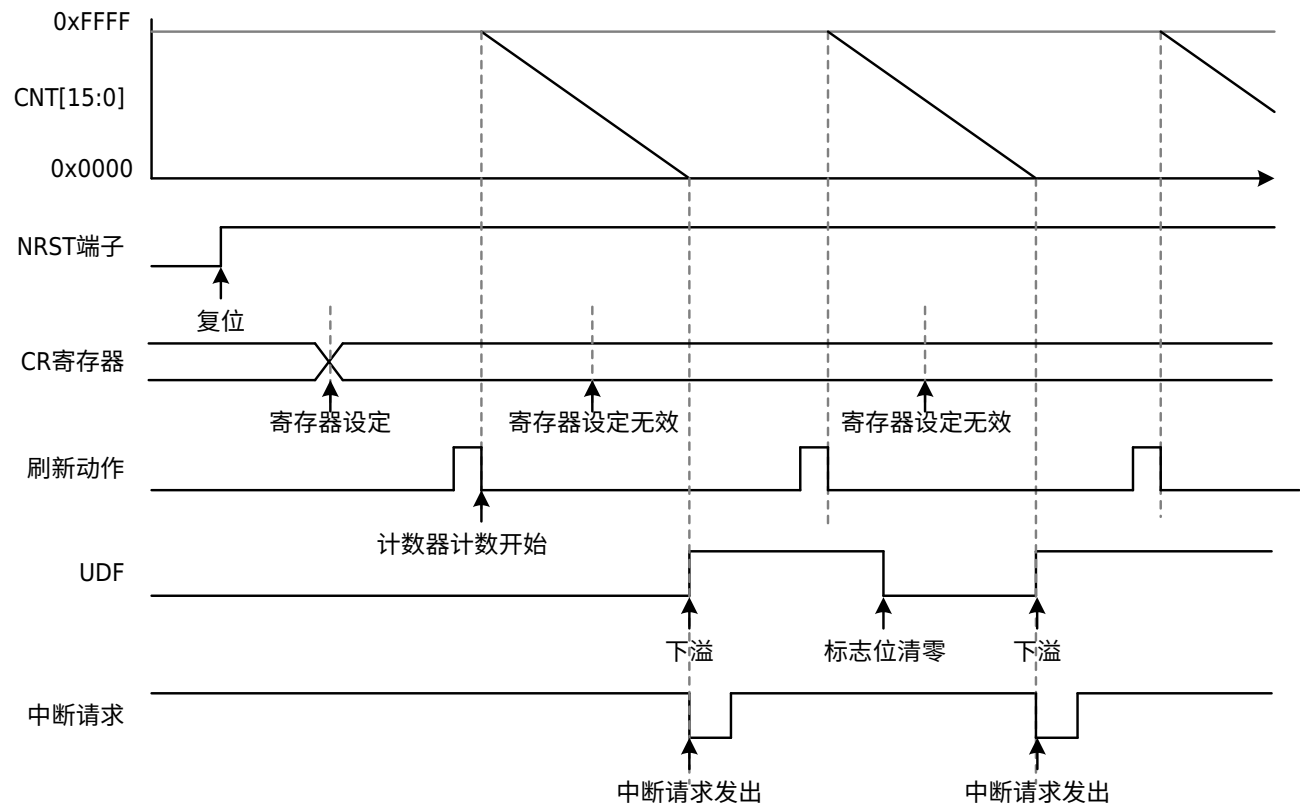


图 28-2 软件启动例

28.3.4 刷新动作

WDT/SWDT_RR 寄存器中先写 0x0123、再写 0x3210 完成一次刷新动作，WDT/SWDT 的计数器就开始计数（软件启动）或重新开始计数。

WDT/SWDT_RR 寄存器在写 0x0123、0x3210 之间，若发生对其他寄存器访问或读取 WDT/SWDT_RR 寄存器等，不影响正常的刷新动作。

下图为动作示例。

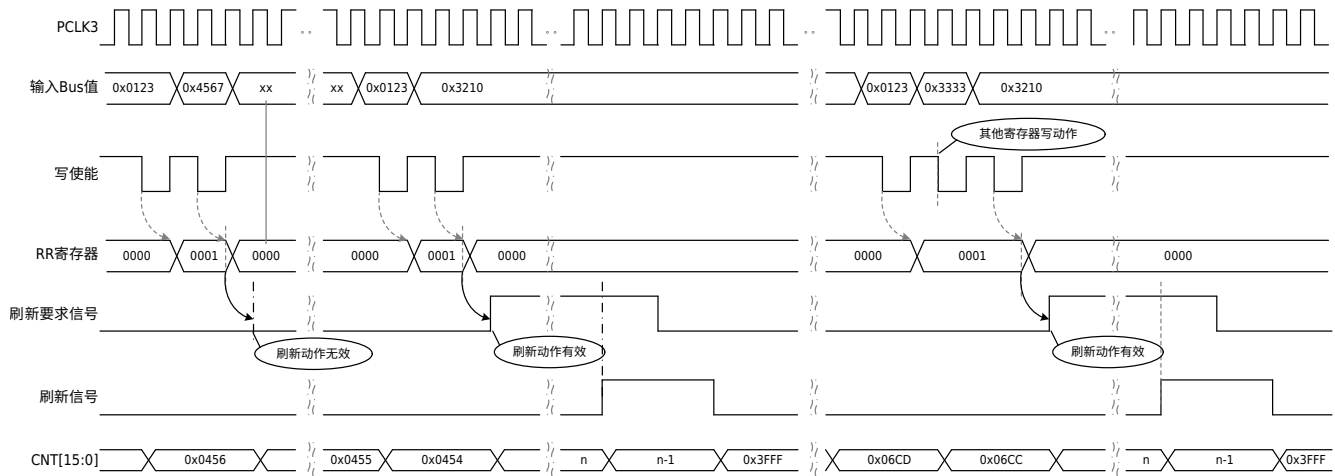


图 28-3 各种刷新动作时序示例（动作确认，刷新要求信号的下降沿等）

刷新动作需要 4 个计数周期完成计数值的更新，所以请在刷新下位窗口和下溢位置的提前 4 个计数值完成刷新寄存器的写入。计数值的确认请读取状态寄存器。

28.3.5 标志位

刷新错误标志位和计数下溢标志位在中断请求的情况下会保持。当进入中断后，可以通过查询标志位来确认中断原因。标志位清零：先读“1”再写“0”。



说明

在发生刷新错误或者下溢错误的一定时间内，对标志位清零操作无效，这段时间为：1 个计数周期+2 个 SWDTLRC（SWDT 模块）；1 个计数周期+2 个 PCLK3（WDT 模块）。

刷新错误或者计数下溢标志位置位时，硬件启动模式看门狗计数不停止；软件启动模式看门狗计数停止。对标志位写“0”时，SWDT 最多需要经过 3 个 SWDTLRC 和 2 个 PCLK3 时间后，寄存器位才能被清零；WDT 最多需要经过 5 个 PCLK3 时间后，寄存器位才能被清零。用户对标志位清零后，需再次读取标志位查询清零操作是否成功。

标志位清零步骤：

- 步骤 1.** 读取 SWDT_SR/WDT_SR，查询标志位；
- 步骤 2.** 对置 1 的标志位写入 0；
- 步骤 3.** 读取 SWDT_SR/WDT_SR，查询标志位是否清零成功，若标志位未清零回到步骤 2，直至清零成功。

28.3.6 中断复位

WDT/SWDT 在计数器计数下溢或刷新错误时，可以选择产生中断请求或复位请求。硬件启动模式下，通过 ICG0 的 WDTITS/SWDTITS 位，来决定产生中断请求还是复位请求。软件启动模式下，通过设定 WDT_CR/SWDT_CR 寄存器 ITS 位，来决定产生中断请求还是复位请求。

WDT/SWDT 的中断复位产生条件有两种。一种是计数器计数产生下溢；一种是在刷新允许区间之外执行刷新动作，产生刷新错误。

28.3.7 计数下溢

如下图所示动作例，在递减计数到零时产生下溢。

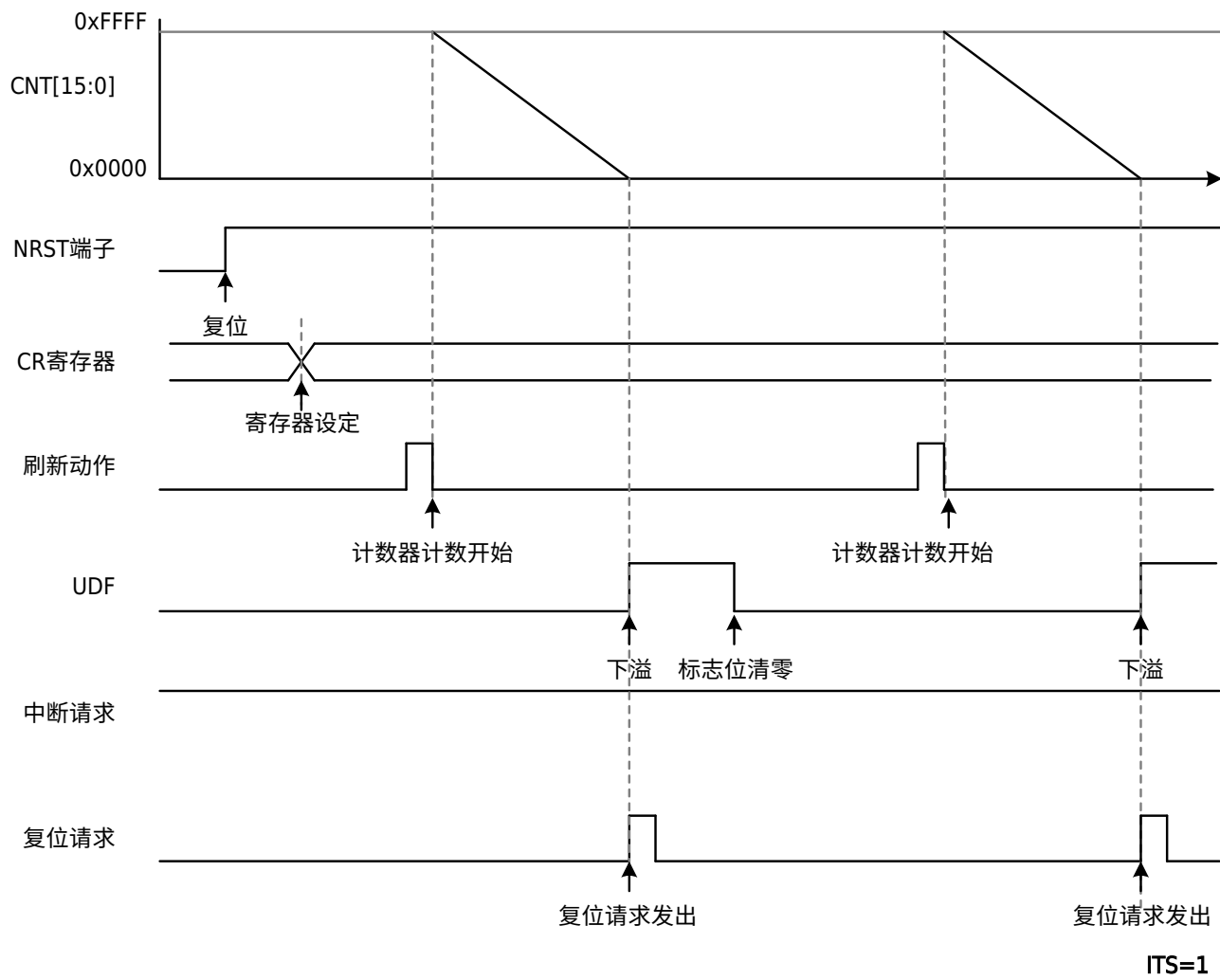


图 28-4 计数器下溢动作例

28.3.8 刷新错误

在设定了窗口区间后，只有在窗口区间内执行刷新动作时计数器才会被刷新、重新开始计数，在窗口区间外执行刷新动作时产生刷新错误。下图为刷新动作例。

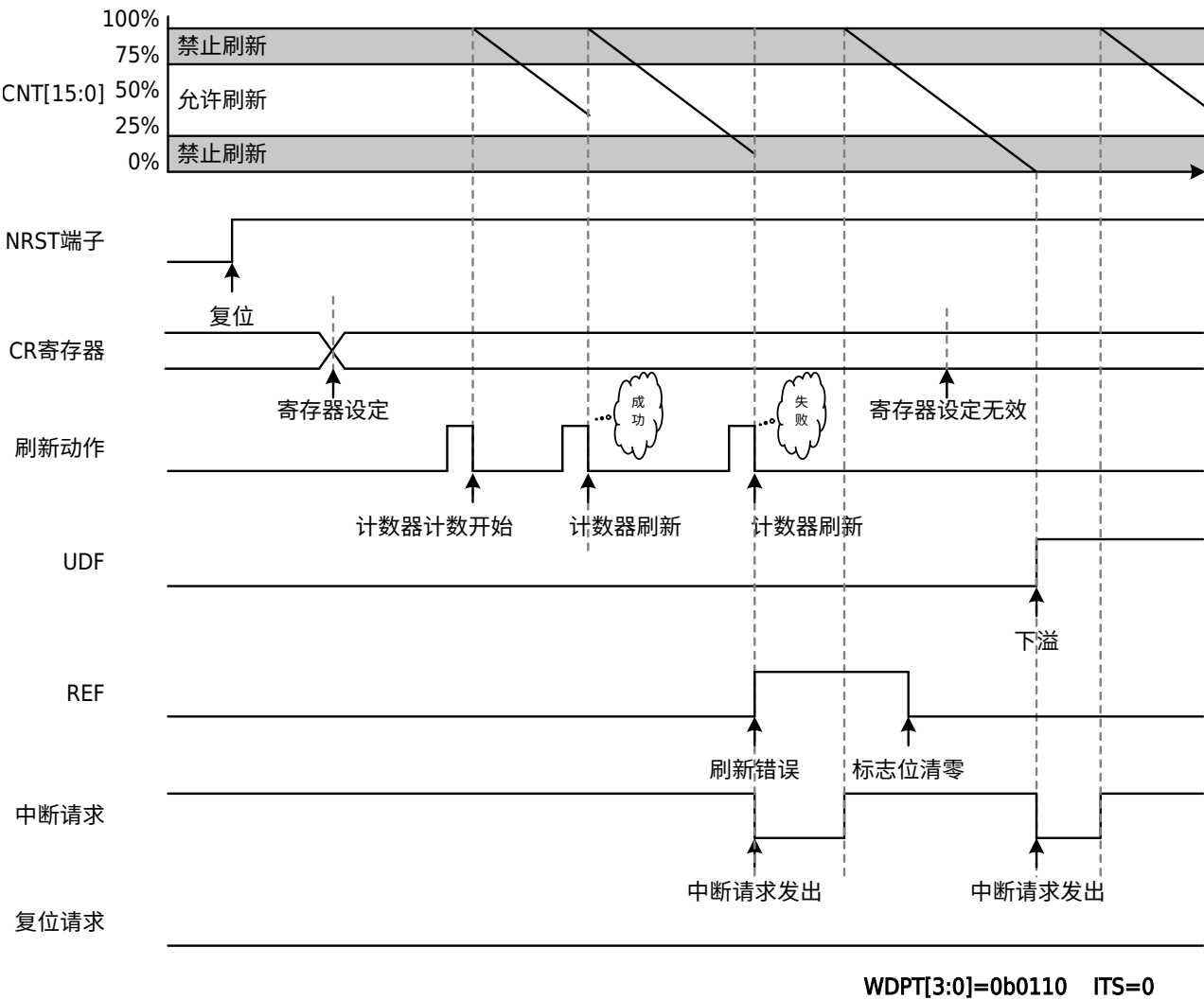


图 28-5 计数器刷新动作例

28.4 寄存器

28.4.1 寄存器总表

WDT 基地址：0x40049000

表 28-2 看门狗计数器 (WDT) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	WDT_CR	WDT 控制寄存器	32
0x04	WDT_SR	WDT 状态寄存器	32
0x08	WDT_RR	WDT 刷新寄存器	32

SWDT 基地址：0x40049400

表 28-3 看门狗计数器 (SWDT) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	SWDT_CR	SWDT 控制寄存器	32
0x04	SWDT_SR	SWDT 状态寄存器	32
0x08	SWDT_RR	SWDT 刷新寄存器	32

28.4.2 控制寄存器 (SWDT_CR、WDT_CR)

Offset	Bit Position																																
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x80010FF3																																
Name	ITS	Reserved														SLPOFF	Reserved						WDPT				CKS				Reserved		PERI
Access		RW														RW	RW						RW				RW				RW		RW

位/位域	标记	位名	功能描述	读写
31	ITS	刷新错误/溢出中断/复位选择	0b0: 中断请求 0b1: 复位请求	RW
30:17	Reserved	保留	-	-
16	SLPOFF	WDT/SWDT 在低功耗模式下计数禁止	WDT_CR: 0b0: WDT 在 Sleep 模式下计数许可 0b1: WDT 在 Sleep 模式下计数禁止 SWDT_CR: 0b0: SWDT 在 Sleep/Stop 模式下计数许可 0b1: SWDT 在 Sleep/Stop 模式下计数禁止	RW
15:12	Reserved	保留	-	-
11:8	WDPT	刷新允许区域计数值百分比	0b0000: 0%~100% 0b0001: 0%~25% 0b0010: 25%~50% 0b0011: 0%~50% 0b0100: 50%~75% 0b0101: 0%~25%, 50%~75% 0b0110: 25%~75% 0b0111: 0%~75% 0b1000: 75%~100% 0b1001: 0%~25%, 75%~100% 0b1010: 25%~50%, 75%~100% 0b1011: 0%~50%, 75%~100% 0b1100: 50%~100% 0b1101: 0%~25%, 50%~100% 0b1110: 25%~100% 0b1111: 0%~100%	RW
7:4	CKS	计数时钟	WDT_CR: 0b0010: PCLK3/4 0b0110: PCLK3/64 0b0111: PCLK3/128 0b1000: PCLK3/256 0b1001: PCLK3/512 0b1010: PCLK3/1024 0b1011: PCLK3/2048 0b1101: PCLK3/8192	RW

位/位域	标记	位名	功能描述	读写
7:4	CKS	计数时钟	- 其他：禁止设置 SWDT_CR: 0b0000: SWDTCLK 0b0100: SWDTCLK/16 0b0101: SWDTCLK/32 0b0110: SWDTCLK/64 0b0111: SWDTCLK/128 0b1000: SWDTCLK/256 0b1011: SWDTCLK/2048 - 其他：禁止设置	RW
3:2	Reserved	保留	-	-
1:0	PERI	计数周期	0b00: 256 cycle 0b01: 4096 cycle 0b10: 16384 cycle 0b11: 65536 cycle	RW

28.4.3 状态寄存器 (SWDT_SR、WDT_SR)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Name	Reserved																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Access															RW	REF		RW	UDF																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	REF	刷新错误标志	0b0: 没有刷新错误 0b1: 发生刷新错误 对该位读出 1 后写入 0, 该位清零。	RW
16	UDF	计数下溢标志	0b0: 没有计数下溢 0b1: 发生计数下溢 对该位读出 1 后写入 0, 该位清零。	RW
15:0	CNT	计数值	计数器当前计数值	RO

28.4.4 刷新寄存器 (SWDT_RR、WDT_RR)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RF															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RF	刷新值	依次写入 0x0123、0x3210 之后，完成刷新动作。 写入 0x0123 后，该寄存器读出于 0x00000001；其 余情况读出于 0x00000000。	RW

28.5 注意事项

SWDT 动作时，周边时钟 PCLK3 的动作频率必须大于或等于计数时钟频率的 4 倍，即 PCLK3 频率≥计数时钟频率×4。

29 通用同步异步收发器 (USART)

29.1 简介

本产品搭载通用同步异步收发器 (USART) 模块 10 个单元，能够灵活地与外部设备进行全双工数据交换；本产品搭载的 USART 支持通用异步串行通信接口 (UART)，时钟同步通信接口，智能卡接口 (ISO/IEC7816-3) 和 LIN 通信接口。支持调制解调器操作 (CTS/RTS 操作)，多处理器操作，DE 功能。与 Timer0 模块配合支持 UART 接收超时功能。USART1 支持通过 RX 线唤醒 STOP 模式功能。

29.2 主要特性

功能特性	模块支持情况									
	USART1	USART2	USART3	USART4	USART5	USART6	USART7	USART8	USART9	USART10
UART	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
多处理器通信	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
时钟同步通信	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RX 线唤醒 STOP 模式功能	✓	-	-	-	-	-	-	-	-	-
小数波特率	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LIN	-	-	-	-	✓	-	-	-	-	✓
智能卡	✓	✓	✓	✓	-	✓	✓	✓	✓	-
UART 接收超时功能	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RS485 Driver Enable	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

USART 主要特性

- 支持全双工异步通信，全双工时钟同步通信
- 支持 LIN 总线
- 支持智能卡接口 (ISO/IEC7816-3)
- 发送器和接收器具有独立使能位
- 内置双缓冲器
- LSB/MSB 可选
- 支持调制解调器操作 (CTS/RTS)
- 传输标志：发送数据寄存器空、发送数据完成、接收数据寄存器满、接收错误标志、UART 接收超时标志、LIN 唤醒信号检出标志、LIN 间隔段检出标志、LIN 总线错误标志
- 支持 RS485 Driver Enable 功能

UART 主要特性

- 数据长度可编程：8 位/9 位
- 校验功能可配置：奇校验/偶校验/无校验
- 停止位可配置：1 位/2 位
- 时钟源可选：内部时钟源 (内部波特率生成器生成的时钟) /外部时钟源 (USARTn_CK 管脚输入的时钟)
- 接收错误：校验错误、帧错误、上溢错误
- 支持多个处理器间通信
- 内置数字滤波器
- 支持接收数据 TIMEOUT 功能
- 单元 1 支持停止模式唤醒功能

- 支持全双工/半双工通信方式

时钟同步模式主要特性

- 数据长度：8 位
- 接收错误：上溢错误
- 时钟源：内部时钟源（内部波特率生成器生成的时钟）/外部时钟源（USARTn_CK 管脚输入的时钟）
- 支持全双工通信方式

智能卡接口的主要特性

- 数据长度：8 位
- 检测到校验错误时能自动送出错误信号
- 支持数据重发

LIN 的主要特性

- 数据长度：8 位
- 支持唤醒信号的检测
- 支持 10/11 位同步间隔段（BF：Break Field）的检测
- 支持同步段测量，寄存器记录测量值
- 支持 10/11/13/14 位的间隔段（BF）发送
- 支持总线冲突检测
- 支持回环模式

29.3 功能说明

29.3.1 系统框图

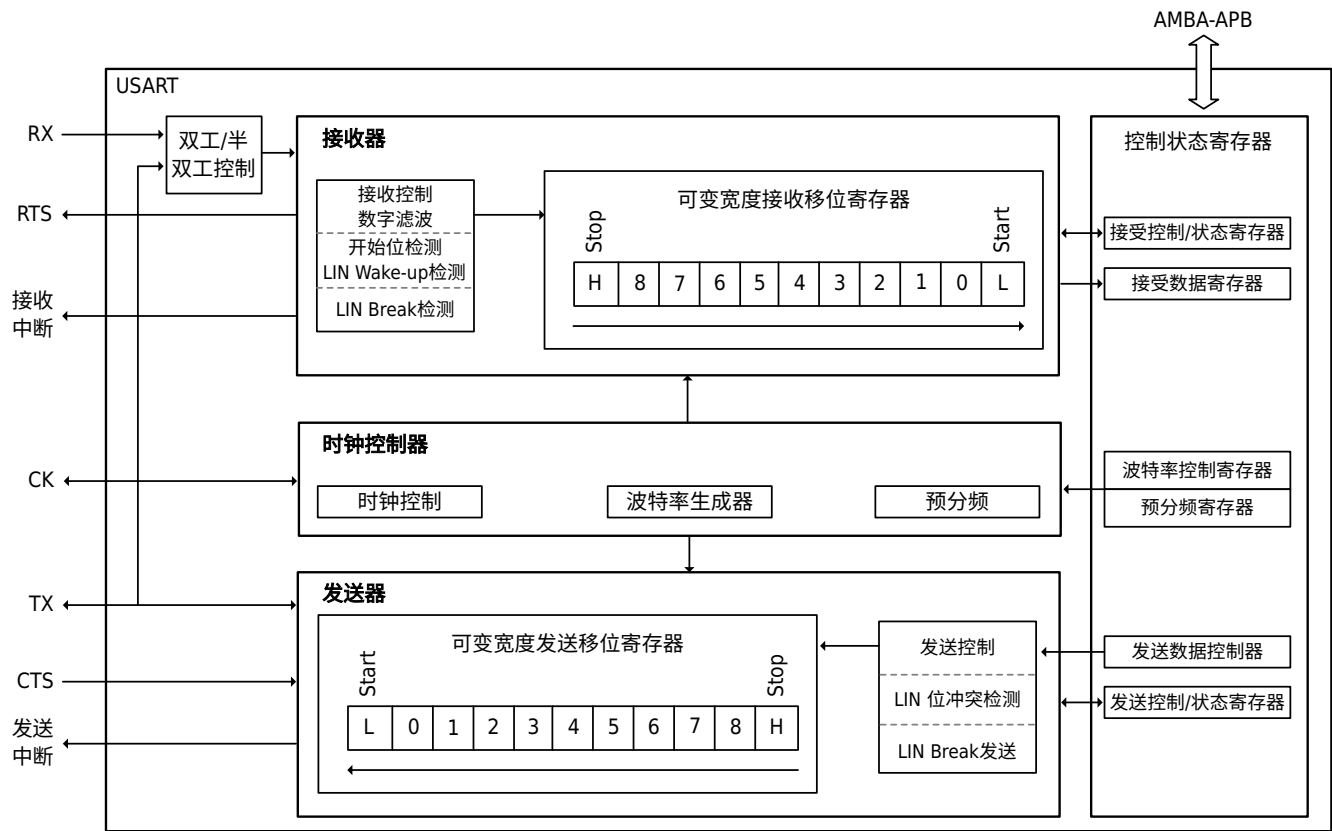


图 29-1 USART 系统框图

表 29-2 USART 管脚说明

管脚名	方向	功能描述
USARTn_CK	输入/输出	时钟
USARTn_TX	输入/输出	发送数据管脚 半双工时也作为接收数据管脚
USARTn_RX	输入	接收数据管脚
USARTn_CTS	输入	调制解调器操作管脚 清除发送管脚
USARTn_RTS/DE	输出	调制解调器操作管脚 请求发送管脚/Driver Enable 管脚

 **说明**
n 取值为 1~10。

29.3.2 UART 功能

29.3.2.1 时钟

UART 可以选择内部波特率生成器生成的时钟（内部时钟源）或 USARTn_CK 管脚输入的时钟（外部时钟源）作为通信的时钟源。

内部时钟源

USARTn_CR2.CLKC[1:0]位设定为 0b00 或者 0b01 时选择时钟源为内部时钟源即内部波特率生成器生成的时钟。

USARTn_CR2.CLKC[1:0]=0b00 时 USARTn_CK 管脚不作为时钟管脚使用，可以作为普通 IO 使用。

USARTn_CR2.CLKC[1:0]=0b01 时从 USARTn_CK 管脚输出与通信波特率相同频率的时钟。

当 USARTn_PR.ULBREN 为 0 时，内部波特率生成器的时钟源由 USARTn_PR.PSC[1:0]位的设定选择为 PCLK1，PCLK1/4，PCLK1/16，PCLK1/64，当 USARTn_PR.ULBREN 为 1 时，超低波特率功能使能，内部波特率生成器的时钟源由 USARTn_PR.PSC[1:0]位的设定选择为 PCLK1/128，PCLK1/256，PCLK1/512，PCLK1/1024。

外部时钟源

USARTn_CR2.CLKC[1:0]位设定为 0b10 或者 0b11 时选择时钟源为从 USARTn_CK 管脚输入的外部时钟，输入时钟的频率为波特率的 16 倍（USARTn_CR1.OVER8=0）或者 8 倍（USARTn_CR1.OVER8=1）。

最高波特率

内部时钟源时，内部波特率生成器生成的波特率计算公式为：

$$B = \frac{C}{8 \times (2 - \text{OVER8}) \times (\text{DIV\_Integer} + 1)}$$

其中：

- B：波特率，单位：Mbps
- C：当 USARTn_PR.ULBREN 为 0 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1，PCLK1/4，PCLK1/16，PCLK1/64），单位：MHz
当 USARTn_PR.ULBREN 为 1 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1/128，PCLK1/256，PCLK1/512，PCLK1/1024），单位：MHz
- OVER8：USARTn_CR1.OVER8 设定值
- DIV_Integer：USARTn_BRR.DIV_Integer 设定值

最高波特率为 PCLK1/8（Mbps）。

外部时钟源时，外部输入 UART 时钟的最高频率要求为 PCLK1/4（MHz），所以时钟源为外部输入时钟时：当 USARTn_CR1.OVER8=0 时，最高波特率为 PCLK1/64（Mbps）；当 USARTn_CR1.OVER8=1 时，最高波特率为 PCLK1/32（Mbps）。

需要注意的是，UART 最高通信波特除了以上描述的基于 PCLK1 的计算方法外，还需要参考《数据手册》“电气特性”章节规定的最高通信波特率。

29.3.2.2 数据格式

UART 模式时一帧数据是由开始位，数据位，校验位和停止位组成。

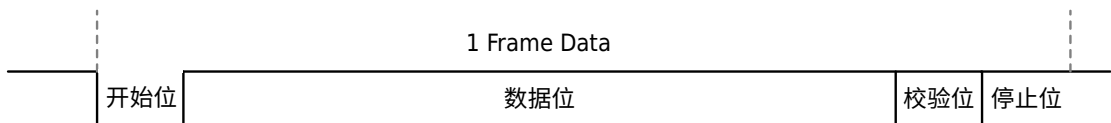


图 29-2 UART 数据格式

- 开始位 (ST)：开始位固定由一位的低电平构成。
- 数据位 (D)：数据位可以配置成 8 位或者 9 位。

- **校验位 (P)**：校验位可以配置成 1 位偶校验或 1 位奇校验或无校验位。
- **停止位 (SP)**：停止位固定为高电平，可以配置成 1 位或者 2 位。

29.3.2.3 调制解调器操作

调制解调器操作包括 CTS 功能和 RTS 功能。USARTn_CR3.RTSE=1 时 RTS 功能有效，USARTn_CR3.CTSE=1 时 CTS 功能有效。

CTS 功能

CTS 功能是通过 USARTn_CTS 管脚的输入来控制数据的发送，只有当 USARTn_CTS 管脚输入低电平时才可以发送数据，发送数据过程中如果 USARTn_CTS 输入高电平，正在发送的数据不受影响。

RTS 功能

RTS 功能是指通过 USARTn_RTS 管脚输出低电平，请求对方发送数据。

USARTn_RTS 管脚输出低电平需要满足以下全部条件：

- 接收使能(USARTn_CR1.RE=1)，且不在接收数据
- USARTn_RDR.RDR 寄存器中没有未读取的接收数据
- 无任何接收错误，包括帧错误、校验错误和上溢错误

29.3.2.4 发送器

发送器可发送 8 位或 9 位的数据，具体取决于 USARTn_CR1.M 位的设定值。

发送器使能位 (USARTn_CR1.TE) 置 1，写入发送数据后，发送数据在 TX 管脚上串行输出；相应的时钟脉冲可以选择在 USARTn_CK 管脚输出或者不输出。

发送数据的顺序为：开始位->数据位 (MSB/LSB) ->校验位 (有或者无) ->停止位。

发送数据寄存器 TDR 和内部的发送移位寄存器组成双缓冲器结构，可以连续发送数据。

通过发送数据寄存器空中断或者 DMA 写入发送数据时，为保证发送的正确性，一次请求只能写入一次数据。

发送数据设定步骤

- 步骤 1.** 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2.** 设定 UART 所需要使用的管脚；
- 步骤 3.** 通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 步骤 4.** 设定 USARTn_CR1，USARTn_CR2，USARTn_CR3 寄存器；
- 步骤 5.** 设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6.** 使能发送器 (USARTn_CR1.TE=1)，如果需要使用发送数据寄存器空中断，则设置 USARTn_CR1.TXEIE=1；
- 步骤 7.** 等待发送数据寄存器空，写通信数据到 USARTn_TDR.TDR，数据传输到发送移位寄存器，发送开始；



说明

CTS 功能有效时，USARTn_CTS 输入为低电平时数据传输到发送移位寄存器，发送开始。

- 步骤 8.** 如果需要连续发送数据时，重复步骤 7；
- 步骤 9.** 通过确认 USARTn_SR.TC 位确认发送是否完成。连续发送数据并使用发送中断的情况，可通过 TI 中断写入最后一个发送数据，并将 USARTn_CR1.TXEIE 写 0，USARTn_CR1.TCIE 写 1，最后一帧数据发送结束后，产生发送完成中断。

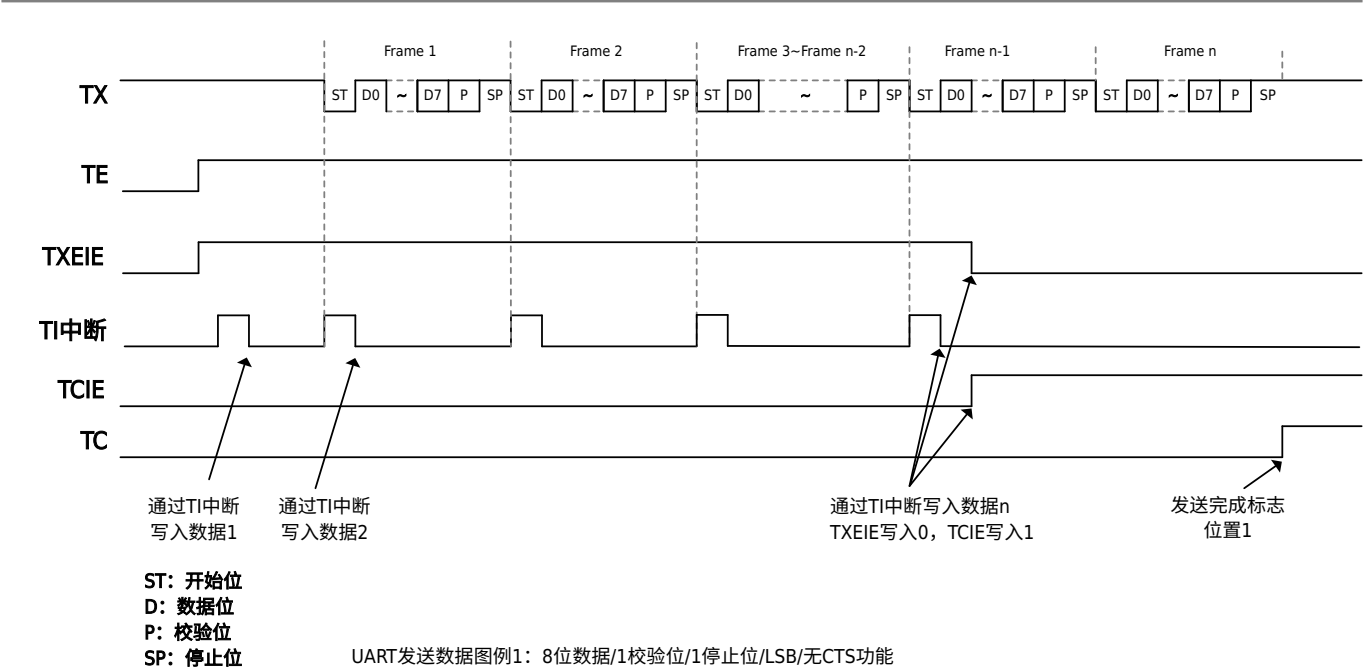


图 29-3 USART 发送数据图例 1

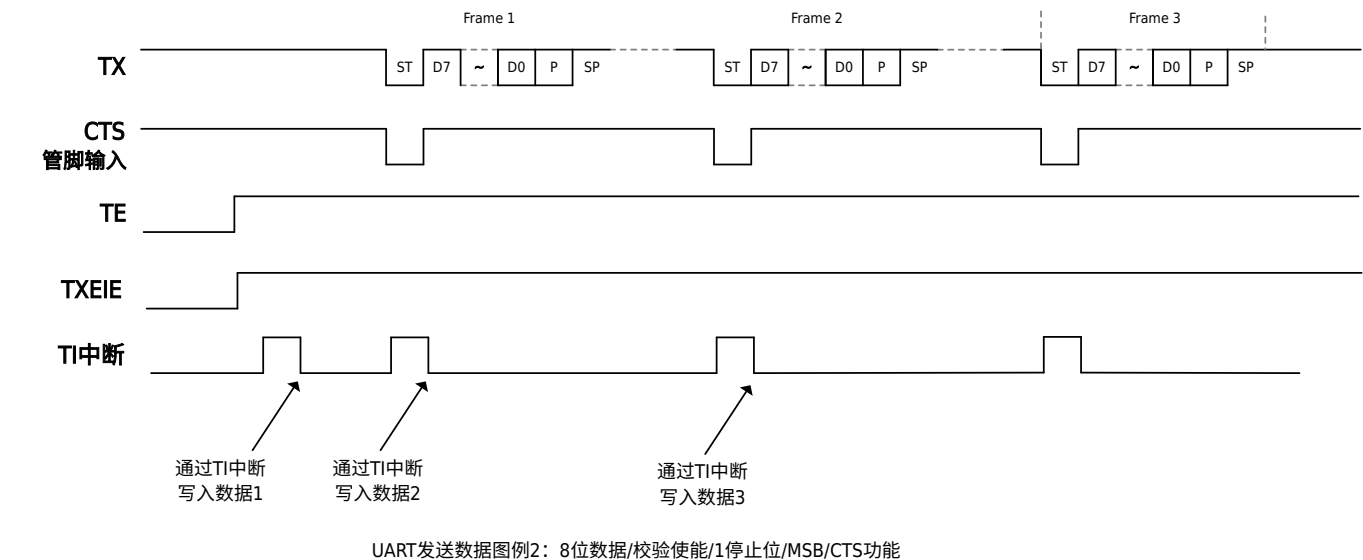


图 29-4 USART 发送数据图例 2

发送器中断

UART 模式发送器支持两种中断，发送数据寄存器空中断 TI 和发送完成中断 TCI。
TXEIE=1，USARTn_TDR.TDR 寄存器的值传送到发送移位寄存器时 TI 中断发生。
TCIE=1，发送数据的最后一位时 USARTn_TDR.TDR 寄存器没有更新则 TCI 中断发生。

29.3.2.5 接收器

接收器可接收 8 位或 9 位的数据，具体取决于 USARTn_CR1.M 位的设定值。接收器使能位 (USARTn_CR.RE) 置 1 并检测到开始位后，RX 管脚上数据接收到接收移位寄存器，收满一帧数据，数据从接收移位寄存器传送到接收数据寄存器 USARTn_RDR.RDR。
接收数据的顺序为：开始位->数据位 (MSB/LSB) ->校验位 (有或者无) ->停止位。
接收数据寄存器 USARTn_RDR.RDR 寄存器和内部的接收移位寄存器组成双缓冲器结构，可以连续接收数据。
通过接收数据寄存器满中断或者 DMA 读取接收数据时，一次请求只能读取一次数据。

开始位检测

开始位检测可以选择低电平方式或下降沿方式，具体取决于 USARTn_CR1.SBS 位，USARTn_CR1.SBS=0 时为低电平检测，USARTn_CR1.SBS=1 时为下降沿检测。

采样和接收容差

检测到开始条件（低电平或下降沿）后，USART 会基于内部基本时钟对接收数据进行时钟同步，从而开始数据接收。
数据的采样在数据中央，USARTn_CR1.OVER8=0 时在第 8 个内部基本时钟采样，USARTn_MR.OVER8=1 时在第 4 个内部基本时钟采样。

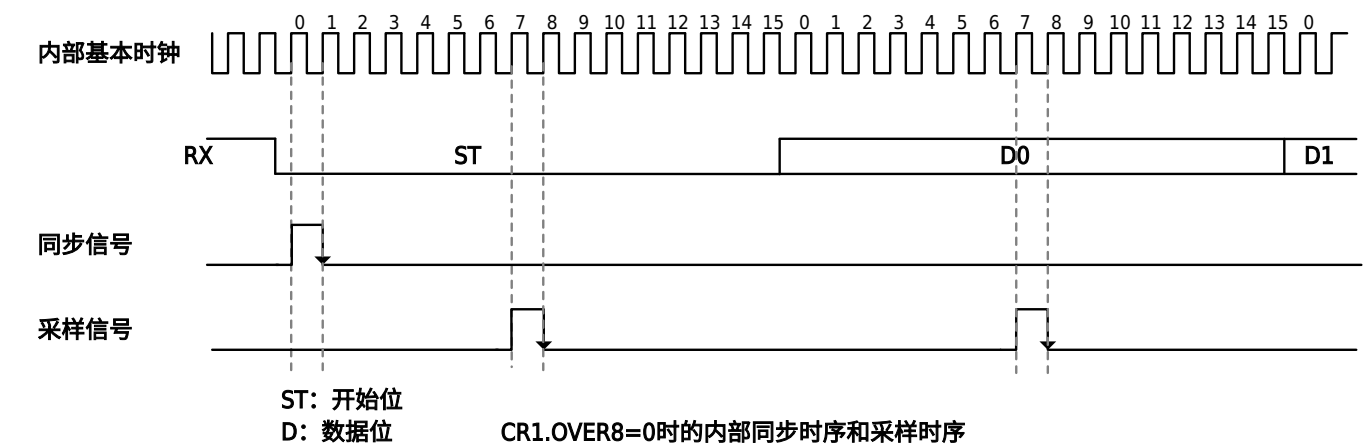


图 29-5 USART 内部同步和采样时序

仅当总时钟系统偏差小于 USART 接收器的容差时，USART 异步接收器才能正常工作。影响总偏差的因素包括：

- 发送器误差引起的偏差（其中还包括发送器本地振荡器的偏差）
 - 接收器的波特率量化引起的误差
 - 接收器本地振荡器的偏差
 - 传输线路引起的偏差
- 对于正确接收数据，USART 异步接收器所容许的最大偏差值具体取决于以下选项：
- 数据长度 FL (Frame Length) 。FL 由 USART_CR1 寄存器中 M 位定义的 8 或 9 数据位和 PCE 位定义的校验使能位决定
 - 由 USART_CR1 寄存器中 OVER8 位定义的 8 倍或 16 倍过采样
 - 由 USART_CR1 寄存器中 FBME 位定义的是否使用小数波特率

表 29-3 USART 接收器的容差

FL	DIV_Fraction 为 0		DIV_Fraction 不为 0	
	OVER8 位=0	OVER8 位=1	OVER8 位=0	OVER8 位=1
10	4.375%	3.75%	3.88%	3%
11	3.97%	3.41%	3.53%	2.73%
12	3.646%	3.125%	3.23%	2.5%

 **说明**
在特殊情况下，当 RX 管脚接收到数据长度 FL 时间的连续高电平时，上表中指定的数据可能会略微不同。

接收数据设定步骤

- 步骤 1. 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2. 设定 UART 所需要使用的管脚；
- 步骤 3. 通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 步骤 4. 设定 USARTn_CR1, USARTn_CR2, USARTn_CR3 寄存器；
- 步骤 5. 设定 USARTn_PR 选择预分频值, USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6. 使能接收器（USARTn_CR1.RE=1），如果需要使用接收中断，则设置 USARTn_CR1.RIE=1；
- 步骤 7. 当检测到开始位后，接收器将数据接收到接收移位寄存器，并检查校验位和停止位。
 - 校验错误时，接收到的数据传送到 USARTn_RDR.RDR 寄存器中并置位 USARTn_SR.PE 标志
 - 停止位不为高电平时，发生帧错误，接收到的数据传送到 USARTn_RDR.RDR 寄存器中并置位 USARTn_SR.FE 标志
 - 发生上溢错误时，数据丢失并置位 USARTn_SR.ORE 标志
 - 无错误发生时，接收到的数据传送到 USARTn_RDR.RDR 寄存器中，并置位 USARTn_SR.RXNE 标志，读取接收到的数据后重复步骤 7 可连续接收数据

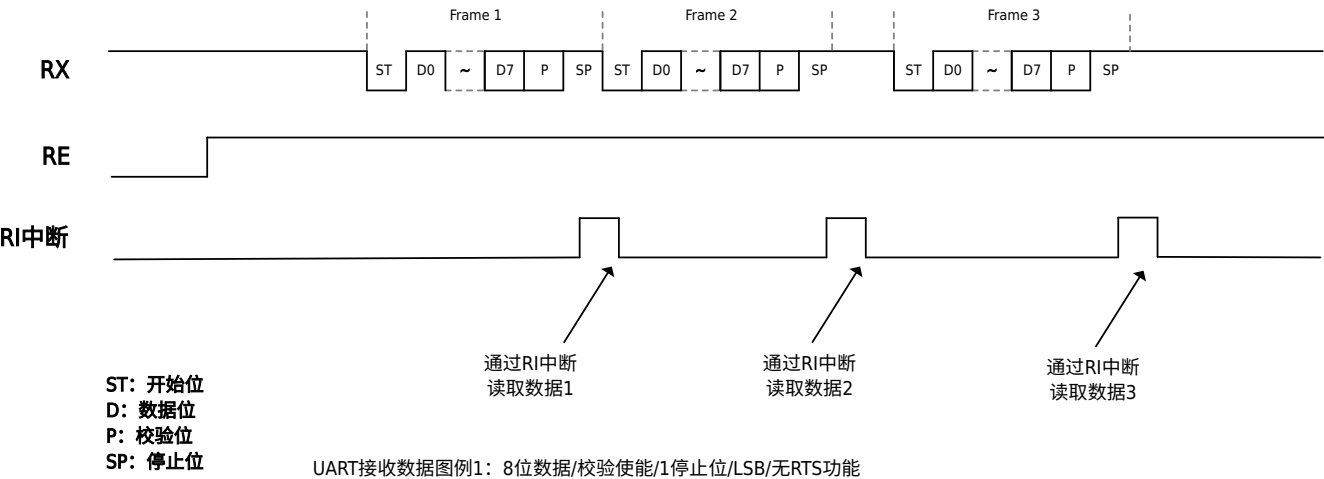


图 29-6 UART 接收数据图例 1

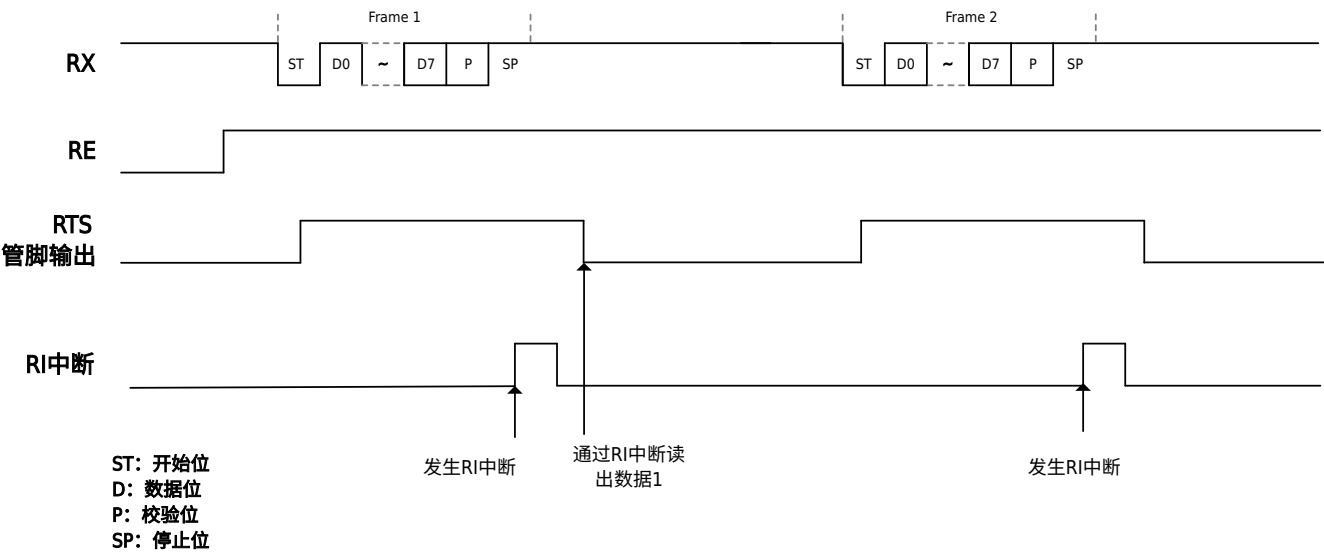


图 29-7 UART 接收数据图例 2

错误处理

接收数据时有三种类型的接收错误，分别为上溢错误 (USARTn_SR.ORE)，校验错误 (USARTn_SR.PE) 和帧错误 (USARTn_SR.FE)。发生任何一种接收错误都不能再进行数据的接收。可以通过将所有的错误标志清零来重启数据接收，清零方法是写对应清零寄存器。

上溢错误发生的条件是 USARTn_RDR.RDR 寄存器值未被读取的情况下又接收到一帧新的数据，所以应该在收到当前帧最后一位之前将接收到的前一帧数据读取。

校验错误发生的条件是发生了奇偶校验错误。

帧错误发生的条件是停止位为低电平，2 个停止位的情况也只检查第一个停止位。

发生上溢错误时接收到的数据丢失，RI 中断不发生。

发生校验错误时接收到的数据传送给 USARTn_RDR.RDR，RI 中断不发生。

发生帧错误时接收到的数据传送给 USARTn_RDR.RDR，RI 中断不发生。

接收器中断

UART 模式接收器支持两种中断，接收数据寄存器满中断 RI 和接收错误中断 EI。

USARTn_CR.RIE=1，未发生任何接收错误，数据从接收移位寄存器传送到接收数据寄存器时 RI 中断发生。

USARTn_CR.RIE=1，接收过程中发生上溢错误，校验错误或者帧错误时 EI 中断发生。

29.3.2.6 UART 接收 TIMEOUT 功能

UART 接收数据停止位被检测时 TIMEOUT 计数器启动，经过设定的 TIMEOUT 时间（设定单位为接收位数）后未检测到下一帧接收数据时，发生 TIMEOUT，如果此时 CR1.RE=1，则 TIMEOUT 状态位

USARTn_SR.RTOF 置位，如果此时 USARTn_CR1.RE=0，则等待 USARTn_CR1.RE=1 后 TIMEOUT 状态位 USARTn_SR.RTOF 置位。

TIMEOUT 计数器采用 Timer0 模块的计数器，Timer0 计数方式需设置为异步计数方式，异步计数时钟源设置为 XTAL32（Timer0 计数时钟由 UART 波特率生成器产生的时钟 tick 提供，可以不存在 XTAL32 晶振），具体对应关系如下：

USART1	USART2	USART3	USART4	USART5	USART6	USART7	USART8	USART9	USART10
Timer0_1 A 通道	Timer0_1 B 通道	Timer0_3 A 通道	Timer0_3 B 通道	Timer0_4 A 通道	Timer0_2 A 通道	Timer0_2 B 通道	Timer0_4 B 通道	Timer0_5 A 通道	Timer0_5 B 通道

TIMEOUT 功能 Timer0 比较计数器值设定

Timer0 为 16 位计数器，计数时钟最大可以选择 1024 分频，TMR0_CMPAR 值设定计算公式如下：

$$CMPAR = \frac{RTB}{2^{CKDIRA}} - \alpha$$

其中：

- CMPAR：TMR0_CMPAR 寄存器值，计算结果请向上进位取整。
- α：Timer0 异步计数同步电路带来的延迟
计数时钟不分频时，α=7；
计数时钟 2 分频时，α=5；
计数时钟 4、8、16 分频时，α=3；
计数时钟 32 分频及以上，α=2。
- RTB：Receive Timeout Bits，最小值=接收数据帧长度+α×2^{CKDIRA}
实际发生 TIMEOUT 的时间与 RTB 值之间有一定误差，误差≤2^{CKDIRA}。
- CKDIRA：TMR0_BCONR.CKDIRA位寄存器值

TIMEOUT 功能设定步骤

- 步骤 1. 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2. 设定 UART 所需要使用的管脚；

- 步骤 3. 通过 USARTn_CR2.CLKC[1:0]位选择时钟源（如果选择内部时钟源时需设置 CR2.CLKC[1:0]=0b01）；
- 步骤 4. 设定 USARTn_CR1, USARTn_CR2, USARTn_CR3 寄存器；
- 步骤 5. 设定 USARTn_PR 选择预分频值, USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6. USARTn_CR1.RTOE=1, 如果需要使用中断, 则设定 USARTn_CR1.RTOIE=1；
- 步骤 7. 设置 TMR0_BCONR.CSTA=0；
- 步骤 8. 设置 TMR0_CNTAR 为 0, 设置 TMR0_CMPAR 寄存器和 TMR0_BCONR.CKDIVA寄存器决定 TIMEOUT 时间；
- 步骤 9. 设置 TMR0_BCONR.HCLEA=1, TMR0_BCONR.HSTAA=1, TMR0_BCONR.ASYNCLKA=1, TMR0_BCONR.SYNCLKA=0, TMR0_BCONR.SYNSA=1；
- 步骤 10. 使能接收器 (USARTn_CR1.RE=1), 如果需要使用接收中断, 则设置 USARTn_CR1.RIE=1；
- 步骤 11. 检测到 TIMEOUT 后按照以下步骤设置关闭 TMR0 定时器和清除 USARTn_SR.RTOF 状态位。
同时设置 TMR0_BCONR.SYNCLKA=1 和 TMR0_BCONR.SYNA=0, 再同时设置 TMR0_BCONR.CSTA=0, TMR0_BCONR.SYNCLKA=0 和 TMR0_BCONR.SYNSA=1, 通过写 USARTn_CR1.CRTOF 清除 USARTn_SR.RTOF 状态位。

29.3.2.7 RX 线唤醒停止模式功能

UART 通信空闲时, 可以让系统进入停止模式以节省电流消耗, 在不改变 UART PORT 设定的情况下, UART 单元 1 可以通过 RX 线来唤醒系统的停止模式。具体步骤如下:

- 步骤 1. UART 通信空闲时, 设定 USART_1_WUPI 中断向量以及 INTC_WKEN.RXWKEN 位使能 UART 接收信号线唤醒停止模式功能；
 - 步骤 2. 系统进入停止模式；
 - 步骤 3. 系统检测到 RX 线下降沿时, 从停止模式返回, 在 USART_1_WUPI 中断处理程序中关闭该功能。
- 需要注意的是, 当通信方需要唤醒本系统时, 需要发送一帧唤醒数据 (建议为 0x00), 该数据不会被 UART 接收且不置位相关的标志。并且通信方需要经过系统停止模式唤醒需要的时间后再进行 UART 通信数据的传输。

UART RX 线唤醒功能, 可以对 RX 线上的噪声进行过滤, 详细信息请参考 [USART_NFC 寄存器](#)。

29.3.2.8 UART 半双工通信方式

UART 模式支持单线半双工模式, UART 模式下通过设定 USARTn_CR3.HDSEL=1 启用单线半双工模式。

单线半双工模式时:

- TX 和 RX 线从内部相连接, 不再使用 RX 管脚。
- 无数据传输且发送功能禁止时, TX 管脚处于释放状态。因此使用单线半双工模式时, 通过上拉 TX 线来避免无数据传输时的浮空输入。
- 无数据传输且发送功能使能时, TX 管脚输出为高电平。

除此以外, 半双工模式与正常 UART 模式通信相似。需要注意的是, 发送过程不会被硬件封锁, 只要数据在 USARTn_CR1.TE=1 时写入, 发送就会进行。因此线路上的冲突必须有软件进行管理。

半双工通信时, 为防止 UART 接收到自己发送的数据, 建议 USARTn_CR1.RE=0。

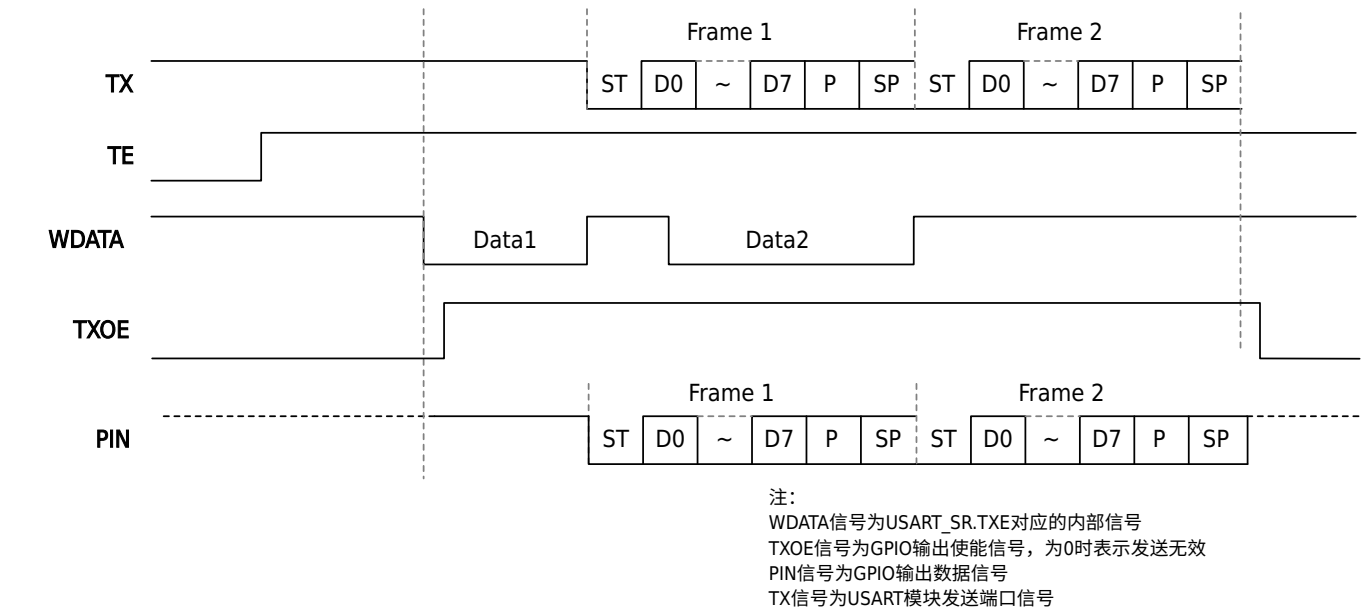


图 29-8 半双工工作时序

29.3.2.9 RS485 Driver Enable 功能

UART 模式支持 Driver Enable 模式, 与 RTS 共用输出信号, 需要先将 USART_CR3.DEM 配置为 1, 再把 USART_CR3.RTSE 配置为 1。

通过在 USART_CR3 控制寄存器中设置 DEM 开启驱动器启用功能, 允许用户通过 DE (驱动器启用) 信号激活外部收发器控制。使用 USART_CR3 控制寄存器中的 DEA[3:0]位字段编程的断言时间是 DE 信号激活和 START 位开始之间的时间。

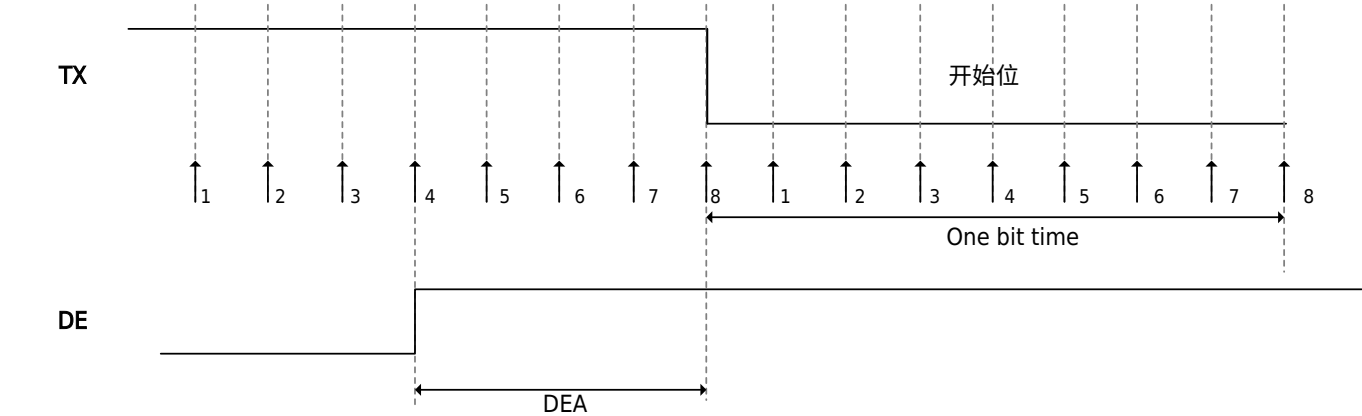


图 29-9 OVER8=1 DE 断言

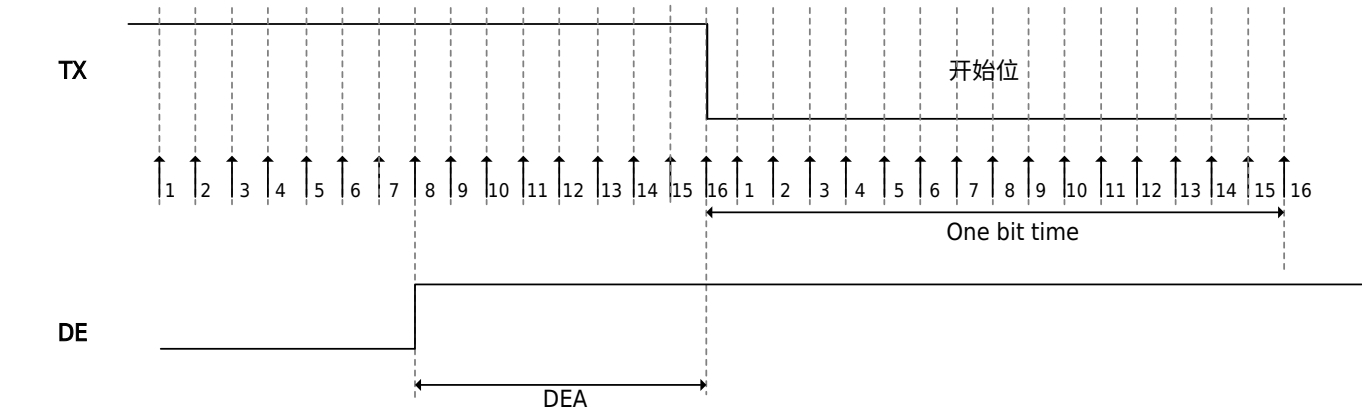


图 29-10 OVER8=0 DE 断言

使用 USART_CR3 控制寄存器中的 DED[3:0]位字段编程的去断言时间是最后一个停止位结束与 DE 信号失效之间的时间。DE 信号的极性可以使用 USART_CR3 控制寄存器中的 DEP 位来配置。

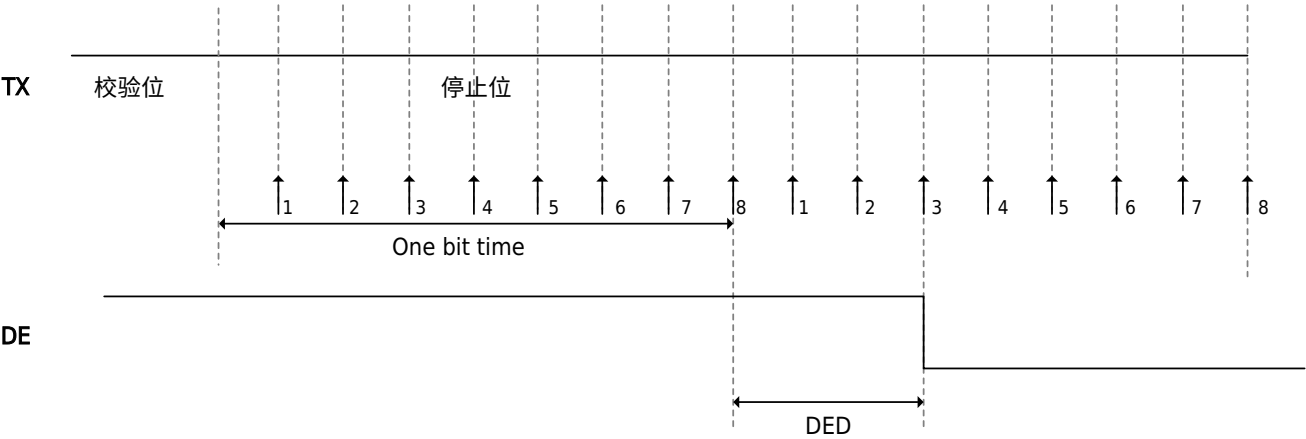


图 29-11 OVER8=1 DE 去断言

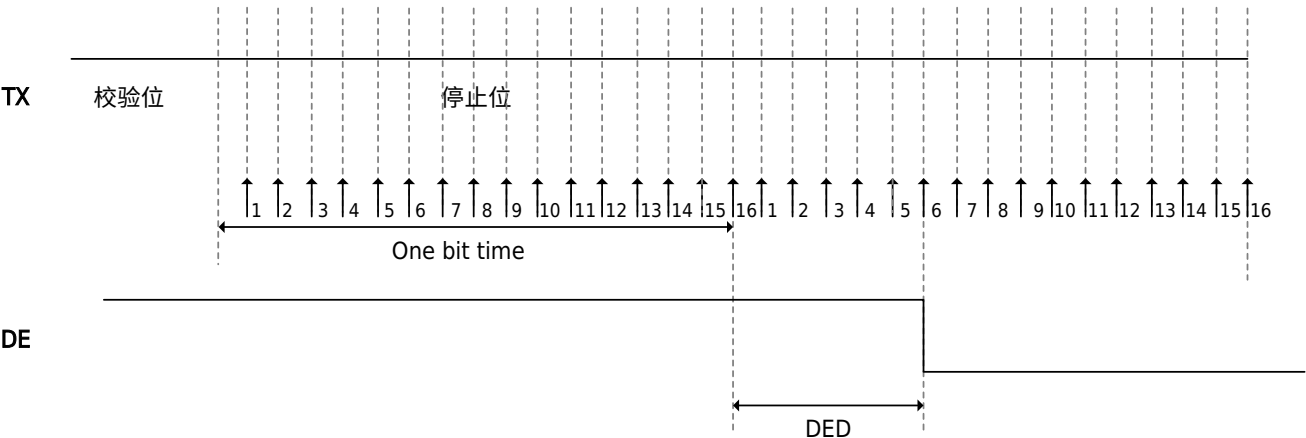


图 29-12 OVER8=0 DE 去断言

29.3.2.10 中断和事件

表 29-5 UART 中断/事件表

功能名称	使能位（仅中断）	标志位	可否作为事件源
接收错误中断	RIE	ORE, FE, PE	可
接收数据寄存器满中断	RIE	RXNE	可
发送数据寄存器空中断	TXEIE	TXE	可
发送完成中断	TCIE	TC	可
发送完成中断	TENDIE	TEND	可
TIMEOUT 中断	RTOIE	RTOF	可
RX 线唤醒停止模式中断	INTC_WKEN.RXWKEN	-	不可

29.3.3 多处理器通信

29.3.3.1 功能简介

多处理器通信模式是指多个处理器间共用通信线的一种通信方式，处理器分为发送站和接收站，每个接收站都有自己固有的 ID。发送站发送数据的类型有接收站 ID 和通信数据两种。通过在数据格式中添加 MPB 位来区分当前发送的是接收站的 ID 还是通信数据。MPB 位为 0 时当前帧为通信数据，MPB 位为 1 时当前帧为接收站的 ID。所有接收站都能接收发送站发送的 ID 并与自己的 ID 比较，如果一致，则接收数据，不一致则进入静默模式（既不接收数据也不置位接收相关标志）直到再次接收到 ID。

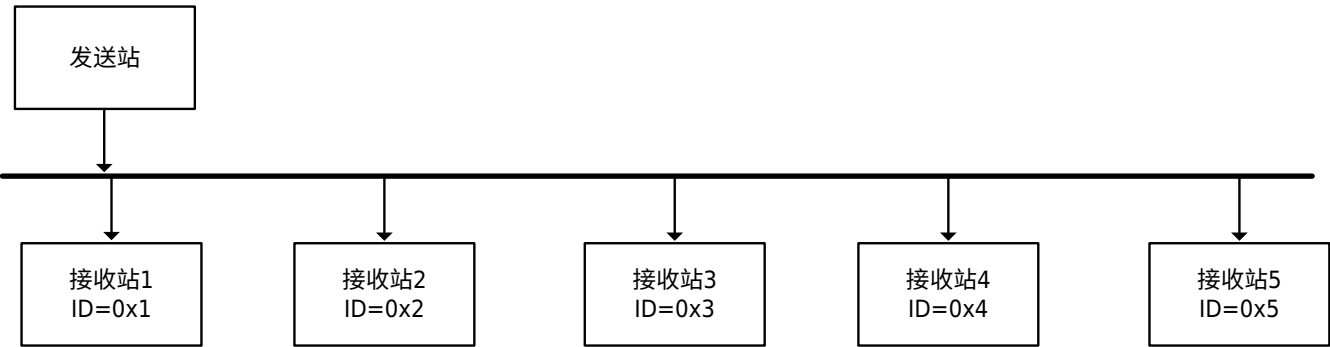


图 29-13 多处理器通信图例

29.3.3.2 数据格式

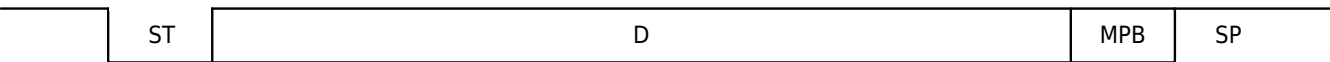


图 29-14 多处理器模式数据格式

- 开始位 (ST)：开始位固定由一位的低电平构成。
- 数据位 (D)：数据位可以配置成 8 位或者 9 位。
- 多处理器位 (MPB)：多处理器位可以配置成发送通信数据或者发送 ID。
- 停止位 (SP)：停止位固定为高电平，可以配置成 1 位或者 2 位。

29.3.3.3 动作说明

多处理器模式时校验位功能无效，增加了多处理器位功能，其余功能如时钟、中断等与 UART 模式相同。

发送站动作

- 步骤 1. 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2. 设定所需要使用的管脚；
- 步骤 3. 通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 步骤 4. 设定 USARTn_CR1, USARTn_CR2, USARTn_CR3 寄存器；
- 步骤 5. 设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6. 使能发送器 (USARTn_CR1.TE=1)，如果需要使用发送数据寄存器空中断，则设置 USARTn_CR1.TXEIE=1；
- 步骤 7. 等待发送数据寄存器空，设定 USARTn_TDR.MPID 位为 1（发送 ID），写 ID 值到 USARTn_TDR，发送 ID；



说明

CTS 功能有效时，USARTn_CTS 输入为低电平时数据传输到发送移位寄存器，发送开始。

- 步骤 8. 设定 USARTn_TDR.MPID 位为 0（发送数据），写数据到 USARTn_TDR，发送数据；



说明

CTS 功能有效时，USARTn_CTS 输入低电平时数据传输到发送移位寄存器，发送开始。

- 步骤 9.** 如果需要连续发送数据，重复步骤 8，如果需要改变 ID 后再发送数据重复 7 和 8；
- 步骤 10.** 通过确认 USARTn_SR.TC 位确认发送是否完成。连续发送数据并使用发送中断的情况，可通过 TI 中断写入最后一个发送数据，并将 USARTn_CR1.TXEIE 写 0，USARTn_CR1.TCIE 写 1，最后一帧数据发送结束后，产生发送完成中断。

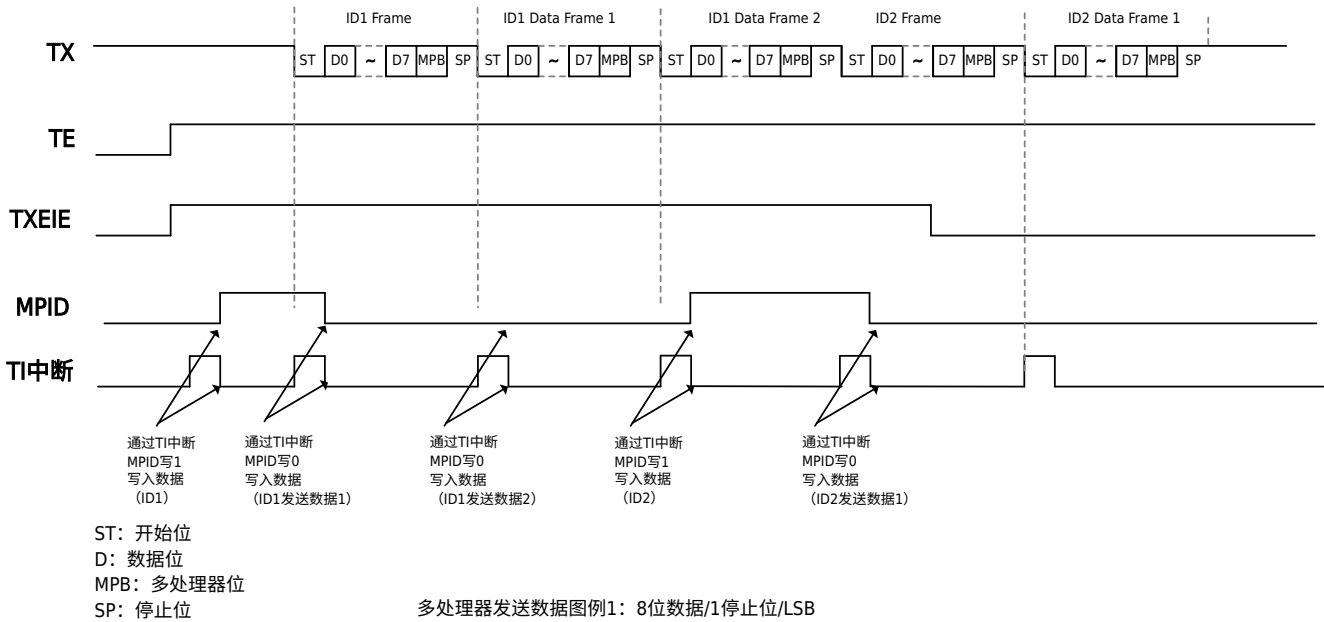


图 29-15 多处理器模式发送数据图例

接收站动作

在多处理器模式时，接收站必须保证能收到每一个 ID 数据，并与自身的 ID 比较，如果一致则接收数据，不一致则进入静默模式（不接收数据，也不置位接收相关标志），直到接收到下一个 ID 数据。通过 USARTn_CR1.SLME 位来实现这个功能。

USARTn_CR1.SLME=0 时正常接收数据。

USARTn_CR1.SLME=1 时，除非接收到 MPB 位为 1（ID）的数据，否则不接收数据，不发生 RI 中断，错误标志 FE，ORE 也不置位。当接收到 MPB 位为 1 的数据时（ID），USARTn_CR1.SLME 位自动清零，正常接收数据和发生中断。

动作步骤：

- 步骤 1.** 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2.** 设定所需要使用的管脚；
- 步骤 3.** 通过 USARTn_CR1.CLKC[1:0]位选择时钟源；
- 步骤 4.** 设定 USARTn_CR1，USARTn_CR2，USARTn_CR3 寄存器；
- 步骤 5.** 设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6.** USARTn_CR1.RE=1，USARTn_CR1.SLME=1（等待接收 ID），如果使用接收中断，则设定 USARTn_CR1.RIE=1；
- 步骤 7.** 当检测到开始位时，接收器将数据接收到接收移位寄存器，并检查 USARTn_SR.MPB 位；
- 步骤 8.** 如果 USARTn_SR.MPB=1，USARTn_CR1.SLME 位自动清零，正常接收数据，软件比较接收的 ID 与自身的 ID。
- a. 如果 ID 一致，则正常接收数据，发生中断，进行错误检测，与 UART 接收数据相同；

b. 如果 ID 不一致，软件再次将 USARTn_CR1.SLME 位写 1，重复 8 的动作。

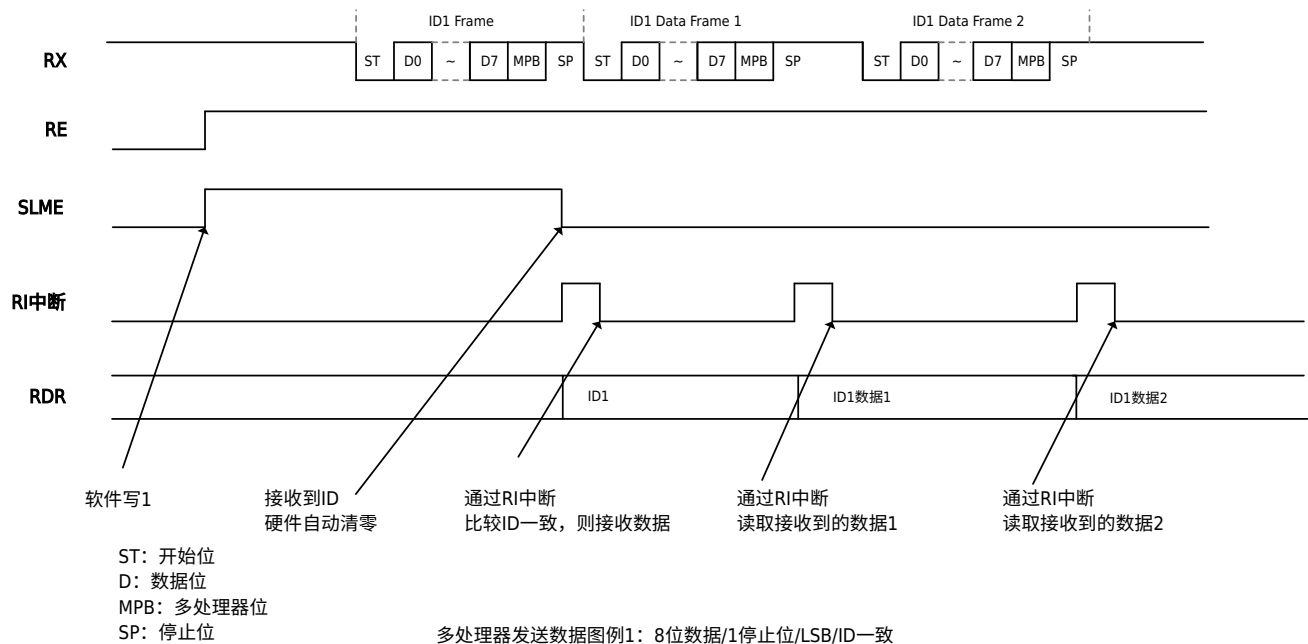


图 29-16 多处理器模式接收数据图例 1

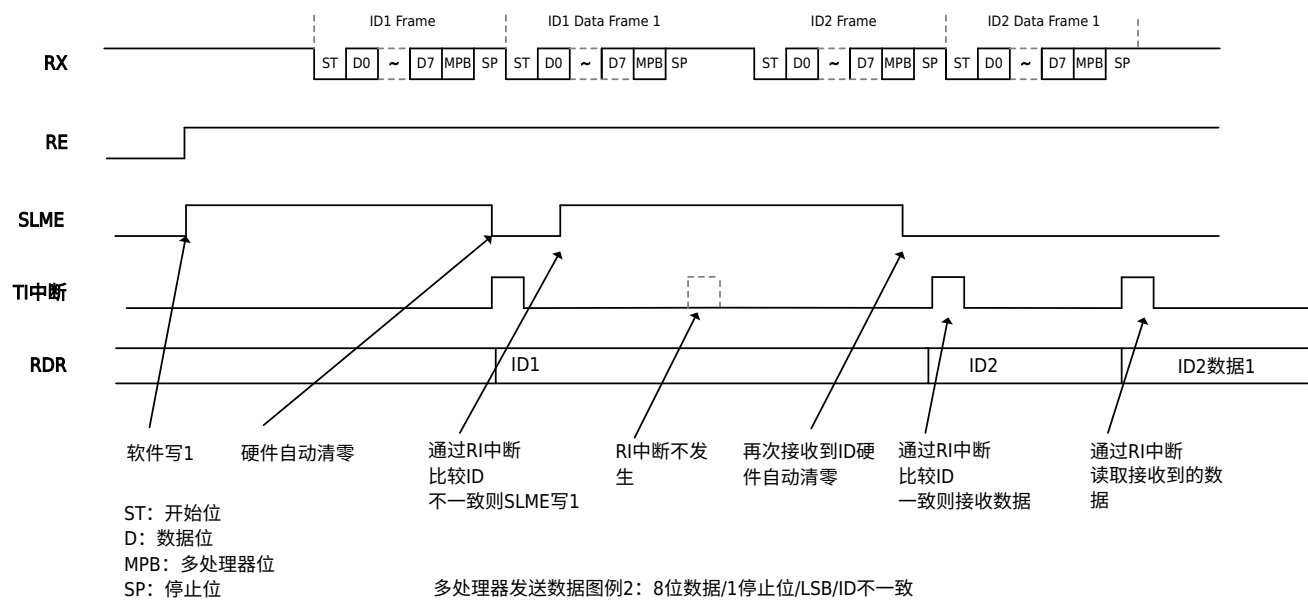


图 29-17 多处理器模式接收数据图例 2

29.3.3.4 中断和事件

多处理器模式除了无校验错误外，中断处理与 UART 模式相同。

表 29-6 多处理器模式中中断/事件表

功能名称	使能位（仅中断）	标志	可否作为事件源
接收错误中断	RIE	ORE, FE	可
接收数据寄存器满中断	RIE	RXNE	可
发送数据寄存器空中断	TXEIE	TXE	可
发送完成中断	TCIE	TC	可

功能名称	使能位（仅中断）	标志	可否作为事件源
发送完成中断	TENDIE	TEND	可

29.3.4 UART-LIN

29.3.4.1 功能简介

LIN 是 Local Interconnect Network 的简称，是为降低汽车网络成本的低速（1~20kbps）串行通信协议。

29.3.4.2 LIN 数据格式

LIN 的一帧数据是由开始位+8 位数据+1 停止位组成，以 LSB 方式发送和接收数据。LIN 总线上的数据行为如下图所示。

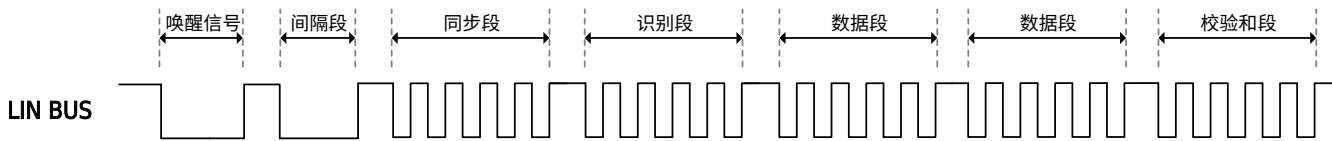


图 29-18 LIN 总线数据行为

29.3.4.3 LIN 发送

唤醒信号帧的发送

通过发送数据 0x80 对应信号帧的发送。

间隔段发送

同步间隔段表示一帧数据的开始，本产品硬件支持 LIN 间隔段的发送。

LIN 间隔段低电平的宽度由寄存器 USARTn_CR2.SBK 寄存器设定。

USARTn_CR2.SBKM=0 时，USARTn_CR2.SBK 写 1，硬件自动送出间隔段。

USARTn_CR2.SBKM=1 时，USARTn_CR2.SBK 写 1 后，写数据 0x00 到发送数据寄存器 USARTn_TDR.TDR 开始发送间隔段。

间隔段发送完成后，USARTn_CR2.SBK 自动清零。

同步段发送

通过发送数据 0x55 实现同步段数据的发送。

总线错误检测

USARTn_CR2.BEE=1 时，硬件检测到总线上的数据和发送的数据不一致时，置位 USARTn_SR.BE 标志，USARTn_CR2.BEIE=1 时，产生对应的中断。

29.3.4.4 LIN 接收

唤醒信号帧的检测

USARTn_CR2.WKUPE=1 时，硬件自动检测唤醒信号，检测到 RX 线低电平宽度上大于等于 2.5 位数据宽度（19.2kbps 时为 130μs）时，置位 USARTn_SR.WKUP 标志，USARTn_CR2.WKUPE=1 时，产生对应的中断。检测到唤醒信号帧后需要清除 USARTn_SR.WKUP 标志，并读取 USARTn_RDR.RDR 寄存器。

USARTn_CR2.WKUPE 寄存器仅在系统需要等待唤醒信号时设定为 1，其他情况需要设定为 0。

同步间隔段的检测

本产品硬件支持同步间隔段检测，当检测到通信线低电平宽度大于等于 USARTn_CR2.LBDL 寄存器设定的值，并检测到 break delimiter 时，置位 USARTn_SR.LBD 标志，USARTn_CR2.LBDIE=1 时，产生对应的中断。

需要注意的是，同步间隔段检测时，会置位 USARTn_SR.FE 标志，清除 USARTn_SR.LBD 标志的同时，需要清除 USARTn_SR.FE 标志，并读取 USARTn_RDR.RDR 寄存器（读取值为 0x00）。

同步段的检测与波特率测量

当检测到间隔段后，硬件自动测量同步段的频率。测量计数器的时钟由 USARTn_PR.LBMPSC 位设定，计数器的值保存在 USARTn_LBMC 寄存器中，选择的计数器时钟频率除以 USARTn_LBMC 寄存器值即可得到主节点的波特率。

需要注意的是，USARTn_LBMC 寄存器的读取需要在同步段接收完成后再读取。

接收数据设定步骤

- 1. 将 USARTn_CR1 寄存器设定为复位值；
- 2. 设定 UART 所需要使用的管脚；
- 3. 通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 4. 设定 USARTn_CR1 (RE=1, RIE=0) , USARTn_CR2, USARTn_CR3 寄存器；
- 5. （可选）唤醒信号帧的检测：
 - a. 设定 USARTn_CR2.WKUPE=1，等待唤醒信号；
 - b. 检测到唤醒信号，USARTn_CR2.WKUPE=0，等待同步段间隔段。
- 6. （可选）同步段间隔场检测：
 - a. 检测到同步间隔断后，清除 USARTn_SR.FE/LBD 标志，并确认接收到的数据为 0x00。
- 7. （可选）同步段频率测量与通信波特率计算：
 - a. 如果需要使用接收中断，则设置 USARTn_CR1.RIE=1；
 - b. 检测到同步间隔场，并完成同步段接收后，读取 USARTn_LBMC 寄存器计算波特率。接收数据若为 0x55，说明波特率误差在可接受范围内；若接收数据不为 0x55，用户校准波特率为必须步骤，用户根据 USARTn_LBMC 寄存器来重新校准波特率。
- 8. 接收识别段（过程与 UART 接收数据过程相同）；
- 9. 接收数据（过程与 UART 接收数据过程相同）；
- 10. 接收校验和（过程与 UART 接收数据过程相同）。

29.3.4.5 中断和事件

表 29-7 LIN 中断/事件表

功能名称	使能位（仅中断）	标志位	可否作为事件源
错误中断	RIE	ORE, FE	可
	BEIE	BE	可
唤醒信号/间隔段检测中断	WKUPIE	WKUP	可
	LBDIE	LBD	可
接收数据寄存器满中断	RIE	RXNE	可
发送数据寄存器空中断	TXEIE	TXE	可
发送完成中断	TCIE	TC	可
发送完成中断	TENDIE	TEND	可

29.3.5 智能卡

29.3.5.1 连接示意图

支持 ISO/IEC 7816-3 规定的智能卡通信协议。下图为智能卡模式的连接示意图。

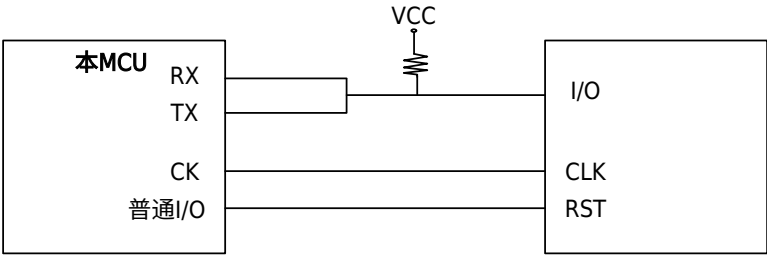


图 29-19 智能卡连接示意图

29.3.5.2 时钟

内部时钟源

智能卡模式时只能使用内部波特率生成器生成的时钟作为时钟源。
一位数据传输的基本时钟数为 USARTn_CR3.BCN[2:0]设定值。
通过设定寄存器 USARTn_CR2.CLKC[1:0]位控制智能卡模式的时钟输出。

最高波特率

内部时钟源时，内部波特率生成器生成的波特率计算公式为：

$$B = \frac{C}{2 \times BCN \times (DIV\_Integer + 1)}$$

其中：

- B：波特率，单位：Mbps
- C：当 USARTn_PR.ULBREN 为 0 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1，PCLK1/4，PCLK1/16，PCLK1/64），单位：MHz
当 USARTn_PR.ULBREN 为 1 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1/128，PCLK1/256，PCLK1/512，PCLK1/1024），单位：MHz
- DIV_Integer：USARTn_BRR.DIV_Integer 设定值
- BCN：USARTn_CR3.BCN 寄存器设定值

当 C 为 PCLK1，DIV_Integer=0，BCN=0 时，波特率为最高波特率为 PCLK1/64（Mbps）。

采样和接收容差

检测到 RX 的下降后，USART 会基于内部基本时钟对接收数据进行时钟同步，从而开始数据接收。接收数据将在数据中央被采样。

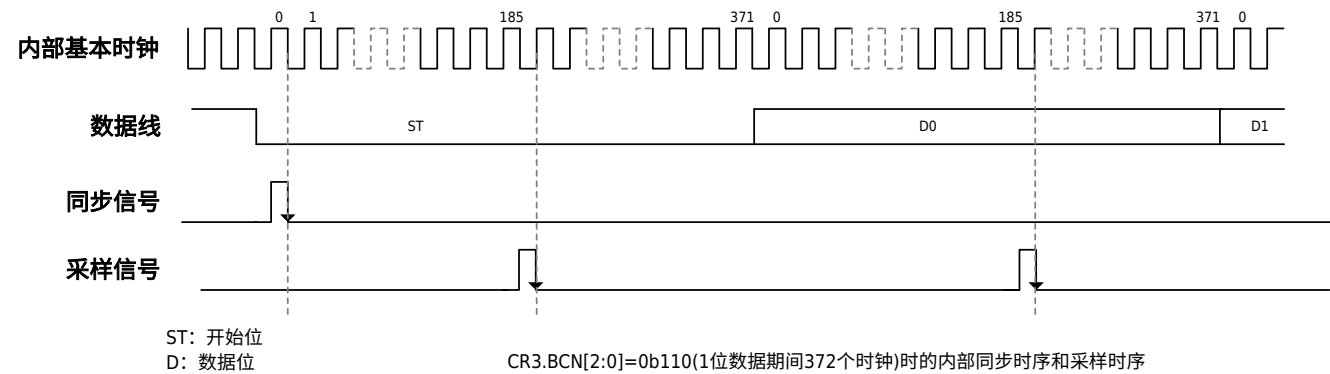


图 29-20 智能卡模式同步时序和采样时序图

接收容差计算公式：

$$RM = \{0.5 \times (1 - \frac{1}{BCN}) - 9.5CFD\} \times 100\%$$

其中：

- RM：接收容差
- BCN：一位数据传输所需要的时钟数 (USARTn.CR3.BCN[2:0]设定值)
- CFD：时钟频率偏差

29.3.5.3 数据格式

智能卡模式时一帧数据由起始位，数据位和校验位组成。

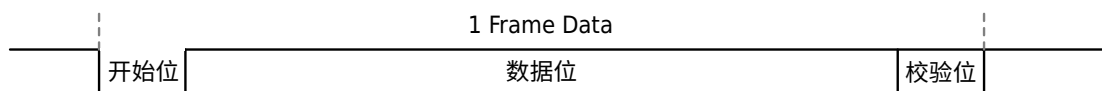


图 29-21 智能卡模式数据格式

- 开始位 (ST)：开始位固定由一位的低电平构成。
- 数据位 (D)：数据位固定为 8 位数据。
- 校验位 (P)：校验位需配置成 1 位偶校验。

29.3.5.4 智能卡的初始化设定步骤

- 步骤 1. 将 USARTn_CR1 寄存器设定为复位值；
- 步骤 2. 设定所需要使用的管脚；
- 步骤 3. 状态寄存器确认，USARTn_SR 寄存器设定为复位值；
- 步骤 4. 设定 USARTn_CR1、USARTn_CR2、USARTn_CR3 寄存器；
- 步骤 5. 设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率；
- 步骤 6. USARTn_CR2.CLKC[1:0]位设定时钟控制；
- 步骤 7. USARTn_CR1 寄存器 (TE、RE、RIE、TXEIE 位) 设定，除了自测外，TE 和 RE 不要同时设定为 1。

由发送模式切换到接收模式，或者由接收模式切换到发送模式时，需要重新设定上述步骤 1 到步骤 7。

29.3.5.5 智能卡模式动作说明

智能卡模式时，TI 中断（发送数据空中断）的标志位为 USARTn_SR.TC 位。USARTn_SR.TC=1 而且 USARTn_CR1.TXEIE=1 时产生 TI 中断。

功能概述

发送数据时两帧数据间（从校验位结束到下一帧的起始位开始）有 2etu（Elementary Time Unit）以上的保护时间。

发送数据时如果检测到接收方发送的错误信号则经过 2etu 后自动重发数据。

接收数据发生校验错误时，发送 1etu 的低电平即错误信号，错误信号送出的时序为接收开始经过 10.5etu。

发送说明

一帧数据发送完成后，如果检测到接收方发送的错误信号，则 USARTn_SR.FE 置 1（如果 USARTn_CR1.RIE=1，则发生错误中断），USARTn_SR.TC 标志不置 1，数据自动重发。USARTn_SR.FE 位必须在接收到下一帧校验位之前清零。

一帧数据无错误发送完成后，USARTn_SR.TC 标志置位，USARTn_CR1.TXEIE=1 时发生 TI 中断。再次写入数据，则可实现连续发送数据。

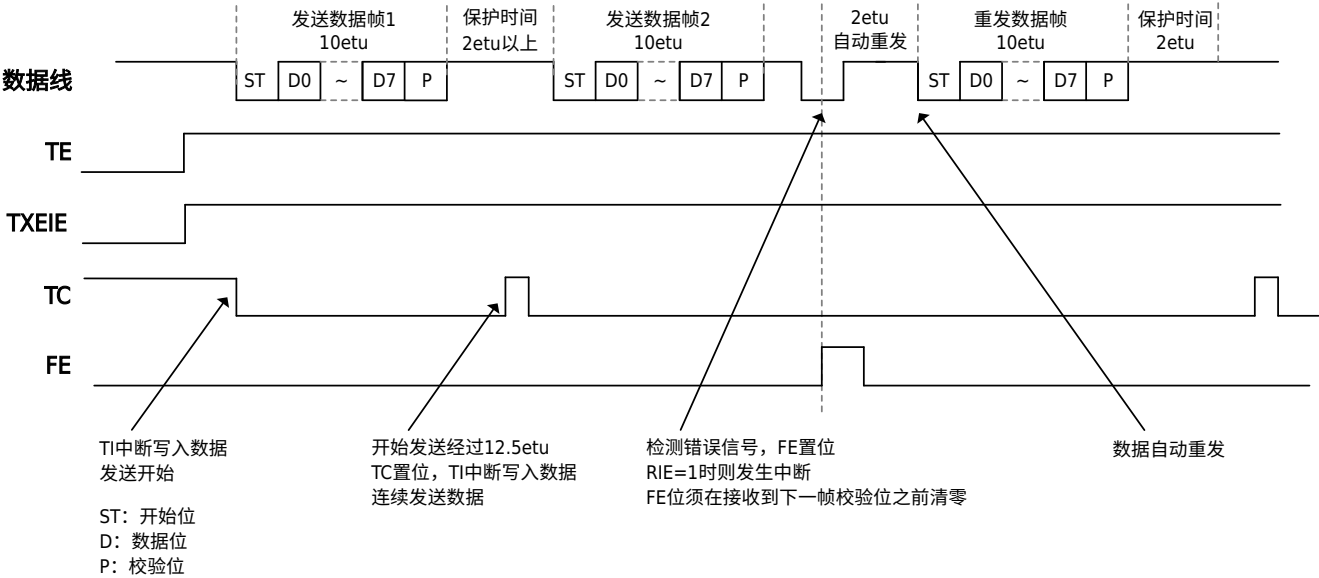


图 29-22 智能卡模式发送数据图例

接收说明

接收数据时, 如果检测到校验错误, 则 USARTn_SR.PE 置位, 中断使能时, 发生 EI 中断。USARTn_SR.PE 位需要在接收到下一帧校验位之前清零。

发生校验错误时, 会发送 1etw 的低电平, 即错误信号, 要求发送方重新发送数据。

正常接收数据, 可以通过 RI 中断读取接收到的数据, 连续接收。

接收数据时, 检测上溢错误。

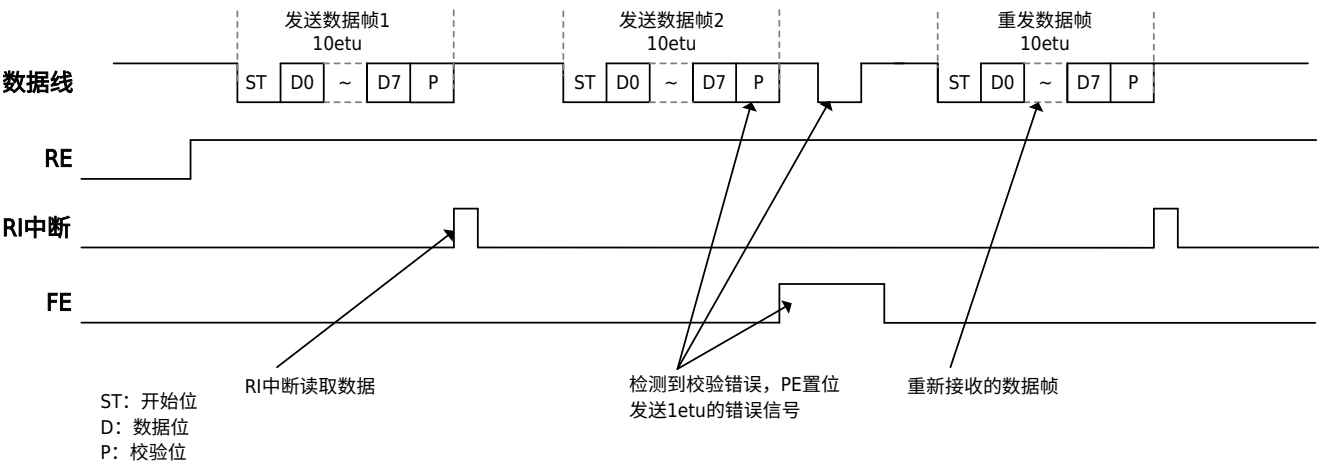


图 29-23 智能卡模式接收数据图例

29.3.5.6 中断和事件

表 29-8 智能卡模式中断/事件表

功能名称	使能位 (仅中断)	标志位	可否作为事件源
错误中断	RIE	ORE, PE, FE	可
接收数据满中断	RIE	RXNE	可
发送数据空中断	TXEIE	TC	可

29.3.6 时钟同步模式

29.3.6.1 时钟

时钟同步模式可以选择内部波特率生成器生成的时钟（内部时钟源）或 USARTn_CK 管脚输入的时钟（外部时钟源）作为通信的时钟源。

内部时钟源

同步时钟从 USARTn_CK 管脚输出，一帧数据输出 8 个时钟脉冲，既不发送数据也不接收数据时，时钟输出固定为高电平。

外部时钟源

外部时钟源即从 USARTn_CK 管脚输入时钟作为通信时钟。

最高波特率

内部时钟源时，内部波特率生成器生成的波特率计算公式为：

$$B = \frac{C}{4 \times (DIV\_Integer + 1)}$$

其中：

- B：波特率，单位：Mbps
- C：当 USARTn_PR.ULBREN 为 0 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1，PCLK1/4，PCLK1/16，PCLK1/64），单位：MHz
当 USARTn_PR.ULBREN 为 1 时，USARTn_PR.PSC[1:0]位设定的时钟（PCLK1/128，PCLK1/256，PCLK1/512，PCLK1/1024），单位：MHz
- DIV_Integer：USARTn_BRR.DIV_Integer 设定值

内部时钟源时，当 C 为 PCLK1，DIV_Integer=1 时，最高波特率为 PCLK1/8（Mbps）。

 **注意**
同步模式时 DIV_Integer 禁止设置为 0。

外部时钟源时，外部输入时钟的最大频率要求为 PCLK1/6（MHz），所以最高波特率为 PCLK1/6（Mbps）。需要注意的是，同步模式最高通信波特除了以上描述的基于 PCLK1 的计算方法外，还需要参考《数据手册》“电气特性”章节规定的最高通信波特率。

29.3.6.2 数据格式

时钟同步模式一帧数据固定有 8 位组成，一帧数据的发送和接收需要 8 个同步时钟脉冲。发送数据时数据在同步时钟的下降沿送出，接收数据时数据在同步时钟的上升沿被采样。

同步时钟在没有数据传输时固定为高电平，最后一位发送完后，通信线保持最后一位的值。

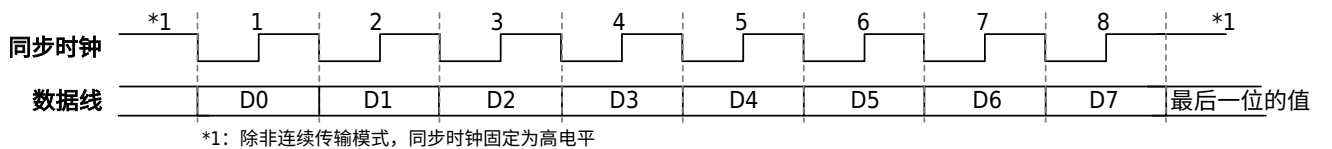


图 29-24 时钟同步模式数据格式

29.3.6.3 调制解调器操作

调制解调器操作包括 CTS 功能和 RTS 功能。USARTn_CR3.RTSE=1 时 RTS 功能有效，USARTn_CR3.CTSE=1 时 CTS 功能有效。

CTS 功能

CTS 功能是通过 USARTn_CTS 管脚的输入来控制数据的发送，只有当 USARTn_CTS 管脚输入低电平时才可以发送数据，发送数据过程中如果 USARTn_CTS 输入高电平，正在发送的数据不受影响。

RTS 功能

RTS 功能是指通过 USARTn_RTS 管脚输出低电平，请求对方发送数据。

USARTn_RTS 管脚输出低电平需要满足以下全部条件：

- 接收使能 (USARTn_CR1.RE=1)，且不在接收数据
- USARTn_RDR.RDR 寄存器中没有未读取的数据 (USARTn_CR1.RE=1 时)
- USARTn_TDR.TDR 更新完成 (USARTn_CR1.TE=1 时)
- 无任何接收错误

如果不能同时满足以上全部条件，USARTn_RTS 则输出高电平。

29.3.6.4 发送器

发送器使能位 (USARTn_CR1.TE) 置 1 时，发送移位寄存器中的数据在 USARTn_TX 管脚串行输出，相应的时钟脉冲在 USARTn_CK 管脚输出。

发送数据寄存器 USARTn_TDR.TDR 和内部的发送移位寄存器组成双缓冲器结构，可以连续发送数据。

通过发送数据寄存器空中断或者 DMA 写入发送数据时，为保证发送的正确性，一次请求只能写入一次数据。

发送数据设定步骤

- 步骤 1.

将 USARTn_CR1, USARTn_SR1 寄存器设定为复位值；
- 步骤 2.

设定所需要使用的管脚；
- 步骤 3.

通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 步骤 4.

设定 USARTn_CR1, USARTn_CR2, USARTn_CR3 寄存器；
- 步骤 5.

设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6.

使能发送器 (USARTn_CR1.TE=1)，如果需要使用发送数据寄存器空中断，则设置 USARTn_CR1.TXEIE=1；
- 步骤 7.

等待发送数据寄存器空，写通信数据到 USARTn_TDR.TDR，数据传输到发送移位寄存器，发送开始；
- 说明

CTS 功能有效时，USARTn_CTS 输入为低电平时数据传输到发送移位寄存器，发送开始。
- 步骤 8.

如果需要连续发送数据时，重复步骤 7；
- 步骤 9.

通过确认 USARTn_SR.TC 位确认发送是否完成。连续发送数据并使用发送中断的情况，可通过 TI 中断写入最后一个发送数据，并将 USARTn_CR1.TXEIE 写 0，USARTn_CR1.TCIE 写 1，最后一个数据发送结束后，产生发送完成中断。

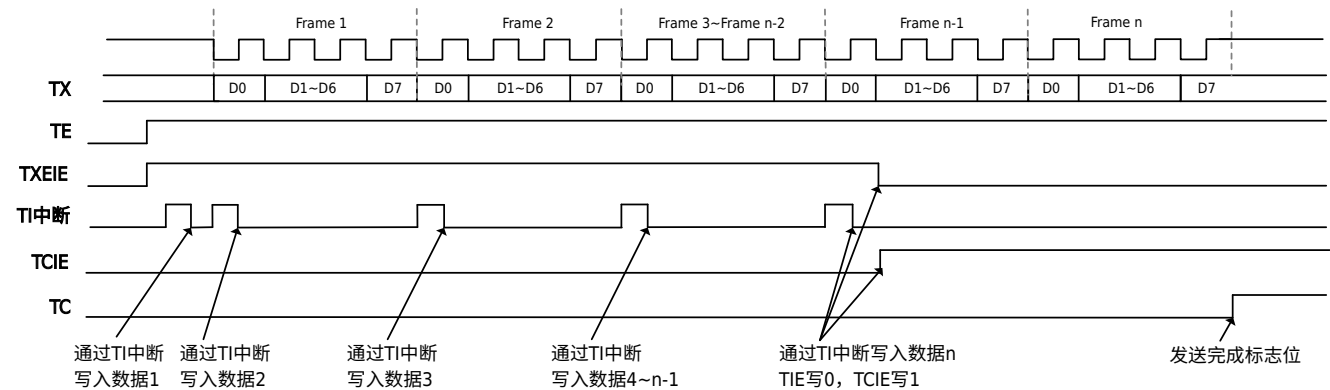


图 29-25 时钟同步模式发送数据图例 1 (LSB/无 CTS 功能)

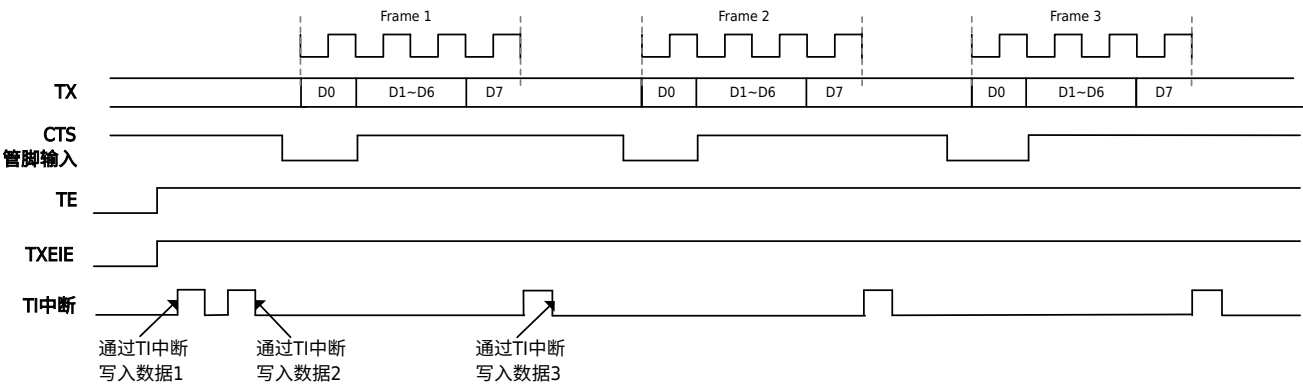


图 29-26 时钟同步模式发送数据图例 2 (LSB/CTS 功能)

发送器中断

时钟同步模式发送器支持两种中断，发送数据寄存器空中断 TI 和发送完成中断 TCI。
TXEIE=1，USARTn_TDR.TDR 寄存器的值传送到发送移位寄存器时 TI 中断发生。
TCIE=1，发送数据的最后一位时 USARTn_TDR.TDR 寄存器没有更新则 TCI 中断发生。

29.3.6.5 接收器

接收数据设定步骤

- 步骤 1. 将 USARTn_CR1, USARTn_SR 寄存器设定为复位值；
- 步骤 2. 设定所需要使用的管脚；
- 步骤 3. 通过 USARTn_CR2.CLKC[1:0]位选择时钟源；
- 步骤 4. 设定 USARTn_CR1, USARTn_CR2, USARTn_CR3 寄存器；
- 步骤 5. 设定 USARTn_PR 选择预分频值，USARTn_BRR 寄存器设定通信波特率（时钟源为外部时钟源时不需要设定）；
- 步骤 6. 使能接收器（USARTn_CR1.RE=1），如果需要使用接收中断，则设置 USARTn_CR1.RIE=1；



说明

使用 RTS 功能时，RE=1 后 USARTn_RTS 输出低电平。

- 步骤 7. 同步于输入同步时钟或者内部生成的同步时钟开始接收数据，接收数据到接收移位寄存器。
 - 发生上溢错误时，数据丢失并置位 USARTn_SR.ORE 标志；
 - 无错误发生时，接收到的数据传送到 USARTn_RDR.RDR 寄存器中，置位 USARTn_SR.RXNE 标志，将当前接收到的数据在接收到下一帧数据最后一位前读取后重复步骤 7 可实现连续接收数据功能。



说明

使用 RTS 功能时，数据读取后 USARTn_RTS 输出低电平。

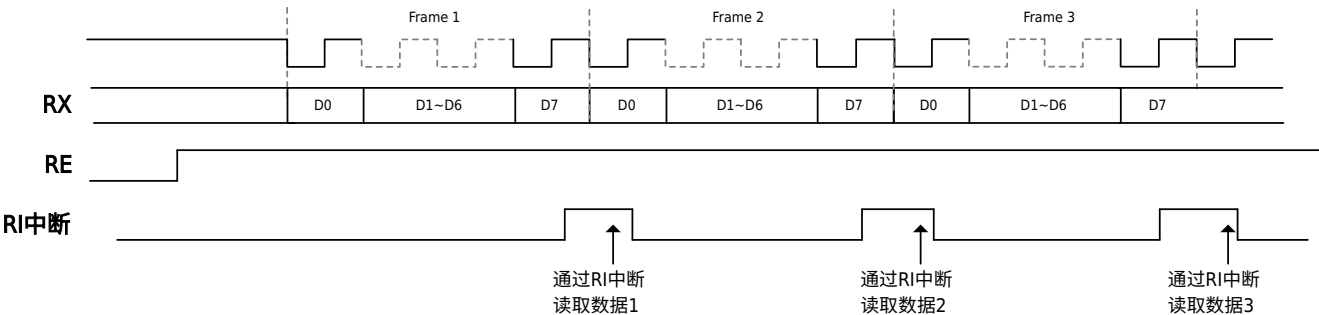


图 29-27 时钟同步模式接收数据图例 1 (LSB/无 RTS 功能)

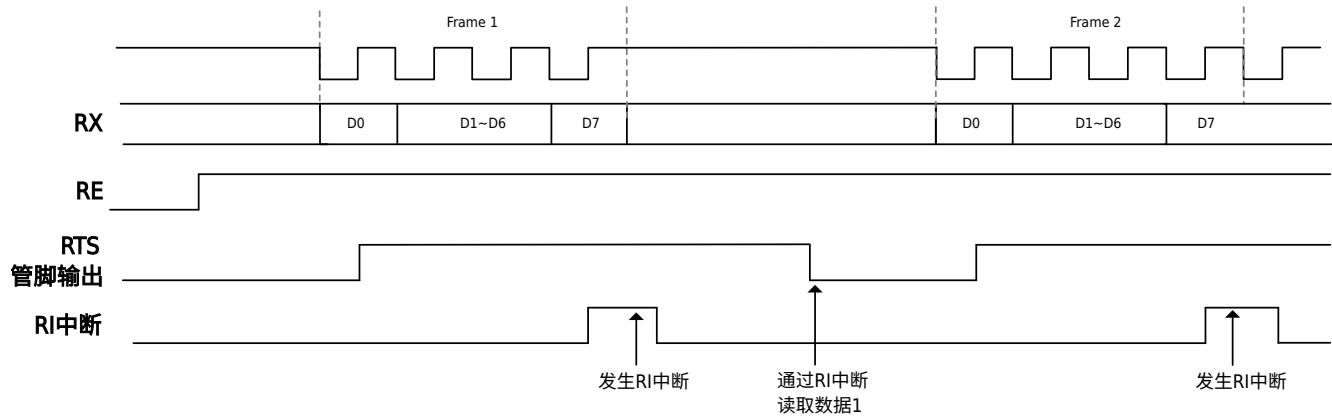


图 29-28 时钟同步模式接收数据图例 2 (LSB/RTS 功能)

错误处理

时钟同步模式接收数据时接收错误为上溢错误 (USARTn_SR.ORE)。发生接收错误不能再进行数据的接收和发送。可以通过将错误标志清零来重启数据传输，清除方法是写对应的清零寄存器。

上溢错误发生的条件是 USARTn_RDR.RDR 值未被读取的情况下又收到新的数据，所以应该在收到当前帧最后一位之前将前一帧收到的数据读取。发生上溢错误时接收到的数据丢失，RI 中断不发生。

接收器中断

时钟同步模式接收器支持两种中断，接收数据寄存器满中断 RI 和接收错误中断 EI。

RIE=1，数据从接收移位寄存器传送到接收数据寄存器时 RI 中断发生。

RIE=1，接收数据发生错误（上溢错误）时 EI 中断发生。

29.3.6.6 同时发送接收数据

USART 时钟同步模式支持全双工动作，同时发送接收数据。同时发送接收数据时需要一个命令把 RE，TE，RIE，TXEIE 写 1，其他设定流程与发送器和接收器相同。

29.3.6.7 中断和事件

表 29-9 时钟同步模式中断/事件表

中断名称	使能位（仅中断）	标志位	可否作事件源
接收错误中断	RIE	ORE	可
接收数据寄存器满中断	RIE	RXNE	可
发送数据寄存器空中断	TXEIE	TXE	可
发送完成中断	TCIE	TC	可
发送完成中断	TENDIE	TEND	可

29.3.7 数字滤波功能

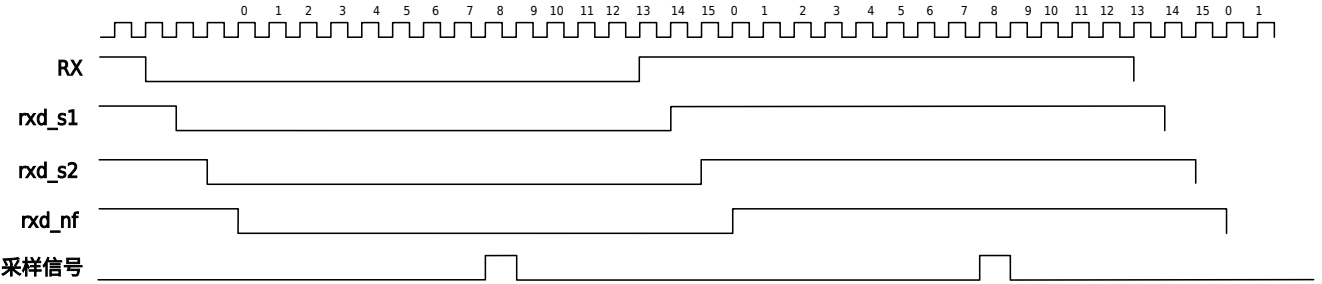
USARTn_CR1.NFE=1 时，内置数字滤波器功能有效。数字滤波器仅在 UART 模式时有效，可以除去接收数据线 RX 上的噪音。

内置数字滤波器可以滤除小于一位数据的 3/16 (USARTn_CR1.OVER8=0) 宽度或者 3/8 宽度 (USARTn_CR1.OVER8=1) 的噪音。

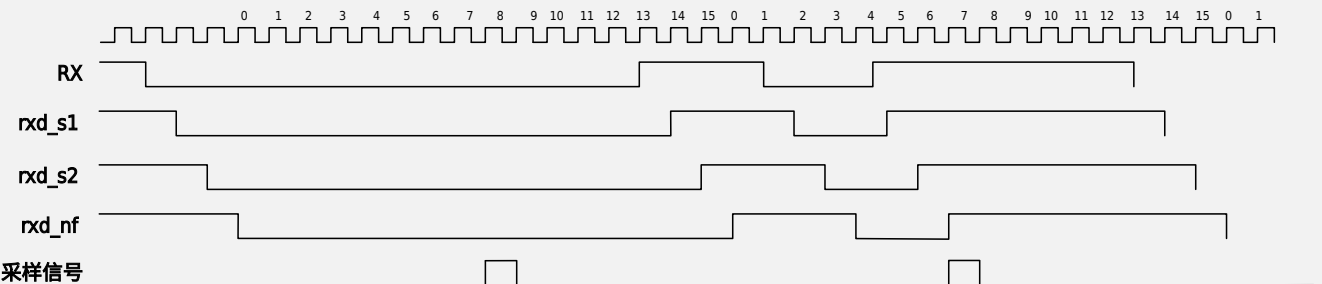
如果数字滤波器的时钟停止后再开始时，数字滤波器从时钟停止时保持的状态继续工作。

USARTn_CR.TE=0 且 USARTn_CR.RE=0，将数字滤波器内部的 Flip-Flop 状态复位为 1。

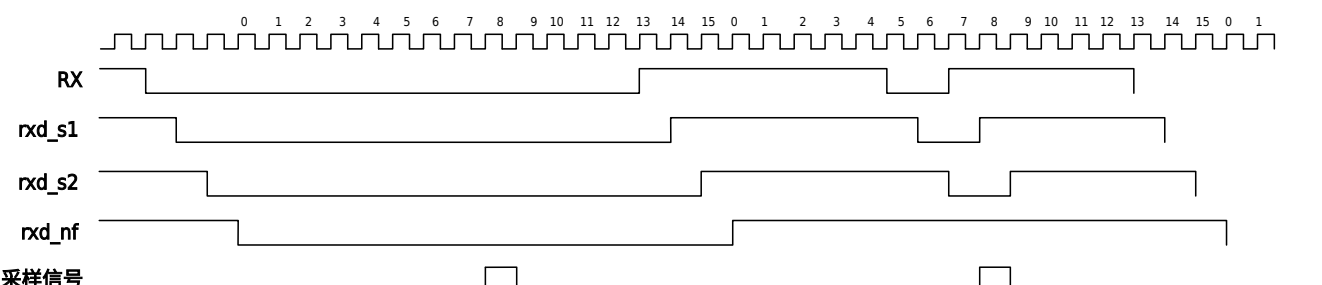
启用滤波，无干扰



启用滤波，未过滤干扰，不影响采样（采样信号有效时，未采到干扰信号）



启用滤波，过滤干扰，不影响采样（干扰信号被过滤掉，采样时采集到正确信号）



启用滤波，未过滤干扰，影响采样（干扰未过滤掉，采样时采集到干扰信号）

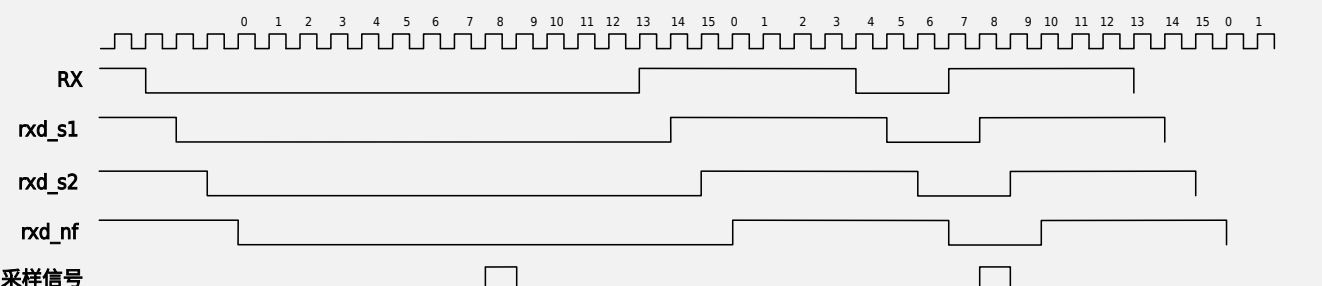


图 29-29 数字滤波功能图例

- rx_d_s1: 信号 RX 延迟 1 个采样时钟的内部信号
- rx_d_s2: 信号 RX 延迟 2 个采样时钟的内部信号
- rx_d_nf: 实际被采样的信号，该信号通过比较输入信号 RX、rx_d_s1 和 rx_d_s2 来确定

打开 NF 功能，实际上是对输入 RX 信号进行了两次延迟操作，得到三个信号。通过这两个延迟，将输入信号与两个延迟信号必须保持一致后，才能在模块内部进行翻转。

29.3.8 中断

表 29-10 USART 整体中断一览表

中断名称	符号	说明
错误中断	USART_n_EI n=1~10	UART/多处理器/智能卡/时钟同步模式接收错误中断 LIN 错误中断
接收数据寄存器满中断	USART_n_RI n=1~10	UART/多处理器/LIN/智能卡/时钟同步模式接收数据寄存器满中断 LIN 错误中断
发送数据寄存器空中断	USART_n_TI n=1~10	UART/多处理器/LIN/智能卡/时钟同步模式发送数据寄存器空中断
发送完成中断	USART_n_TCI n=1~10	UART/多处理器/LIN/智能卡/时钟同步模式发送完成中断 (向发送数据寄存器写入发送数据可清中断)
发送完成中断	USART_n_TENDI n=1~10	UART/多处理器/LIN/智能卡/时钟同步模式发送完成中断 (向寄存器 CR1.CTEND 位写入 1 或向发送数据寄存器写入发送数据可清中断)
唤醒信号/间隔段检测中断	USART_n_BRKWKPI n=5/10	LIN 唤醒信号/间隔段检测中断
UART 接收 TIMEOUT 中断	USART_n_RTO n=1~10	UART 接收 TIMEOUT 中断
RX 线唤醒停止模式中断	USART_1_WUPI	USART1 RX 线唤醒停止模式中断

29.4 寄存器

本章将详细描述 USART 模块控制状态等相关寄存器。功能寄存器位只有该通道配置了对应的功能才有效，否则默认为 0。

29.4.1 寄存器总表

USART1_NFC 基地址：0x40055400

表 29-11 USART1 滤波控制寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x1C	USART1_NFC	USART1 滤波控制寄存器	32

- USART1 基地址：0x4001CC00
- USART2 基地址：0x4001D000
- USART3 基地址：0x4001D400
- USART4 基地址：0x4001D800
- USART5 基地址：0x4001DC00
- USART6 基地址：0x40020C00
- USART7 基地址：0x40021000
- USART8 基地址：0x40021400
- USART9 基地址：0x40021800
- USART10 基地址：0x40021C00

表 29-12 通用同步异步收发器 (USART) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	USART_SR	状态寄存器	32
0x04	USART_TDR	发送数据寄存器	16
0x06	USART_RDR	接收数据寄存器	16
0x08	USART_BRR	波特率寄存器	32
0x0C	USART_CR1	控制寄存器 1	32
0x10	USART_CR2	控制寄存器 2	32
0x14	USART_CR3	控制寄存器 3	32
0x18	USART_PR	预分频寄存器	32
0x1C	USART_LBMC	LIN 波特率测量计数寄存器	32

29.4.2 状态寄存器 (USART_SR)

Offset	Bit Position																																
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x000000C0																																
Name	Reserved																MPB	Reserved				TEND	LBD	WKUP	RTOF	TXE	TC	RXNE	BE	ORE	Reserved	FE	PE
Access																	RO					RO	RO	RO	RO	RO	RO	RO	RO	RO		RO	RO

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	MPB	多处理器位标志	0b0: 当前接收数据为通信数据 0b1: 当前接收数据为 ID  说明 MPB 位只在多处理器模式时有效。	RO
15:12	Reserved	保留	-	-
11	TEND	发送完成标志	0b0: 发送数据中 0b1: 发送数据完成 UART 模式，多处理器模式，LIN 模式，时钟同步模式 - TEND 置位条件： - 送出一帧数据的最后一位时，发送数据寄存器的值没有被更新。 - TEND 清零条件： - TE=1 时，向发送数据寄存器写入发送数据； - 清零寄存器 CR1.CTEND 位写入 1。 智能卡模式 - TEND 置位条件： - 最后 1 字节的数据送出后经过特定时间后，FE=0 而且发送数据寄存器的值没有被更新。 - TEND 清零条件： - TE=1 时，向发送数据寄存器写入发送数据； - 清零寄存器 CR1.CTEND 位写入 1。	RO
10	LBD	LIN 间隔段检出标志	仅 USART5/USART10 有效： 0b0: 未检测到间隔段 0b1: 检测到间隔段 LBD 置位条件： - CR2.WKUPE=0 时，检测到 RX 线低电平宽度上大于等于 CR2.LBDL 寄存器设定的宽度后，并检测到 break delimiter 时。 清零条件： - 清零寄存器 CR1.CLBD 位写入 1。	RO

位/位域	标记	位名	功能描述	读写
10	LBD	LIN 间隔段检出标志	 说明 RE=0 并不能复位 LBD 位。	RO
9	WKUP	LIN Wakeup 信号检出标志	仅 USART5/USART10 有效： ● 0b0：未检测到 Wakeup 信号 ● 0b1：检测到 Wakeup 信号 WKUP 置位条件： ● CR2.WKUPE=1 时，检测到 RX 线低电平宽度上大于等于 2.5 位数据宽度（19.2kbps 时为 130μs）。 清零条件： ● 清零寄存器 CR1.CWKUP 位写入 1。  说明 RE=0 并不能复位 WKUP 位。 检测到 WKUP 信号为 1 后，表示本 LIN 节点已经从休眠模式退出，设定寄存器 CR2.WKUPE=0 后，进行后续间隔场检测等操作。	RO
8	RTOF	UART 接收 TIMEOUT 标志位	● 0b0：无 UART 接收 TIMEOUT ● 0b1：发生 UART 接收 TIMEOUT RTOF 置位条件： ● 从检测到最后一帧数据的 STOP 位起经过设定时间后没有检测到新的接收数据。 RTOF 清零条件： ● 清零寄存器 CR1.CRTOF 位写入 1。  说明 RTOF 为由硬件置 1，且只有在 CR1.RE=1 且 CR1.RTOE=1 时置 1。CR1.RE=0 时，TIMEOUT 功能有效，但 RTOF 不置 1。	RO
7	TXE	发送数据寄存器空标志	● 0b0：数据未传输到移位寄存器，发送数据寄存器空 ● 0b1：数据传输到移位寄存器，发送数据寄存器空 TXE 位由硬件置 1 和清 0，数据未传输到移位寄存器时硬件将 TXE 清 0，数据传输到移位寄存器时硬件将 TXE 置 1。  说明 TXE 位在 UART/时钟同步模式/LIN 模式时有效。	RO
6	TC	发送完成标志	● 0b0：发送数据中 ● 0b1：发送数据完成 UART 模式，多处理器模式，LIN 模式，时钟同步模式 ● TC 置位条件： ► TE=0 发送禁止时； ► 送出一帧数据的最后一位时，发送数据寄存器的值没有被更新。 ● TC 清零条件： ► TE=1 时，向发送数据寄存器写入发送数据 智能卡模式	RO

位/位域	标记	位名	功能描述	读写
6	TC	发送完成标志	● TC 置位条件： ▶ TE=0 发送禁止时； ▶ 最后 1 字节的数据送出后经过特定时间后，FE=0 而且发送数据寄存器的值没有被更新。 - TC 置位的具体时序为：校验位送出后经过 2.5 位时间。 ● TC 清零条件： ▶ TE=1 时，向发送数据寄存器写入发送数据  说明 TE 位由 0 变化为 1 时 TC 为保持为 1。	RO
5	RXNE	接收数据寄存器不为空标志	● 0b0：未接收到数据 ● 0b1：准备好读取接收到的数据 RXNE 位由硬件置 1 和清 0，准备好读取接收到的数据时硬件将 RXNE 置 1，读取接收数据后硬件将 RXNE 清 0。	RO
4	BE	LIN 总线检错误标志	仅 USART5/USART10 有效： ● 0b0：无总线错误 ● 0b1：发生总线错误 BE 置位条件： ● 发送数据时，检测到总线数据和发送数据不一致，检测位置为每一位的 13/16 处（16 位过采样）。 BE 清零条件： ● 清零寄存器 CR1.CBE 位写入 1。  说明 TE=0 并不能复位 BE 位。 BE=1 时，正在发送的数据发送完成后，不会开始新的数据发送，需要软件清除 BE 标志，重启发送动作。	RO
3	ORE	接收上溢错误标志	● 0b0：无接收上溢错误 ● 0b1：发生接收上溢错误 ORE 置位条件： ● 接收数据寄存器未被读取的情况下，又接收到一帧新的数据。 ORE 清零条件： ● 清零寄存器 CR1.CORE 位写入 1。  说明 RE=0 并不能复位 ORE 位。 ORE=1 之前接收到的数据会被保持，ORE=1 时接收到的数据会丢掉。 ORE=1 后不能继续接收数据，时钟同步模式下也不能发送数据。	RO
2	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
1	FE	接收帧错误标志	● 0b0: 无接收帧错误 ● 0b1: 发生接收帧错误 UART 模式, 多处理器模式, LIN 模式 ● FE 置位条件: ▶ 接收的数据帧的停止位为低电平, 两个停止位的情况只检查第一个停止位。 ● FE 清零条件: ▶ 清零寄存器 CR1.CFE 位写入 1。  说明 RE=0 并不能复位 FE 位。 FE=1 时收到的数据会保留但是 RI 中断不会发生, FE=1 后不能继续接收数据。 在 LIN 模式接收到间隔段时, 会产生该标志位, 不代表帧错误; 在接收到同步段时, 若波特率误差较大, 也会产生该标志位。 智能卡模式 ● FE 置位条件: ▶ 采样到低电平错误信号标志。 ● FE 清零条件: ▶ 清零寄存器 CR1.CFE 位写入 1。  说明 RE=0 并不能复位 FE 位。	RO
0	PE	接收数据校验错误标志	● 0b0: 无接收数据校验错误 ● 0b1: 发生接收数据校验错误 PE 置位条件: ● 接收数据发生奇偶校验错误时。 PE 清零条件: ● 清零寄存器 CR1.CPE 位写入 1。  说明 RE=0 并不能复位 PE 位。 PE=1 时收到的数据会保留但是 RI 中断不会发生, PE=1 后不能继续接收数据。	RO

29.4.3 发送数据寄存器 (USART_TDR)

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x01FF															
Name	Reserved						MPID	TDR								
Access							RW	RW								

位/位域	标记	位名	功能描述	读写
15:10	Reserved	保留	-	-
9	MPID	多处理器模式 ID 位	多处理器模式时，发送通信数据或者发送 ID 的选择位： ● 0b0：发送数据 ● 0b1：发送 ID  说明 MPID 位只在多处理器模式时有效，其他模式须设定为复位值。	RW
8:0	TDR	发送数据	发送数据寄存器  说明 最高位 TDR[8]仅在 UART 模式而且数据长度设定为 9 位时有效。	RW

29.4.4 接收数据寄存器 (USART_RDR)

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved						RDR									
Access							RO									

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8:0	RDR	接收数据	接收数据寄存器  说明 最高位 RDR[8]仅在 UART 模式而且数据长度设定为 9 位时有效。	RO

29.4.5 波特率寄存器 (USART_BRR)

Offset	Bit Position																																				
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x0000FFFF																																				
Name	Reserved																DIV_Integer				Reserved	DIV_Fraction															
Access																	RW					RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	DIV_Integer	整数分频	整数分频寄存器 说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
7	Reserved	保留	读出为 1，写入时必须写 1	-
6:0	DIV_Fraction	小数分频	小数分频寄存器 说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定，而且仅在 FBME=1 时设定值有效。	RW

表 29-21 波特率计算公式（小数波特率无效 FBME=0）

模式	波特率计算公式	误差 E 计算公式
UART 模式/多处理器模式	$B = \frac{C}{8 \times (2 - \text{OVER8}) \times (\text{DIV\_Integer} + 1)}$	$E = \left\{ \frac{C}{8 \times (2 - \text{OVER8}) \times (\text{DIV\_Integer} + 1) \times B} - 1 \right\} \times 100\%$
时钟同步模式	$B = \frac{C}{4 \times (\text{DIV\_Integer} + 1)}$	-
智能卡模式	$B = \frac{C}{2 \times \text{BCN} \times (\text{DIV\_Integer} + 1)}$	$E = \left\{ \frac{C}{2 \times \text{BCN} \times (\text{DIV\_Integer} + 1) \times B} - 1 \right\} \times 100\%$

表 29-22 波特率计算公式（小数波特率有效 FBME=1）

模式	波特率计算公式	误差 E (%) 计算公式
UART 模式/多处理器模式	$B = \frac{C \times (128 + \text{DIV\_Fraction})}{8 \times (2 - \text{OVER8}) \times (\text{DIV\_Integer} + 1) \times 256}$	$E = \left\{ \frac{C \times (128 + \text{DIV\_Fraction})}{8 \times (2 - \text{OVER8}) \times (\text{DIV\_Integer} + 1) \times 256 \times B} - 1 \right\} \times 100\%$
时钟同步模式	$B = \frac{C \times (128 + \text{DIV\_Fraction})}{4 \times (\text{DIV\_Integer} + 1) \times 256}$	-
智能卡模式	$B = \frac{C \times (128 + \text{DIV\_Fraction})}{2 \times \text{BCN} \times (\text{DIV\_Integer} + 1) \times 256}$	$E = \left\{ \frac{C \times (128 + \text{DIV\_Fraction})}{2 \times \text{BCN} \times (\text{DIV\_Integer} + 1) \times 256 \times B} - 1 \right\} \times 100\%$

其中：

- B：波特率（Mbps）
- C：PR.ULBREN 与 PR.PSC[1:0]位设定的时钟（MHz）
- BCN：CR3.BCN 寄存器设定值

29.4.6 控制寄存器 1 (USART_CR1)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80000000																															
Name	SBS	NFE	FBME	ML	Reserved	CTEND	MS	CLBD	CWKUP	CBE	CRTOF	CORE	Reserved	CFE	CPE	OVER8	Reserved	M	Reserved	PCE	PS	TENDIE	TXEIE	TCIE	RIE	SLME	TE	RE	RTOIE	RTOE		
Access	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW		RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	SBS	UART 模式接收数据开始位检测方式设定位	UART 模式接收数据时，开始位检测方式设定位： 0b0：开始位检测方式为 RX 管脚低电平 0b1：开始位检测方式为 RX 管脚下降沿  说明 非 UART 模式时该位须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
30	NFE	数字滤波使能位	0b0：禁止 0b1：使能  说明 时钟同步模式、LIN 模式检测同步帧时该位须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
29	FBME	小数波特率功能使能	0b0：禁止 0b1：使能  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
28	ML	MSB/LSB 选择位	UART 模式/时钟同步模式/智能卡模式时，MSB/LSB 方式选择位： 0b0：LSB 方式 0b1：MSB 方式  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
27:26	Reserved	保留	-	-
25	CTEND	TEND 清零位	写 1 清除 TEND 标志，写 0 无效；读出恒为 0	RW
24	MS	通信模式选择位	0b0：UART 模式 0b1：时钟同步模式  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定，智能卡模式 MS 需写入复位值。	RW
23	CLBD	LBD 清零位	仅 USART5/USART10 有效： 写 1 清除 LBD 标志，写 0 无效；读出恒为 0	RW

位/位域	标记	位名	功能描述	读写
22	CWKUP	WKUP 清零位	仅 USART5/USART10 有效： 写 1 清除 WKUP 标志，写 0 无效；读出恒为 0	RW
21	CBE	BE 清零位	仅 USART5/USART10 有效： 写 1 清除 BE 标志，写 0 无效；读出恒为 0	RW
20	CRTOF	RTOF 清零位	写 1 清除 RTOF 标志，写 0 无效；读出恒为 0	RW
19	CORE	ORE 标志清零位	写 1 清除 ORE 标志，写 0 无效；读出恒为 0	RW
18	Reserved	保留	-	-
17	CFE	FE 标志清零位	写 1 清除 FE 标志，写 0 无效；读出恒为 0	RW
16	CPE	PE 标志清零位	写 1 清除 PE 标志，写 0 无效；读出恒为 0	RW
15	OVER8	UART 过采样模式	UART 过采样模式设定，即一位数据传输期间的基本时钟数： ● 0b0：16 位 ● 0b1：8 位  说明 智能卡模式、时钟同步模式时该位须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
14:13	Reserved	保留	-	-
12	M	数据长度设定位	UART 模式时，发送/接收数据长度设定位： ● 0b0：8 位 ● 0b1：9 位  说明 LIN 模式、智能卡模式、时钟同步模式时该位须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
11	Reserved	保留	-	-
10	PCE	校验使能位	UART 模式时，奇偶校验使能位： ● 0b0：无校验 ● 0b1：校验  说明 智能卡模式时该位必须为 1，LIN 模式、多处理器模式、时钟同步模式时该位必须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
9	PS	校验位	UART 模式时，奇偶校验选择位： ● 0b0：偶校验 ● 0b1：奇校验  说明 该位只在 PCE=1 时有效。 智能卡模式时该位必须保持为 0。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
8	TENDIE	发送完成中断使能位	● 0b0：TENDI 中断请求无效，TENDI 中断不发生 ● 0b1：TENDI 中断请求使能，TENDI 中断发生	RW

位/位域	标记	位名	功能描述	读写
7	TXEIE	发送数据寄存器空中断使能位	0b0: TI 中断请求无效, TI 中断不发生 0b1: TI 中断请求有效, TI 中断发生  说明 TE=0 时写 TXEIE=1, 则需要等待 TE=1 时发生 TI 中断。 TE=1 时写 TXEIE=1, 则需要等待 SR.TC=1 时才能写入。	RW
6	TCIE	发送完成中断使能位	0b0: TCI 中断请求无效, TCI 中断不发生 0b1: TCI 中断请求使能, TCI 中断发生  说明 非智能卡模式读取有效, 智能卡模式读取为 0。	RW
5	RIE	接收中断、错误中断使能位	0b0: 接收中断、错误中断请求无效, RI 和 EI 中断不发生 0b1: 接收中断、错误中断请求有效, RI 和 EI 中断发生	RW
4	SLME	静默模式使能位	多处理器操作时, 静默模式使能位: 0b0: 正常模式 0b1: 静默模式 SLME=1 时, MPB 位为 0 的通信数据不会从接收移位寄存器读取到接收数据寄存器, 同时错误标志 ORE 和 FE 位也不置位。接收到 MPB 为 1 的 ID 数据时, SLME 自动清零, 开始正常的数据接收动作。  说明 该位只在 UART 多处理器模式时有效, 其他模式时该位必须保持复位值。	RW
3	TE	发送器使能位	0b0: 发送器禁止 0b1: 发送器使能  说明 时钟同步模式时该位只能在 TE=0&RE=0 (发送/接收禁止) 时写 1。	RW
2	RE	接收器使能位	0b0: 接收器禁止 0b1: 接收器使能  说明 时钟同步模式时该位只能在 TE=0&RE=0 (发送/接收禁止) 时写 1。	RW
1	RTOIE	UART TIMEOUT 中断使能位	0b0: UART TIMEOUT 中断请求无效, RTOI 中断不发生 0b1: UART TIMEOUT 中断请求有效, RTOI 中断发生	RW
0	RTOE	UART TIMEOUT 功能使能位	0b0: UART TIMEOUT 功能禁止 0b1: UART TIMEOUT 功能使能	RW

29.4.7 控制寄存器 2 (USART_CR2)

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000600																															
Name	Reserved														SBKM	SBK	Reserved	LINEN	STOP	CLKC	Reserved	WKUPE	SBKL	LBDL	LBDIE	BEE	BEIE	WKUPIE	MPE			
Access															RW	RW		RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW		

位/位域	标记	位名	功能描述	读写															
31:18	Reserved	保留	-	-															
17	SBKM	间隔段发送模式	仅 USART5/USART10 有效： 具体设置请参见 SBK 位描述。  说明 该位只在 LINEN=1 时有效。	RW															
16	SBK	间隔段发送使能位	仅 USART5/USART10 有效： <table><tr><th>SBK</th><th>SBKM</th><th>间隔段发送</th></tr><tr><td>0b0</td><td>0b0</td><td>不发送间隔段</td></tr><tr><td>0b0</td><td>0b1</td><td>不发送间隔段</td></tr><tr><td>0b1</td><td>0b0</td><td>SBK 写 1 时，自动发送间隔段，发送完成后，SBK 自动清零</td></tr><tr><td>0b1</td><td>0b1</td><td>写数据 0x00 到 USART_TDR.TDR，开始发送间隔段，发送完成后，SBK 自动清零</td></tr></table>  说明 该位只在 LINEN=1 时有效。	SBK	SBKM	间隔段发送	0b0	0b0	不发送间隔段	0b0	0b1	不发送间隔段	0b1	0b0	SBK 写 1 时，自动发送间隔段，发送完成后，SBK 自动清零	0b1	0b1	写数据 0x00 到 USART_TDR.TDR，开始发送间隔段，发送完成后，SBK 自动清零	RW
SBK	SBKM	间隔段发送																	
0b0	0b0	不发送间隔段																	
0b0	0b1	不发送间隔段																	
0b1	0b0	SBK 写 1 时，自动发送间隔段，发送完成后，SBK 自动清零																	
0b1	0b1	写数据 0x00 到 USART_TDR.TDR，开始发送间隔段，发送完成后，SBK 自动清零																	
15	Reserved	保留	-	-															
14	LINEN	LIN 功能使能	LIN 功能使能位，仅 USART5/USART10 有效： ● 0b0：禁止 LIN 功能 ● 0b1：使能 LIN 功能  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW															
13	STOP	停止位设定位	UART 模式时，停止位长度设定位： ● 0b0：1 停止位 ● 0b1：2 停止位  说明 LIN 模式、智能卡模式、时钟同步模式时该位须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW															

位/位域	标记	位名	功能描述	读写
12:11	CLKC	时钟控制位	UART 模式 0b00: 时钟源为内部波特率生成器生成的时钟, 时钟不输出到 USARTn_CK 管脚, USARTn_CK 管脚可以当作普通 IO 使用 0b01: 时钟源为内部波特率生成器生成的时钟, 时钟输出到 USARTn_CK 管脚, 输出时钟频率和波特率相同 0b10/0b11: 时钟源为外部输入时钟, 输入时钟的频率为波特率的 16 倍 (OVER8=0) 或者 8 倍 (OVER8=1) 时钟同步模式 0b00/0b01: 时钟源为内部波特率生成器生成的时钟, 输出到 USARTn_CK 管脚 0b10/0b11: 时钟源为外部输入时钟, 输入时钟的频率和波特率相同 智能卡模式 0b00: 时钟源为内部波特率生成器生成的时钟, 时钟不输出到 USARTn_CK 管脚, USARTn_CK 管脚可以当作普通 IO 使用 0b01: 时钟源为内部波特率生成器生成的时钟, 时钟输出到 USARTn_CK 管脚 0b10/0b11: 禁止设置  说明 该位只能在 TE=0&RE=0 (发送/接收禁止) 时设定。	RW
10:9	Reserved	保留	所有位读出的 1, 写入时必须写 1	-
8	WKUPE	LIN 唤醒信号检测使能位	仅 USART5/USART10 有效: 0b0: 禁止唤醒信号检测 0b1: 使能唤醒信号检测  说明 仅在系统需要唤醒信号检测时, 设置 WKUPE 为 1, 检测到唤醒信号后, 需要软件清零 WKUPE。	RW
7:6	SBKL	LIN 间隔段低电平宽度设定位	仅 USART5/USART10 有效: 0b00: 10 位 0b01: 11 位 0b10: 13 位 0b11: 14 位  说明 非 UART-LIN 模式时该位必须保持复位值。 该位只能在 TE=0&RE=0 (发送/接收禁止) 时设定。	RW
5	LBDL	LIN 间隔段低电平宽度检测阈值设定位	仅 USART5/USART10 有效: 0b0: ≥ 10 位 0b1: ≥ 11 位  说明 非 UART-LIN 模式时该位必须保持复位值。 该位只能在 TE=0&RE=0 (发送/接收禁止) 时设定。	RW

位/位域	标记	位名	功能描述	读写
4	LBDIE	LIN 间隔段检测中断使能位	仅 USART5/USART10 有效： ● 0b0：LIN 间隔段检测中断请求无效，不发生中断 ● 0b1：LIN 间隔段检测中断请求有效，发生中断  说明 非 UART-LIN 模式时该位必须保持复位值。	RW
3	BEE	LIN 总线错误检测功能使能位	仅 USART5/USART10 有效： ● 0b0：禁止 LIN 总线错误功能检测功能 ● 0b1：使能 LIN 总线错误功能检测功能  说明 非 UART-LIN 模式时该位必须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
2	BEIE	LIN 总线错误中断使能位	仅 USART5/USART10 有效： ● 0b0：LIN 总线错误中断请求无效，不发生中断 ● 0b1：LIN 总线错误中断请求有效，发生中断  说明 非 UART-LIN 模式时该位必须保持复位值。	RW
1	WKUPIE	LIN 唤醒信号检测中断使能位	仅 USART5/USART10 有效： ● 0b0：LIN 唤醒信号检测中断请求无效，不发生中断 ● 0b1：LIN 唤醒信号检测中断请求有效，发生中断  说明 非 UART-LIN 模式时该位必须保持复位值。	RW
0	MPE	多处理器功能使能位	UART 模式时，多处理器功能使能位： ● 0b0：禁止 ● 0b1：使能  说明 非 UART 模式时该位必须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW

29.4.8 控制寄存器 3 (USART_CR3)

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DEP	DEM	Reserved	DEA				Reserved	BCN				Reserved				DED				CTSE	RTSE	Reserved	SCEN	LOOP	HDSEL	Reserved					
Access	RW	RW		RW					RW								RW				RW	RW		RW	RW	RW						

位/位域	标记	位名	功能描述	读写
31	DEP	DE 功能极性	Driver Enable 极性模式： 0b0：USART 在 IDLE 时 DE 信号为高 0b1：USART 在 IDLE 时 DE 信号为低 启动 USART TE 或者 RE 时，无法写入此位字段。	RW
30	DEM	DE 功能使能	启用 Driver Enable 模式，该位用于通过 DE 信号激活外部收发器控制： 0b0：DE 功能被禁用 0b1：DE 功能启用 启用 USART TE 或者 RE 时，禁止写入此位字段。	RW
29	Reserved	保留	-	-
28:25	DEA	DE 断言启用时间	这些位用于定义 DE（驱动器启用）信号激活与起始位开始之间的时间，它以采样时间单位（1/8 或者 1/16 位时间）表示，由 OVER8 位配置：为 1 时最大可配置为 0x7，为 0 时最大可配置为 0xF。 最低配置为 0。配置为 0 时，DE 信号与开始位同时发生；配置为 0x7 或者 0xF 时，DE 信号发生在开始位前 7 或者 15 个时间单位。 启用 USART TE 或者 RE 时，禁止写入此位字段。	RW
24	Reserved	保留	-	-
23:21	BCN	基本时钟数	智能卡模式时，设定一位数据传输期间的基本时钟数： 0b000：32 0b001：64 0b010：禁止设置 0b011：128 0b100：禁止设置 0b101：256 0b110：372 0b111：禁止设置  说明 非智能卡模式时该位必须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。 USART5/USART10 不支持该功能，读出为 0。	RW
20:14	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
13:10	DED	DE 断言取消时间	这些位用于定义传输消息中最后一个停止位的结束与 DE（驱动器启用）信号的停用之间。它以采样时间单位（1/8 或 1/16 位时间）表示，由 OVER8 位配置：为 1 时最大可配置为 0x7，为 0 时最大可配置为 0xF。 最小可配置为 0。配置为 0 时，DE 信号变换发生在结束位完成后的 2 个时间单位后；配置为 0x7 或者 0xF 时，DE 信号发生在结束位完成后的 9 或者 17 个时间单位。 启用 USART TE 或者 RE 时，禁止写入此位字段。	RW
9	CTSE	CTS 功能使能位	● 0b0: CTS 功能无效 ● 0b1: CTS 功能有效  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
8	RTSE	RTS 功能使能位	● 0b0: RTS 功能无效 ● 0b1: RTS 功能有效  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
7:6	Reserved	保留	-	-
5	SCEN	智能卡模式使能位	● 0b0: 禁止智能卡模式 ● 0b1: 使能智能卡模式  说明 非智能卡模式时该位必须保持复位值。 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。 USART5/USART10 不支持该功能，读出为 0。	RW
4	LOOP	LIN 回环模式使能位	仅 USART5/USART10 有效： ● 0b0: 正常模式 ● 0b1: 回环模式  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
3	HDSEL	UART 单线半双工模式使能位	● 0b0: UART 全双工模式 ● 0b1: UART 半双工模式  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
2:0	Reserved	保留	-	-

29.4.9 预分频寄存器 (USART_PR)

Offset	Bit Position																																	
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																												ULBREN		LBMPSC		PSC	
Access																													RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4	ULBREN	超低波特率使能位	内部时钟源时，超低波特率功能使能位： ● 0b0：超低波特率功能无效 ● 0b1：超低波特率功能有效 且根据 PSC[1:0]的值预分频器分频值为： ● 0b00：PCLK1/128 ● 0b01：PCLK1/256 ● 0b10：PCLK1/512 ● 0b11：PCLK1/1024  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
3:2	LBMPSC	LIN 波特率测量计数器时钟源选择	仅 USART5/USART10 有效： ● 0b00：PCLK1 ● 0b01：PCLK1/2 ● 0b10：PCLK1/4 ● 0b11：PCLK1/8  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW
1:0	PSC	预分频器值	内部时钟源时，预分频器分频值选择位： ● 0b00：PCLK1 ● 0b01：PCLK1/4 ● 0b10：PCLK1/16 ● 0b11：PCLK1/64  说明 该位只能在 TE=0&RE=0（发送/接收禁止）时设定。	RW

29.4.10 LIN 波特率测量计数寄存器 (USART_LBMC)

Offset	Bit Position																															
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																LBMC															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	LBMC	LIN 波特率测量结果计数器	仅 USART5/USART10 有效。 LIN 从节点，用 USARTn_PR.LBMPSC 寄存器选择的时钟源去测量得到的主节点发送的同步段的频率。 USARTn_PR.LBMPSC 选择的计数时钟频率除以 LBMC 的值，即可得到测量到的 LIN 通信波特率。  说明 LBMC 的值在同步段接收完成后读取才有意义。	RO

29.4.11 USART1 滤波控制寄存器 (USART1_NFC)

Offset	Bit Position																															
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																												USART1_NFE		USART1_NFS	
Access																													RW		RW	

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	USART1_NFE	USART1 滤波使能控制	用于控制 STOP 模式下，RX 线上的模拟滤波器的开关： ● 0b0：模拟滤波器禁止 ● 0b1：模拟滤波器使能，滤波范围请参考 USART1_NFS[1:0]的设置	RW
1:0	USART1_NFS	USART1 滤波范围设置	用于控制 STOP 模式下，RX 线上的模拟滤波器的滤波范围： ● 0b00：滤波宽度档位 1 ● 0b01：滤波宽度档位 2 ● 0b10：滤波宽度档位 3 ● 0b11：滤波宽度档位 4 各档位具体值请参见《数据手册》“电气特性-工作条件-USART1 STOP 模式下 RX 滤波特性”章节相关描述。	RW

29.5 注意事项

29.5.1 UART 注意事项

发送器

UART 模式发送器发送禁止时 (USARTn_CR1.TE=0)，则 TX 管脚可以当作普通 IO 使用，可以设定输出值和方向。如果输出 0，则会使接收方产生帧错误，从而中断数据传输。如果输出 1，则使接收方检测不到开始位从而无法开始数据传输。

接收器

UART 模式产生帧错误时，可以软件检测后续 RX 线是否为低电平，从而判断发送方是否想中断传输。

如果接收数据开始位检测方式为低电平检测，则在清除错误标志后继续接收全为低电平数据，接收错误会再次发生。

29.5.2 时钟同步模式注意事项

1. 使用外部输入时钟发送数据时，USARTn_TDR.TDR 的更新需要在时钟输入之前完成，写入数据后，至少需要等待一位数据时间再输入时钟。
2. 连续发送数据时，下一帧数据需要在当前帧最后一位发送前完成更新。

29.5.3 其他注意事项

为了防止发送禁止时 TX 通信线 Hi-Z 状态，可以采用以下方法：

- 通信线上拉；
- 发送数据结束时，USARTn_CR1.TE=0 之前，将 TX 管脚设为普通 IO 输出；
- 发送数据开始前，USARTn_CR1.TE=1 之后，将 IO 设为 TX 功能。

30 集成电路总线 (I2C)

30.1 简介

I2C (集成电路总线) 用作微控制器和 I2C 串行总线之间的接口。提供多主模式功能, 可以控制所有 I2C 总线的协议、仲裁。支持标准模式、快速模式。还支持 SMBus 总线。

30.2 主要特性

- I2C 总线方式、SMBUS 总线方式可选。主机模式、从机模式可选。自动确保与传送速率相对于的各种准备时间、保持时间和总线空闲时间
- 标准模式最大 100kbps, 快速模式最大 400kbps
- 自动生成开始条件、重新开始条件和停止条件, 并能检测到总线的开始条件, 重新开始条件和停止条件
- 最大支持 128 个从机模式地址。支持 7 位地址格式和 10 位地址格式。能检测到广播呼叫地址、SMBus 主机地址、SMBus 设备默认地址、SMBus 报警地址
- 发送时可以自动判定应答位。接收时可以自动发送应答位
- 握手功能
- 仲裁功能
- 超时功能, 可以检测 SCL 时钟长时间停止
- SCL 输入和 SDA 输入内置数字滤波器, 滤波能力可编程
- 通信错误, 接收数据满, 发送数据空, 一帧发送结束, 地址匹配一致中断
- 2 级发送 FIFO 和 2 级接收 FIFO

30.3 功能说明

30.3.1 I2C 系统框图

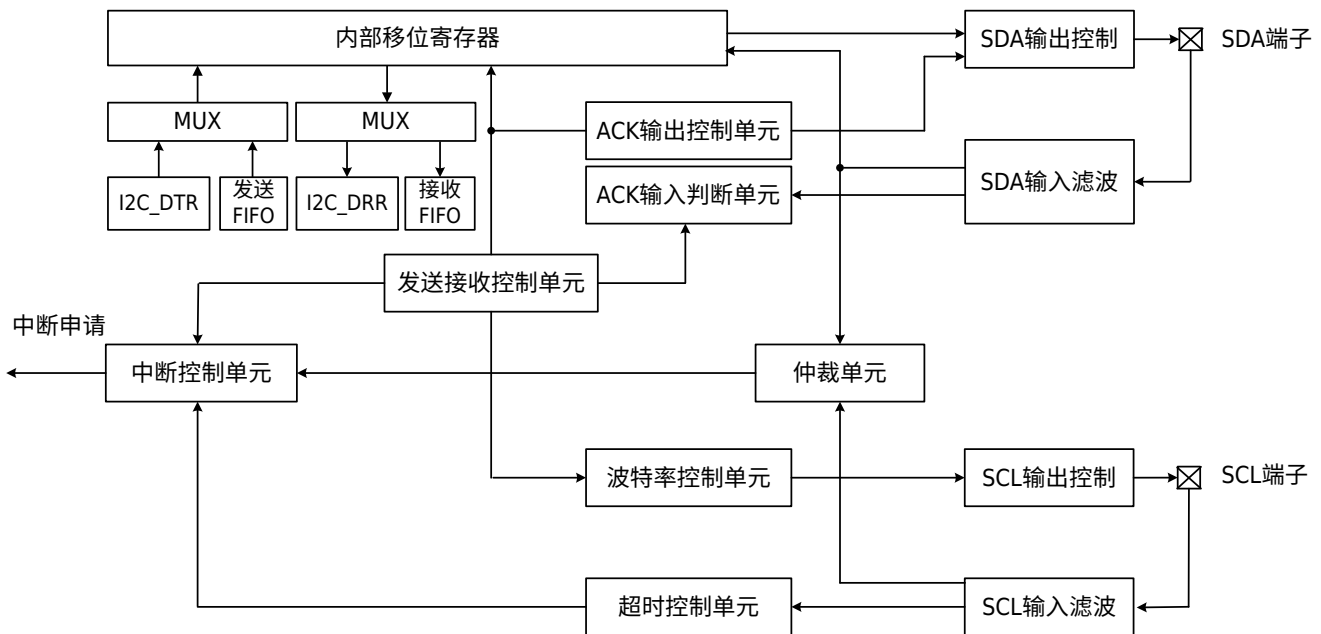


图 30-1 I2C 系统框图

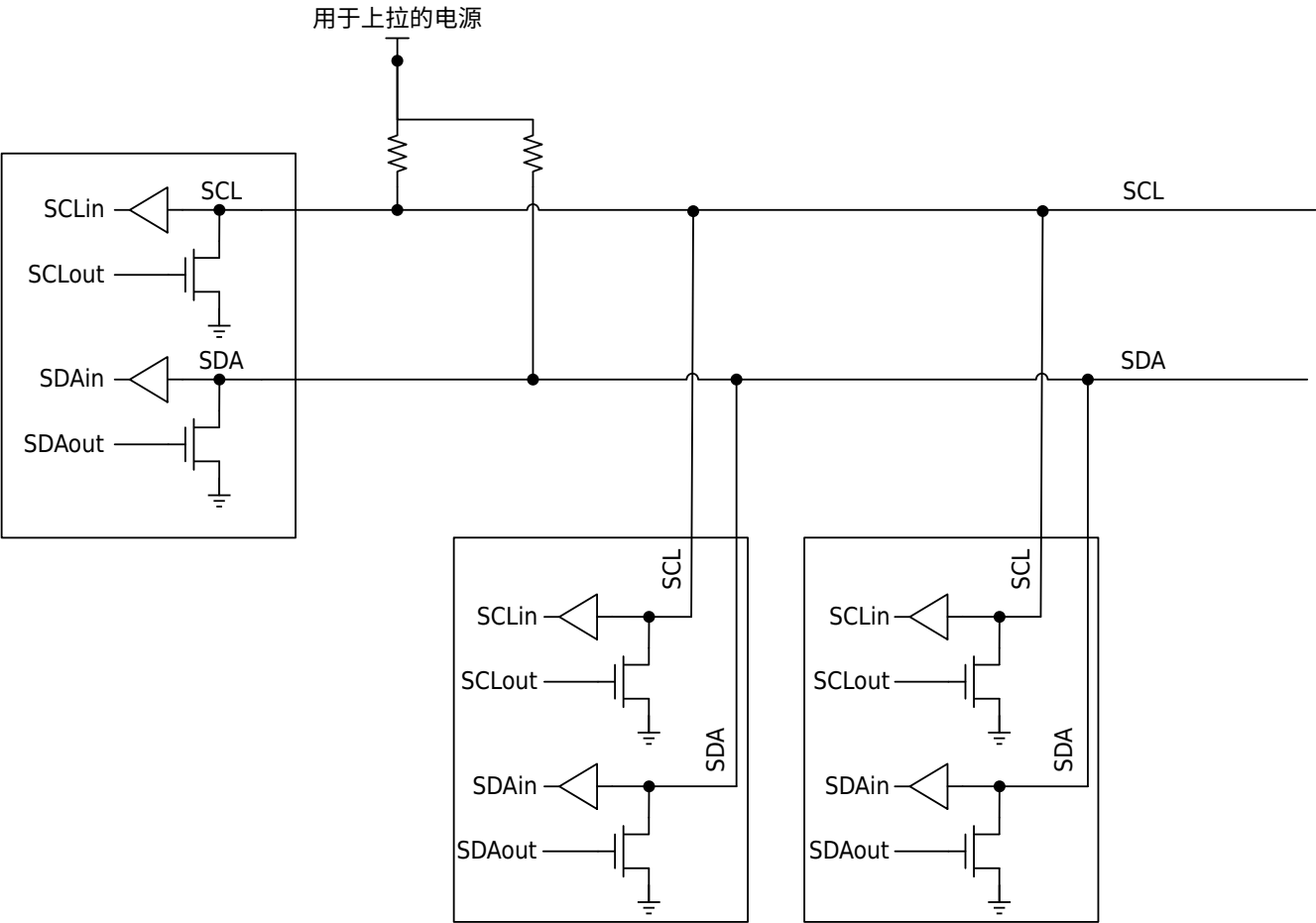


图 30-2 I2C 总线的结构例

表 30-1 输入/输出引脚

引脚名	输入/输出	功能
SCL	输入/输出	串行时钟的输入/输出引脚
SDA	输入/输出	串行数据的输入/输出引脚

当选择 I2C 总线时，SCL/SDA 输入电平为 Schmitt 电平（兼容 CMOS 电平）。当选择 SMBus 时，请通过设定《通用 IO（GPIO）》“寄存器-通用控制寄存器（PCRxy）”的 PCRxy.CINSEL，将 SCL/ SDA 输入电平设定为 CMOS 电平。

30.3.2 I2C 协议

30.3.2.1 数据传输

I2C 总线由一根时钟线（SCL），一根数据线（SDA）构成。所有的连接器件必须是漏极开路输出。SCL 和 SDA 线外接上拉电阻，电阻阻值取决于系统应用。

通常情况下，一个完整的通信过程包括下列 4 部分：

- 1. 开始条件
- 2. 地址传送
- 3. 数据传送
- 4. 停止条件

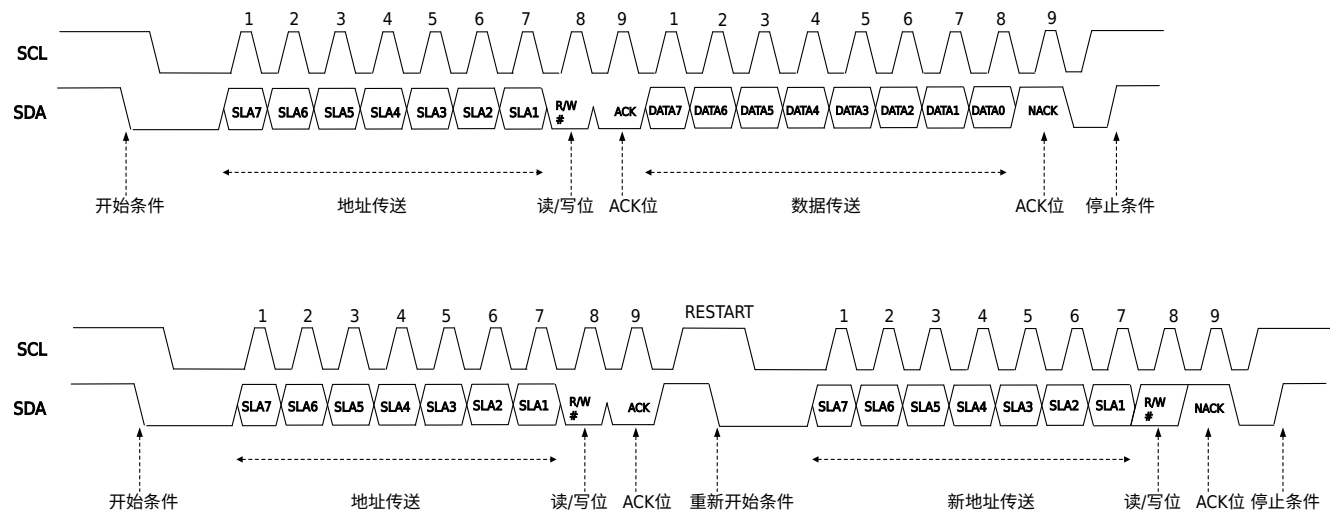


图 30-3 I2C 总线的时序图

开始条件

当总线上的主机都不驱动总线，总线进入空闲状态。SCL 和 SDA 都为高电平。总线空闲状态下总线上设备都可以通过发送开始条件启动通信。

在 I2C_SR.BUSY 标志为 0（总线空闲）的状态下如果将 START 位置 1，就发送开始条件。如果检测到开始条件，就自动将 I2C_SR.BUSY 标志和 I2C_SR.STARTF 标志置 1，并且自动将 START 位清 0。此时，如果在 START 位为 1 的状态下发送的 SDA 信号和 SDA 线的信号状态相同，并且检测到开始条件，就视为通过 START 位正确地发送了开始条件，再将 I2C_SR.MSL 位和 I2C_SR.TRA 位自动置 1 后变为主控发送模式。另外，I2C_SR.TEMPTYF 因 TRA 位为 1 而自动变为 1。接下来把从机地址写入 I2C_DTR 寄存器，发送地址。

地址传送

开始条件或者重新开始条件后面的帧是地址帧，用于指定主机通信的对象地址。在发送停止条件之前，指定的从机一直有效。

地址帧的高 7 位为从机地址。地址帧第 8 位来决定数据帧传送的方向。

1. 7 位寻址模式见下图[7 位地址格式]

- 主机发送模式，主机发送地址帧第 8 位为 0；
- 主机接收模式，主机发送地址帧第 8 位为 1。

2. 10 位寻址模式见下图[10 位地址格式]

- 主机发送模式，主机第一帧发送头序列（0b11110XX0，其中 XX 表示 10 位地址的高两位），然后第二帧发送低八位从机地址。
- 主机接收模式，主机第一帧发送头序列（0b11110XX0，其中 XX 表示 10 位地址的高两位），然后第二帧发送低八位从机地址。接下来会发送一个重新开始条件，然后再发送一帧头序列（0b11110XX1，其中 XX 表示 10 位地址的高两位）。

7位地址格式

S	SLA（7位）	R/W #	ACK/ NACK	DATA（8位）	ACK	DATA（8位）	ACK/ NACK	P

10位地址格式

S	0b11110+SL A（2位）	W#	ACK	SLA（8位）	ACK	DATA（8位）	ACK	DATA（8位）	ACK/ NACK	P

S	0b11110+SL A（2位）	W#	ACK	SLA（8位）	ACK	Sr	0b11110+SL A（2位）	R	ACK	DATA（8位）	ACK	DATA（8位）	ACK/ NACK	P

图 30-4 I2C 总线的数据格式

- S：表示开始条件
- SLA：表示从机地址
- RW#：表示发送和接收的方向。当 RW#为 1 时，将数据从从机发送到主机；当 RW#为 0 时，将数据从主机发送到从机
- Sr：表示重新开始条件
- DATA：表示发送和接收的数据
- P：表示停止条件

数据传送

地址匹配一致后，总线上的主机根据 RW 定义的方向一帧一帧的传送数据。

所有的地址帧后传送的数据都视为数据帧。即使是 10 位地址格式的低 8 位地址也视为数据帧。

数据帧的长度是 8 位。SCL 的低电平 SDA 变化，SCL 的高电平 SDA 保持，每个时钟周期发送一位数据。数据帧后的第 9 个时钟是应答位，是接收方向发送方传送的握手信号。

如果总线上从机接收数据，在第 9 个时钟周期不响应主机，从机必须发送 NACK。如果总线上主机接收数据，第 9 个周期发送 NACK，从机接收到 NACK，从机停止发送数据。

无论主机还是从机发送了 NACK，数据传送终止。主机可以做下列任一动作：

1. 发送停止条件释放总线；
2. 发送重新开始条件开始一个新的通信。

- 主机发送数据
- 在主机发送模式中，主机输出 SCL 时钟和发送数据，从机接收数据并返回应答。主机发送数据运行时序例如下图所示。

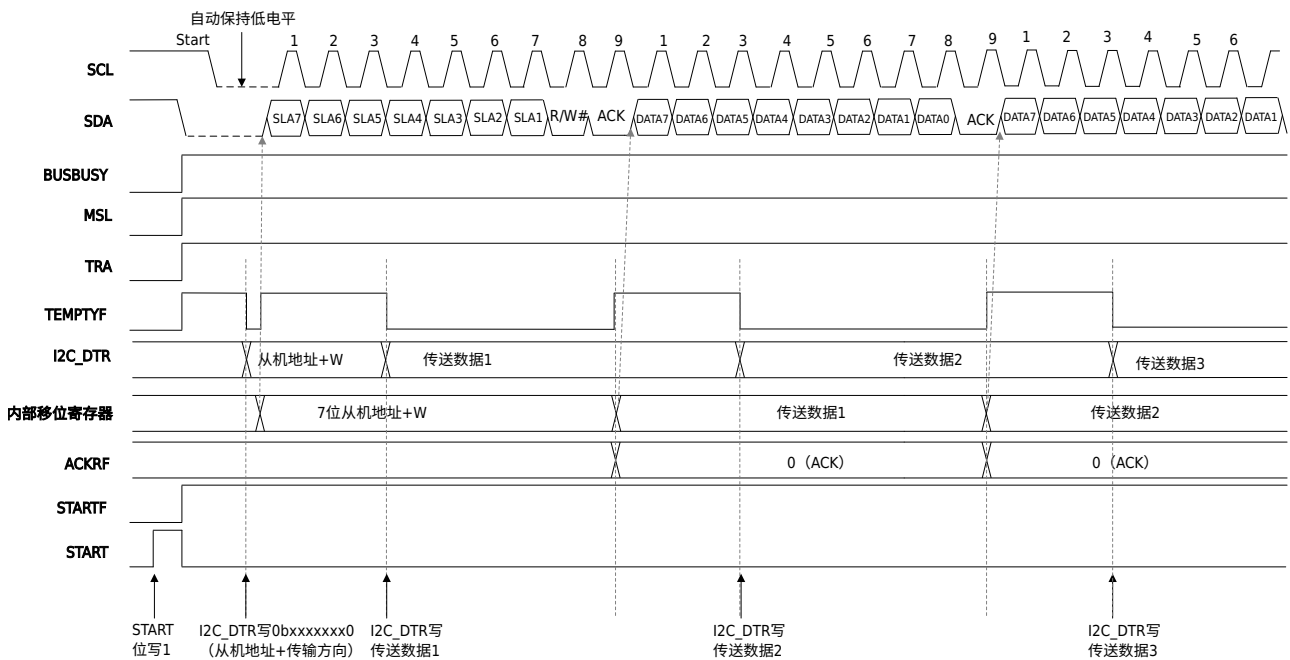


图 30-5 7 位地址格式的主机发送数据时序图（例）

- 主机接收数据
- 在主机接收模式中，主机输出 SCL 时钟，接收从机数据并返回应答。主机接收数据的运行时序例如下图所示。

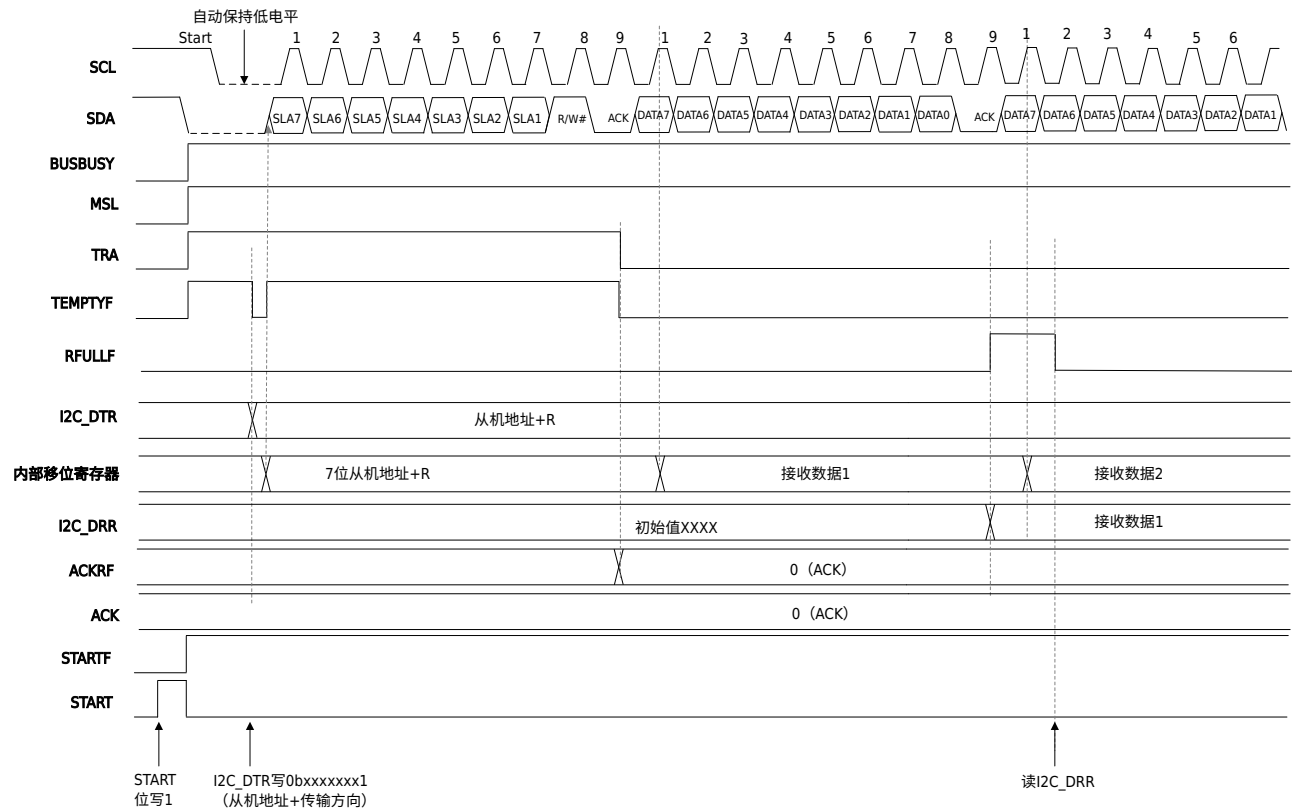


图 30-6 7 位地址格式的主机接收数据的时序图（例）

● 从机发送数据

在从机发送模式中，接收来自主机的 SCL 时钟，本产品为从机发送数据，并且接收主机返回应答。从机发送数据的运行时序例如下图所示。

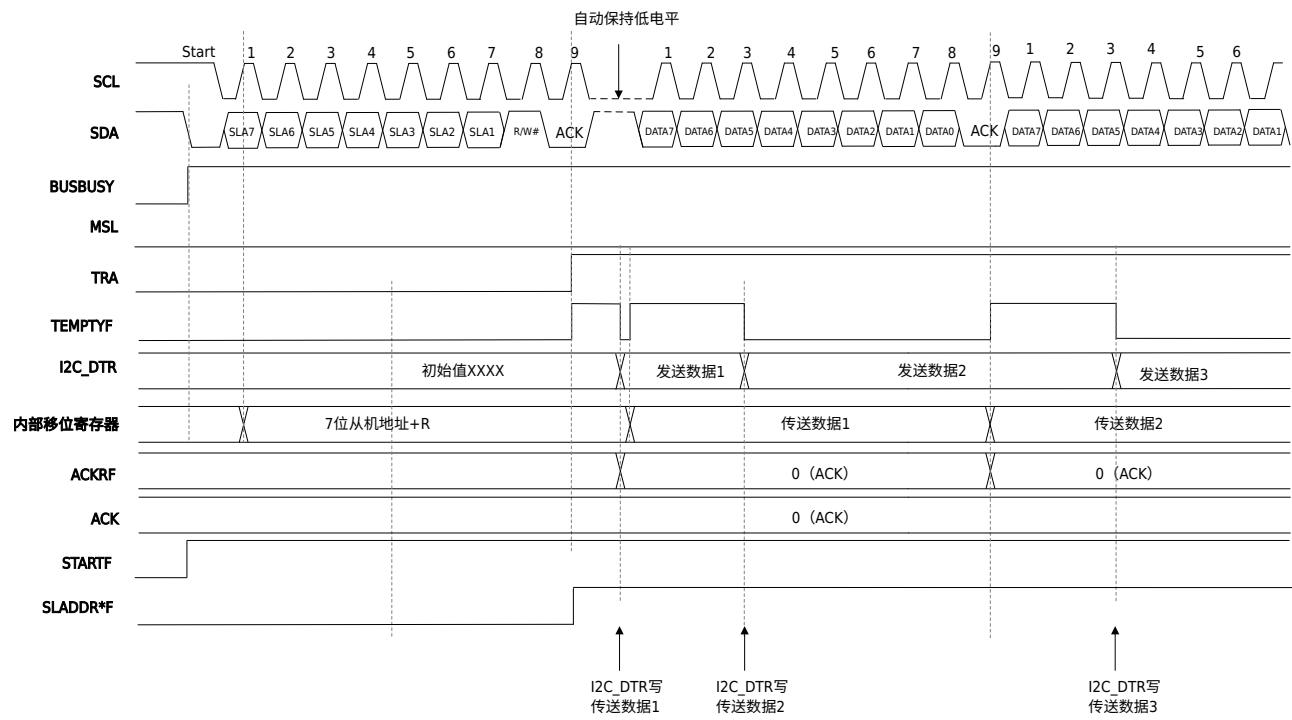


图 30-7 7 位地址格式的从机发送模式时序图（例）

● 从机接收数据

在从机接收模式中，接收来自主机的 SCL 时钟和数据，接收完数据后返回应答。从机接收数据的运行时序例如下图所示。

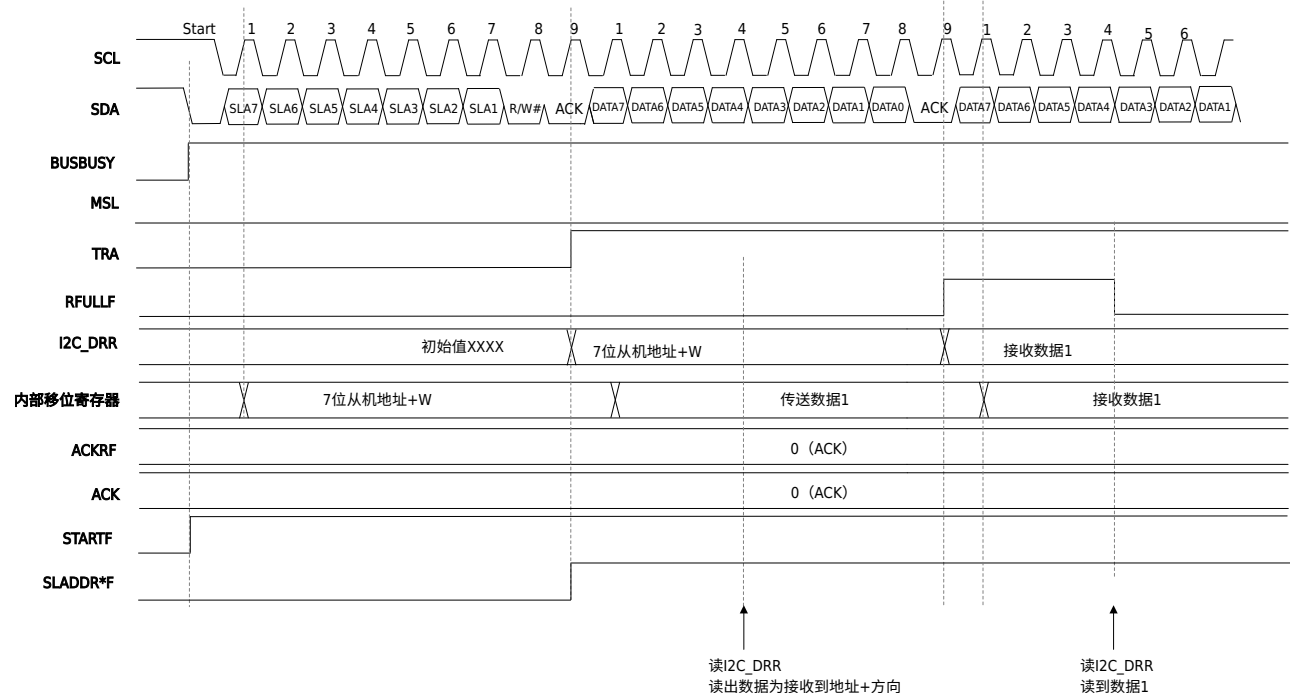


图 30-8 7 位地址格式从机接收模式时序图（例）

停止条件

通过 I2C_CR1.STOP 位发送停止条件。
在 I2C_SR.BUSY 标志为 1（总线忙）并且 I2C_SR.MSL 位为 1（主机模式）的状态下将 STOP 位置 1，就发送停止条件。

重新开始条件

通过 I2C_CR1.RSTART 位生成重新开始条件。
在 I2C_SR.BUSY 标志为 1（总线忙）并且 I2C_SR.MSL 位为 1（主机模式）的状态下 RSTART 位置 1，就生成重新开始条件。
通过重新开始条件，主机可以在不释放 BUS 权的情况下，切换发送/接收模式。也可以在不释放 BUS 权的情况下和另一个从机建立通信。

30.3.2.2 SCL 时钟同步

在多主机模式中使用 I2C 总线时，有可能因和其他主机的竞争而使 SCL 时钟发生冲突。如果 SCL 时钟发生冲突，主机就需要与 SCL 时钟同步，而且需要逐位进行 SCL 时钟的同步。当检测到 SCL 线的上升沿并且在对 I2C_CCR.SHIGHW 寄存器设定的高电平进行计数的过程中，如果因其他主机的 SCL 时钟输出而使 SCL 线下降，就在检测到 SCL 线的下降沿时中止高电平宽度的递增计数，并且在 SCL 线被驱动为低电平的同时开始对 I2C_CCR.SLOWW 设定的低电平宽度进行递增计数，在结束低电平宽度的计数时结束 SCL 线的低电平驱动，并释放 SCL 线。此时，如果其他主机的 SCL 时钟的低电平宽度大于 SLOWW 设定的低电平宽度，就延长 SCL 时钟的低电平宽度。当其他主机结束低电平输出时，释放 SCL 线并且 SCL 时钟上升。因此，在发生 SCL 时钟输出冲突时，SCL 时钟的高电平宽度与短时钟同步，低电平宽度与长时钟同步。

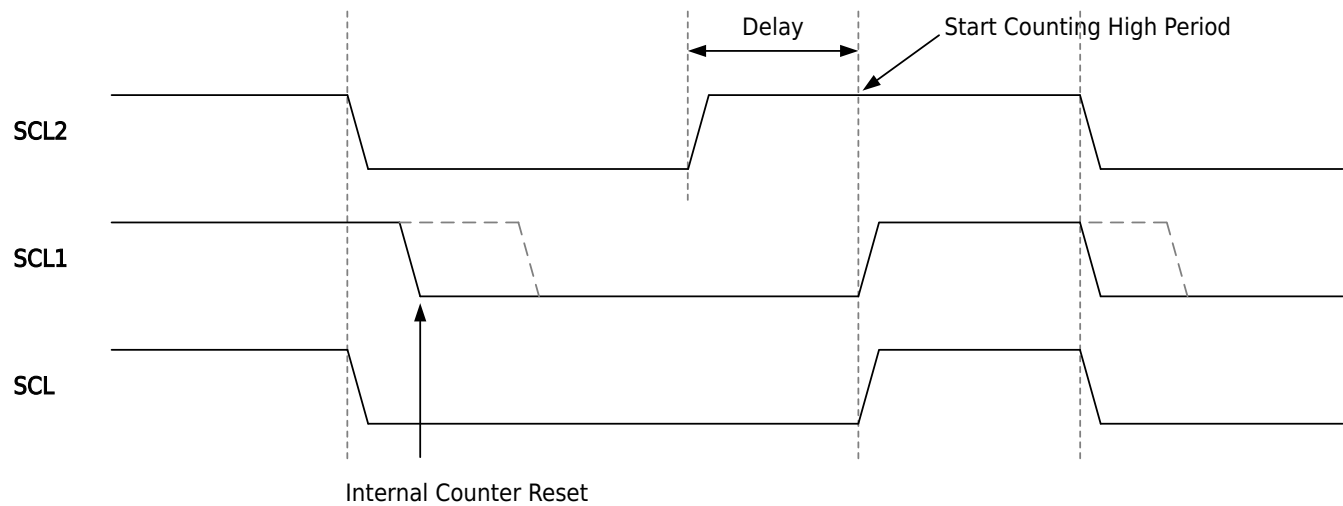


图 30-9 SCL 同步时序

30.3.2.3 仲裁

I2C 总线是一个真正的多主机总线，允许多个主机连接。

如果两个或两个以上的主机试图同时控制总线，SCL 时钟同步过程决定了总线时钟。总线时钟的低周期取决于最长的低电平时钟，高周期取决于最短的高电平时钟。高电平采集到的数据决定了仲裁结果。发送的 SDA 输出为高电平输出（SDA 引脚为高阻抗状态）而检测到 SDA 线为低电平时，就产生仲裁失败。

I2C_SR.AROLF 位会硬件置“1”。如果发生主机仲裁失败，就立即转移到从机接收模式。此时，如果包括广播地址在内的从机地址匹配，就继续从机模式的运行。

30.3.2.4 握手

数据传送过程中通过 SCL 时钟同步机制实现握手。从机在传送完一帧数据后（包含 ACK 位），将 SCL 时钟线维持在低电平。在这种情况下，SCL 时钟的低电平让主机进入等待状态，直到从机释放 SCL 线。

从机发送模式

在发送模式中（I2C_SR.TRA 位=1），如果内部移位寄存器为空状态并且未写发送数据（I2C_DTR 寄存器），就在第 9 时钟和下次传送的第 1 个时钟的低电平区间自动保持 SCL 线的低电平，动作时序如下图所示。

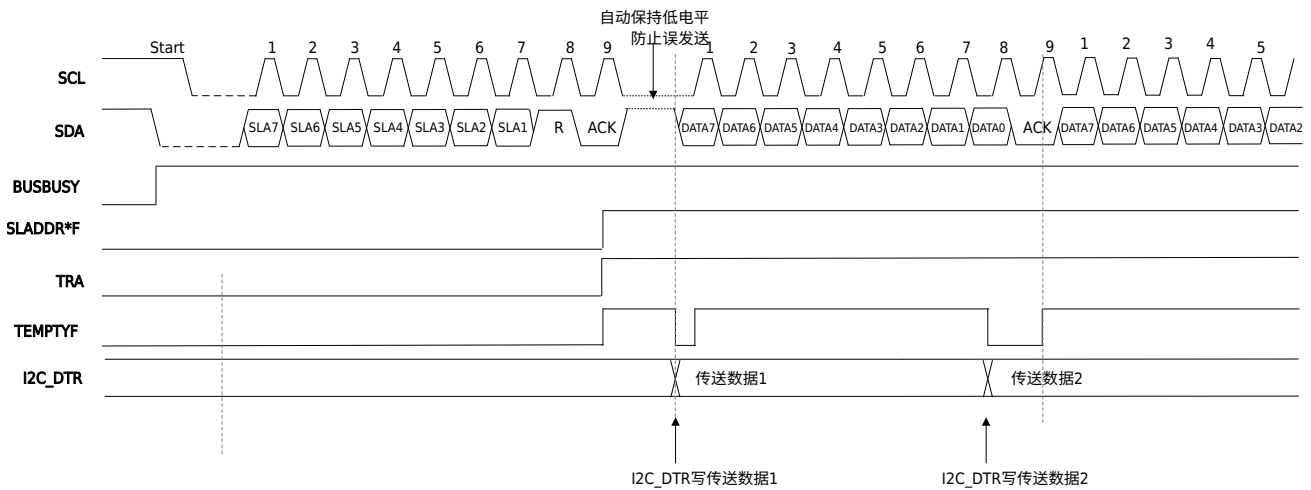


图 30-10 从机发送时序图（1）

在 I2C_SR.NACKF 标志变为 1 或者将最后的发送数据写到 I2C_DTR 寄存器后，在 I2C_SR.EMPTYF 标志为 1 的状态下等到 I2C_SR.TENDF 标志变为 1。当 IC_SR.NACKF 标志或者 TENDF 标志为 1 时，在第 9 个时钟下降后将 SCL 线保持为低电平。此时必须通过读 I2C_DRR 寄存器来结束通信，从而释放 SCL 线。

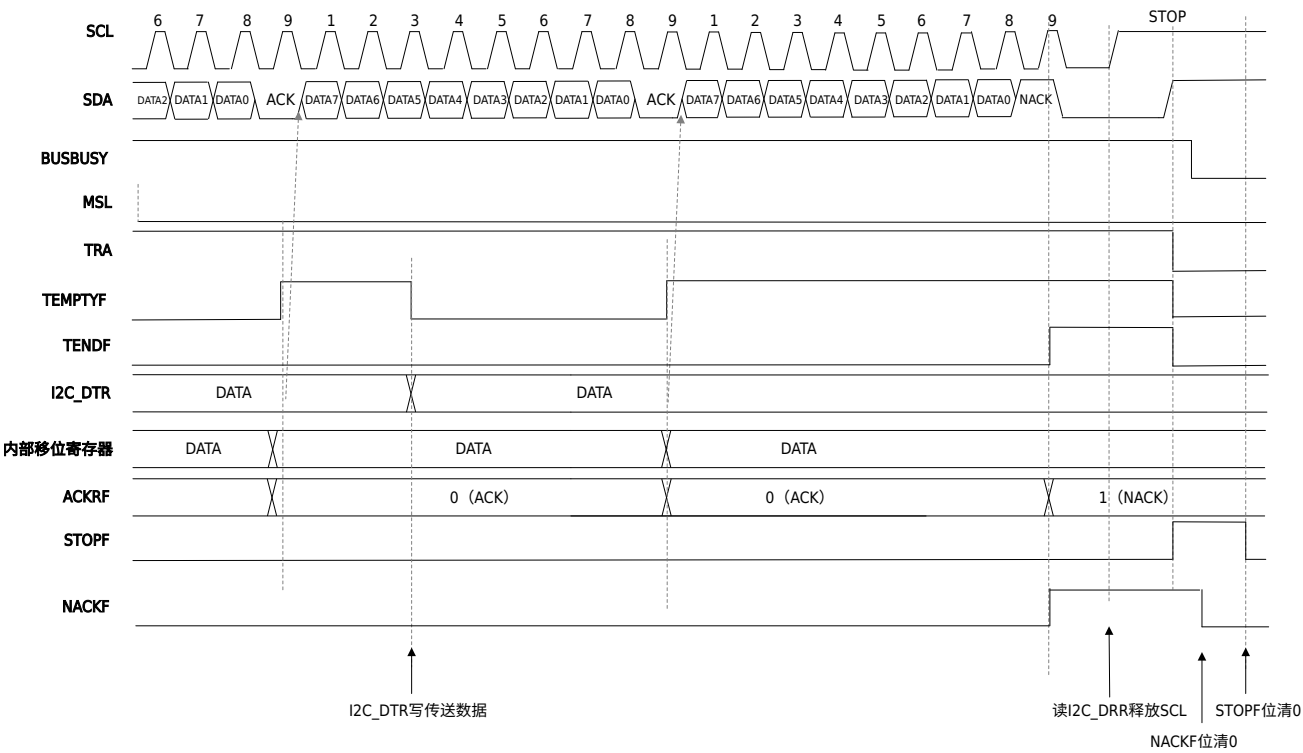


图 30-11 从机发送时序图 (2)

从机接收模式

如果在接收模式中 (I2C_SR.TRA 位=0) 并且接收数据满 (I2C_SR.RFULLF 标志=1) 的状态下, 因至少推迟 1 个传送帧读接收数据 (I2C_DRR 寄存器) 等而发生响应处理延迟, 就在开始下一个数据接收前, 在第 8 个 SCL 和第 9 个 SCL 时钟之间自动保持 SCL 线的低电平, 动作时序如下图所示。

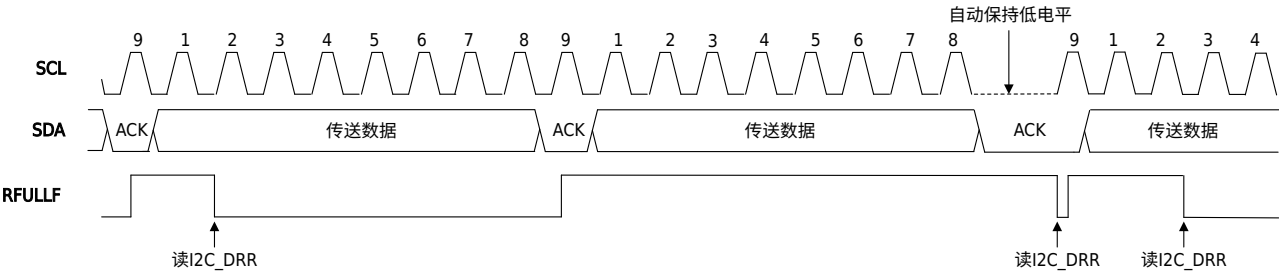


图 30-12 从机接收时序图

快速 ACK/NACK

SMBUS 通信中, 利用系统内置的 CRC 运算器, 计算 SMBUS 的数据包错误码 (PEC) 或者检查接收的数据。在检查 PEC 码的过程中, 在最后字节根据是否匹配发送 ACK 或者 NACK。这就必须在接收的最后字节的 SCL 的第 8 个时钟的下降沿将 SCL 保持低电平。以此来满足软件处理时间。软件根据计算结果, 写 I2C_CR1.ACK 位来解除 SCL 低电平。快速 ACK/NACK 通过 I2C_CR3.FACKEN 位控制, 动作时序如下图所示。

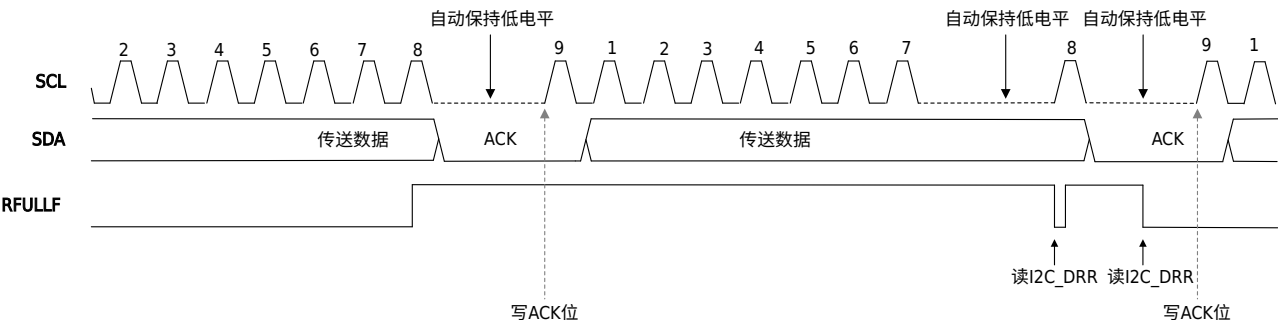


图 30-13 快速 ACK/NACK 时序图

30.3.3 地址匹配

作为从机，可以设定除广播地址和主机通知地址以外的 2 种地址，从机地址能设定 7 位地址或者 10 位地址格式。

30.3.3.1 从机地址匹配

I2C 总线能设定 2 种从机地址，有分别对应的从机地址检测功能。当 SLADDR1EN、SLADDR0EN 为 1 时，能检测到 I2C_SLR1、I2C_SLR0 寄存器设定的从属地址。

如果设定的从机地址匹配一致，就在 SCL 时钟的第 9 个时钟的上升沿将对应的 SLADDR1F、SLADDR0F 置 1，然后根据后续的数据位将 I2C_SR.RFULLF 标志或者 I2C_SR.EMPTIF 标志置 1。由此，就能产生接收数据满中断或者发送数据空中断，并且能通过确认 I2C_SR.SLADDR1F、SLADDR0F 标志判断指定了哪个从机地址。

I2C_SR.SLADDR1F、SLADDR0F 标志变为 1 的时序分别如下图所示。

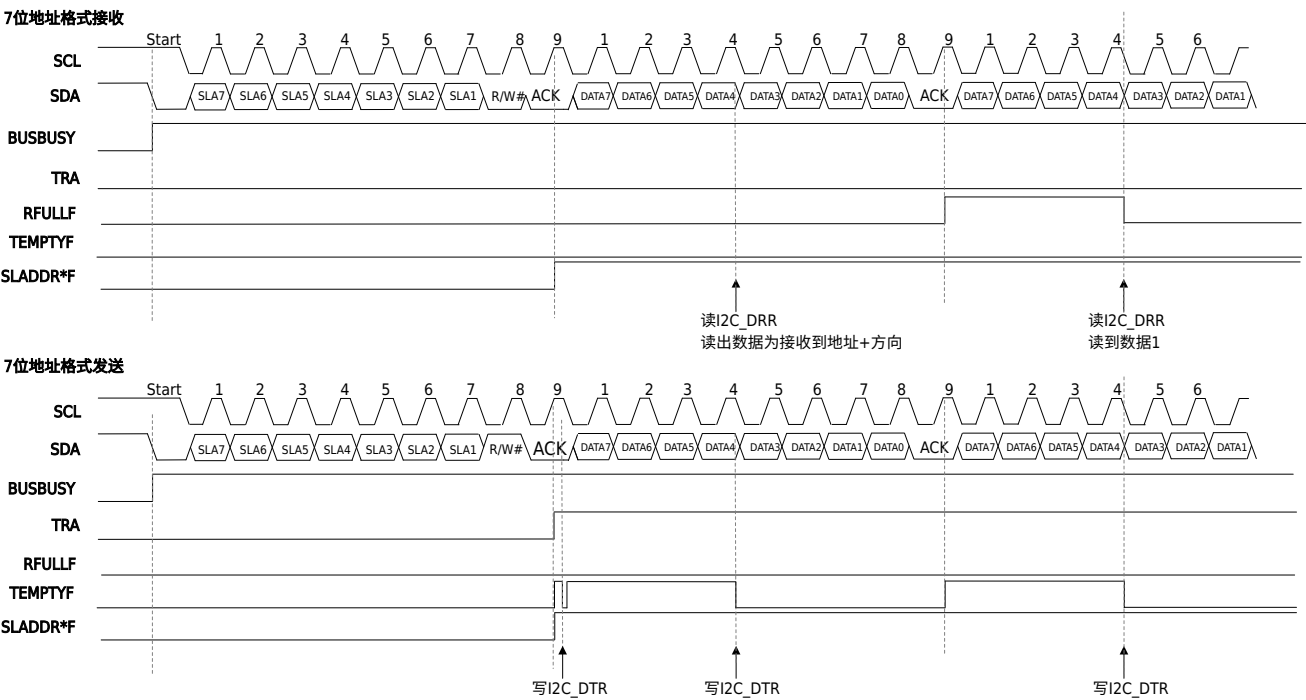


图 30-14 选择 7 位地址格式时的时序

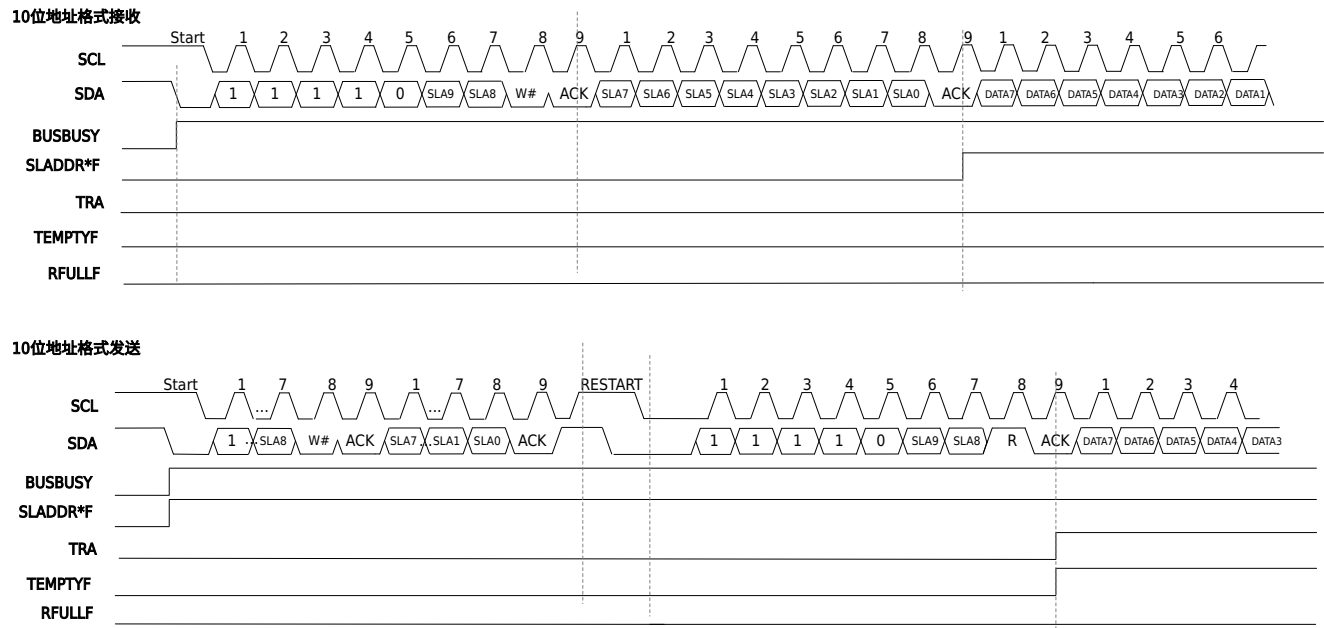


图 30-15 选择 10 位地址格式时的时序

30.3.3.2 广播地址匹配

当 I2C_CR1.GCEN 位为 1 时，能检测广播地址 (0b0000000+0[W])。

但是开始条件或者重新开始条件后的地址为 0b0000000+1[R] (开始字节)，就将此地址视为 All“0”的从机地址而不视为广播地址。

如果匹配到广播地址，就在 SCL 时钟的第 9 个时钟的上升沿将 I2C_SR.GENCALLF 标志置 1。

广播地址匹配一致后的运行和普通的从机接收运行相同。

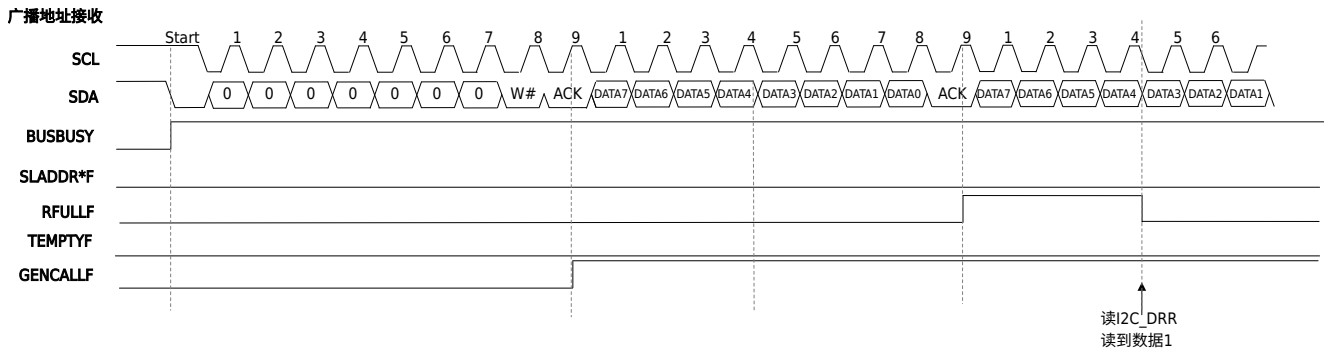


图 30-16 广播地址匹配时序图

30.3.3.3 SMBus 主机地址匹配

本产品具有 SMBus 运行时的主机地址检测功能。如果在 I2C_CR1.SMBUS 位为 1 时将 I2C_CR1.SMBHOSTEN 位置 1，就能在从机接收模式 (I2C_SR.MSL 位 TRA 位为 0b00) 中检测主机地址 (0b0001000)。

如果检测到 SMBUS 主机地址，就在 SCL 时钟的第 9 个时钟的上升沿将 I2C_SR.SMBHOSTF 标志置 1。

即使接在 SMBUS 主机地址 (0b0001000) 后面的位是 Rd 位 (RW#位接收到 1)，也能检测 SMBUS 主机地址。SMBUS 主机地址检测后的运行和普通的从机模式运行相同。

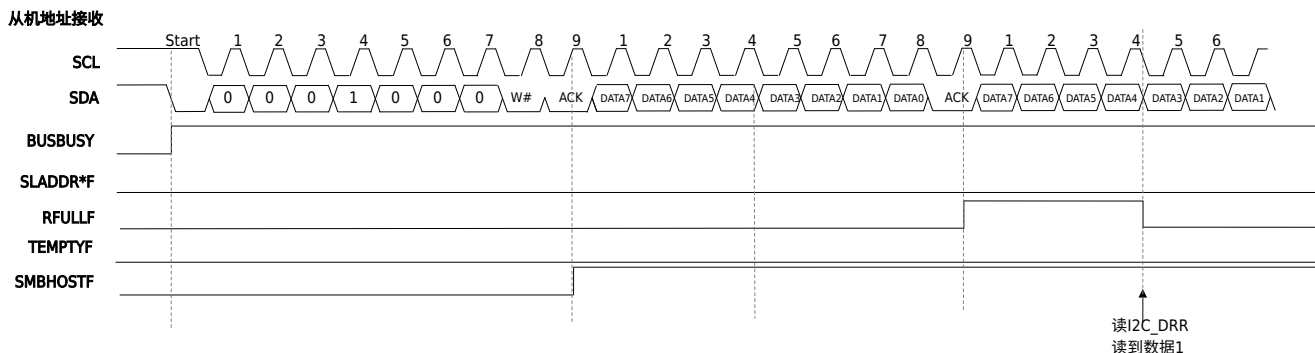


图 30-17 SMBus 主机地址匹配时序图

30.3.3.4 SMBus 报警响应地址匹配

本产品具有 SMBus 运行时的报警响应地址检测功能。如果在 I2C_CR1.SMBUS 位为 1 时将 I2C_CR1.SMBARLERTEN 位置 1，就能在从机接收模式（I2C_SR.MSL 位 TRA 位为 0b00）中检测 SMBUS 报警响应地址（0b0001100）。

如果检测到 SMBUS 报警响应地址，就在 SCL 时钟的第 9 个时钟的上升沿将 I2C_SR.SMBALRTF 标志置 1。SMBUS 报警响应地址检测后的运行和普通的从机模式运行相同。

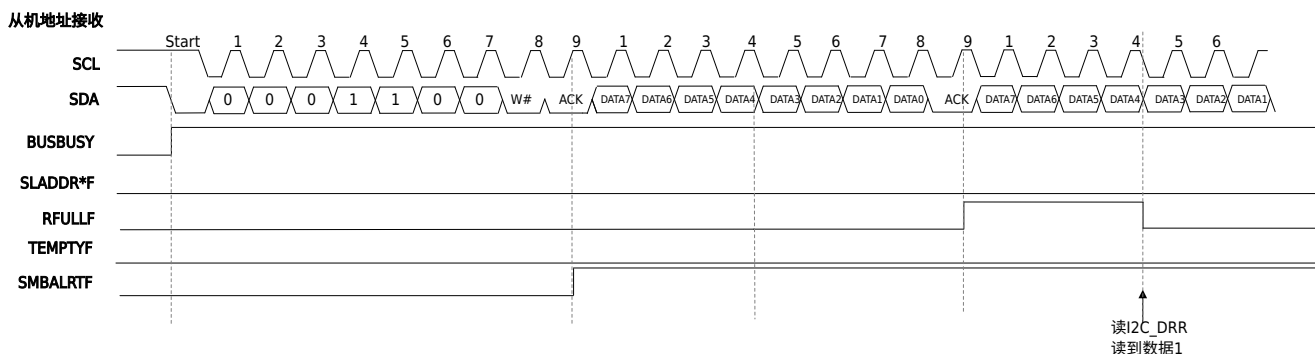


图 30-18 SMBus 报警响应地址匹配时序图

30.3.3.5 SMBus 默认地址匹配

本产品具有 SMBus 运行时的默认地址检测功能。如果在 I2C_CR1.SMBUS 位为 1 时将 I2C_CR1.SMBDEFAULTEN 位置 1，就能在从机接收模式（I2C_SR.MSL 位 TRA 位为 0b00）中检测 SMBUS 默认地址（0b1100001）。

如果检测到 SMBUS 默认地址，就在 SCL 时钟的第 9 个时钟的上升沿将 I2C_SR.SMBDEFAULTF 标志置 1。SMBUS 默认地址检测后的运行和普通的从机模式运行相同。

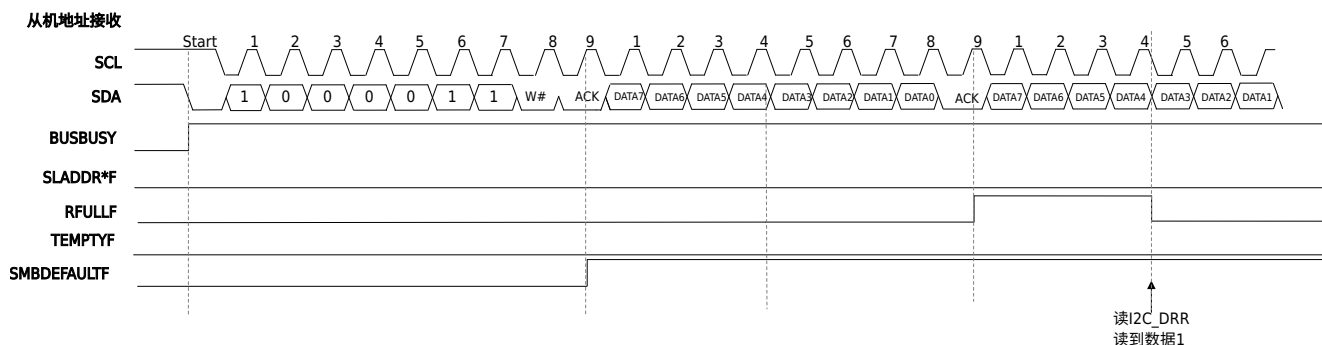


图 30-19 SMBus 默认地址时序图

30.3.4 SMBus 动作

此 I2C 接口能进行以 SMBus (Ver.2.0) 为基准的通信。要进行 SMBus 通信时, 必须将 I2C_CR1.SMBUS 位置 1。必须通过设定 I2C_CCR 寄存器, 将传送速度设定在 SMBus 规格的 10kbps~100kbps 范围内。

30.3.4.1 SMBus 超时测量

SCL 电平超时测量

工作在从机模式下时, 地址匹配一致后检测到 SCL 线的低电平或者高电平被固定了一定的时间以上, 并且检测到总线的异常状态。

工作在主机模式下时, 起始条件生成后检测到 SCL 先的低电平或者高电平被固定了一定的时间以上, 并且检测到总线的异常状态。

超时检测功能监视 SCL 线的状态, 通过内部计数器对高电平或者低电平的时间进行计数。如果 SCL 线有变化(上升/下降), 就对内部计数器进行复位, 否则就继续进行计数。如果在 SCL 线没有变化的状态下内部计数器计数到 TOUTHIGH/TOUTLOW 设定值, 就能检测到超时并且通知总线的异常状态。

对于内部计数器的计数, 能通过设定 HTMOUT、LTMOUT 位选择是在 SCL 线的低电平还是在高电平的状态下进行计数, 或者在低电平和高电平的状态下都进行计数。如果将 HTMOUT、LTMOUT 位都置 0, 就不进行内部计数。

从机的超时测量

SMBus 通信的从属设备需要测量以下所示的区间(超时间隔: $T_{\text{LOW:SEXT}}$):

- 开始条件到停止条件的区间

在通过从属设备进行超时测量时, 使用开始条件检测中断和停止条件检测中断并且通过芯片定时器, 测量从检测到开始条件到检测到停止条件的时间。此超时的测量时间必须在 SMBus 规格的时钟低电平的累积时间[从属设备] $T_{\text{LOW:SEXT}}$: 25ms (max) 以内。

如果定时器测量的时间超过 SMBus 规格的时钟 Low 电平检测的超时 T_{TIMEOUT} : 25ms (min), 从机就需要释放总线。

主机的超时测量

SMBus 通信的主控设备需要测量以下所示的区间(超时间隔: $T_{\text{LOW:MEXT}}$):

- 开始条件到应答位的区间
- 应答位到下一个应答位的区间
- 应答位到停止条件的区间

在主机进行超时测量时, 使用开始条件检测中断、停止条件检测中断以及发送结束中断或者接收数据满中断, 通过芯片定时器测量各区间的时间。此超时的测量时间必须在 SMBus 规格的时钟低电平的累积时间[主机] $T_{\text{LOW:MEXT}}$: 10ms (max) 以内, 开始条件到停止条件的全部 $T_{\text{LOW:MEXT}}$ 的累加结果必须在 $T_{\text{LOW:SEXT}}$: 25ms (max) 以内。

如果定时器测量的时间超过 SMBus 规格的时钟低电平的累积时间[主控设备] $T_{\text{LOW:MEXT}}$: 10ms (max), 或者各测量时间的累加结果超过 SMBus 规格的时钟低电平检测的超时 T_{TIMEOUT} : 25ms (min), 主机就需要中止处理。在主机发送时, 必须立即中止发送(写 I2C_DTR 寄存器)。通过发送停止条件中止主机的处理。

30.3.4.2 数据包错误码 (PEC)

通信中, 利用 CPU 运算 CRC, 发送 SMBus 的数据包错误码 (PEC) 或者检查接收数据。

30.3.5 复位

具有对通信模块进行复位的功能。有 2 种复位, 一种是通信复位: I2C 内部的全部寄存器 (I2C_FSTR[3:0]除外) 及内部状态复位; 另一种内部状态复位: I2C_SR (包括 BUSY 位), 内部移位寄存器和内部状态机进行复位。

复位后, 必须将 I2C_CR1.SWRST 位置 0。

因为无论进行哪种复位都要解除 SCL 引脚/SDA 引脚的输出状态而变为高阻抗状态，所以也能用于解除总线意外停机状态。

从属模式中的复位会引起与主控设备不同步，因此尽量避免使用。



注意

在复位（I2C_CR1.PE 位和 I2C_CR1.SWRST 位为 0b01）过程中不能监视开始条件等的总线状态。

30.3.6 可编程数字滤波

SCL 引脚和 SDA 引脚的状态经由模拟滤波器电路和数字滤波器进入内部。数字滤波器电路的框图下图所示。

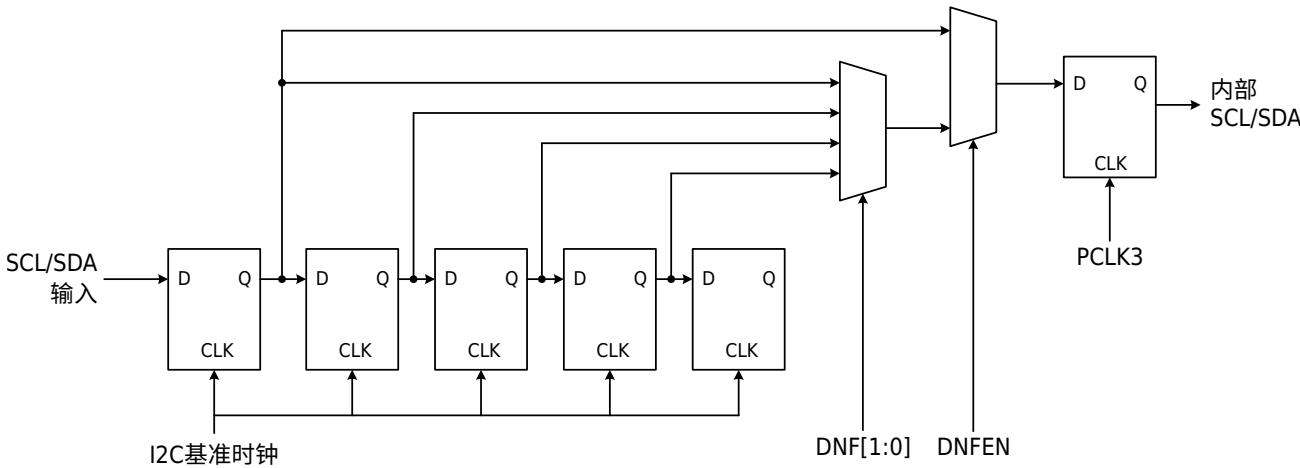


图 30-20 数字滤波电路框图

内部数字滤波器电路由 4 段串联的触发电路和匹配检测电路构成。

通过 I2C_FLTR.DNF 位选择数字滤波器的有效段数，根据选择的有效段数，噪声消除能力为 1~4 个 I2C 周期。

在 I2C 内部时钟的下降沿对 SCL 引脚的输入信号（或者 SDA 引脚的输入信号）进行采样，如果 I2C_FLTR.DNF 位设定的有效段数的触发电路输出全部匹配，就将该电平作为内部信号进行传输，否则就保持原来的值。

30.3.7 中断和事件

I2C 具有 4 种中断和用于触发启动其他外围电路的事件输出供用户选择。包括：通信错误的发生（仲裁失败检测、NACK 检测、超时检测、开始条件检测、停止条件检测）、接收结束、发送数据空、发送结束。

中断一览表如下所示。

表 30-2 中断一览表

名称	中断源	中断标志	中断条件
I2C_EEI	通信错误/通信事件	ARLOF	ARLOF=1&ARLOIE=1
		SLADDR0F	SLADDR0F=1&SLADDR0IE=1
		SLADDR1F	SLADDR1F=1&SLADDR1IE=1
		SMBALRTF	SMBALRTF=1&SMBALRTIE=1
		SMBHOSTF	SMBHOSTF=1&SMBHOSTFIE=1
		SMBDEFAULTF	SMBDEFAULTF=1&SMBDEFAULTIE=1

名称	中断源	中断标志	中断条件
I2C_EEI	通信错误/通信事件	GENCALLF	GENCALLF=1&GENCALLIE=1
		NACKF	NACKF=1&NACKIE=1
		TMOUTF	TMOUTF=1&TMOUTIE=1
		STARTF	STARTF=1&STARTIE=1
		STOPF	STOPF=1&STOPIE=1
I2C_RXI	接收数据满	RFULLF	RFULLF=1&RFULLIE=1&~FEN
	接收 FIFO 数据非空	RFREQF	RFREQF=1&RFREQIE=1&FEN
I2C_TXI	发送数据空	EMPTYF	EMPTYF=1&EMPTYIE=1
I2C_TEI	发送结束	TENDF	TENDF=1&TENDIE=1

事件信号输出一览表如下所示。

表 30-3 事件信号输出一览表

名称	事件源	事件条件
I2C_EEI	通信错误/通信事件	ARLOF=1
		SLADDR0F=1
		SLADDR1F=1
		SMBALRTF=1
		SMBHOSTF=1
		SMBDEFAULTF=1
		GENCALLF=1
		NACKF=1
		TMOUTF=1
		STARTF=1
		STOPF=1
I2C_RXI	接收数据满	RFULLF=1
I2C_TXI	发送数据空	EMPTYF=1
I2C_TEI	发送结束	TENDF=1

30.4 操作示例

30.4.1 I2C 初始化流程

在开始发送或者接收数据时，必须按以下步骤进行初始化：

- 步骤 1. PE 位设定为 0；
- 步骤 2. SWRST 设定为 1，通信复位；
- 步骤 3. PE 位设定为 1，内部状态复位；
- 步骤 4. 设定从机地址格式和地址；
- 步骤 5. 设定波特率；
- 步骤 6. 根据需要设定控制寄存器功能及中断；
- 步骤 7. SWRST 位设定为 0，解除内部状态复位；
- 步骤 8. 初始化结束。可发送接收数据。

30.5 寄存器

30.5.1 寄存器总表

- I2C1 基地址: 0x4004E000
- I2C2 基地址: 0x4004E400
- I2C3 基地址: 0x4008E800
- I2C4 基地址: 0x4008EC00
- I2C5 基地址: 0x4008F000
- I2C6 基地址: 0x4008F400

表 30-4 集成电路总线 (I2C) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	I2C_CR1	I2C 控制寄存器 1	32
0x04	I2C_CR2	I2C 控制寄存器 2	32
0x08	I2C_CR3	I2C 控制寄存器 3	32
0x0C	I2C_CR4	I2C 控制寄存器 4	32
0x10	I2C_SLR0	I2C 从机地址寄存器 0	32
0x14	I2C_SLR1	I2C 从机地址寄存器 1	32
0x18	I2C_SLTR	I2C SCL 电平超时控制寄存器	32
0x1C	I2C_SR	I2C 状态寄存器	32
0x20	I2C_CLR	I2C 状态清零寄存器	32
0x24	I2C_DTR	I2C 数据发送寄存器	8
0x28	I2C_DRR	I2C 数据接收寄存器	8
0x2C	I2C_CCR	I2C 时钟控制寄存器	32
0x30	I2C_FLTR	I2C 滤波控制寄存器	32
0x34	I2C_FSTR	I2C FIFO 状态寄存器	32
0x38	I2C_SLVADRR	I2C 从机地址接收寄存器	32

30.5.2 I2C 控制寄存器 1 (I2C_CR1)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																
Reset	0x00000040																																																																																																																																																																																																																																																																																																																																																															
Name	Reserved																SWRST	Reserved																ACK	STOP	START	RESTART	GCEN	Reserved	SMBHOSTEN	SMBDEFAULTEN	SMBALRTEN	SMBUS	PE																																																																																																																																																																																																																																																																																																																				
Access																	RW																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写									
31:16	Reserved	保留	-	-									
15	SWRST	软件复位	● 0b0：解除复位 ● 0b1：软件复位 通过和 PE 位组合，选择内部状态复位或者通信复位： <table><tr><th>SWRST</th><th>PE</th><th>复位内容</th></tr><tr><td>0b1</td><td>0b0</td><td>通信复位：I2C 内部的全部寄存器（I2C_FSTR[3:0]除外）及内部状态复位</td></tr><tr><td>0b1</td><td>0b1</td><td>内部状态复位：I2C_SR（包括 BUSY 位），内部移位寄存器和内部状态机进行复位</td></tr></table>	SWRST	PE	复位内容	0b1	0b0	通信复位：I2C 内部的全部寄存器（I2C_FSTR[3:0]除外）及内部状态复位	0b1	0b1	内部状态复位：I2C_SR（包括 BUSY 位），内部移位寄存器和内部状态机进行复位	RW
SWRST	PE	复位内容											
0b1	0b0	通信复位：I2C 内部的全部寄存器（I2C_FSTR[3:0]除外）及内部状态复位											
0b1	0b1	内部状态复位：I2C_SR（包括 BUSY 位），内部移位寄存器和内部状态机进行复位											
14:11	Reserved	保留	-	-									
10	ACK	发送应答	● 0b0：应答位发送“0”（发送 ACK） ● 0b1：应答位发送“1”（发送 NACK）	RW									
9	STOP	停止条件生成位	● 0b0：不生成停止条件 ● 0b1：生成停止条件 此位可软件置位和清零。 硬件清零条件： ● 检测到停止条件 ● 仲裁失败 ● 检测到开始条件 ● 通信复位	RW									
8	START	起始条件生成位	● 0b0：不生成起始条件 ● 0b1：生成起始条件 此位可软件置位和清零。 硬件清零条件： ● 检测到开始条件 ● 仲裁失败时 ● 通信复位	RW									

位/位域	标记	位名	功能描述	读写
7	RESTART	重复起始条件生成位	● 0b0: 不生成重复起始条件 ● 0b1: 生成重复起始条件 此位可软件置位和清零。 硬件清零条件: ● 检测到开始条件 ● 仲裁失败时 ● 通信复位	RW
6	GCEN	广播呼叫使能	● 0b0: 广播地址检测无效 ● 0b1: 广播地址检测有效	RW
5	<i>Reserved</i>	<i>保留</i>	-	-
4	SMBHOSTEN	允许匹配 SMBUS 主机地址位	● 0b0: 禁止匹配 SMBUS 主机地址 ● 0b1: 允许匹配 SMBUS 主机地址	RW
3	SMBDEFAULTEN	允许匹配 SMBUS 默认地址位	● 0b0: 禁止匹配 SMBUS 默认地址 ● 0b1: 允许匹配 SMBUS 默认地址	RW
2	SMBALRTEN	允许匹配 SMBUS 报警响应地址位	● 0b0: 禁止 SMBUS 报警响应地址 ● 0b1: 允许 SMBUS 报警响应地址	RW
1	SMBUS	SMBUS/I2C 总线模式选择位	● 0b0: I2C 总线模式 ● 0b1: SMBUS 总线模式	RW
0	PE	I2C 功能使能	通过和 SWRST 位组合, 选择内部状态复位或者通信复位: ● 0b0: I2C 功能禁止 ● 0b1: I2C 功能允许	RW

30.5.3 I2C 控制寄存器 2 (I2C_CR2)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
Name	Reserved								SMBALRTIE				SMBHOSTIE				SMBDEFAULTIE				GENCALLIE				Reserved								TMOUTIE		Reserved		NACKIE		RFREQIE		Reserved		ARLOIE		Reserved		EMPTYIE		RFULLIE		Reserved		STOPIE		TENDIE		SLADDR1IE		SLADDR0IE		STARTIE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									</

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	SMBALRTIE	SMBUS 报警响应地址匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
22	SMBHOSTIE	SMBUS 主机地址匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
21	SMBDEFAULTIE	SMBUS 默认地址匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
20	GENCALLIE	广播呼叫地址匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
19:15	Reserved	保留	-	-
14	TMOUTIE	超时中断使能	0b0: 中断禁止 0b1: 中断允许	RW
13	Reserved	保留	-	-
12	NACKIE	NACK 中断使能	0b0: 中断禁止 0b1: 中断允许	RW
11	RFREQIE	FIFO 接收请求中断使能	0b0: 中断禁止 0b1: 中断允许	RW
10	Reserved	保留	-	-
9	ARLOIE	仲裁失败中断使能	0b0: 中断禁止 0b1: 中断允许	RW
8	Reserved	保留	-	-
7	EMPTYIE	发送数据空中断使能	0b0: 中断禁止 0b1: 中断允许	RW
6	RFULLIE	接收数据满中断使能	0b0: 中断禁止 0b1: 中断允许	RW
5	Reserved	保留	-	-
4	STOPIE	停止条件中断使能	0b0: 中断禁止 0b1: 中断允许	RW

位/位域	标记	位名	功能描述	读写
3	TENDIE	发送一帧数据结束中断使能	0b0: 中断禁止 0b1: 中断允许	RW
2	SLADDR1IE	从机地址 1 匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
1	SLADDR0IE	从机地址 0 匹配一致中断使能	0b0: 中断禁止 0b1: 中断允许	RW
0	STARTIE	开始条件/重新开始条件中断使能	0b0: 中断禁止 0b1: 中断允许	RW

30.5.4 I2C 控制寄存器 3 (I2C_CR3)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000006																															
Name	Reserved																							FACKEN	Reserved			HTMOUT	LTMOUT	TMOUTEN		
Access																								RW				RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	FACKEN	RFULLF 标志位置位时间点选择	0b0: 在 SCL 时钟的第 9 个时钟上升时, RFULLF 标志位为“1” (在第 8 个时钟的下降沿, SCL 线不保持低电平) 0b1: 在 SCL 时钟的第 8 个时钟上升时, RFULLF 标志位为“1” (在第 8 个时钟的下降沿, SCL 线保持低电平)  说明 通过写 ACK 位来解除保持的低电平。	RW
6:3	Reserved	保留	-	-
2	HTMOUT	高电平超时检测允许	0b0: 在 SCL 线为高电平时, 禁止超时检测 0b1: 在 SCL 线为高电平时, 允许超时检测	RW
1	LTMOUT	低电平超时检测允许	0b0: 在 SCL 线为低电平时, 禁止超时检测 0b1: 在 SCL 线为低电平时, 允许超时检测	RW
0	TMOUTEN	超时功能允许位	0b0: 检测 SCL 电平超时功能禁止 0b1: 检测 SCL 电平超时功能允许	RW

30.5.5 I2C 控制寄存器 4 (I2C_CR4)

Offset	Bit Position																																		
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00301B07																																		
Name	Reserved																				BUSFREE_CLREN	Reserved	BUSWAIT	Reserved											
Access																					RW		RW												

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21:20	Reserved	保留	所有位读出为 1，写入时必须写 1	-
19:13	Reserved	保留	-	-
12	BUSFREE_CLREN	总线结束状态清除位	0b0：禁止 I2C 总线 STOP 标志对 I2C_SLVADRR 从机接收地址寄存器以及 I2C_SR 状态寄存器中的 SMBALRTF，SMBHOSTF，SMBDEFAULTF，GENCALLF，MSL，SLADDR1F，SLADDR0F，STARTF 位进行清除 0b1：使能 I2C 总线 STOP 标志对 I2C_SLVADRR 从机接收地址寄存器以及 I2C_SR 状态寄存器中的 SMBALRTF，SMBHOSTF，SMBDEFAULTF，GENCALLF，MSL，TENDF，SLADDR1F，SLADDR0F，STARTF 位进行清除	RW
11	Reserved	保留	读出为 1，写入时必须写 1	-
10	BUSWAIT	总线等待位	0b0：I2C_DRR 接收满、内部移位寄存器为空时，在第 9 个时钟和下次传送的第 1 个时钟之间不保持低电平，继续接收下一个数据 0b1：I2C_DRR 接收满、内部移位寄存器为空时，在第 9 个时钟和下次传送的第 1 个时钟之间保持低电平，通过读 I2C_DRR 寄存器来解除保持的低电平	RW
9:8	Reserved	保留	所有位读出为 1，写入时必须写 1	-
7:3	Reserved	保留	-	-
2:0	Reserved	保留	所有位读出为 1，写入时必须写 1	-

30.5.6 I2C 从机地址寄存器 0 (I2C_SLR0)

Offset	Bit Position																																
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00001000																																
Name	Reserved		MASK0EN	MSLADDR0												ADDRMOD0	Reserved		SLADDR0EN	Reserved		SLADDR0											
Access			RW	RW												RW		RW		RW													

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26	MASK0EN	从机地址 0 掩码有效位	0b0: 从机地址寄存器 0 掩码设定值无效 0b1: 从机地址寄存器 0 掩码设定值有效	RW
25:24	MSLADDR0[9:8]	10 位从机地址的高位掩码地址	设定从机地址 0 的高位掩码地址： - 当 MASK0EN=0, MSLADDR0[9:8]设定的掩码无效。 - 当 MASK0EN=1, MSLADDR0[9:8]设定的掩码有效。其有效范围与 ADDRMOD0 所设值成正相关即： - 当 ADDRMOD0=0 时, MSLADDR0[9:8]位无效； - 当 ADDRMOD0=1 时, MSLADDR0[9:8]为 10 位从机地址的高两位掩码地址。	RW
23:16	MSLADDR0[7:0]	7 位地址/10 位地址的低位掩码地址	设定从机地址 0 的低位掩码地址： - 当 MASK0EN=0, MSLADDR0[7:0]设定的掩码无效。 - 当 MASK0EN=1, MSLADDR0[7:0]设定的掩码有效。其有效范围与 ADDRMOD0 所设值成正相关即： - 当 ADDRMOD0=0 时, MSLADDR0[7:1]为 7 位从机地址掩码, MSLADDR0[0]位无效； - 当 ADDRMOD0=1 时, MSLADDR0[7:0]为 10 位从机地址的低 8 位地址掩码。	RW
15	ADDRMOD0	7 位/10 位地址格式选择位	0b0: 选择 7 位地址格式 0b1: 选择 10 位地址格式	RW
14:13	Reserved	保留	-	-
12	SLADDR0EN	从机地址 0 有效位	0b0: 从机地址寄存器 0 设定值无效 0b1: 从机地址寄存器 0 设定值有效	RW
11:10	Reserved	保留	-	-
9:8	SLADDR0[9:8]	10 位从机地址的高位	设定从机地址： - 当 ADDRMOD0=0 时, SLADDR0[9:8]设定无效； - 当 ADDRMOD0=1 时, SLADDR0[9:8]为 10 位从机地址的高两位。	RW

位/位域	标记	位名	功能描述	读写
7:0	SLADDR0[7:0]	7 位地址/10 位地址的低位	设定从机地址： ● 当 ADDRMOD0=0 时，SLADDR0[7:1]为 7 位从机地址，SLADDR0[0]位无效； ● 当 ADDRMOD0=1 时，SLADDR0[7:0]为 10 位从机地址的低 8 位地址。	

30.5.7 I2C 从机地址寄存器 1 (I2C_SLR1)

Offset	Bit Position																																
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved		MASK1EN	MSLADDR1												ADDRMOD1	Reserved		SLADDR1EN	Reserved		SLADDR1											
Access			RW	RW												RW			RW			RW											

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26	MASK1EN	从机地址 1 掩码有效位	0b0: 从机地址寄存器 1 掩码设定值无效 0b1: 从机地址寄存器 1 掩码设定值有效	RW
25:24	MSLADDR1[9:8]	10 位从机地址的高位掩码地址	设定从机地址 1 的高位掩码地址: - 当 MASK1EN=0, MSLADDR1[9:8]设定的掩码无效。 - 当 MASK1EN=1, MSLADDR1[9:8]设定的掩码有效。其有效范围与 ADDRMOD1 所设值成正相关即: - 当 ADDRMOD1=0 时, MSLADDR1[9:8]位无效; - 当 ADDRMOD1=1 时, MSLADDR1[9:8]为 10 位从机地址的高两位掩码地址。	RW
23:16	MSLADDR1[7:0]	7 位地址/10 位地址的低位掩码地址	设定从机地址 1 的低位掩码地址: - 当 MASK1EN=0, MSLADDR1[7:0]设定的掩码无效。 - 当 MASK1EN=1, MSLADDR1[7:0]设定的掩码有效。其有效范围与 ADDRMOD1 所设值成正相关即: - 当 ADDRMOD1=0 时, MSLADDR1[7:1]为 7 位从机地址掩码, MSLADDR1[0]位无效; - 当 ADDRMOD1=1 时, MSLADDR1[7:0]为 10 位从机地址的低 8 位地址掩码。	RW
15	ADDRMOD1	7 位/10 位地址格式选择位	0b0: 选择 7 位地址格式 0b1: 选择 10 位地址格式	RW
14:13	Reserved	保留	-	-
12	SLADDR1EN	从机地址 1 有效位	0b0: 从机地址寄存器 1 设定值无效 0b1: 从机地址寄存器 1 设定值有效	RW
11:10	Reserved	保留	-	-
9:8	SLADDR1[9:8]	10 位从机地址的高位	设定从机地址: - 当 ADDRMOD1=0 时, SLADDR1[9:8]设定无效; - 当 ADDRMOD1=1 时, SLADDR1[9:8]为 10 位从机地址的高两位。	RW

位/位域	标记	位名	功能描述	读写
7:0	SLADDR1[7:0]	7 位地址/10 位地址的低位	设定从机地址： - 当 ADDRMOD1=0 时，SLADDR1[7:1]为 7 位从机地址，SLADDR1[0]位无效； - 当 ADDRMOD1=1 时，SLADDR1[7:0]为 10 位从机地址的低 8 位地址。	

30.5.8 I2C SCL 电平超时控制寄存器 (I2C_SLTR)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	TOUTHIGH																TOUTLOW															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	TOUTHIGH	SCL 高电平超时周期	SCL 高电平超时时间=TOUTHIGH*I2C 基准时钟周期	RW
15:0	TOUTLOW	SCL 低电平超时周期	SCL 低电平超时时间=TOUTLOW*I2C 基准时钟周期	RW

30.5.9 I2C 状态寄存器 (I2C_SR)

Offset	Bit Position																																
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x05000000																																
Name	RFREQ	Reserved		TFST		RFFULL	RFEMPTY	TFFULL	TFEMPTY	RO SMBALRTF	RO SMBHOSTF	RO SMBDEFAULTF	RO GENCALLF	Reserved	TRA	RO BUSY	RW MSL	Reserved	TMOUTF	Reserved	NACKF	Reserved	ACKRF	ARLOF	Reserved	TEMTYF	RO RFULLF	Reserved	RO STOPF	RO TENDF	RO SLADDR1F	RO SLADDR0F	RO STARTF
Access	RO			RO		RO	RO	RO	RO	RO	RO	RO	RO		RW	RO	RW		RO		RO		RO	RO		RO	RO		RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31	RFREQ	接收 FIFO 请求标志位	0b0: 当前接收 FIFO 无有效数据 0b1: 当前接收 FIFO 有有效数据	RO
30	Reserved	保留	-	-
29:28	TFST	发送 FIFO 状态标志位	0b00: 当前发送 FIFO 已空 0b01: 当前发送 FIFO 有 1Byte 有效数据 0b10: 当前发送 FIFO 有 2Bytes 有效数据 0b11: Reserved	RO
27	RFFULL	接收 FIFO 满标志位	0b0: 当前接收 FIFO 未滿 0b1: 当前接收 FIFO 已滿	RO
26	RFEMPTY	接收 FIFO 空标志位	0b0: 当前接收 FIFO 未空 0b1: 当前接收 FIFO 已空	RO
25	TFFULL	发送 FIFO 满标志位	0b0: 当前发送 FIFO 未滿 0b1: 当前发送 FIFO 已滿	RO
24	TFEMPTY	发送 FIFO 空标志位	0b0: 当前发送 FIFO 未空 0b1: 当前发送 FIFO 已空	RO
23	SMBALRTF	SMBUS 报警响应地址匹配一致标志位	0b0: 未匹配到 SMBUS 报警响应地址 0b1: 检测到主机地址 置位条件: - 接收到的地址和 0b0001100 匹配一致 清零条件: - SMBALRTFCLR 写“1” - 检测到停止条件 - 通信复位 - 内部状态复位	RO
22	SMBHOSTF	SMBUS 主机地址匹配一致标志位	0b0: 未匹配到 SMBUS 主机地址 0b1: 匹配到 SMBUS 主机地址 置位条件: - 接收到的地址和 0b0001000 匹配一致 清零条件: - SMBHOSTFCLR 写“1” - 检测到停止条件 - 通信复位 - 内部状态复位	RO

位/位域	标记	位名	功能描述	读写															
21	SMBDEFAULTF	SMBUS 默认地址匹配一致标志位	0b0：未匹配到 SMBUS 默认地址 0b1：匹配到 SMBUS 默认地址 置位条件： - 接收到的地址和 0b1100001 匹配一致 清零条件： - SMBDEFAULTFCLR 写“1” - 检测到停止条件 - 通信复位 - 内部状态复位	RO															
20	GENCALLF	广播呼叫地址匹配一致标志位	0b0：未匹配到广播呼叫地址 0b1：匹配到广播呼叫地址 置位条件： - 接收的从机地址与广播呼叫地址（All“0”）匹配一致 清零条件： - GENCALLFCLR 写“1” - 检测到停止条件 - 通信复位 - 内部状态复位	RO															
19	Reserved	保留	-	-															
18	TRA	发送接收选择位	0b0：接收数据 0b1：发送数据 通过和 MSL 位的组合，表示 I2C 的运行模式： <table border="1"><thead><tr><th>MSL</th><th>TRA</th><th>I2C 运行模式</th></tr></thead><tbody><tr><td>0b0</td><td>0b0</td><td>从机接收模式</td></tr><tr><td>0b0</td><td>0b1</td><td>从机发送模式</td></tr><tr><td>0b1</td><td>0b0</td><td>主机接收模式</td></tr><tr><td>0b1</td><td>0b1</td><td>主机发送模式</td></tr></tbody></table> 此位可软件置位和清零。 硬件置位条件： - 检测到开始条件 - 主机模式下，发送的 RW 位为 0 - 从机模式下，地址匹配且接收的 RW 位为 1 硬件清零条件： - 检测到停止条件 - 主机模式下，发送的 RW 位为 1 - 从机模式下，地址匹配且接收的 RW 位为 0 - 通信复位 - 内部状态复位	MSL	TRA	I2C 运行模式	0b0	0b0	从机接收模式	0b0	0b1	从机发送模式	0b1	0b0	主机接收模式	0b1	0b1	主机发送模式	RW
MSL	TRA	I2C 运行模式																	
0b0	0b0	从机接收模式																	
0b0	0b1	从机发送模式																	
0b1	0b0	主机接收模式																	
0b1	0b1	主机发送模式																	
17	BUSY	总线忙标志位	0b0：空闲状态，总线上无通信 0b1：占有状态，总线正在通信 置位条件： - 检测到总线上开始条件 清零条件： - 检测到总线停止条件 - 通信复位	RO															

位/位域	标记	位名	功能描述	读写
16	MSL	主从机选择位	● 0b0: 从机模式 ● 0b1: 主机模式 通过和 TRA 位的组合, 表示 I2C 的运行模式。 此位可软件置位和清零。 硬件置位条件: ● START 位为 1 的状态下, 检测到开始条件 硬件清零条件: ● 检测到停止条件 ● 仲裁失败 ● 通信复位 ● 内部状态复位	RW
15	Reserved	保留	-	-
14	TMOUTF	超时标志位	● 0b0: 未检测到 SCL 电平超时 ● 0b1: SCL 电平超时 置位条件: ● I2C_SLTR 设定的周期内, SCL 未翻转 清零条件: ● TMOUTFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
13	Reserved	保留	-	-
12	NACKF	NACK 标志位	● 0b0: 未接收到 NACK ● 0b1: 接收到 NACK 置位条件: ● 发送模式下, 接收到 NACK 清零条件: ● NACKFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
11	Reserved	保留	-	-
10	ACKRF	接收应答位	● 0b0: 接收到应答位为“0” (接收 ACK) ● 0b1: 接收到应答位为“1” (接收 NACK) 置位条件: ● 发送模式下, 接收到 NACK 清零条件: ● 发送模式下, 接收到 ACK ● 通信复位 ● 内部状态复位	RO
9	ARLOF	仲裁失败标志位	● 0b0: 未发生仲裁失败 ● 0b1: 仲裁失败 置位条件: ● 仲裁失败 清零条件: ● ARLOFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
8	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
7	EMPTYF	发送数据空标志位	● 0b0: I2C_DTR 寄存器满 ● 0b1: I2C_DTR 寄存器空 置位条件: ● I2C_DTR 数据传送到内部移位寄存器 ● TRA 位置 1 清零条件: ● 写 I2C_DTR ● TRA 位清 0 ● 通信复位 ● 内部状态复位	RO
6	RFULLF	接收数据满标志位	● 0b0: I2C_DRR 寄存器空 ● 0b1: I2C_DRR 寄存器满 置位条件: ● 接收的数据从内部移位寄存器传送到 I2C_DRR 清零条件: ● 读 I2C_DRR ● RFULLFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
5	<i>Reserved</i>	<i>保留</i>	-	-
4	STOPF	停止条件标志位	● 0b0: 总线未检测到停止条件 ● 0b1: 总线检测到停止条件 置位条件: ● 检测到停止条件 清零条件: ● STOPFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
3	TENDF	发送数据结束标志位	● 0b0: 内部移位寄存器发送中 ● 0b1: 内部移位寄存器发送结束 置位条件: ● EMPTYF=1 的条件下, SCL 的第 9 个上升沿此位置“1” 清零条件: ● 检测到停止条件 ● 写 I2C_DTR ● TENDFCLR 写“1” ● 通信复位 ● 内部状态复位	RO
2	SLADDR1F	从机地址寄存器 1 匹配一致标志位	● 0b0: 未检测到从机地址寄存器 1 一致地址 ● 0b1: 检测到从机地址寄存器 1 一致地址 置位条件: ● 当 I2C_SLR1.ADDRM0D1=0 时, 接收到的从机地址与 I2C_SLR1.SLADDR1[7:1]匹配一致 ● 当 I2C_SLR1.ADDRM0D1=1 时, 接收 10 位从机地址的第一个字节地址与 0b11110+I2C_SLR1.SLADDR1[9:8]匹配一致	RO

位/位域	标记	位名	功能描述	读写
2	SLADDR1F	从机地址寄存器 1 匹配一致标志位	并且第二个字节地址与 I2C_SLR1.SLADDR1[7:0]匹配一致 清零条件： ● 检测到停止条件 ● SLADDR1FCLR 写“1” ● 通信复位 ● 内部状态复位	RO
1	SLADDR0F	从机地址寄存器 0 匹配一致标志位	● 0b0：未检测到从机地址寄存器 0 一致地址 ● 0b1：检测到从机地址寄存器 0 一致地址 置位条件： ● 当 I2C_SLR0.ADDRM0=0 时，接收到的从机地址与 I2C_SLR0.SLADDR0[7:1]匹配一致 ● 当 I2C_SLR0.ADDRM0=1 时，接收 10 位从机地址的第一个字节地址与 0b11110+I2C_SLR0.SLADDR0[9:8]匹配一致并且第二个字节地址与 I2C_SLR0.SLADDR0[7:0]匹配一致 清零条件： ● 检测到停止条件 ● SLADDR0FCLR 写“1” ● 通信复位 ● 内部状态复位	RO
0	STARTF	开始条件/重新开始条件标志位	● 0b0：总线未检测到开始条件 ● 0b1：总线检测到开始条件 置位条件： ● 检测到开始条件 清零条件： ● 检测到停止条件 ● STARTFCLR 写“1” ● 通信复位 ● 内部状态复位	RO

30.5.10 I2C 状态清零寄存器 (I2C_CLR)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
Name	Reserved								SMBALRTFCLR				SMBHOSTFCLR				SMBDEFAULTFCLR				GENCALLFCLR				Reserved								TMOUTFCLR				Reserved				NACKFCLR				Reserved				RFREQCLR				ARLOFCLR				Reserved				RFULLFCLR				Reserved				STOPFCLR				TENDFCLR				SLADDR1FCLR				SLADDR0FCLR				STARTFCLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
Access									WO				WO				WO				WO												WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO								WO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	SMBALRTFCLR	SMBUS 报警响应地址匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
22	SMBHOSTFCLR	SMBUS 主机地址匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
21	SMBDEFAULTFCLR	SMBUS 默认地址匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
20	GENCALLFCLR	广播呼叫地址匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
19:15	Reserved	保留	-	-
14	TMOUTFCLR	超时标志清除	写 1 清除标志位, 写 0 无效	WO
13	Reserved	保留	-	-
12	NACKFCLR	NACK 标志清除	写 1 清除标志位, 写 0 无效	WO
11	Reserved	保留	-	-
10	RFREQCLR	接收 FIFO 数据请求标志清除	写 1 清除 RFREQF, 写 0 无效	WO
9	ARLOFCLR	仲裁失败标志清除	写 1 清除标志位, 写 0 无效	WO
8:7	Reserved	保留	-	-
6	RFULLFCLR	接收数据满标志清除	写 1 清除标志位, 写 0 无效	WO
5	Reserved	保留	-	-
4	STOPFCLR	停止条件标志清除	写 1 清除标志位, 写 0 无效	WO
3	TENDFCLR	发送数据结束标志清除	写 1 清除标志位, 写 0 无效	WO
2	SLADDR1FCLR	从机地址寄存器 1 匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
1	SLADDR0FCLR	从机地址寄存器 0 匹配一致标志清除	写 1 清除标志位, 写 0 无效	WO
0	STARTFCLR	开始条件/重新开始条件标志清除	写 1 清除标志位, 写 0 无效	WO

30.5.11 I2C 数据发送寄存器 (I2C_DTR)

Offset	Bit Position							
0x24	7	6	5	4	3	2	1	0
Reset	0xFF							
Name	DT							
Access	RW							

位/位域	标记	位名	功能描述	读写
7:0	DT	发送数据	如果内部移位寄存器为空，就将写在 I2C_DTR 寄存器的发送数据传送到内部移位寄存器，发送模式时开始发送数据到 SDA 上	RW

内部移位寄存器和 I2C_DTR 寄存器是双缓冲结构，在内部移位寄存器数据发送过程中，如果预先写 I2C_DTR 寄存器的数据，就能进行连续发送数据。

I2C_DTR 寄存器可读可写。请在发送数据空中断请求发生时，仅写一次 I2C_DTR 寄存器。

I2C 工作在 FIFO 模式时，访问 I2C_DTR 寄存器地址可以直接访问发送 FIFO，如果当前 TEMPTYF 有效，针对 I2C_DTR 寄存器地址的写操作可以将当前的总线数据写入发送 FIFO。当发送 FIFO 有效并且当前 I2C 允许发送则可以将发送 FIFO 数据传送到内部移位寄存器并开始发送数据到 SDA 上。

发送数据源的选择通过 I2C 中 FIFO 状态寄存器 (I2C_FSTR) 里面的 FEN 位来选择：

- 当 FEN 为 1 时选择外部数据总线到 FIFO 数据通路以及 FIFO 到内部移位寄存器的数据通路。
- 当 FEN 为 0 时选择外部数据总线到 I2C_DTR 寄存器通路以及 I2C_DTR 寄存器到内部移位寄存器的数据通路。

默认选择非 FIFO 模式。

30.5.12 I2C 数据接收寄存器 (I2C_DRR)

Offset	Bit Position							
0x28	7	6	5	4	3	2	1	0
Reset	0x00							
Name	DR							
Access	RO							

位/位域	标记	位名	功能描述	读写
7:0	DR	接收数据	如果接收到 1 帧数据，就能将接收数据从内部移位寄存器转存到 I2C_DRR 寄存器，进而可以进入到下一个数据接收状态	RO

内部移位寄存器和 I2C_DRR 寄存器是双缓冲结构，在内部移位寄存器数据接收过程中，如果读取了 I2C_DRR 寄存器的数据，就能进行连续接收数据。

禁止对 I2C_DRR 寄存器写。请在接收数据满中断要求发生时，仅读一次 I2C_DRR 寄存器。

在 I2C_SR.RFULLF 标志位为 1 的状态下，如果不读取 I2C_DRR 寄存器的数据，而立即接收下一个数据，SCL 时钟将在该数据接收的第 7 个或第 8 个下降沿后（参考[从机接收时序图](#)和[快速 ACK/NACK 时序图](#)）自动保持为低电平，直至 I2C_DRR 寄存器的数据被读取。

I2C 工作在 FIFO 模式时，访问 I2C_DRR 寄存器地址可以直接访问接收 FIFO：

- 如果接收到 1 帧数据，就能将接收数据从内部移位寄存器转存到接收 FIFO。当接收 FIFO 收到有效数据之后将输出 RFREQ 请求状态信号，此时通过针对 I2C_DRR 寄存器地址的读操作就可以直接访问接收 FIFO 中的数据。
- 在 I2C_SR.RFULLF 标志位为 1 的状态下（即接收 FIFO 已经存储了 2Bytes 数据，此时 FIFO 已满），如果不读取接收 FIFO 中的数据而立即接收下一个数据，SCL 时钟将在该数据接收的第 7 个或第 8 个下降沿后自动保持为低电平，直至接收 FIFO 的数据被读取。

I2C 接收通路的选择通过 I2C 中 FIFO 状态寄存器里面的 FEN 位来选择：

- 当 FEN 为 1 时选择内部移位寄存器到接收 FIFO 数据通路以及接收 FIFO 到数据总线的通路。
- 当 FEN 为 0 时选择内部移位寄存器到 I2C_DRR 的数据通路以及 I2C_DRR 到数据总线的通路。

默认选择非 FIFO 功能。

30.5.13 I2C 时钟控制寄存器 (I2C_CCR)

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00001F1F																															
Name	Reserved														CKDIV	SHIGHW								SLOWW								
Access															RW	RW								RW								

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18:16	CKDIV	I2C 基准时钟频率设定位	0b000: I2C 基准时钟频率=PCLK3 0b001: I2C 基准时钟频率=PCLK3/2 0b010: I2C 基准时钟频率=PCLK3/4 0b011: I2C 基准时钟频率=PCLK3/8 0b100: I2C 基准时钟频率=PCLK3/16 0b101: I2C 基准时钟频率=PCLK3/32 0b110: I2C 基准时钟频率=PCLK3/64 0b111: I2C 基准时钟频率=PCLK3/128	RW
15:8	SHIGHW	设定 SCL 高电平宽度位	设定 SCL 时钟的高电平宽度	RW
7:0	SLOWW	设定 SCL 低电平宽度位	设定 SCL 时钟的低电平宽度	RW

波特率计算如下：

I2C_FLTR.DNFEN	I2C_CCR.CKDIV	波特率
0b0	0b000	$B = \frac{1}{\frac{(SHIGHW + 3) + (SLOWW + 3)}{\Phi_{I2C}} + t_R + t_F}$
0b1	0b000	$B = \frac{1}{\frac{(SHIGHW + 3 + DNF) + (SLOWW + 3 + DNF)}{\Phi_{I2C}} + t_R + t_F}$
0b0	非 0	$B = \frac{1}{\frac{(SHIGHW + 2) + (SLOWW + 2)}{\Phi_{I2C}} + t_R + t_F}$
0b1	非 0	$B = \frac{1}{\frac{(SHIGHW + 2 + DNF) + (SLOWW + 2 + DNF)}{\Phi_{I2C}} + t_R + t_F}$

其中：

- B：波特率，单位：Mbps
- Φ_{I2C} ：I2C 基准时钟频率，单位：MHz
- DNF：数字滤波器滤波能力，通过寄存器 I2C_FLTR.DNF 设置
- t_R ：SCL 线的上升时间，单位：ns
- t_F ：SCL 线的下降时间，单位：ns



说明

- 在主机模式下，SHIGHW 用于设定 SCL 时钟的高电平宽度。在从机模式下，设定无效。
- 在主机模式下，SLOWW 用于设定 SCL 时钟的低电平宽度。在从机模式下，设定值要大于数据 Setup 时间 ($t_{SU,DAT}$)，对于标准模式为 250ns，对于快速模式为 100ns。
- SCL 线的上升时间 (t_R) 和下降时间 (t_F) 取决于总线的总电容量 (C_b) 和上拉电阻 (R_p)，详细内容请参见 I2C 总线规范。

30.5.14 I2C 滤波控制寄存器（I2C_FLTR）

Offset	Bit Position																															
0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000010																															
Name	Reserved																												Reserved			
																											RW	ANFEN			RW	DNFEN
Access																											RW	RW		RW		

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	ANFEN	模拟滤波功能允许位	● 0b0：模拟滤波功能禁止 ● 0b1：模拟滤波功能允许	RW
4	DNFEN	数字滤波功能允许位	● 0b0：数字滤波功能禁止 ● 0b1：数字滤波功能允许	RW
3:2	Reserved	保留	-	-
1:0	DNF	数字滤波器滤波能力选择	● 0b00：滤波能力 1 个 I2C 基准时钟周期 ● 0b01：滤波能力 2 个 I2C 基准时钟周期 ● 0b10：滤波能力 3 个 I2C 基准时钟周期 ● 0b11：滤波能力 4 个 I2C 基准时钟周期	RW

30.5.15 I2C FIFO 状态寄存器 (I2C_FSTR)

Offset	Bit Position																																																											
0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																												
Reset	0x00000000																																																											
Name	Reserved																								RFST		TFST		NACK_TFFLUSH		RFLUSH		TFLUSH		FEN																									
RFLUSH																													TFLUSH		FEN																													
Access																																																	RO		RO		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:6	RFST	接收 FIFO 状态位	0b00: FIFO 空 0b01: FIFO 半满 0b10: FIFO 满 0b11: Reserved	RO
5:4	TFST	发送 FIFO 状态位	0b00: FIFO 空 0b01: FIFO 半空 0b10: FIFO 满 0b11: Reserved	RO
3	NACK_TFFLUSH	NACK 复位发送 FIFO	0b0: 从模式下接收到 NACK 后不执行 TFIFO FLUSH 0b1: 从模式下接收到 NACK 后执行 TFIFO FLUSH	RW
2	RFLUSH	RX FIFO 指针复位	0b0: 写入无效 0b1: 复位 RX FIFO 指针	RW
1	TFLUSH	TX FIFO 指针复位	0b0: 写入无效 0b1: 复位 TX FIFO 指针	RW
0	FEN	FIFO 功能允许位	0b0: FIFO 功能禁止 0b1: FIFO 功能允许	RW

30.5.16 I2C 从机地址接收器寄存器 (I2C_SLVADDR)

Offset	Bit Position																															
0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				SLVADR											
Access																					RO											

位/位域	标记	位名	功能描述	读写
31:10	Reserved	保留	-	-
9:0	SLVADR	从机地址接收寄存器	SLVADR[6:0]: 7 位从机接收地址 SLVADR[9:0]: 10 位从机接收地址	RO

31 串行外设接口 (SPI)

31.1 简介

本产品搭载 6 个通道的串行外设接口 SPI，支持高速全双工串行同步传输，方便地与外围设备进行数据交换。用户可根据需要进行三线/四线，主机/从机及波特率范围的设置。

31.2 主要特性

表 31-1 SPI 主要特性

要点	描述
串行通信功能	● 支持 4 线式 SPI 模式和 3 线式时钟同步运行模式 ● 支持全双工和只发送两种通信方式 ● 可调整通信时钟 SCK 的极性和相位
数据格式	● 可选择数据移位顺序：MSB 开始/LSB 开始 ● 可选择数据宽度：4/5/6/7/8/9/10/11/12/13/14/15/16/20/24/32 位 ● 单次最多可传送或接收 4 帧宽度为 32 位的数据
波特率	● 主机模式下可通过内置专用波特率发生器对波特率进行调整，波特率范围为 PCLK1 的 2 分频~256 分频 ● 从机模式下允许的最大波特率为 PCLK1 的 6 分频
数据缓冲	● 带有 16 字节的数据缓冲器 ● 支持双重缓冲
错误监测	● 模式故障错误监测 ● 数据过载错误监测 ● 数据欠载错误监测 ● 奇偶校验错误监测
片选信号控制	● 每个通道配置四根片选信号线 ● 可对片选信号和通信时钟的相对时序关系进行调整 ● 可对连续两次通信之间的片选信号无效时间进行调整 ● 极性可调
主机模式下的传输控制	● 通过将数据写入数据寄存器启动传输 ● 通信自动挂起功能
中断	● 接收缓冲器已满 ● 发送缓冲器已空 ● SPI 错误（模式/过载/欠载/奇偶校验） ● SPI 空置 ● 传输完成（仅为事件源）
低功耗控制	● 可设置模块停止

要点	描述
其他功能	● SPI 初始化功能

说明

在主接收模式使用通信自动挂起功能时，由于通信时钟停止，将不会发生过载错误。详情请参见“[过载错误](#)”。

31.3 功能说明

31.3.1 系统框图

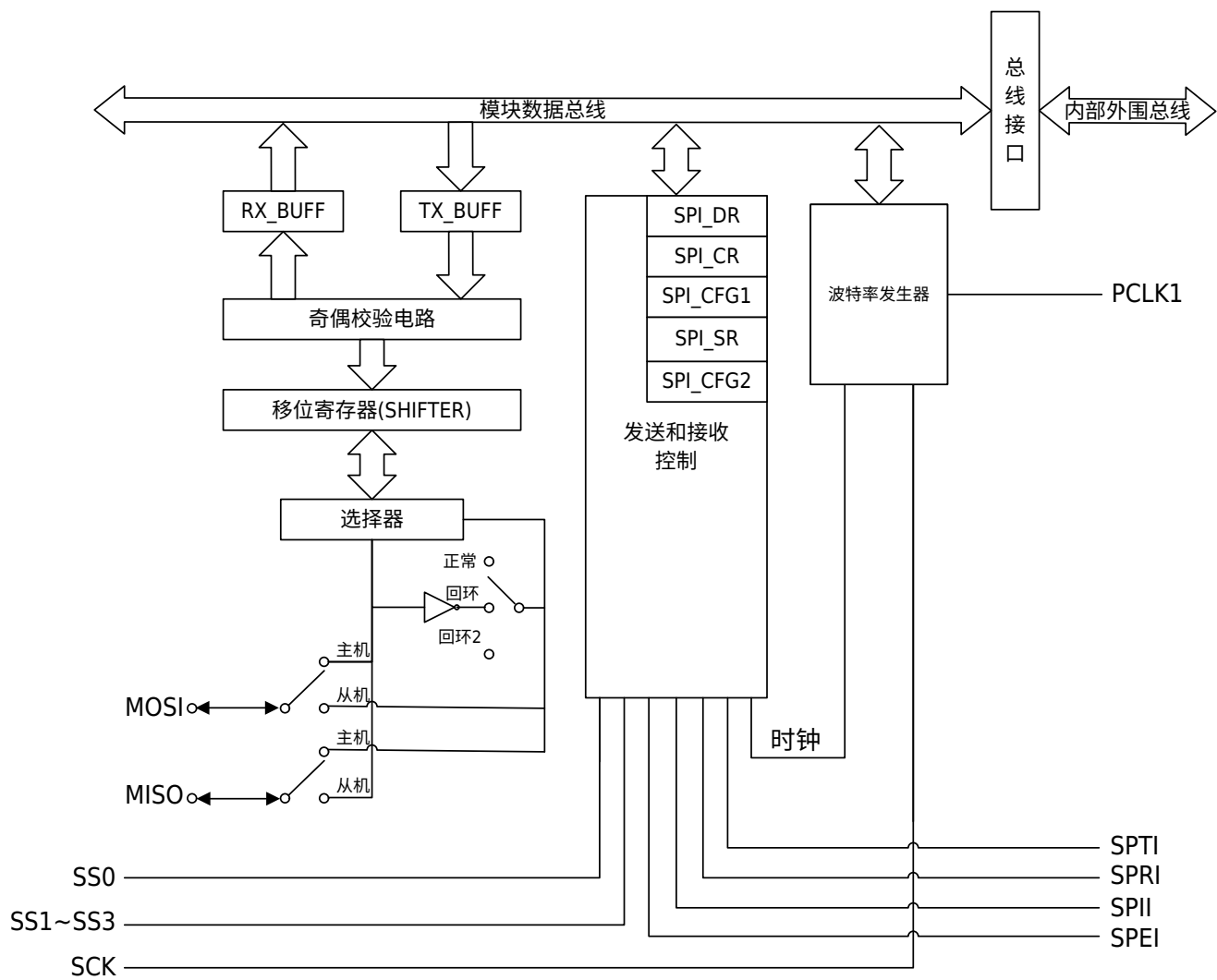


图 31-1 SPI 系统框图

表 31-2 管脚说明

管脚名	端口方向	功能
SCK	输入/输出	通信时钟管脚
MOSI	输入/输出	主机数据传输管脚
MISO	输入/输出	从机数据传输管脚
SS0	输入/输出	从机选择输入/输出管脚

管脚名	端口方向	功能
SS1	输出	从机选择输出管脚
SS2	输出	从机选择输出管脚
SS3	输出	从机选择输出管脚

31.3.2 SPI 动作系统说明

31.3.2.1 主机模式的管脚状态

SPI 工作在主机模式下时，各个管脚的状态如下表所示。

表 31-3 主机模式时 SPI 管脚状态说明

模式		管脚名	管脚状态
SPI 动作（SPIMDS=0）	主机模式（MSTR=1、MODFE=0）	SCK	输出
		SS0~SS3	输出
		MOSI	输出
		MISO	输入
时钟同步运行（SPIMDS=1）	主机模式（MSTR=1）	SCK	输出
		SS0~SS3（不使用）	Hi-Z（可作为通用 I/O）
		MOSI	输出
		MISO	输入

说明

管脚输入类型请设定为 CMOS 输入，输出类型请设定为高驱动力模式，相关设定请参见《通用 IO（GPIO）》“寄存器-通用控制寄存器（PCRxy）”。

31.3.2.2 从机模式的管脚状态

SPI 工作在从机模式下时，各个管脚的状态如下表所示。

表 31-4 从机模式时 SPI 管脚状态说明

模式		管脚名	管脚状态
SPI 动作（SPIMDS=0）	从机模式（MSTR=0、MODFE=0）	SCK	输入
		SS0	输入
		SS1~SS3（不使用）	Hi-Z（可作为通用 I/O）
		MOSI	输入
		MISO	输出/Hi-Z

模式		管脚名	管脚状态
时钟同步运行 (SPIMDS=1)	从机模式 (MSTR=0)	SCK	输入
		SS0~SS3 (不使用)	Hi-Z (可作为通用 I/O)
		MOSI	输入
		MISO	输出

说明

管脚输入类型请设定为 CMOS 输入，输出类型请设定为高驱动力模式，相关设定请参见《通用 IO（GPIO）》“寄存器-通用控制寄存器（PCRxy）”。

31.3.2.3 SPI 系统连接实例

主机模式

在主机-多从机模式的 SPI 系统结构中，主机驱动 SCK、MOSI 和 SS0~SS3。在 SPI 从机设备 0~3 中，当某一从机的 SS 输入为有效电平时，该从机设备驱动 MISO。

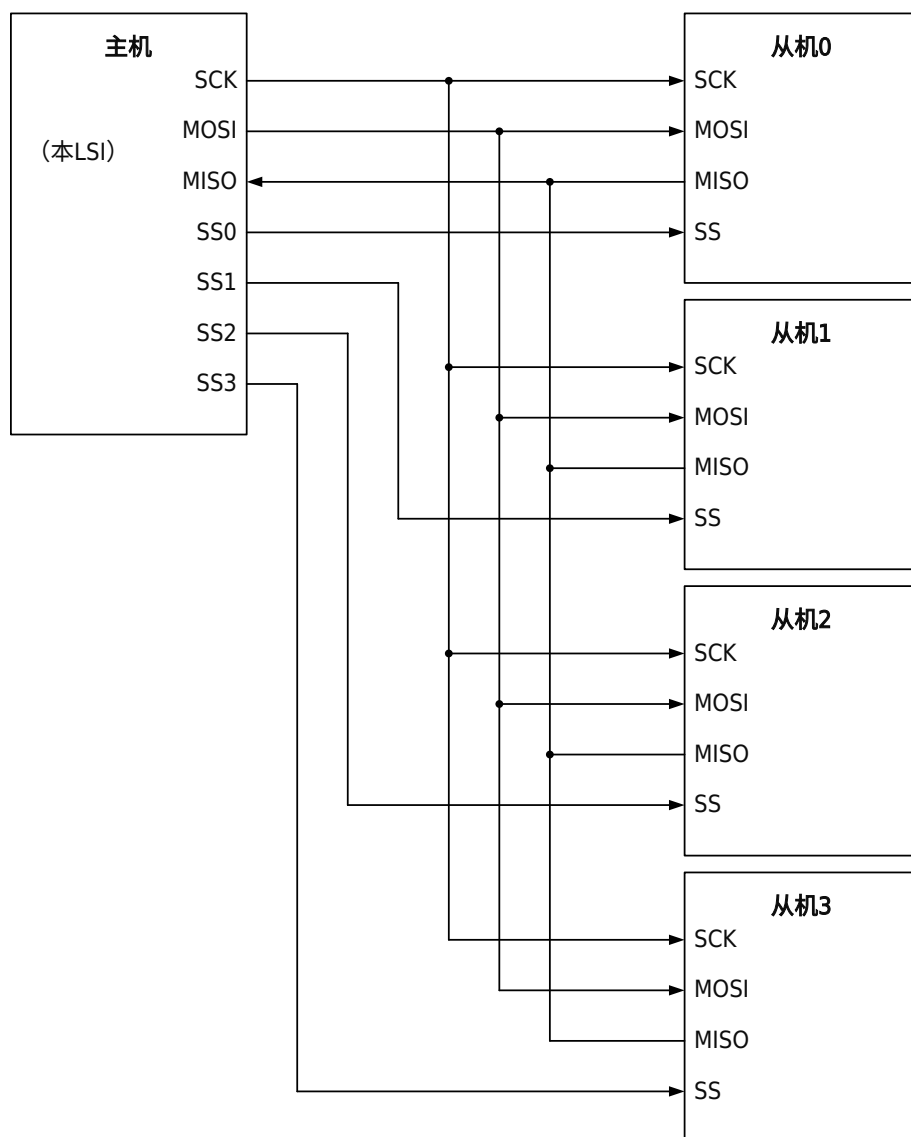


图 31-2 主机模式结构

时钟同步运行

在用作时钟同步运行动作的 SPI 系统结构中，主机设备驱动 SCK 和 MOSI，从机设备驱动 MISO。SS 管脚不作使用。

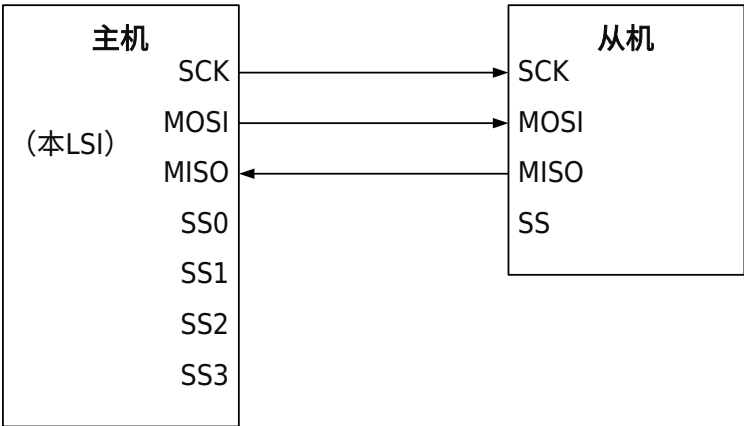


图 31-3 三线式时钟同步运行

31.3.3 数据通信说明

31.3.3.1 波特率

主机模式下，SPI 时钟由内部波特率发生器提供；从机模式下，时钟为 SCK 脚输入。

波特率取决于 SPI_CFG2.MBR[1:0]位的设置。其计算方法如下面公式所示：

波特率 = $\frac{f_{PCLK1}}{2^{N+1} \times (M+1)}$

其中：

- N：MBR[1:0]位的设定值，范围是 0~3；
- M：CLKDIV[3:0]位的设定值，范围是 0~15。

表 31-5 部分设定值的位速

MBR[1:0]位的设定值 (N)	CLKDIV[3:0]位的设定值 (M)	分频比	波特率			
			PCLK1=5MHz	PCLK1=10MHz	PCLK1=20MHz	PCLK1=40MHz
0	0	2	2.50Mbps	5.00Mbps	10.0Mbps	20.0Mbps
0	1	4	1.25Mbps	2.50Mbps	5.00Mbps	10.0Mbps
0	3	8	625kbps	1.25Mbps	2.50Mbps	5.00Mbps
0	7	16	313kbps	625kbps	1.25Mbps	2.50Mbps
0	15	32	156kbps	313kbps	625kbps	1.25Mbps
1	15	64	78kbps	156kbps	313kbps	625kbps
2	15	128	39kbps	78kbps	156kbps	313kbps
3	15	256	20kbps	39kbps	78kbps	156kbps

31.3.3.2 数据格式

SPI 的数据格式取决于 SPI 命令寄存器 SPI_CFG2 和 SPI 控制寄存器 SPI_CR 中的奇偶校验许可位 PAE 的设定值。SPI 将数据寄存器 SPI_DR 中从 LSB 位开始一定长度（数据长度由寄存器 SPI_CFG2 中的 DSIZE[3:0]位设置）的数据作为传送对象进行处理，与 MSB/LSB 移位顺序无关。

SPI_CFG2.DSIZE[3:0]决定数据的位宽，位宽范围为 4~32 位，SPI_CR.PAE 决定数据最末位，PAE 为 1 时最末位作为奇偶校验位，为 0 时则是数据本身最低位。

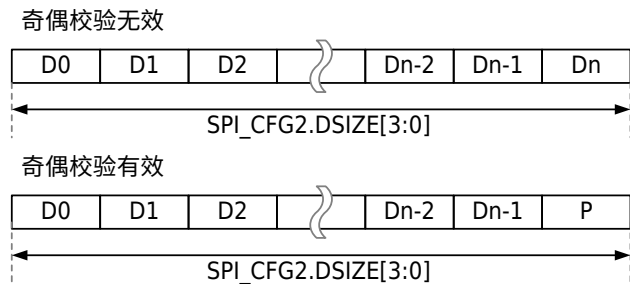


图 31-4 数据格式

SPI 数据发送时，传送数据先进入发送缓冲器（TX_BUFF），再将 TX_BUFF 的数据复制到移位寄存器（SHIFTER），SHIFTER 依次发出数据；SPI 数据接收时，数据从 SHIFTER 依次移入，移入完成后再将 SHIFTER 的数据复制到接收缓冲器（RX_BUFF）。

数据传输时，根据移位顺序控制位 SPI_CFG2.LSBF 和奇偶校验控制位 SPI_CR.PAE 的设置分为 4 种情况：

● MSB 先传，奇偶校验无效

发送时，数据 D31~D0 按照顺序从 TX_BUFF 复制到 SHIFTER，按照 D31~D0 的顺序从 SHIFTER 的最高位移出；

接收时，数据 D31~D0 从 SHIFTER 的最低位移入，等到数据全部移入后再将数据复制到 RX_BUFF。

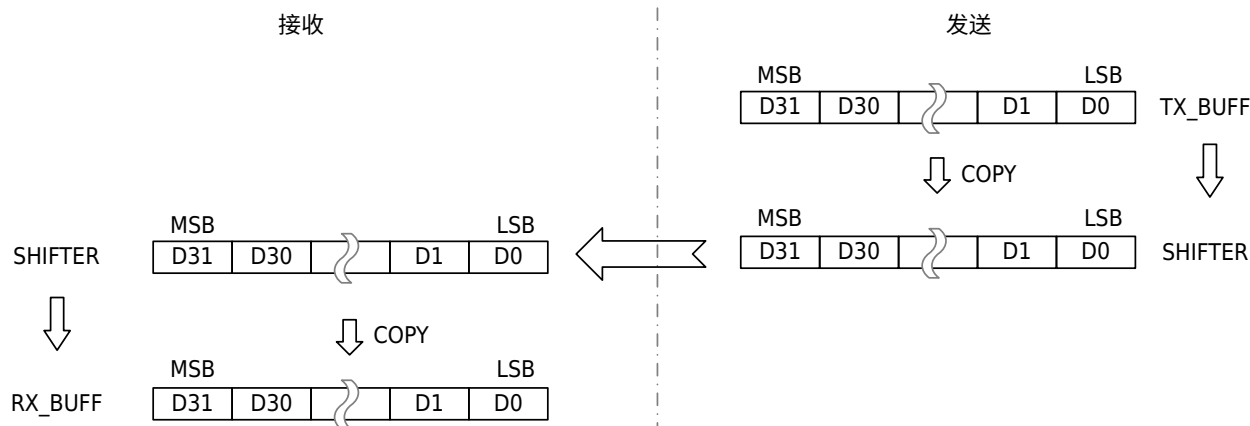


图 31-5 MSB 先传，奇偶校验无效

● LSB 先传，奇偶校验无效

发送时，数据 D31~D0 按照 D0~D31 的顺序从 TX_BUFF 复制到 SHIFTER，按照 D0~D31 的顺序从 SHIFTER 的最高位移出；

接收时，数据 D0~D31 从 SHIFTER 的最低位移入，等到数据全部移入后，按照 D31~D0 的顺序从 SHIFTER 复制到 RX_BUFF。

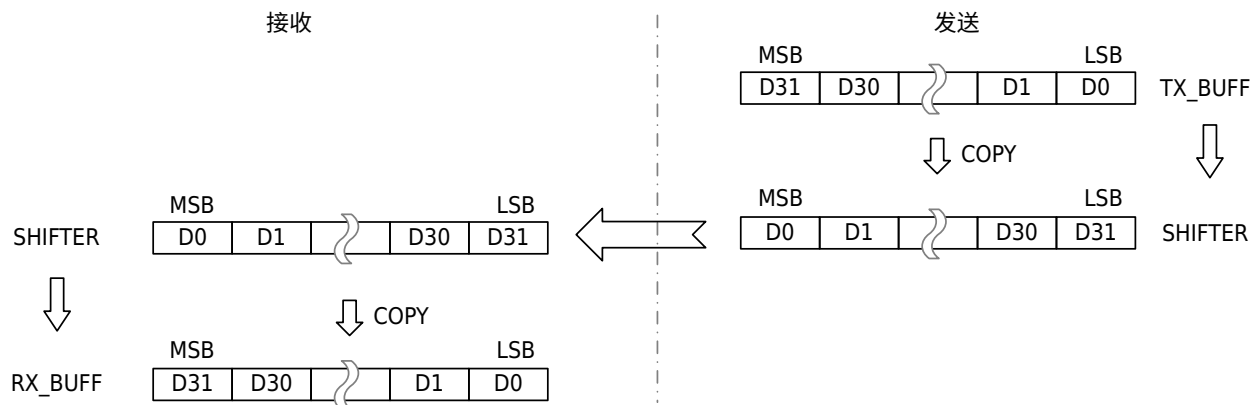


图 31-6 LSB 先传，奇偶校验无效

● MSB 先传，奇偶校验有效

发送时，先根据 D31~D1 的值计算出奇偶校验位 P 的值，然后用 P 代替 D0，按照 D31~D1，P 的顺序复制到 SHIFTER，按照 D31~P 的顺序从 SHIFTER 的最高位移出；
接收时，数据 D31~P 从 SHIFTER 的最低位移入，在数据复制到 SHIFTER 时，进行奇偶校验。最后再将数据复制到 RX_BUFF。

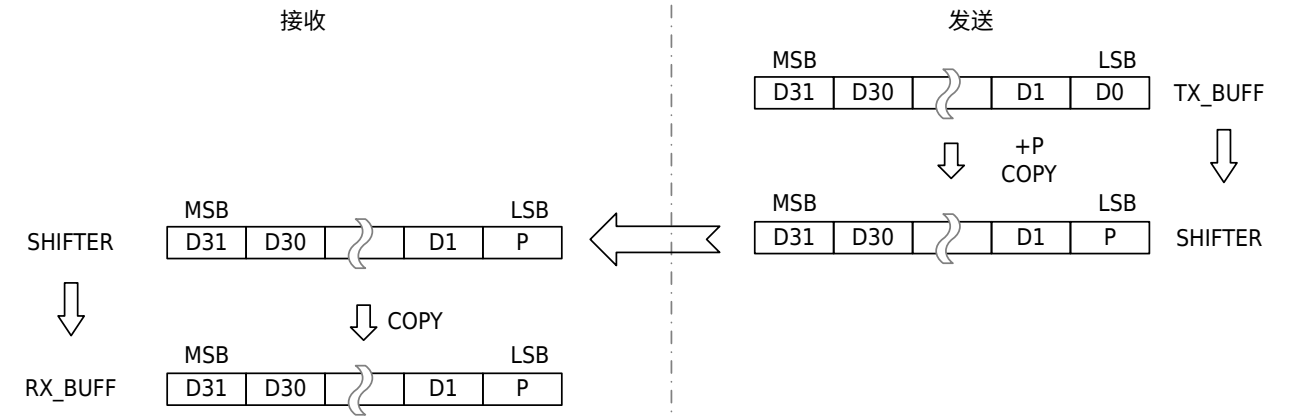


图 31-7 MSB 先传，奇偶校验有效

● LSB 先传，奇偶校验有效

发送时，先根据 D30~D0 的值计算出奇偶校验位 P 的值，然后用 P 代替 D31，按照 D0~P 的顺序从 TX_BUFF 复制到 SHIFTER，按照 D0~P 的顺序从 SHIFTER 的最高位移出；
接收时，数据 D0~P 从 SHIFTER 的最低位移入，在数据复制到 SHIFTER 时，进行奇偶校验。数据 D0~P 在复制时重新排列，按照 P~D0 的顺序复制到 RX_BUFF。

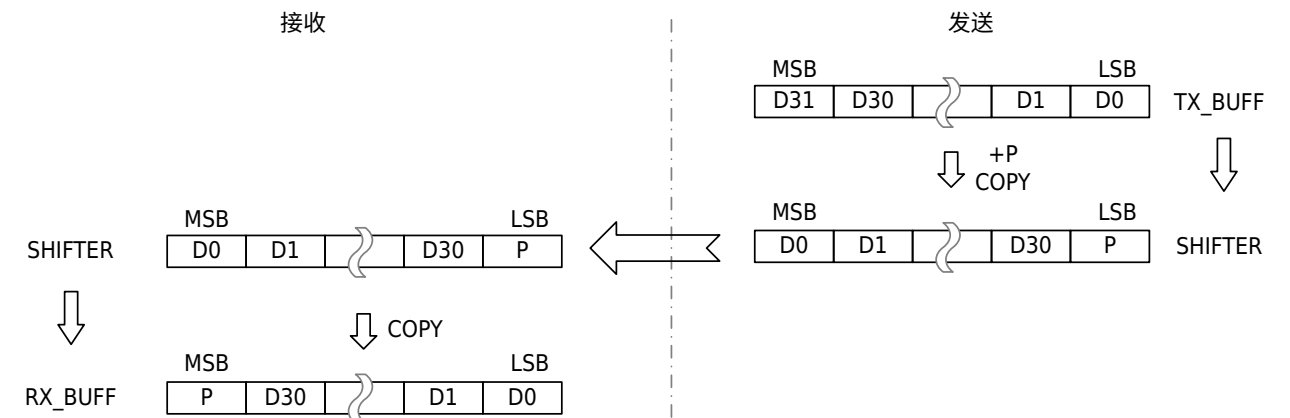


图 31-8 LSB 先传，奇偶校验有效

31.3.3.3 传送格式

CPHA=0 的情况

当 SPI_CFG2.CPHA 位为 0 时，SPI 在 SCK 的奇数边沿进行数据采样，偶数边沿进行数据更新。下图是 CPHA=0 时 SPI 的传送时序图。当 SSI 信号的输入电平变为有效电平时，MOSI/ MISO 开始更新传输数据。在 SSI 信号变为有效后的第一个 SCK 信号边沿进行第一次数据采样，在此之后，每过一个 SCK 周期对数据进行一次采样。每次采样后的 1/2 个 SCK 周期时 MOSI/ MISO 信号上数据进行更新。CPOL 位的设定值不影响 SCK 信号的运行时序，而只影响信号的极性。

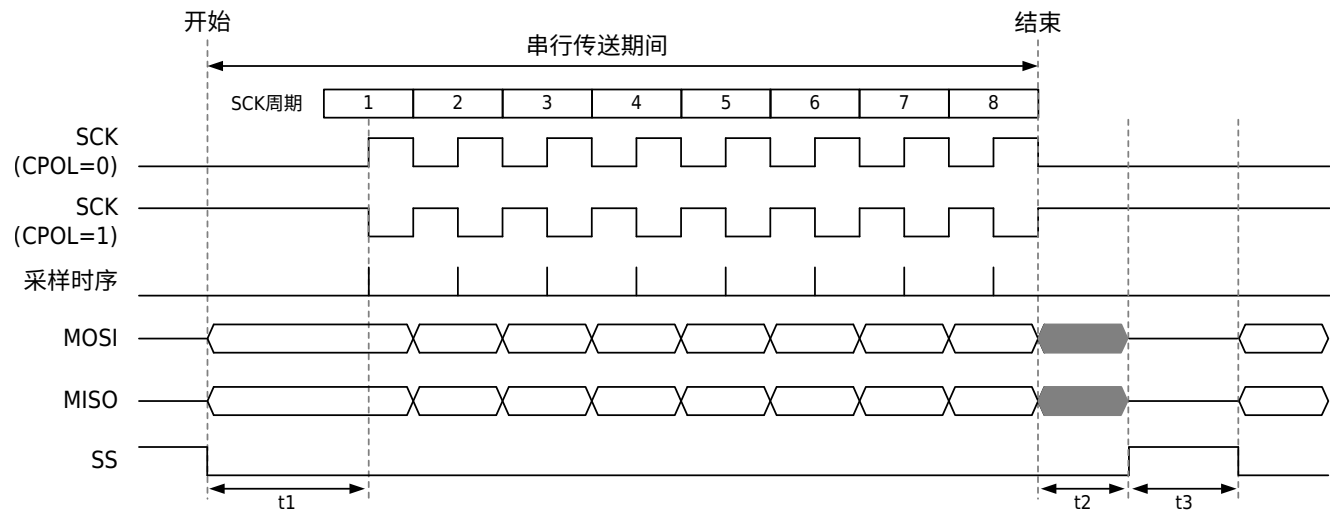


图 31-9 数据传送格式图 (CPHA=0)

上图中 t1 表示从 SS 信号有效到 SCK 振荡的间隔时间 (SCK 延迟时间, 可通过设置 SPI_CFG1.MSSI[2:0]和 SPI_CFG2.MSSIE 进行控制), t2 表示从 SCK 振荡停止到 SS 信号变成无效的间隔期间 (SS 无效延迟时间, 它可以通过设置 SPI_CFG1.MSSDL[2:0]和 SPI_CFG2.MSSDLE 来控制), t3 表示在串行传送结束后到下次传送开始的最小等待时间 (下次存取延迟, 可以通过设置 SPI_CFG1.MIDI[2:0]和 SPI_CFG2.MIDIE 来控制)。t1、t2 和 t3 由 SPI 系统上的主机设备进行控制。详细请参见“SPI 运行模式时的主机动作”。

CPHA=1 的情况

当 SPI_CFG2.CPHA 位为 1 时, SPI 在 SCK 的奇数边沿进行数据更新, 偶数边沿进行数据采样。下图是 CPHA=1 时 SPI 的传送时序图。MOSI/ MISO 在 SSi 信号变为有效电平后的第一个 SCK 信号边沿时开始传输数据的更新。在此之后, 每过一个 SCK 周期对数据进行一次更新。每次更新后的 1/2 个 SCK 周期时对数据进行采样。SPI_CFG2.CPOL 位的设定值不影响 SCK 信号的运行时序而只影响信号的极性。

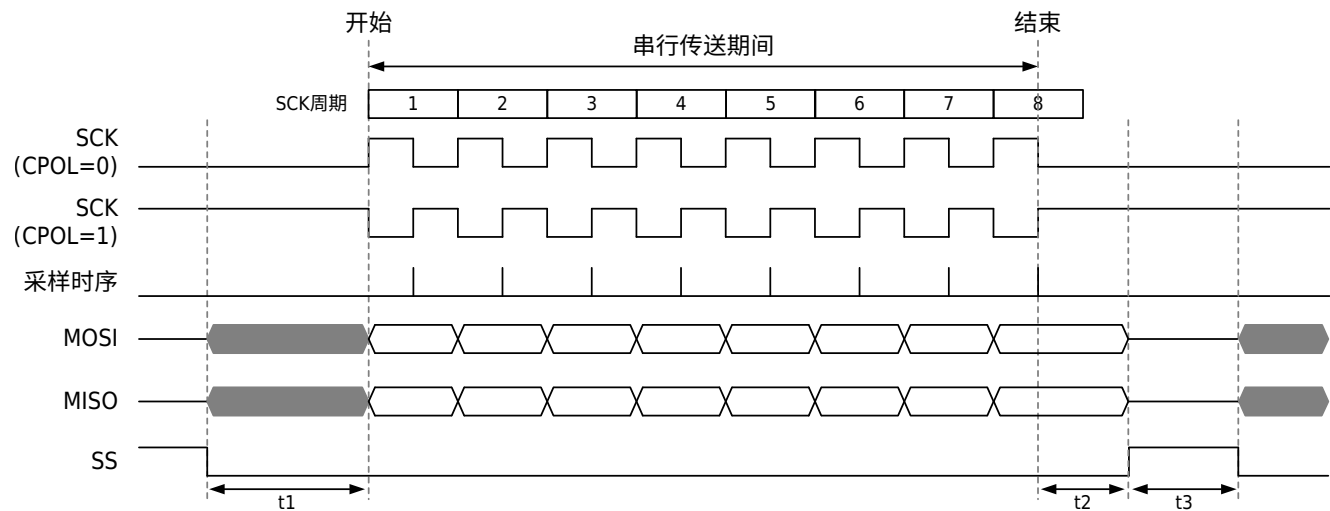


图 31-10 数据传送格式 (CPHA=1)

t1、t2、t3 和 CPHA 位为 0 的情况相同。

31.3.3.4 通信方式

SPI 有全双工同步串行通信和只发送串行通信两种通信方式, 可通过 SPI 控制寄存器 (SPI_CR) 的 TXMDS 位进行选择。

全双工同步串行通信方式

当 SPI_CR.TXMDS 位为 0 时, SPI 运行在全双工同步串行通信方式。如下图所示, SPI_CFG1.FTHLV[1:0]位为 0b00, SPI_CFG2.CPHA 位为 1 并且 SPI_CFG2.CPOL 位为 0, SPI 进行 8 位串行传送。

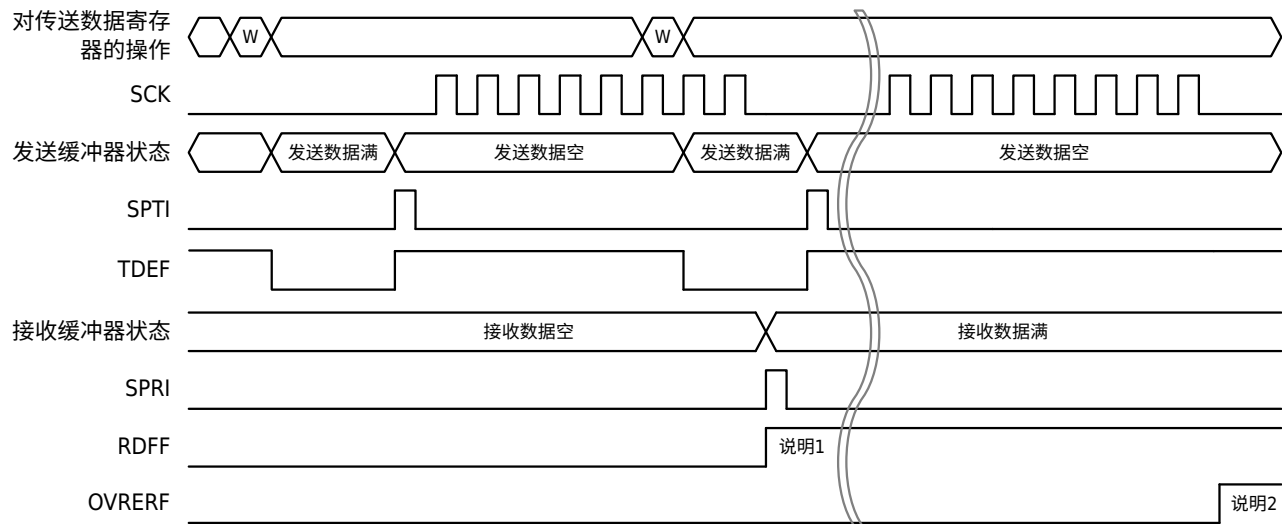


图 31-11 全双工同步串行通信



说明

1. 当本次串行传输结束时，如果接收缓冲器为空，SPI 将会把接收到的数据从移位寄存器复制到接收缓冲器中，接收缓冲器满的标志位 RDFF 被置成 1，并产生一个接收数据满的中断请求 SPRI。
2. 当本次串行传输结束时，如果接收缓冲器中还保持着上次收到的数据而没有被系统读取，SPI 会将数据过载标志位 OVRERF 置成 1，本次数据接收无效，接收移位寄存器中的数据将被丢弃。

只发送通信方式

当 SPI_CR.TXMDS 位为 1 时，SPI 运行在只发送通信方式。如下图所示，SPI_CFG1.FTHLV[1:0]位为 0b00，SPI_CFG2.CPHA 位为 1 并且 SPI_CFG2.CPOL 位为 0，SPI 进行 8 位串行传送。

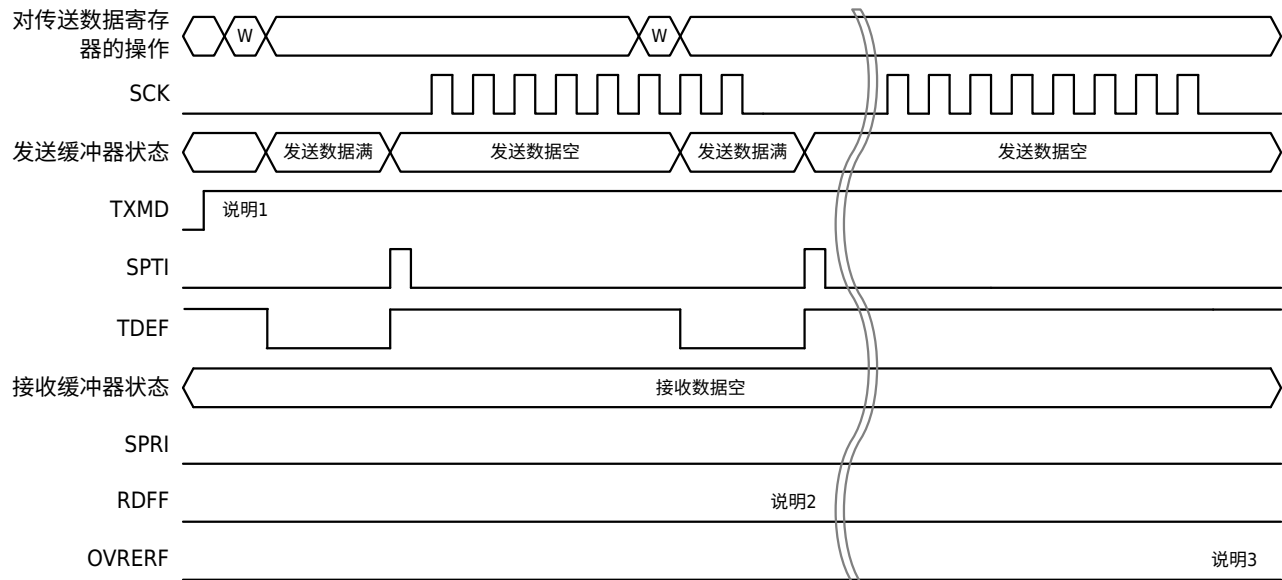


图 31-12 只进行发送通信



说明

1. 在设置进入只发送通信方式之前，请确保接收缓冲器中没有未读取的数据（RDFF=0）且未出现数据过载错误（OVRERF=0）。
2. 在只发送通信方式下，当本次串行传输结束时，即使接收缓冲器为空，也不会有数据被接收，RDFF 将始终保持为 0。
3. 在只发送通信方式下，由于接收缓冲器始终为空，故不会发生数据过载错误，OVRERF 标志位始终保持为 0。

31.3.3.5 通信间隔时间

常规设定下的通信间隔时间

当 SPI_CFG1.CTMDS 位为 0 时，SPI 使用常规通信设定。连续两次通信之间的间隔时间可以通过 SPI_CFG1.MIDI[2:0]和 SPI_CFG2.MIDIE 来进行设定，最小间隔时间为 $1 \times SCK + 2 \times PCLK1$ 。如下图所示，SPI_CFG2.CPHA 位为 1 并且 SPI_CFG2.CPOL 位为 0，SPI 进行串行传送。

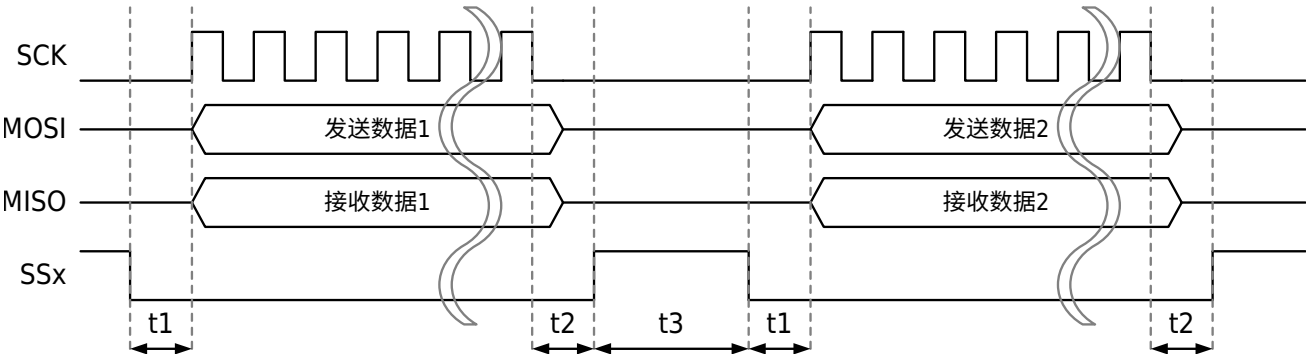


图 31-13 常规通信

在上图中，t1 表示从 SS 信号有效到 SCK 第一边沿的时间间隔（SCK 延迟时间，可通过设置 SPI_CFG1.MSSI[2:0]和 SPI_CFG2.MSSIE 进行控制），t2 表示从 SCK 最后边沿到 SS 信号无效的时间间隔（SS 无效延迟时间，它可以通过设置 SPI_CFG1.MSSDL[2:0]和 SPI_CFG2.MSSDLE 来控制），t3 表示从本次传输结束到下一次传输开始的最小等待时间（下次存取数据延迟，可以通过设置 SPI_CFG1.MIDI[2:0]和 SPI_CFG2.MIDIE 来控制）。

 **说明**

当通信速率为 PCLK1 的 2 分频（SPI_CFG2.MBR[1:0]=0b00 且 SPI_CFG2.CLKDIV[3:0]=0b0000）时，请将 t3 配置为 $2 \times SCK + 2 \times PCLK1$ 及以上，即 SPI_CFG2.MIDIE=1 且 SPI_CFG1.MIDI[2:0]=0x1~0x7。

连续通信设定下的通信间隔时间

当 SPI_CFG1.CTMDS 位为 1 时，SPI 使用连续通信设定。可实现完全连续不间断的数据通信。如下图所示，SPI_CFG2.CPHA 位为 1 并且 SPI_CFG2.CPOL 位为 0，SPI 进行串行传送。

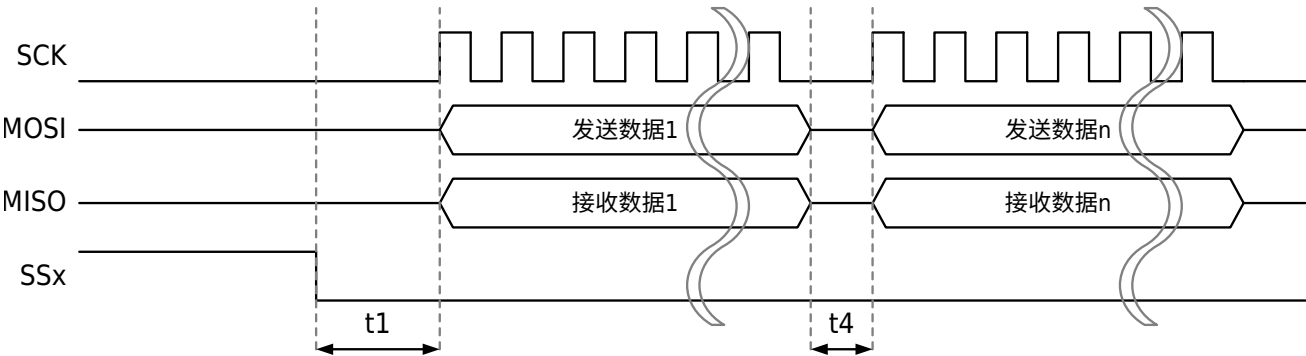


图 31-14 连续通信

在上图中，t1 表示从 SS 信号有效到 SCK 第一边沿的时间间隔（SCK 延迟时间，可通过设置 SPI_CFG1.MSSI[2:0]和 SPI_CFG2.MSSIE 进行控制），t4 表示每次传输之间的时间间隔，可通过设置 SPI_CFG1.MSSI[2:0]和 SPI_CFG2.MSSIE 进行控制。t1 和 t4 之间的关系定义为 $t4 = t1 - 1 \times SCK$ 。



说明

- 连续通信模式下，主机如果没有新的数据写入发送缓冲器，则将进入暂停状态，SCK 停止输出，SS 信号保持有效状态，等待新的发送数据写入发送缓冲器后继续下一帧数据的传输。
- 连续通信模式下，从机如果没有新的数据写入发送缓冲器，则将发生欠载错误，SPI_SR.MODFERF 和 SPI_SR.UDRERF 将置 1，详细描述请参见“欠载错误”。
- 时钟同步模式下的连续通讯，如果使用 CPHA=0 的传送格式，在要保证数据在 CPHA 设置成 0 之前写入。
- 使用 CPHA=0 的传送格式时，从机需要在上一帧数据发送结束前向发送缓冲器写入新的数据，否则无论主机是否开始下一帧数据的通讯，从机都将发生欠载错误。
- 连续通信模式下，也可以通过 MSS1 来调整上一帧数据最后一个 SCK 和下一帧数据第一个 SCK 之间的时间间隔。

31.3.4 运行说明

31.3.4.1 运行模式概述

SPI 支持 4 线式 SPI 模式和 3 线式时钟同步运行模式。每个运行模式下都可作为主机或者从机进行串行通信。设置 SPI 控制寄存器 SPI_CR 中的 MSTR 和 SPIMDS 位设定 SPI 的模式。SPI 的模式和 SPI_CR 寄存器设定的关系以及各模式概要如下表所示。

表 31-6 SPI 模式和寄存器设定关系

模式	主机 (SPI 运行)	从机 (SPI 运行)	主机 (时钟同步运行)	从机 (时钟同步运行)
MSTR 位的设定	1	0	1	0
SPIMDS 位的设定	0	0	1	1
SCK 信号	输出	输入	输出	输入
MOSI 信号	输出	输入	输出	输入
MISO 信号	输入	输出/Hi-Z	输入	输出
SS0 信号	输出	输入	Hi-Z (不使用)	Hi-Z (不使用)
SS1~SS3 信号	输出	Hi-Z	Hi-Z (不使用)	Hi-Z (不使用)
SS 极性变更功能	有	有	-	-
最大传输速率	~PCLK1/2	~PCLK1/6	~PCLK1/2	~PCLK1/6
时钟源	内部波特率发生器	SCK 输入	内部波特率发生器	SCK 输入
时钟极性	2 种	2 种	2 种	2 种
时钟相位	2 种	2 种	2 种	2 种 ^{*3}
开始传送位	MSB/LSB	MSB/LSB	MSB/LSB	MSB/LSB
传送数据长度	4~32 位	4~32 位	4~32 位	4~32 位
SCK 延迟控制	有	无	有	无
SS 无效延迟控制	有	无	有	无
下次存取延迟控制	有	无	有	无

模式	主机 (SPI 运行)	从机 (SPI 运行)	主机 (时钟同步运行)	从机 (时钟同步运行)
传送启动方法	通过发送缓冲器空中断请求, 写发送缓冲器	SS 输入有效或 SCK 时钟边沿	通过发送缓冲器空中断请求, 写发送缓冲器	SCK 振荡
模式故障错误	无	有	无	有
发送缓冲器空检测	有	有	有	有
接收缓冲器满检测	有*1	有*1	有*1	有*1
过载错误检测	有*1	有*1	有*1	有*1
奇偶校验错误检测	有*1 *2	有*1 *2	有*1 *2	有*1 *2
欠载错误检测	无	有	无	有

 **说明**

1. 当 SPI_CR.TXMSDS 位为 1 时, 不进行接收缓冲器满的检测、过载错误的检测和奇偶校验错误的检测。

2. 当 SPI_CR.PAE 位为 0 时, 不进行奇偶校验错误的检测。

3. 当 SPI_CFG2.CPHA 位为 0 时, 仅在连续传输模式下有效, 具体设定步骤请参见“[时钟同步运行模式中的从机动作](#)”。

31.3.4.2 SPI 运行模式时的主机动作

作为主机时的动作说明

在 SPI 数据发送缓冲器 (TX_BUFF) 为空的状态下 (状态寄存器 SPI_SR 中的 TDEF 标志位为 1), 将格式控制寄存器 SPI_CFG1 的 FTHLV[1:0]位所设帧数长度的数据写到 SPI 数据寄存器 (SPI_DR) 后, SPI 会将 SPI_DR 数据更新至 TX_BUFF 中。此时若移位寄存器 (SHIFTER) 为空, SPI 将 TX_BUFF 的数据复制到移位寄存器开始串行传送。

当发送数据复制到 SHIFTER 后, SPI 将会 SHIFTER 状态改为满状态; 当串行传送结束, 改为空状态。SHIFTER 的状态无法进行读取。

当 SPI 发送完最后采样时序所需的 SCK 边沿后, 本次串行传送结束, 结束时间与 SPI_CFG2.CPHA 位无关。在接收缓冲器 (RX_BUFF) 为空的情况下, 串行传送结束后 SPI 将 SHIFTER 中的数据复制到 RX_BUFF 中, 可通过数据寄存器 SPI_DR 进行读取。

最后的采样时序取决于传送数据的位长, 主控模式的 SPI 数据长度取决于 SPI_CFG2.DSIZE[3:0]位的设定值, SS 输出引脚的极性取决于 SPI_CFG1 寄存器的设定值。有关 SPI 传送格式的详细内容请参见“[传送格式](#)”。

SPI 主机模式的初始化

- 步骤 1. 设置通信配置寄存器 1 (SPI_CFG1), 包括有波特率的设定, 使用帧数的设定, 各种迟延时间的设定等;
- 步骤 2. 设置通信配置寄存器 2 (SPI_CFG2), 包括有 SS 电平设定, 数据移位顺序设定, 各种延迟的允许位的设定, 数据格式及时钟极性相位的设定等;
- 步骤 3. 如需要使用中断, 请设置系统的中断寄存器;
- 步骤 4. 如需要使用 DMA, 请设置 DMA 的相关寄存器;
- 步骤 5. 设定输入输出管脚;

- 步骤 6. 设定 SPI 控制寄存器 SPI_CR，包括有模式及运行方式的设定，自诊断功能的设定，奇偶校验的设定等；
- 步骤 7. 确认 SPI_CR 寄存器的设置；
- 步骤 8. 清除各种标志位；
- 步骤 9. 设置中断许可位；
- 步骤 10. 将控制寄存器 SPI_CR 的 SPE 位设置成 1，动作开始。

31.3.4.3 SPI 运行模式时的从机动作

SPI 作为从机时的动作说明

当 SPI_CFG2.CPHA 位为 0 时，如果 SPI 检测到 SS0 输入信号变为有效电平，就需要开始向 MISO 输出信号驱动有效数据。因此，在 CPHA 位为 0 时，将 SS0 输入信号电平从无效变为有效视为开始串行传送的触发信号。

当 CPHA 位为 1 时，如果在 SS0 输入信号为有效电平的状态下 SPI 检测到最初的 SCK 边沿，就需要开始向 MISO 输出信号驱动有效数据。因此，在 CPHA 位为 1 时，将 SS0 信号处于有效电平状态下的首个 SCK 边沿视为开始串行传送的触发信号。

如果 SPI 在 SHIFTER 为空的状态下检测到串行传送的开始，就将 SHIFTER 改为满状态，并且不能在串行传送过程中将数据从 TX_BUFF 传送到 SHIFTER。如果在开始串行传送前 SHIFTER 已满，SPI 就保持 SHIFTER 的满状态。

如果 SPI 检测到最后采样时序的 SCK 边沿，则本次串行传输结束，结束时间与 SPI_CFG2.CPHA 位无关。在 RX_BUFF 为空的情况下，串行传送结束后 SPI 将 SHIFTER 的接收数据复制到 RX_BUFF。可通过访问 SPI_DR 读取该数据。SPI 在串行传送结束后将 SHIFTER 改为空状态，该状态与 RX_BUFF 的状态无关。

在串行传送期间，如果 SPI 检测到 SS0 输入信号无效，就发生模式故障错误。

最后的采样时序取决于传送数据的位长，从属模式的 SPI 的数据长度取决于 SPI_CFG2.DSIZE[3:0]位的设定值，SS0 输入信号的极性取决于 SPI_CFG1.SS0PV 位的设定值。有关 SPI 传送格式的详细内容请参见“[传送格式](#)”。



说明

当 SPI_CFG2.CPHA 位为 0 时，将 SS0 输入信号电平从无效变为有效视为开始串行传送的触发信号。由于在从机模式的结构中 SS0 输入信号被固定为有效状态，此时 SPI 将无法开始正常串行传送。因此，在 SS0 输入信号被固定为有效状态的结构中，要使从机模式的 SPI 正常进行发送和接收，必须将 CPHA 位设置为 1。如果需要将 CPHA 位设定为 0，则不能固定 SS0 输入信号。

SPI 从机模式的初始化

- 步骤 1. 设定通信配置寄存器 1 (SPI_CFG1)，主要包括使用帧数的设定；
- 步骤 2. 设置通信配置寄存器 2 (SPI_CFG2)，包括有传输速率，数据格式及时钟极性相位的设定等；
- 步骤 3. 如需要使用中断，请设置系统的中断寄存器；
- 步骤 4. 如需要使用 DMA，请设置 DMA 的相关寄存器；
- 步骤 5. 设定输入输出管脚；
- 步骤 6. 设定 SPI 控制寄存器 SPI_CR，包括有模式及运行方式的设定，自诊断功能的设定，奇偶校验的设定等；
- 步骤 7. 确认 SPI_CR 寄存器的设置；
- 步骤 8. 清除各种标志位；
- 步骤 9. 设置中断许可位；
- 步骤 10. 将控制寄存器 SPI_CR 的 SPE 位设置成 1，动作开始。

31.3.4.4 时钟同步运行模式中的主机动作

当 SPI 控制寄存器 SPI_CR 中的 SPIMDS 位为 1 时, SPI 处于时钟同步运行模式。在该模式动作时, SPI 只使用 SCK、MOSI 和 MISO 这 3 个管脚进行通信, SSI 管脚被释放可用于普通 I/O 功能。

尽管时钟同步运行模式时不使用 SSI 管脚, 但模块内部的运行和 SPI 运行模式是相同的。

SPI 作为主机时的动作说明

在 SPI 数据发送缓冲器 (TX_BUFF) 为空的状态下 (状态寄存器 SPI_SR 中的 TDEF 标志位为 0), 将格式控制寄存器 SPI_CFG1 的 FTHLV[1:0] 位所设帧数长度的数据写到 SPI 数据寄存器 (SPI_DR) 后, SPI 会将 SPI_DR 数据更新至 TX_BUFF 中。此时若移位寄存器 (SHIFTER) 为空, SPI 将 TX_BUFF 的数据复制到移位寄存器开始串行传送。

当发送数据复制到 SHIFTER 后, SPI 将会 SHIFTER 状态改为满状态; 当串行传送结束, 改为空状态。

SHIFTER 的状态无法进行读取。

当 SPI 发送完最后采样时序所需的 SCK 边沿后, 本次串行传送结束, 结束时间与 SPI_CFG2.CPHA 位无关。在接收缓冲器 (RX_BUFF) 为空的情况下, 串行传送结束后 SPI 将 SHIFTER 中的数据复制到 RX_BUFF 中, 可通过数据寄存器 SPI_DR 进行读取。

最后的采样时序取决于传送数据的位长, 主控模式的 SPI 数据长度取决于 SPI_CFG2.DSIZE[3:0] 位的设定值。有关 SPI 传送格式的详细内容请参见“[传送格式](#)”。

时钟同步运行模式时主机初始化

- 步骤 1. 设置通信配置寄存器 1 (SPI_CFG1), 包括有波特率的设定, 使用帧数的设定, 各种迟延时间的设定等;
- 步骤 2. 设置通信配置寄存器 2 (SPI_CFG2), 包括有数据移位顺序设定, 各种延迟的允许位的设定, 数据格式及时钟极性相位的设定等;
- 步骤 3. 如需要使用中断, 请设置系统的中断寄存器;
- 步骤 4. 如需要使用 DMA, 请设置 DMA 的相关寄存器;
- 步骤 5. 设定输入输出管脚;
- 步骤 6. 设定 SPI 控制寄存器 SPI_CR, 包括有模式及运行方式的设定, 自诊断功能的设定, 奇偶校验的设定等;
- 步骤 7. 确认 SPI_CR 寄存器的设置;
- 步骤 8. 清除各种标志位;
- 步骤 9. 设置中断许可位;
- 步骤 10. 将控制寄存器 SPI_CR 的 SPE 位设置成 1, 动作开始。

31.3.4.5 时钟同步运行模式中的从机动作

SPI 作为从机时的动作说明

当 SPI_CFG2.CPHA 位为 0 时, 需要 SPI 检测到 SS0 输入信号变为有效电平作为开始串行通信的触发信号。由于时钟同步运行模式下不使用 SS0 管脚, 因此, 必须按照以下的顺序设置才能进行正常通信:

- 步骤 1. 进行通信设置, 并将 SPI_CR 的 SPE 位设置为 1;
- 步骤 2. 将 SPI_CFG2.CPHA 设置为 1;
- 步骤 3. 将 SPI_CFG1.CTMDS 设置为 1;
- 步骤 4. 将需要发送的数据写入 SPI_DR;
- 步骤 5. 将 SPI_CFG2.CPHA 设置为 0;
- 步骤 6. 等待主机时钟开始通信。

当 CPHA 位为 1 时, 如果在 SS0 输入信号为有效电平的状态下 SPI 检测到最初的 SCK 边沿, 就需要开始向 MISO 输出信号驱动有效数据。由于时钟同步运行模式下不使用 SS0 管脚, 因此, 在 CPHA 位为 1 时, 将首个 SCK 边沿视为开始串行传送的触发信号。

如果 SPI 在 SHIFTER 为空的状态下检测到串行传送的开始, 就将 SHIFTER 改为满状态, 并且不能在串行传送过程中将数据从 TX_BUFF 传送到 SHIFTER。如果在开始串行传送前 SHIFTER 已满, SPI 就保持 SHIFTER 的满状态。

如果 SPI 检测到最后采样时序的 SCK 边沿, 则本次串行传输结束。在 RX_BUFF 为空的情况下, 串行传送结束后 SPI 将 SHIFTER 的接收数据复制到 RX_BUFF。可通过访问 SPI_DR 读取该数据。SPI 在串行传送结束后将 SHIFTER 改为空状态, 该状态与 RX_BUFF 的状态无关。

最后的采样时序取决于传送数据的位长, 从属模式的 SPI 的数据长度取决于 SPI_CFG2.DSIZE[3:0]位的设定值。

时钟同步运行模式时从机初始化

- 步骤 1. 设定通信配置寄存器 1 (SPI_CFG1), 主要包括使用帧数的设定;
- 步骤 2. 设置通信配置寄存器 2 (SPI_CFG2), 包括有传输速率, 数据格式及时钟极性相位的设定等;
- 步骤 3. 如需要使用中断, 请设置系统的中断寄存器;
- 步骤 4. 如需要使用 DMA, 请设置 DMA 的相关寄存器;
- 步骤 5. 设定输入输出管脚;
- 步骤 6. 设定 SPI 控制寄存器 SPI_CR, 包括有模式及运行方式的设定, 自诊断功能的设定, 奇偶校验的设定等;
- 步骤 7. 确认 SPI_CR 寄存器的设置;
- 步骤 8. 清除各种标志位;
- 步骤 9. 设置中断许可位;
- 步骤 10. 将控制寄存器 SPI_CR 的 SPE 位设置成 1, 动作开始。

31.3.4.6 几种 SPI 动作的处理流程

SPI 作为主机时的数据传送处理流程

- 步骤 1. 等待数据发送缓冲器空的中断或通过轮询方式确认数据发送缓冲器处于空状态;
- 步骤 2. 向数据寄存器 SPI_DR 写入要发送的数据;
- 步骤 3. 重复步骤 1、2, 直到最后一个数据发送完成;
- 步骤 4. 将发送数据寄存器空中断的允许位 TXIE 清零, 同时将 SPI 闲置状态中断允许位 IDIE 设为 1;
- 步骤 5. 硬件产生 SPI 闲置状态中断;
- 步骤 6. 将 SPE 置 0, 停止 SPI 动作, 同时将 IDIE 清零。

数据接收处理流程

- 步骤 1. 等待数据接收缓冲器满的中断或者通过轮询方式确认数据接收缓冲器处于满状态;
- 步骤 2. 通过访问 SPI_DR 从接收缓冲器读取数据;
- 步骤 3. 重复步骤 1、2, 直到最后一个接收数据被读取;
- 步骤 4. 将数据接收缓冲器满的中断允许位 RXIE 清零。

通信错误处理流程

- 步骤 1. 等待通信错误中断或者通过轮询方式确认通信错误标志位 (MODFERF/OVERRF/UDRERF/PERF) 被置成 1;
- 步骤 2. 确认 SS0 状态, 排除模式故障错误;
- 步骤 3. 将 SPE 清零, 停止 SPI 动作;
- 步骤 4. 通过错误标志位确定通信错误种类, 进行通讯错误处理;
- 步骤 5. 将错误标志位清零;
- 步骤 6. 启动 SPI, 重新开始通信。

31.3.5 奇偶校验位自诊断

奇偶校验电路由传输数据的奇偶校验位和接收数据的错误检测部分构成。可按照下图所示流程使用自诊断功能对奇偶校验电路进行故障诊断。

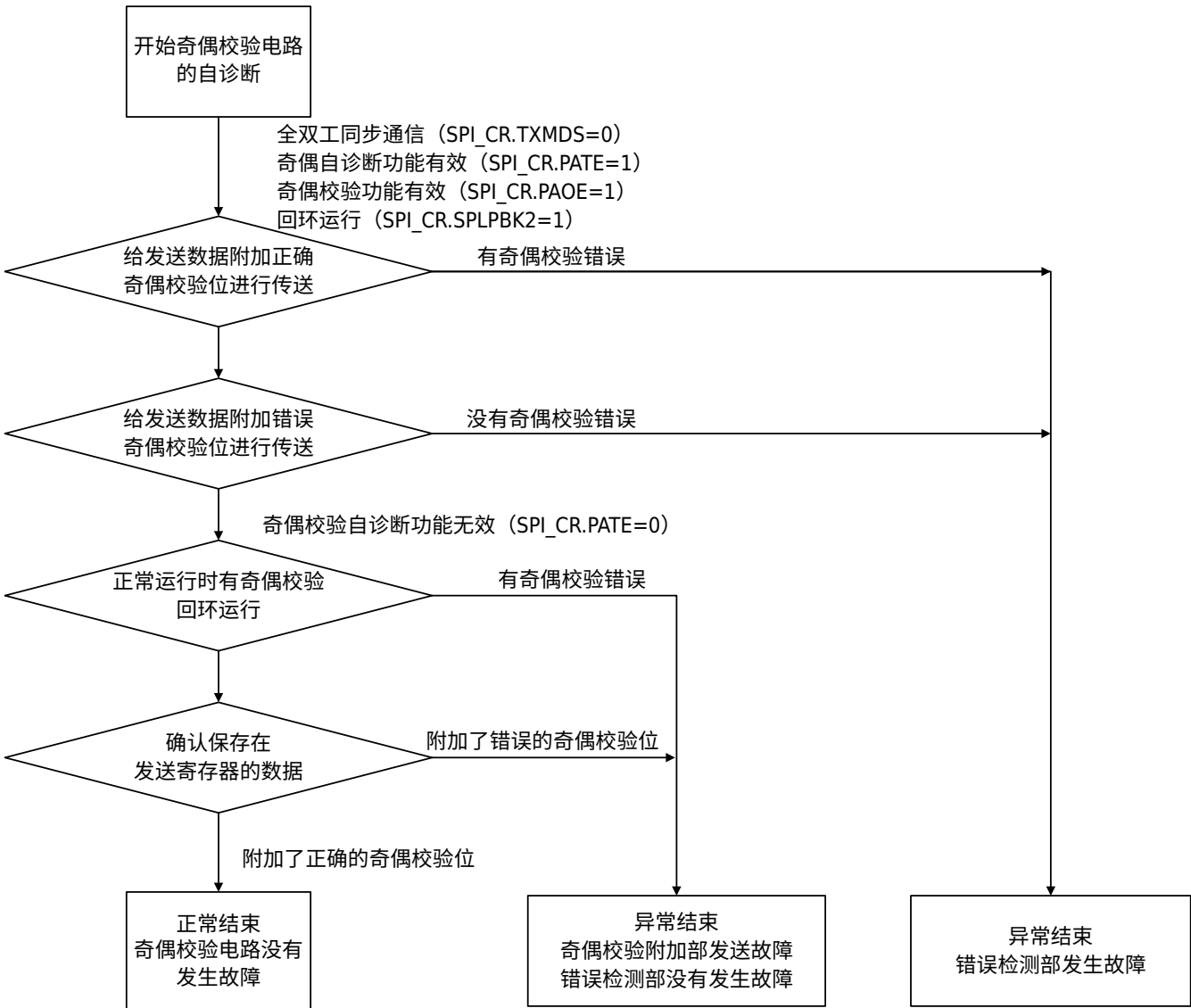


图 31-15 奇偶校验流程

31.3.6 错误检测

31.3.6.1 错误检测概述

在正常的 SPI 串行传送中，系统通过对 SPI_DR 寄存器写操作进行数据串行发送，通过对 SPI_DR 寄存器的读操作获得串行接收的数据。但由于发送/接收缓冲器的状态以及串行传送开始/结束时的 SPI 的状态，某些情况下有可能会发生异常传送。当异常传送发生时，SPI 会将此次传输检测为欠载错误、过载错误、奇偶校验错误或者模式故障错误。异常传送和 SPI 错误检测的对应关系如下表所示。

表 31-7 错误检测对应表

序号	发生条件	SPI 运行	检测错误
①	在发送缓冲器满的状态下写 SPI_DR 寄存器	保持发送缓冲器内容	无
		写数据丢失	

序号	发生条件	SPI 运行	检测错误
②	在接收缓冲器为空的状态下读 SPI_DR 寄存器	输出上次的串行接收数据	无
③	在从机模式中，发送数据未转移到移位寄存器的状态下开始串行传送	中止串行传输	欠载错误
		丢失发送和接收数据	
		停止驱动 MISO 输出信号	
		停止 SPI 功能	
④	从机模式：SS0 管脚有效电平宽度没有达到数据传输所需时间	中止传输 丢失发送和接收数据 停止 SPI 功能	模式错误
⑤	在接收缓冲器满的状态下结束串行传送	保持接收缓冲器内容	过载错误
		接收数据丢失	
⑥	在进行全双工同步串行通信，并且奇偶校验功能有效的状态下，接收到错误的奇偶校验位	奇偶校验错误标志有效	奇偶校验错误

①描述的情况，SPI 不发生检测错误。为防止在写数据到 SPI_DR 寄存器的过程中发生数据遗漏，必须通过发送缓冲器空的中断把数据写入 SPI_DR 寄存器。同理，②的情况下，SPI 也不发生检测错误。为防止无关数据被读入，SPI_DR 的数据读取必须通过接收寄存器满的中断请求来进行。

31.3.6.2 欠载错误

MSTR 位为 0 时，SPI 以从机状态运行，如果 SPE 置为 1 后，SS0 管脚接收到有效电平前，传送数据还未准备好，则 SPI 发生欠载错误，SPI_SR.MODFERF 和 SPI_SR.UDRERF 标志将被置为 1。

当检测到欠载错误后，SPI 将停止驱动信号输出，同时把 SPI_CR.SPE 置为 0。

监测欠载错误可通过直接访问 SPI_SR 寄存器，或者使用 SPI 错误中断读取 SPI_SR 等方式进行。如果不使用错误中断，请使用轮询方式来监测欠载错误。

当 SPI_SR.MODFERF 为 1 时，系统禁止对 SPE 位写 1。要将 SPI_CR.SPE 置成 1 使能 SPI 功能必须先将 MODFERF 标志清零。

31.3.6.3 模式错误

SPI 在主机模式时，SPI_CR.MODFE 请勿设定为 1。在从机模式时，当 SSi 有效电平宽度没有达到传输数据所需时间时，发生模式故障，SPI_SR.MODFERF 置 1，SPI_CR.SPE 置为 0。需要传送时，对 SPI_SR.MODFREF 清零后，再把 SPI_CR.SPE 置 1。

31.3.6.4 过载错误

如果在接收缓冲器满的状态下结束串行传送，SPI 发生过载错误，SPI_SR.OVRERF 标志被置成 1。因为在 OVRERF 标志为 1 的状态下 SPI 不会把移位寄存器的数据复制到接收缓冲器，所以接收缓冲器中保存的是发生错误前的接收数据。需要在 OVRERF 标志为 1 的状态下读 SPI_SR 寄存器后才能给 OVRERF 标志写 0。

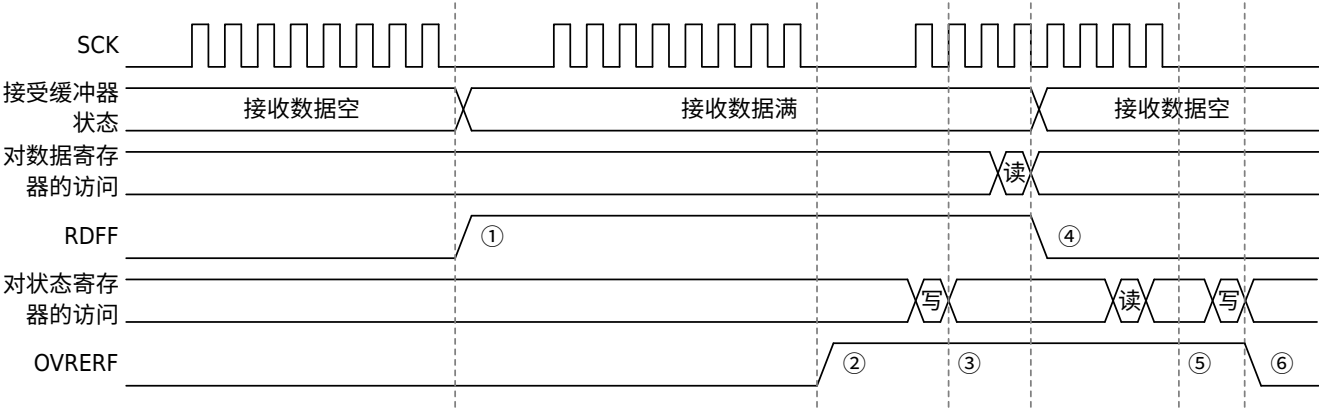


图 31-16 过载错误处理

以下说明在图中的①~⑥所示的时序中标志的运行内容：

- ① 在接收缓冲器空的状态下结束串行传送，SPI 正常动作，将移位寄存器的数据复制到接收缓冲器中，并将 RDFF 标志置 1。
- ② 在接收缓冲器满的状态下结束串行传送，SPI 检测到过载错误并且将 OVRERF 标志置 1。SPI 将不会把移位寄存器的数据复制到接收缓冲器。即使 PAE 位为 1 时，也不检测奇偶校验错误。
- ③ 清零 OVRERF 标志位。
- ④ 对数据寄存器 SPI_DR 进行读取访问，SPI 就能读到接收缓冲器的数据。RDFF 标志变为 0。即使此时接收缓冲器状态为空，OVRERF 标志也不会变为 0。
- ⑤ 在 OVRERF 标志为 1 的状态（过载错误）下结束串行传送，SPI 将不会把移位寄存器的数据复制到接收缓冲器，也不产生接收缓冲器满中断，RDFF 标志保持为 0。即使 PAE 位为 1，也不检测奇偶校验错误。在发生过载错误的状态下，如果在没有将接收数据从移位寄存器复制到接收缓冲器的情况下结束串行传送，SPI 就判断为移位寄存器处于空的状态，允许将数据从发送缓冲器传送到移位寄存器。
- ⑥ 只有在将 OVRERF 标志变为 0 后才可以进行正常的接收运行。

监测过载错误能通过直接访问 SPI_SR 寄存器或者使用 SPI 错误中断访问 SPI_SR 寄存器等方式进行。在进行串行传送时，必须通过诸如读取完 SPI_DR 寄存器后立即读取 SPI_SR 寄存器等方法，尽早检测到过载错误的发生。

在主机模式下如果启用通信自动挂起功能（将 SPI_CR.CSUSPE 位设成 1），则 SPI 将在发生过载错误前的最后一个采样周期将通信时钟暂停，此时由于移位寄存器尚未完成最后一位的接收，SPI 保持在正常通信状态，过载错误不会发生。在通信时钟暂停期间可以对接收缓冲器进行读取，读取后接收缓冲器状态变为空，SPI 再重启通信时钟完成最后一位的数据接收。

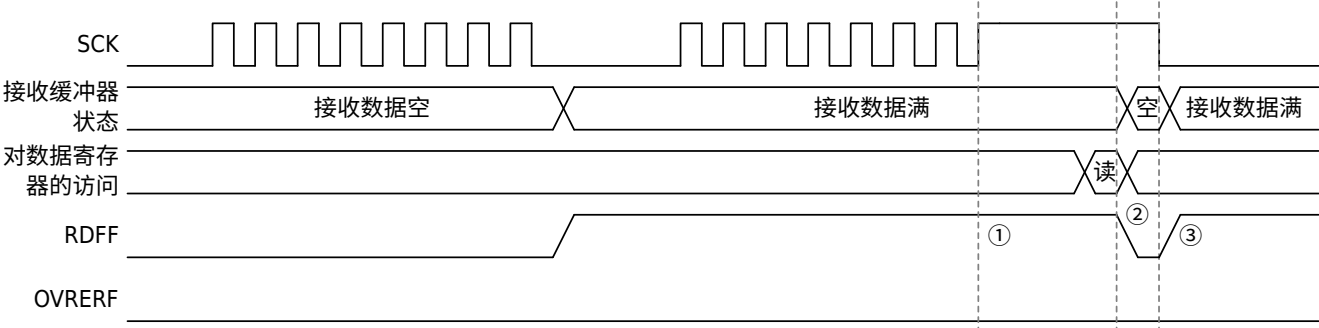


图 31-17 启用时钟自动停止功能时的动作示意图 (CPHA=1)

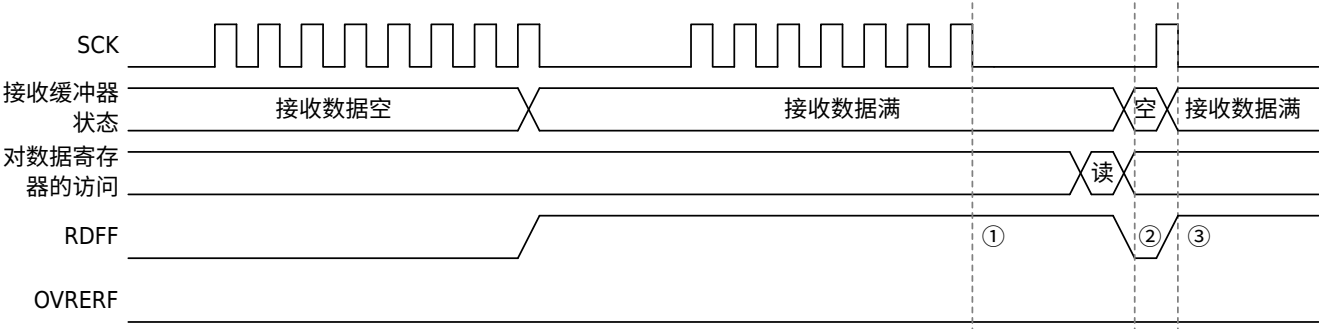


图 31-18 启用时钟自动停止功能时的动作示意图 (CPHA=0)

以下说明在图中的①~③所示的时序中标志的运行内容：

- ① 当处于接收缓冲器满状态时，SPI 在最后一位数据接收前暂停通信时钟。此时将不会发生过载错误。
- ② 当通过访问 SPI_DR 读取接收缓冲器中的数据后，接收缓冲器变为空状态，RDFF 标志为清零，SPI 重启通信时钟完成最后一位的数据通信。
- ③ 最后一位数据通信完成，接收缓冲器再次变为满状态，RDFF 标志置 1，可通过访问 SPI_DR 读取接收到的数据。

31.3.6.5 奇偶校验错误

在 SPI_CR.TXMDS 位为 0 并且 SPI_CR.PAE 位为 1 的状态下，SPI 将在全双工同步串行通信结束时进行奇偶校验。当 SPI 检测到接收数据有奇偶校验错误时，将 SPI_SR.PERF 标志置 1。在 SPI_SR.OVRERF 位为 1 的状态下，因为 SPI 不会将移位寄存器的数据复制到接收缓冲器，所以不对接收数据进行奇偶校验错误的检测。要将 PERF 标志清零时，需要在 PERF 标志为 1 的状态下读 SPI_SR 寄存器后再给 PERF 标志写 0。

OVRERF 标志和 PERF 标志的运行实例如下图所示。在图中的例子中，SPI 在 SPI_CR.TXMDS 位为 0 并且 SPI_CR.PAE 位为 1 的状态下进行全双工同步串行通信的 8 位串行传送。

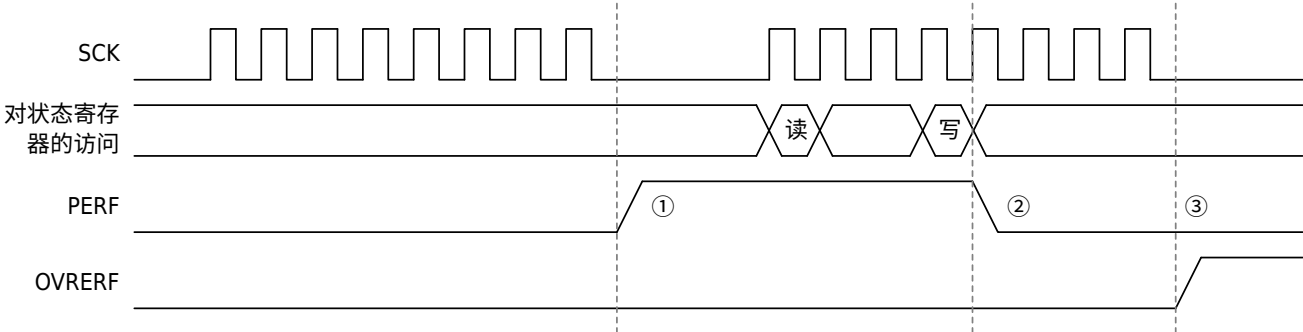


图 31-19 奇偶校验错误

以下说明在图中①~③所示的时序中标志的运行内容：

- ①SPI 未检测到过载错误，串行传送正常结束。SPI 将移位寄存器的数据复制到接收缓冲器。此时，SPI 对接收数据进行奇偶校验。如果检测到奇偶校验错误，则将 PERF 标志置 1。
- ②清零 RERF 标志。
- ③SPI 检测到过载错误，此时 SPI 不会将移位寄存器的数据复制到接收缓冲器，SPI 不会对数据进行奇偶校验，不会发生奇偶校验错误。

可通过直接访问 SPI_SR 寄存器或者经 SPI 错误中断读取 SPI_SR 寄存器等方式监测奇偶校验错误的发生。在进行串行传送时，必须通过访问状态寄存器 SPI_SR 等方法，尽早监测到奇偶校验错误的发生。

31.3.7 SPI 的初始化

通过写操作或者模式故障错误检测将 SPE 位清零，可以使 SPI 功能无效并且对部分 SPI 功能进行初始化。如果发生系统复位或使用功能复位寄存器 RMU_FRST1，则对全部 SPI 功能进行初始化。

清除 SPE 位进行初始化

当 SPI_CR.SPE 位为 0 时，SPI 进行以下的初始化操作：

- 中止正在进行的串行传送。
- 若处于从机状态则停止驱动输出信号（状态变为 Hi-Z）。
- 对 SPI 内部状态进行初始化。
- 清空发送缓冲器，SPI_SR.TDEF 标志置为 1。

通过将 SPE 位清零来进行初始化时，不会对 SPI 的控制位进行初始化。因此，只要重新将 SPE 位置为 1，就能以和初始化前相同的传送模式启动 SPI。

清除 SPE 位不会对错误标志位和序列状态进行初始化。因此，在即使在 SPE 被清零以后，也能通过读取接收缓冲器的数据来确认 SPI 传送时的错误发生状况。

由于清除 SPE 位会清空发送缓冲器，并将 SPI_SR.TDEF 标志置为 1。因此，如果在初始化后将 SPI_CR.TXIE 位设成 1，就会产生 SPI 发送缓冲器为空的中断。要避免系统发生此中断，必须在将 SPE 位清零的同时也将 TXIE 位设成 0。

复位初始化

通过系统复位或使用功能复位寄存器 RMU_FRST1 进行初始化，将对 SPI 所有的控制位、状态位和数据寄存器进行初始化。系统复位和功能复位寄存器的内容请参见《复位控制（RMU）》章节相关描述。

31.3.8 中断和事件

SPI 的接收缓冲器满、发送缓冲器空、模式故障（过载、欠载、奇偶校验错误）和 SPI 空闲即可做为中断源也可做为内部触发源，传送完成只可以作为内部触发源。

过载、欠载和奇偶校验错误的中断被集成为 SPI 错误中断 SPEI，所以需要通过标志判断实际发生的中断源。SPI 中断/内部触发源的具体说明如下表所示。一旦条件成立，就产生相应的中断/内部触发请求。对于接收缓冲器满和发送缓冲器空的中断源，需要通过数据传送改变缓冲器状态来清除。

表 31-8 SPI 中断源和触发源说明

中断/内部触发源	略称	中断源	内部触发源	条件
接收缓冲器满	SPI_SPRI	√	√	在 SPI_CR.RXIE 位为 1 的状态下接收缓冲器变满时
发送缓冲器空	SPI_SPTI	√	√	在 SPI_CR.TXIE 位为 1 的状态下发送缓冲器变空时
SPI 错误（过载、欠载、奇偶校验错误）	SPI_SPEI	√	√	在 SPI_CR.EIE 位为 1 的状态下 SPI_SR.OVRERF、SPI_SR.PERF 或 SPI_SR.MODFERF 及 SPI_SR.UDRERF 标志变为 1 时
SPI 空闲	SPI_SPII	√	√	在 SPI_CR.IDIE 位为 1 的状态下： ● 主机模式：数据传输完成或者 SPI_CR.SPE 由 1 写为 0。 ● 从机模式：SPI_CR.SPE 由 1 写为 0。
传送完成	SPI_SPEND	-	√	● 主机模式：数据传输完成。 ● 从机模式：四线式，SSL 管脚为无效电平时；三线式，SCK 管脚时钟达到数据传输所需时钟数

用户可将上表中可用作触发源的向量写入不同的触发对象寄存器，以实现各种事件触发功能。上述事件触发源对应的向量请参见《中断控制器（INTC）》的“中断事件表”。

31.4 寄存器

31.4.1 寄存器总表

- SPI1 基地址：0x4001C000
- SPI2 基地址：0x4001C400
- SPI3 基地址：0x4001C800
- SPI4 基地址：0x40020000
- SPI5 基地址：0x40020400
- SPI6 基地址：0x40020800

表 31-9 串行外设接口（SPI）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	SPI_DR	SPI 数据寄存器	32
0x04	SPI_CR	SPI 控制寄存器	32
0x0C	SPI_CFG1	SPI 通信配置寄存器 1	32
0x14	SPI_SR	SPI 状态寄存器	32
0x18	SPI_CFG2	SPI 通信配置寄存器 2	32

31.4.2 SPI 数据寄存器 (SPI_DR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SPD																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	SPD	串行数据	SPI 数据存储	RW

31.4.3 SPI 控制寄存器 (SPI_CR)

Offset	Bit Position																																						
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x00000000																																						
Name	Reserved																																						
Access																	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	PAE	奇偶校验允许	0b0: 发送数据不附加奇偶校验位, 接收数据不进行奇偶校验 0b1: 发送数据附加奇偶校验位, 接收数据进行奇偶校验 (SPI_CR.TXMDS=0); 发送数据附加奇偶校验位, 接收数据不进行奇偶校验 (SPI_CR.TXMDS=1)	RW
14	PAOE	奇偶校验模式选择	0b0: 选择偶校验进行发送和接收 0b1: 选择奇校验进行发送和接收	RW
13	PATE	奇偶校验自诊断	0b0: 奇偶校验自诊断功能无效 0b1: 奇偶校验自诊断功能有效	RW
12	MODFE	模式故障错误检测允许	0b0: 禁止模式故障错误检测 0b1: 允许模式故障错误检测	RW
11	IDIE	SPI 空闲中断允许	0b0: 禁止空闲中断请求产生 0b1: 允许空闲中断请求产生	RW
10	RXIE	SPI 接收中断允许	0b0: 禁止 SPI 接收中断请求产生 0b1: 允许 SPI 接收中断请求产生	RW
9	TXIE	SPI 发送中断允许	0b0: 禁止 SPI 发送中断请求产生 0b1: 允许 SPI 发送中断请求产生	RW
8	EIE	SPI 错误中断允许	0b0: 禁止 SPI 错误中断请求产生 0b1: 允许 SPI 错误中断请求产生	RW
7	CSUSPE	通讯自动挂起功能允许	0b0: 通讯自动挂起功能无效 0b1: 通讯自动挂起功能有效	RW
6	SPE	SPI 功能允许	0b0: SPI 功能无效 0b1: SPI 功能有效	RW
5	SPLPBK2	SPI 回环 2 位	0b0: 正常模式 0b1: 回环模式 (发送数据=接收数据)	RW
4	SPLPBK	SPI 回环位	0b0: 正常模式 0b1: 回环模式 (发送数据的反相=接收数据)	RW
3	MSTR	SPI 主从模式选择	0b0: 从机模式 0b1: 主机模式	RW
2	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
1	TXMDS	通信模式选择	● 0b0: 全双工同步串行通信 ● 0b1: 只进行发送串行通信	RW
0	SPIMDS	SPI 模式选择	● 0b0: SPI 运行 (4 线式) ● 0b1: 时钟同步运行 (3 线式)	RW

31.4.4 SPI 通信配置寄存器 1 (SPI_CFG1)

Offset	Bit Position																																	
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000010																																	
Name	Reserved	MIDI		Reserved	MSSDL		Reserved	MSSI		Reserved					CLKDIV					SS3PV		SS2PV		SS1PV		SS0PV		Reserved	SPRDTD		Reserved		CTMDS	FTHLV
Access		RW			RW			RW							RW					RW		RW		RW			RW				RW	RW		

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:28	MIDI	主机下次存取数据间隔闲置时间设定位	0b000: 1 个 SCK+2 个 PCLK1 0b001: 2 个 SCK+2 个 PCLK1 0b010: 3 个 SCK+2 个 PCLK1 0b011: 4 个 SCK+2 个 PCLK1 0b100: 5 个 SCK+2 个 PCLK1 0b101: 6 个 SCK+2 个 PCLK1 0b110: 7 个 SCK+2 个 PCLK1 0b111: 8 个 SCK+2 个 PCLK1	RW
27	Reserved	保留	-	-
26:24	MSSDL	主机 SS 无效延迟设定位	0b000: 1 个 SCK 0b001: 2 个 SCK 0b110: 7 个 SCK 0b111: 8 个 SCK	RW
23	Reserved	保留	-	-
22:20	MSSI	主机 SS 闲置时间设定位	0b000: 1 个 SCK 0b001: 2 个 SCK 0b110: 7 个 SCK 0b111: 8 个 SCK	RW
19:16	Reserved	保留	-	-
15:12	CLKDIV	通讯基准时钟选择位	0000: PCLK1/2 0001: PCLK1/4 0010: PCLK1/6 0011: PCLK1/8 0100: PCLK1/10 0101: PCLK1/12 0110: PCLK1/14 0111: PCLK1/16 1000: PCLK1/18 1001: PCLK1/20 1010: PCLK1/22 1011: PCLK1/24 1100: PCLK1/26	RW

位/位域	标记	位名	功能描述	读写
15:12	CLKDIV	通讯基准时钟选择位	● 1101: PCLK1/28 ● 1110: PCLK1/30 ● 1111: PCLK1/32	RW
11	SS3PV	SS3 信号极性设定	● 0b0: 低电平有效 ● 0b1: 高电平有效	RW
10	SS2PV	SS2 信号极性设定	● 0b0: 低电平有效 ● 0b1: 高电平有效	RW
9	SS1PV	SS1 信号极性设定	● 0b0: 低电平有效 ● 0b1: 高电平有效	RW
8	SS0PV	SS0 信号极性设定	● 0b0: 低电平有效 ● 0b1: 高电平有效	RW
7	<i>Reserved</i>	<i>保留</i>	-	-
6	SPRDTD	数据寄存器读取对象选择	● 0b0: SPI_DR 读接收缓冲器 ● 0b1: SPI_DR 读发送缓冲器 (必须在 TDEF=1 时才可以读取)	RW
5	<i>Reserved</i>	<i>保留</i>	-	-
4	<i>Reserved</i>	<i>保留</i>	读出为 1, 写入时必须写 1	-
3	<i>Reserved</i>	<i>保留</i>	-	-
2	CTMDS	连续通信模式选择	● 0b0: 常规通信模式 ● 0b1: 连续通信模式	RW
1:0	FTHLV	帧数设定位	● 0b00: 1 帧 ● 0b01: 2 帧 ● 0b10: 3 帧 ● 0b11: 4 帧	RW

31.4.5 SPI 状态寄存器 (SPI SR)

[illegible]

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	RDFF	接收缓冲器满标志	硬件置位和清零 0b0：接收缓冲器数据未滿 0b1：接收缓冲器数据已滿	RO
6	FMIDLF	连续模式主机 IDLE 状态标志	硬件置位和清零 0b0：连续模式主机处于正常动作状态 0b1：连续模式主机处于空闲状态	RO
5	TDEF	发送缓冲器空标志	硬件置位和清零 0b0：发送缓冲器有数据 0b1：发送缓冲器无数据	RO
4	UDRERF	欠载错误标志	硬件置位；读取该位为 1 后写 0 将状态位清零 0b0：未发生欠载错误 0b1：欠载错误发生（MODFERF=1）  说明 当 MODFERF=0，此位会被初始化。	RW
3	PERF	奇偶校验错误标志	硬件置位；读取该位为 1 后写 0 将状态位清零 0b0：未发生奇偶校验错误 0b1：发生奇偶校验错误	RW
2	MODFERF	模式故障错误标志	硬件置位；读取该位为 1 后写 0 将状态位清零 0b0：未发生模式故障错误 0b1：发生模式故障错误	RW
1	IDLNF	SPI 空闲标志	硬件置位和清零 0b0：SPI 为空闲状态 0b1：SPI 为传送状态	RO
0	OVRERF	过载错误标志	硬件置位；读取该位为 1 后写 0 将状态位清零 0b0：未发生过载错误 0b1：发生过载错误	RW

31.4.6 SPI 通信配置寄存器 2 (SPI_CFG2)

Offset	Bit Position																																			
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000F1D																																			
Name	Reserved																MSSIE	MSSDLE	MIDIE	LSBF	DSIZE				SSA				Reserved	MBR				CPOL	CPHA	
Access																	RW	RW	RW	RW	RW				RW					RW				RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	MSSIE	SCK 延迟允许	0b0: SCK 延迟为 1 个 SCK 0b1: SCK 延迟为 MSSIE 的设定值	RW
14	MSSDLE	SS 无效延迟允许	0b0: SS 无效延迟为 1 个 SCK 0b1: SS 无效延迟为 MSSDL 的设定值	RW
13	MIDIE	SPI 下次存取延迟允许	0b0: 下次存取延迟为 1 个 SCK+2 个 PCLK1 0b1: 下次存取延迟为 MIDI 的设定值	RW
12	LSBF	SPI LSB first 位	0b0: MSB first 0b1: LSB first	RW
11:8	DSIZE	SPI 数据长度设定位	0b0000: 4 位 0b0001: 5 位 0b0010: 6 位 0b0011: 7 位 0b0100: 8 位 0b0101: 9 位 0b0110: 10 位 0b0111: 11 位 0b1000: 12 位 0b1001: 13 位 0b1010: 14 位 0b1011: 15 位 0b1100: 16 位 0b1101: 20 位 0b1110: 24 位 0b1111: 32 位	RW
7:5	SSA	SS 信号有效设定位	0b000: SS0 0b001: SS1 0b010: SS2 0b011: SS3 - 其他: 禁止设置	RW
4	Reserved	保留	读出为 1, 写入时必须写 1	-

位/位域	标记	位名	功能描述	读写
3:2	MBR	位速率分频设定位	● 0b00: 选择基准通讯时钟 ● 0b01: 选择基准通讯时钟的 2 分频 ● 0b10: 选择基准通讯时钟的 4 分频 ● 0b11: 选择基准通讯时钟的 8 分频	RW
1	CPOL	SCK 极性设定位	● 0b0: 空闲时的 SCK 为 Low 电平 ● 0b1: 空闲时的 SCK 为 High 电平	RW
0	CPHA	SCK 相位设定位	● 0b0: 在奇数边沿进行数据采样, 在偶数边沿数据发生变化 ● 0b1: 在奇数边沿数据发生变化, 在偶数边沿进行数据采样	RW

32 四线式串行外设接口（QSPI）

32.1 简介

四线式串行外设接口（QSPI）是一个存储器控制模块，主要用于和带 SPI 兼容接口的串行 ROM 进行通信。其对象主要包括有串行闪存，串行 EEPROM 以及串行 FeRAM。

32.2 主要特性

表 32-1 QSPI 主要规格

参数	规格
通道数	1 通道
SPI	支持扩展 SPI，二线式 SPI 和四线式 SPI 等多种协议
	支持 SPI 模式 0 和 SPI 模式 3
	地址线宽度可选择 8 位/16 位/24 位/32 位
时序调整	可通过时序调整以支持各种串行闪存
闪存读取	支持多种读取方式： 1. 标准读/快速读 2. 二线式输出快速读取/二线式输入输出快速读取 3. 四线式输出快速读取/四线式输入输出快速读取
	自由设置指令
	数量可调的虚拟周期
	16 字节的预读取功能
	状态查询功能
	SPI 总线周期延长功能
	XIP 控制功能
直接通信功能	灵活而广泛的支持大量串行闪存软件控制指令，包括擦、写、ID 读取及掉电控制等。
中断源	硬件错误中断
模块停止功能	可通过模块停止降低功耗

32.3 内存映射

32.3.1 内部总线空间

串行闪存及相关的控制寄存器在 AHB 总线空间的位置由总体的地址范围配置来决定。

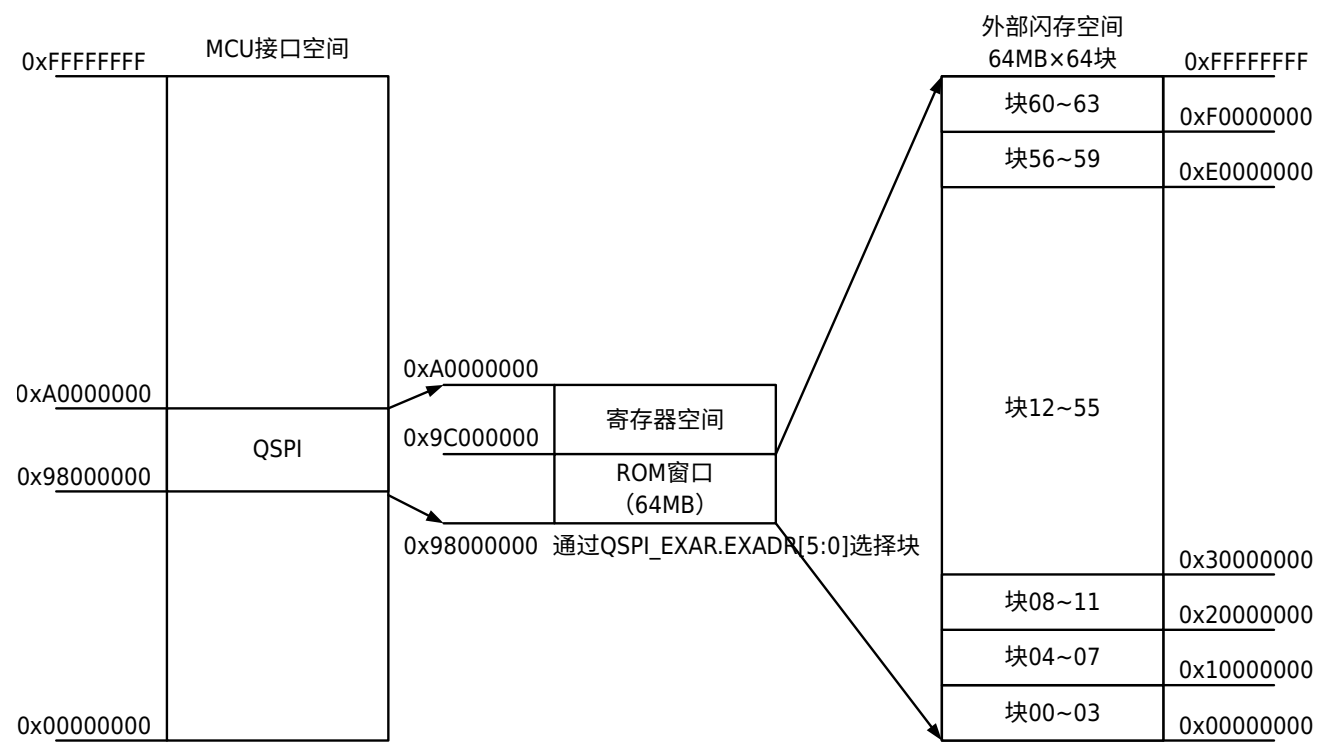


图 32-1 默认区域设定及 AHB 总线空间内存映射关系图

32.3.2 ROM 空间及总线的地址宽度

QSPI 的拥有 32 位地址总线宽度来配合串行闪存。每当对 QSPI 的 ROM 空间进行读访问时，QSPI 总线自动开始工作，将从串行闪存内读到的数据传送过来。

QSPI 不仅可以使用 32 位地址总线宽度，还可以通过设置 QSPI_FCR 寄存器内的 AWSL[1:0]来选择使用 8 位/16 位/24 位地址总线宽度。

如果选择了 8 位/16 位/24 位的地址总线宽度，那么只有地址与之匹配的低位空间可以被正常访问，也就是说，访问 QSPI 中高位的串行闪存镜像空间将会反复出现低位空间的内容。

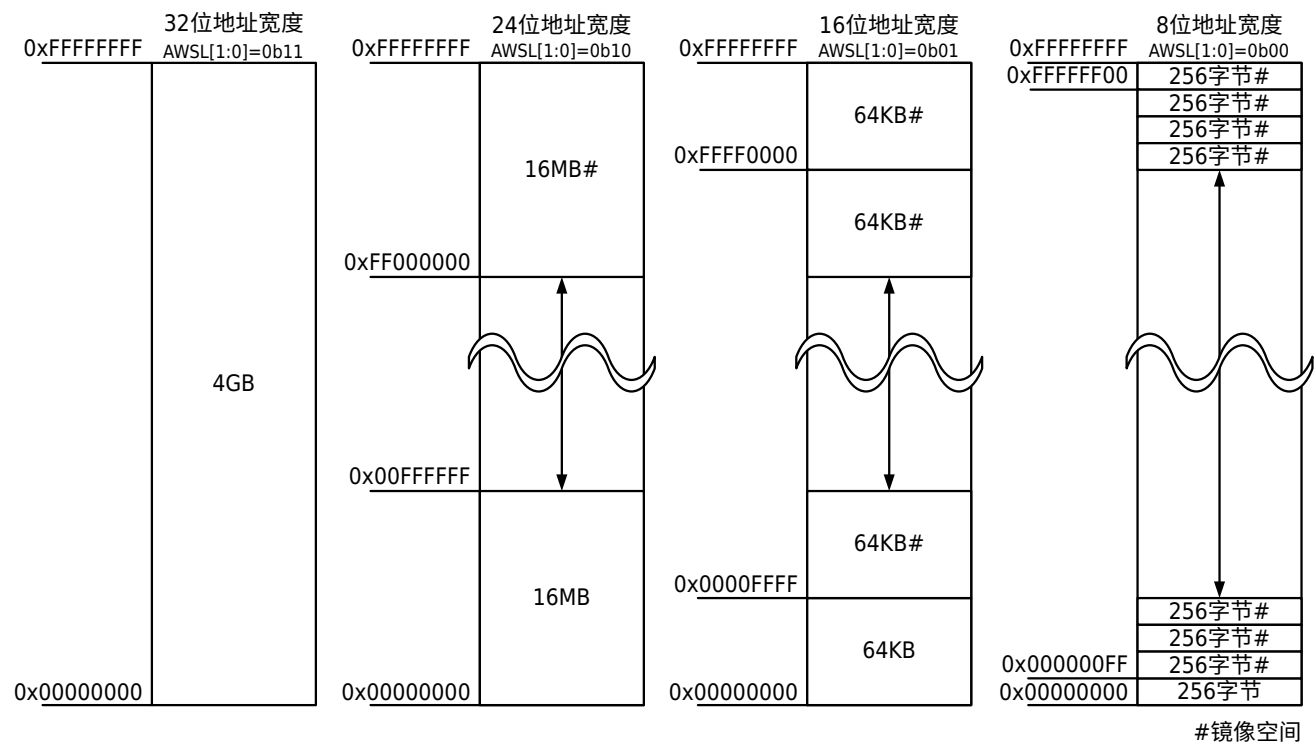


图 32-2 QSPI-ROM 空间内存映像图



说明

- 地址总线宽度可以通过设置 QSPI_FCR 寄存器内的 AWSL[1:0]来选择使用 8 位/16 位/24 位/32 位。
- 选择 8 位地址宽度时，使用标准读指令仅支持访问低 256 字节空间，不支持访问镜像空间。

32.4 功能说明

32.4.1 基本框图

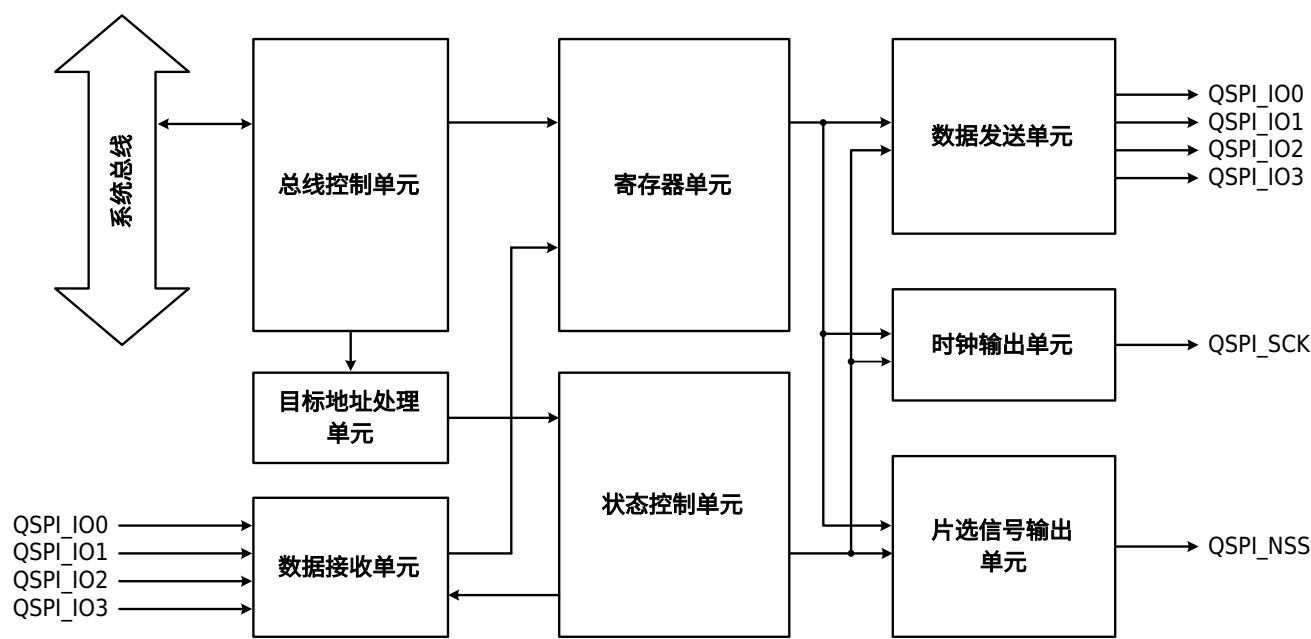


图 32-3 QSPI 的模块构成图

表 32-2 QSPI 管脚

管脚名	输入/输出	功能描述
QSPI_SCK	输出	QSPI 时钟输出管脚
QSPI_NSS	输出	QSPI 从机选择管脚
QSPI_IO0	输入/输出	数据线 0
QSPI_IO1	输入/输出	数据线 1
QSPI_IO2	输入/输出	数据线 2
QSPI_IO3	输入/输出	数据线 3

说明

QSPI_SCK、QSPI_NSS、QSPI_IO0、QSPI_IO1、QSPI_IO2、QSPI_IO3 以下分别简称 QSCK、QSSN、QSIO0、QSIO1、QSIO2、QSIO3。

32.4.2 QSPI 总线

32.4.2.1 SPI 协议

本 QSPI 支持扩展式 SPI、二线式 SPI 和四线式 SPI 三种协议。初始的默认协议是扩展式 SPI 协议。扩展式 SPI 协议只用 QSIO0 管脚单线进行指令输出，之后的地址及数据则根据具体的读取模式指令使用单线式/二线式/四线式输出。

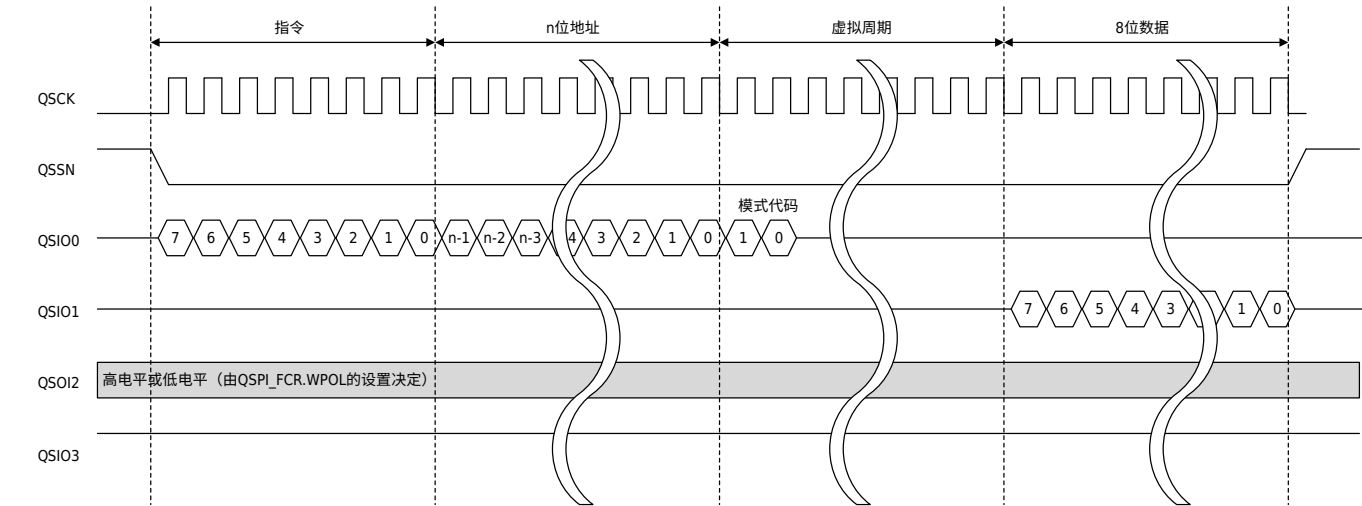


图 32-4 扩展式 SPI 协议动作示意图 1（快速读模式）

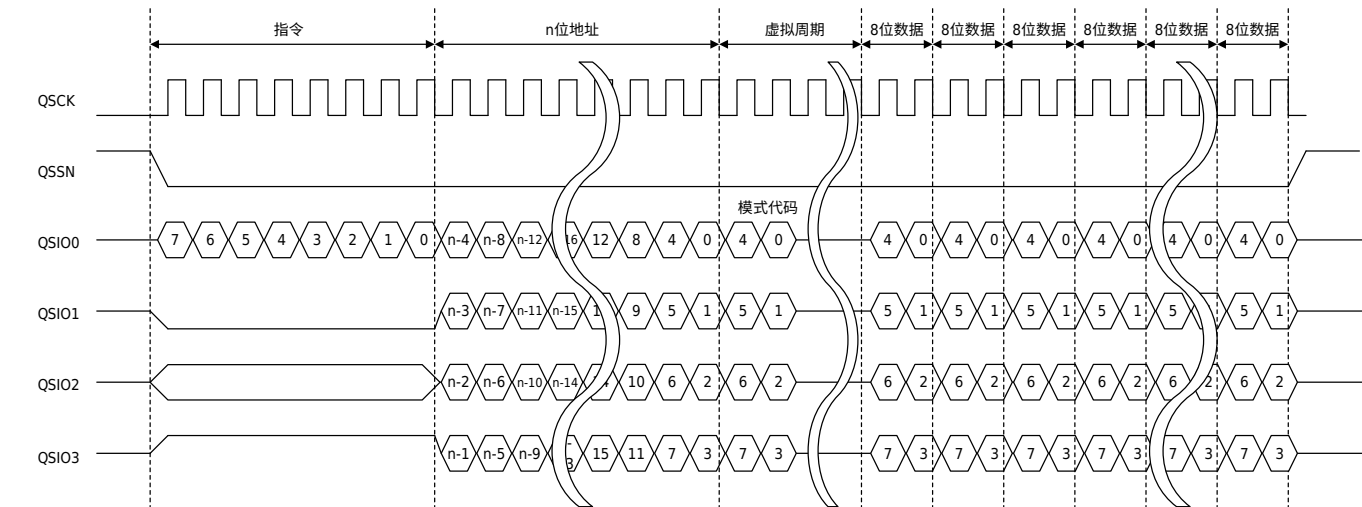


图 32-5 扩展式 SPI 协议动作示意图 2（四线式输入输出快速读模式）

QSPI 读取模式选择为自定义标准读或自定义快速读时，可通过设置 QSPI_CR 寄存器中的 DPRSL[1:0]/APRSL[1:0]/IPRSL[1:0]位来分别配置各个阶段的协议。
二线式 SPI 协议使用 QSIO0，QSIO1 两个管脚实行相应的操作，包括发出指令、地址、接收数据等等。

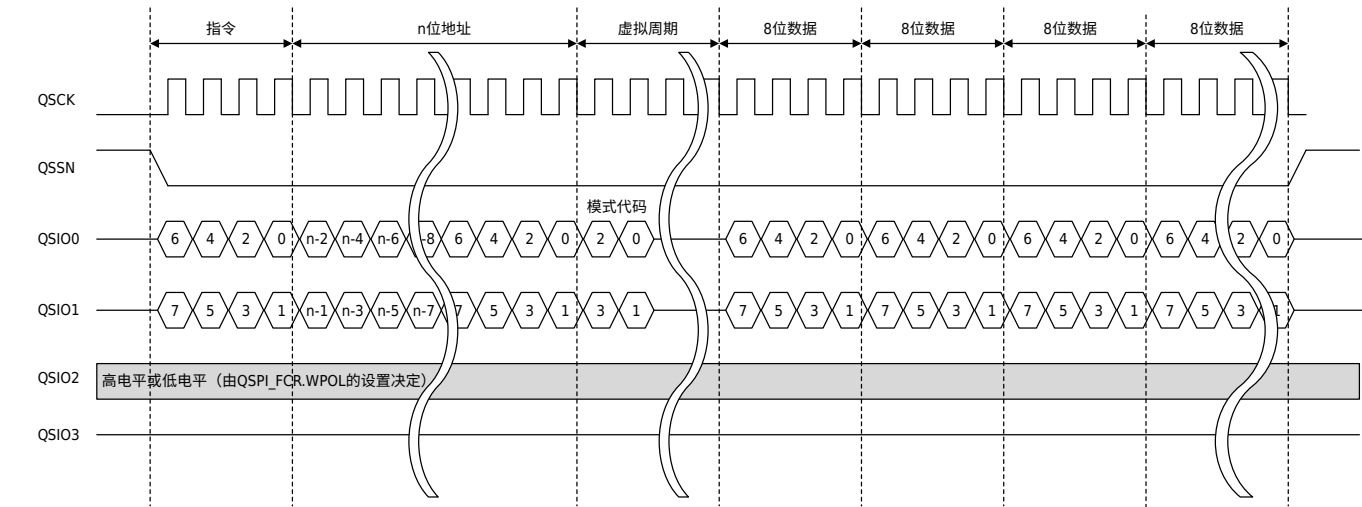


图 32-6 二线式 SPI 协议动作示意图（快速读模式）

四线式 SPI 协议使用 QSI00, QSI01, QSI02, QSI03 四个管脚实行发出指令, 地址, 接收数据等所有相关操作。

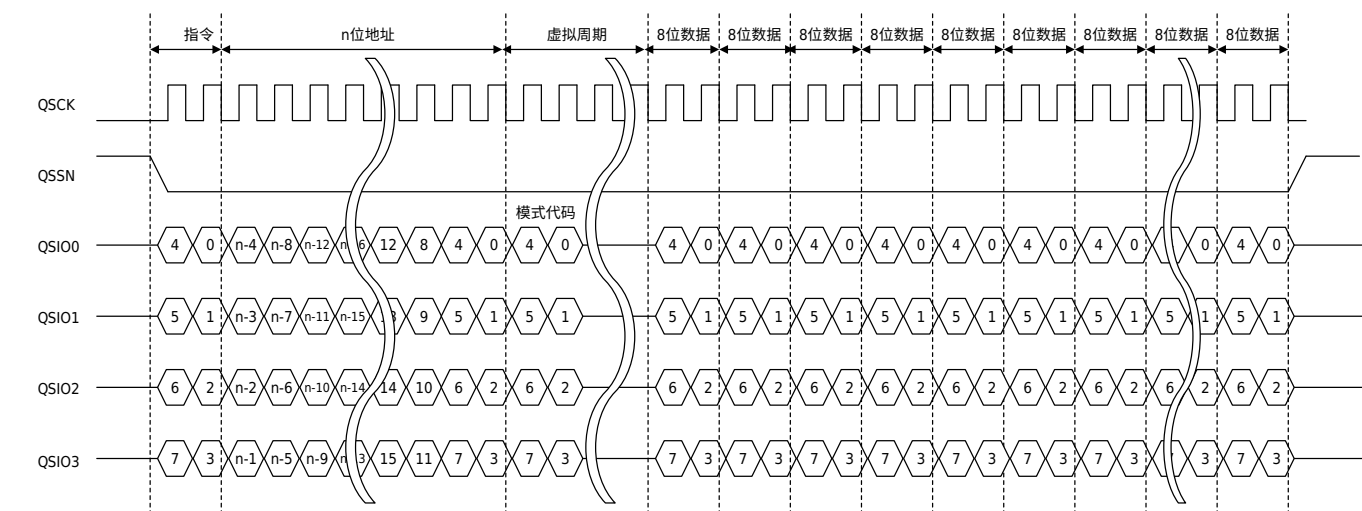


图 32-7 四线式 SPI 协议动作示意图 (快速读模式)

32.4.2.2 SPI 模式

SPI 模式有模式 0 和模式 3 两种, 可通过设置 QSPI_CR 寄存器中的 SPIMD3 位来实现模式切换。SPI 模式 0 和模式 3 的区别在于空闲状态时 QSCCK 的电平不同。在 SPI 模式 0 时 QSCCK 的空闲状态电平为低电平, 而模式 3 时空闲状态电平则是高电平。

串行数据在串行时钟的下降沿从 QSPI 输出并在上升沿被读入外部闪存。而外部闪存则在串行时钟的下降沿输出串行数据并在下一个时钟的下降沿被 QSPI 读入。

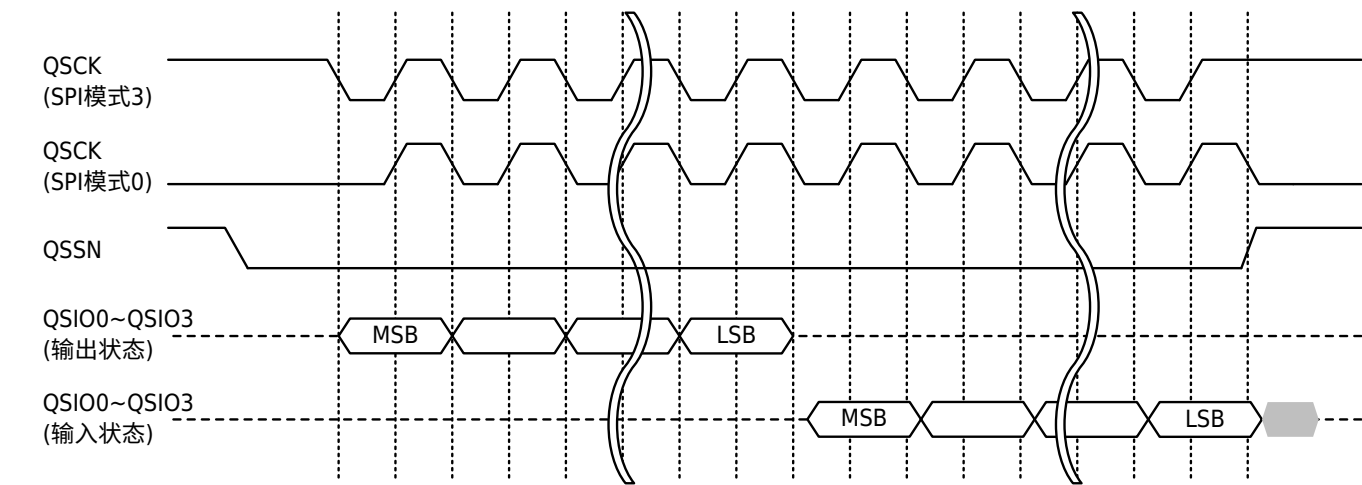


图 32-8 串行接口的基本时序图

32.4.3 QSPI 总线的时序调整

QSPI 总线信号可以通过寄存器进行时序微调, 微调后的时序可更好的应用于各种 QSPI 总线访问, 无论是 ROM 访问模式还是直接通信模式。

32.4.3.1 QSPI 总线基准时钟

QSPI 总线的基准时钟是由 HCLK 经过分频后得到的。通过设置 QSPI_CR 寄存器的 DIV[5:0]位可以选择 HCLK 的 2 分频到 64 分频等多种时钟源作为 QSPI 总线的基准时钟。

32.4.3.2 QSPI 通信时钟占空比修正

当基准时钟选择了 HCLK 的偶数倍分频时，QSCCK 信号的高低电平时间是一致的，如果选择了奇数倍分频，那么 QSCCK 信号的高电平时间将比低电平多一个 HCLK 周期。

如果希望选择奇数倍分频时的 QSCCK 也能输出 50%左右占空比的时钟信号，可以将 QSPI_FCR 寄存器中 DUTY 位设置成 1。通过这个设置，QSCCK 信号的上升沿的输出时间将比调整前晚半个 HCLK 周期，下降沿输出时间保持不变。由此可以得到占空比为 50%的 QSCCK 信号。当基准时钟选择 HCLK 的偶数倍分频时，DUTY 位请设置为 0。初始状态下 DUTY 位默认为 1。

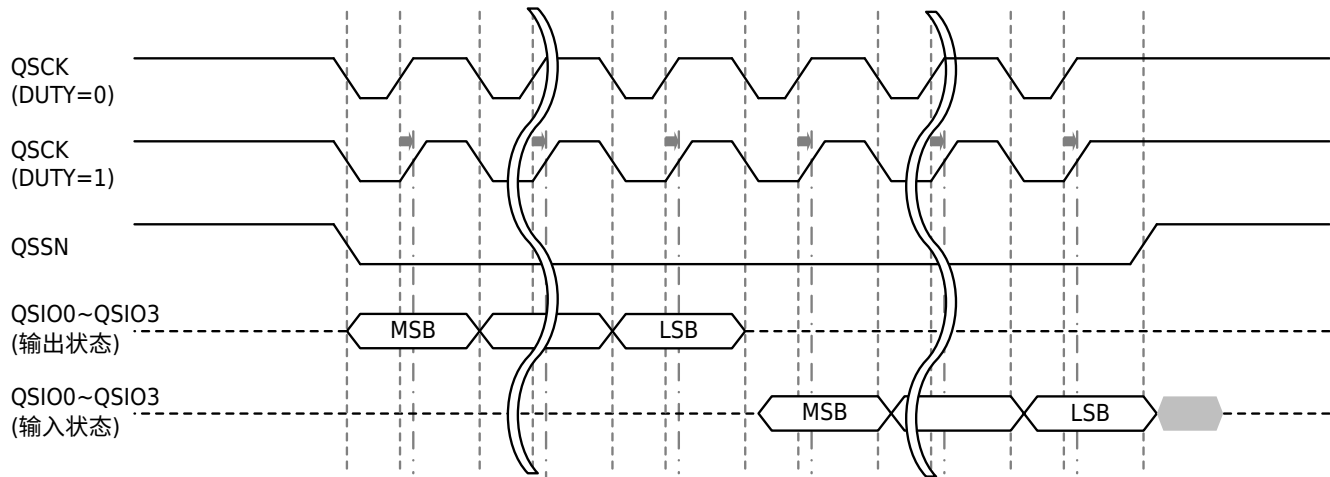


图 32-9 基准时钟选择 HCLK 三分频时输出时钟占空比修正示意图

32.4.3.3 QSSN 信号最小高电平宽度

为满足串行闪存所要求的取消时间要求，相邻的两次 QSPI 总线周期之间 QSSN 信号必须保持足够长时间的高电平状态（即闲置状态）。通过设置 QSPI_CSCR 寄存器中的 SSHW[3:0]位可以选择 QSSN 的最小高电平宽度，选择范围为 1 个到 16 个 QSPI 基准时钟周期。

32.4.3.4 QSSN 的建立时间

从 QSSN 信号开始输出低电平（即变为有效状态）到 QSCCK 信号输出第一个边沿之间的时间称为 QSSN 的建立时间，该时间可以通过寄存器设置进行配置以满足外接串行闪存的要求。设置寄存器 QSPI_FCR 的 SSNLD 位来选择 QSSN 的建立时间是 0.5 个或是 1.5 个 QSPI 基准时钟周期。该设置同样可用于配置数据管脚从数据输出许可到 QSCCK 信号输出第一个边沿之间的建立时间，可根据需要合理运用。

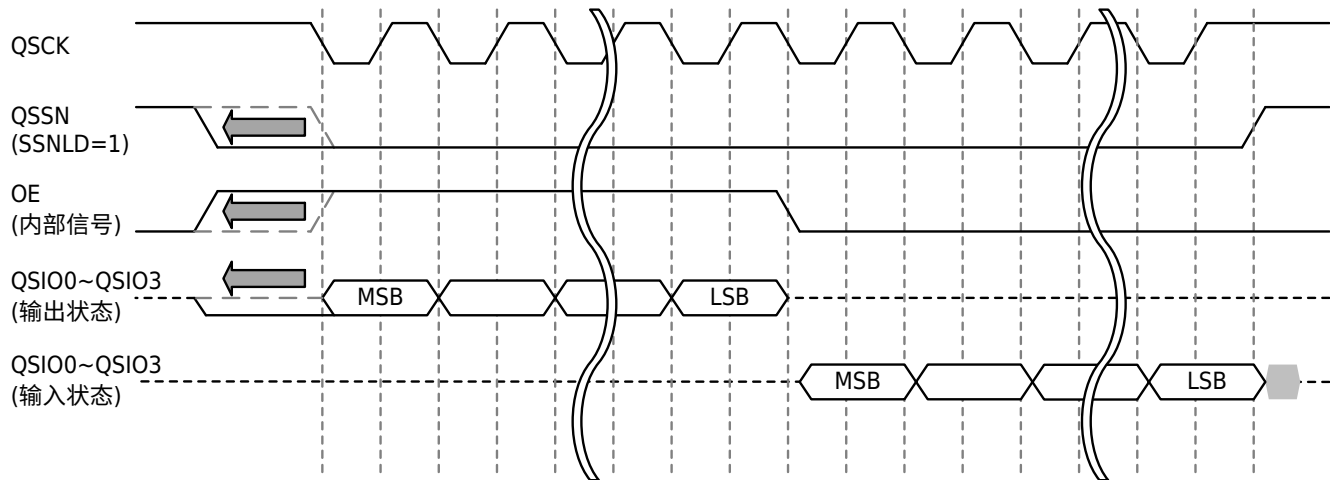


图 32-10 QSSN 建立时间配置示意图

32.4.3.5 QSSN 的保持时间

从 QSCK 信号输出最后一个边沿到 QSSN 信号开始输出高电平（即变为闲置状态）之间的时间称为 QSSN 的保持时间，该时间可以通过寄存器设置进行配置以满足外接设备的要求。设置寄存器 QSPI_FCR 的 SSNHD 位来选择 QSSN 的保持时间是 0.5 个或是 1.5 个 QSPI 基准时钟周期。

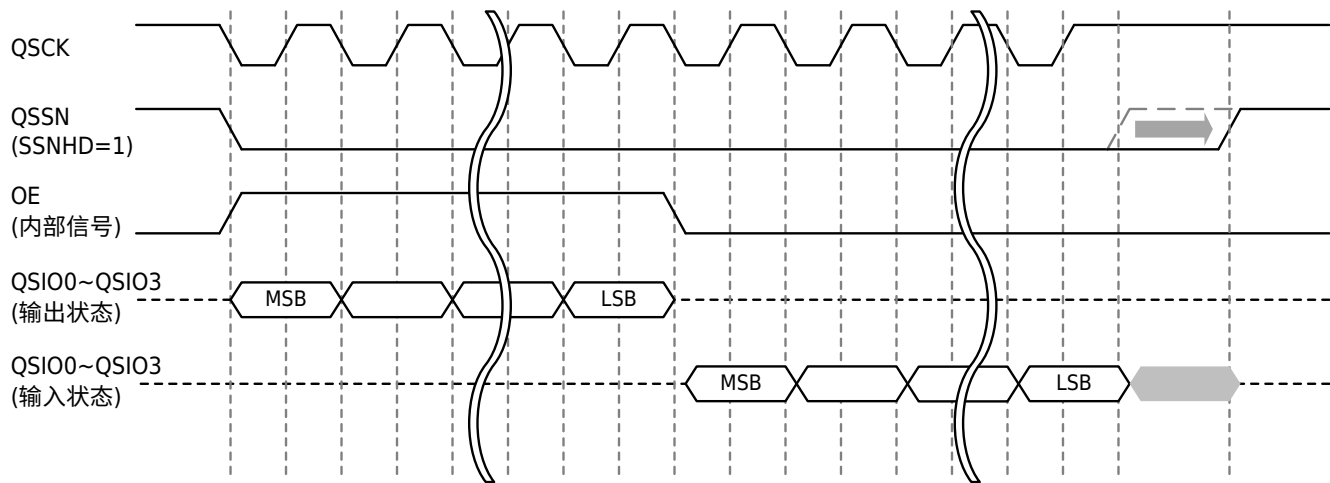


图 32-11 QSSN 保持时间配置示意图

32.4.3.6 串行数据接收延迟

串行闪存的数据和 QSCK 的下降沿同步输出，QSPI 在下一个 QSCK 的下降沿接收该数据。从串行闪存开始输出数据到该数据被 QSPI 接收之间的这段时间称之为接收延迟。QSPI 在第一个数据接收周期之前加入了一个延迟调整周期。站在串行闪存角度，该周期可视为对手方数据接收周期的增加。该延迟调整周期只会在数据接收动作时产生。

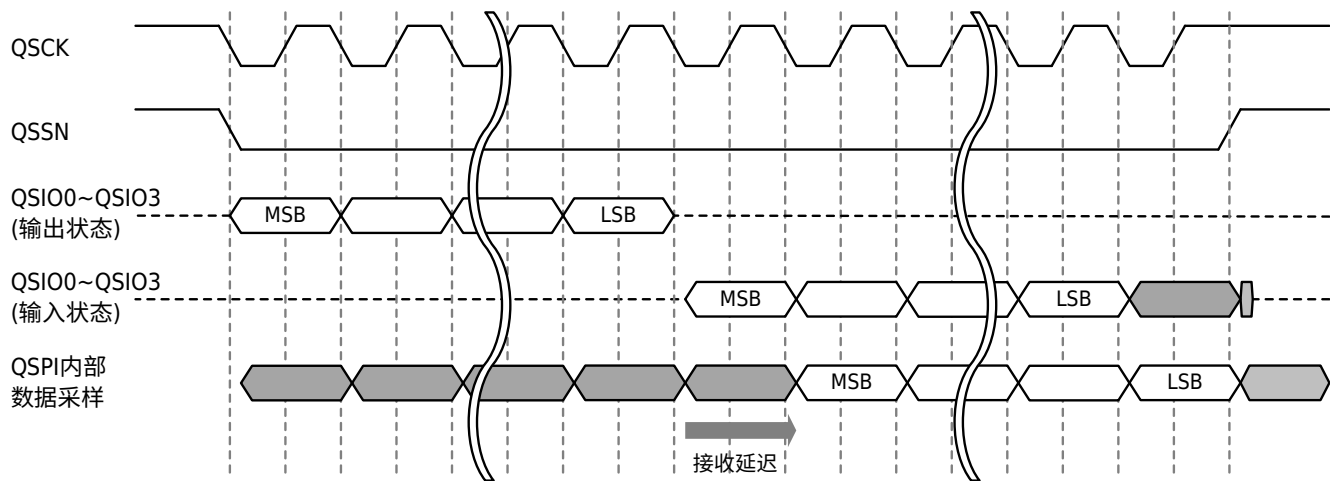


图 32-12 数据接收延迟示意图

32.4.4 用于 ROM 访问的 SPI 指令介绍

32.4.4.1 现有的 QSPI-ROM 指令参考

表 32-3 参考指令一览表

模式名		指令代码	说明
4-Bytes 地址模式	标准读	0x13	-
	快速读	0x0C	-

模式名		指令代码	说明
4-Bytes 地址模式	二线式输出快速读	0x3C	-
	二线式输入输出快速读	0xBC	-
	四线式输出快速读	0x6C	-
	四线式输入输出快速读	0xEC	-
	退出 4-Bytes 模式	0xE9	-
3-Bytes 地址模式	标准读	0x03	-
	快速读	0x0B	-
	二线式输出快速读	0x3B	-
	二线式输入输出快速读	0xBB	-
	四线式输出快速读	0x6B	-
	四线式输入输出快速读	0xEB	-
	进入 4-Bytes 模式	0xB7	-
-	写模式	0x06	-

串行闪存进行“自定义标准读”和“自定义快速读”访问时，指令需通过指令寄存器 QSPI_CCMD 来进行设置。

32.4.4.2 标准读指令

标准读指令是绝大多数串行闪存都支持的一种常用读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，随即 QSPI 就输出该指令的指令代码（0x03/ 0x13），紧接着输出目标地址，该地址的宽度可以通过 QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。然后就可以接收到数据。QSPI 的初始状态选择的指令就是标准读指令。

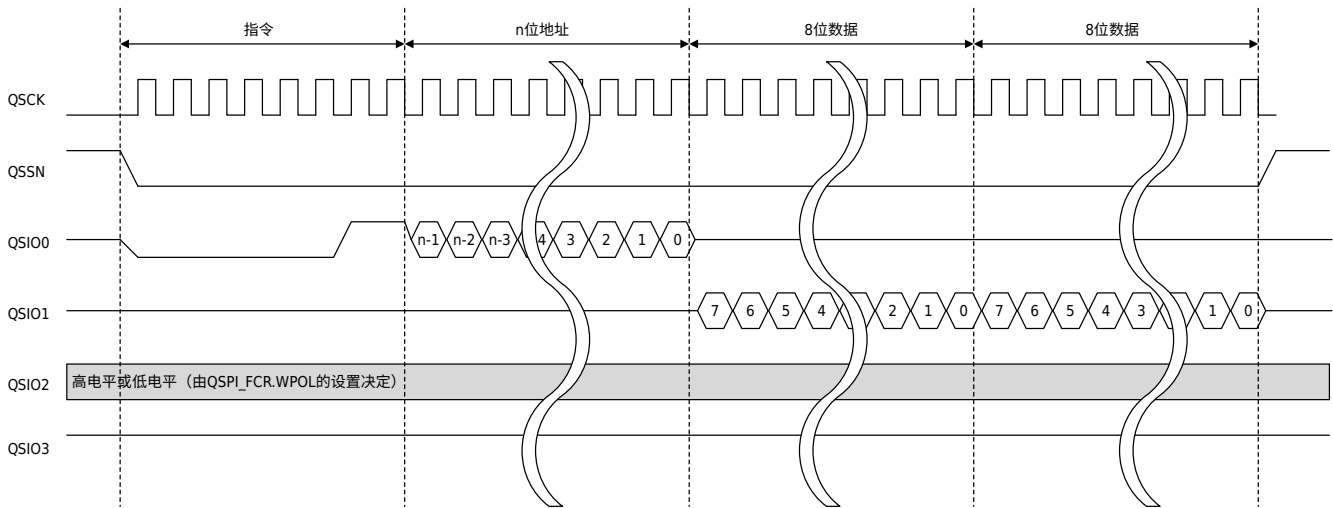


图 32-13 标准读总线周期示意图

32.4.4.3 快速读指令

快速读指令是一种支持更快的通信时钟的读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，随即 QSPI 就输出该指令的指令代码（0x0B/0x0C），紧接着输出目标地址，地址宽度可以通过

QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。地址输出后是有一定数量的虚拟周期，其具体数量由 QSPI_FCR 寄存器中的 DMCYCN[3:0]决定。紧接着就是数据的接收。

虚拟周期最初的两个周期用于决定是否选择 XIP 模式。当选择 XIP 模式时，本次传输所使用的指令将被应用于下一个 SPI 总线周期，在下一个 SPI 总线周期时指令传输部分将会被省略。详细可参见“XIP 控制”。

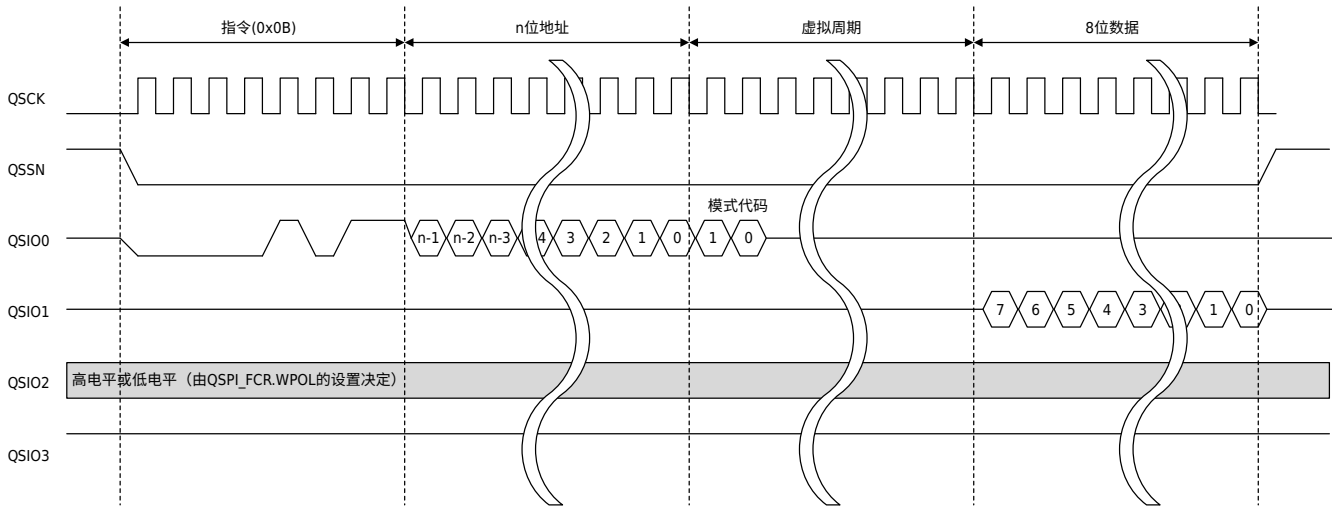


图 32-14 快速读总线周期示意图

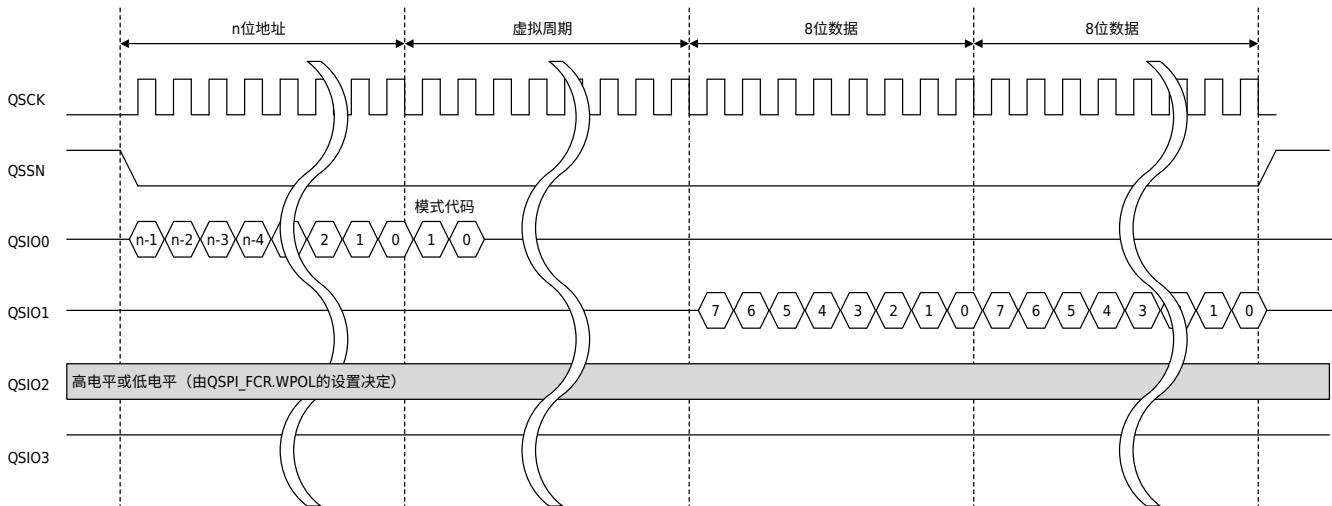


图 32-15 选择 XIP 模式的快速读总线周期示意图



说明

要使用快速读指令请确保使用支持快速读功能的串行闪存。

32.4.4.4 二线式输出快速读指令

二线式输出快速读是一种使用两根信号线进行数据接收的读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSIO0 管脚输出该指令的指令代码 (0x3B/0x3C) 和目标地址，地址宽度可以通过 QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。在这之后是一定数量的虚拟周期，其具体数量由 QSPI_FCR 寄存器中的 DMCYCN[3:0]决定。然后开始通过 QSIO0 和 QSIO1 两根管脚进行数据接收。偶位数据在 QSIO0 接收，奇位在 QSIO1。

虚拟周期最初的两个周期用于决定是否选择 XIP 模式。当选择 XIP 模式时，本次传输所使用的指令将被应用于下一个 QSPI 总线周期，在下一个 QSPI 总线周期时指令传输部分将会被省略。详细可参见“XIP 控制”。

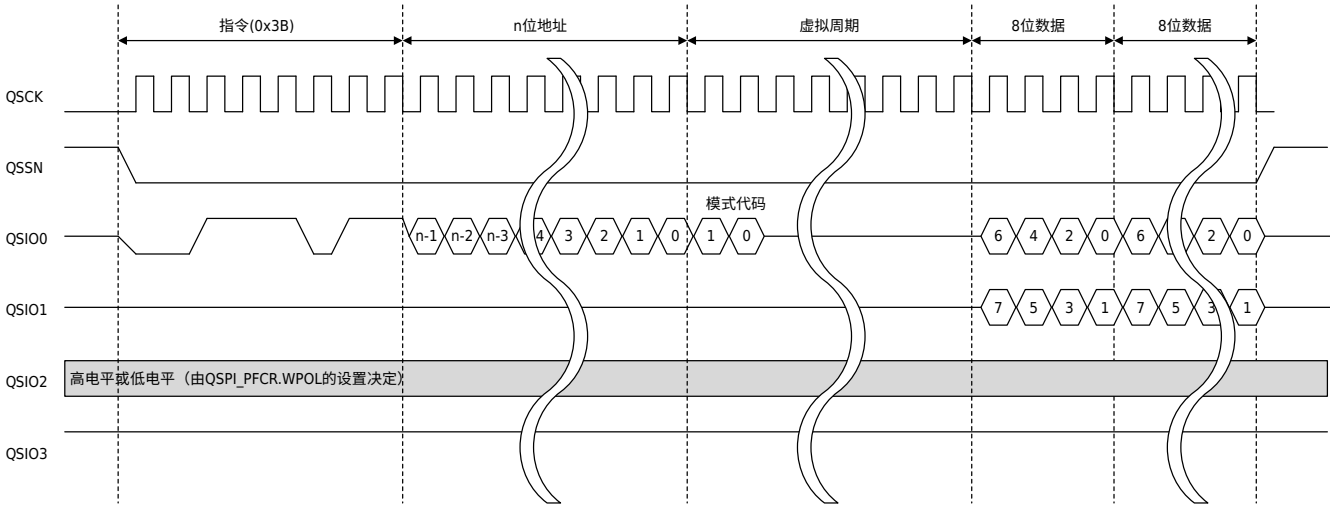


图 32-16 二线式输出快速读总线周期示意图

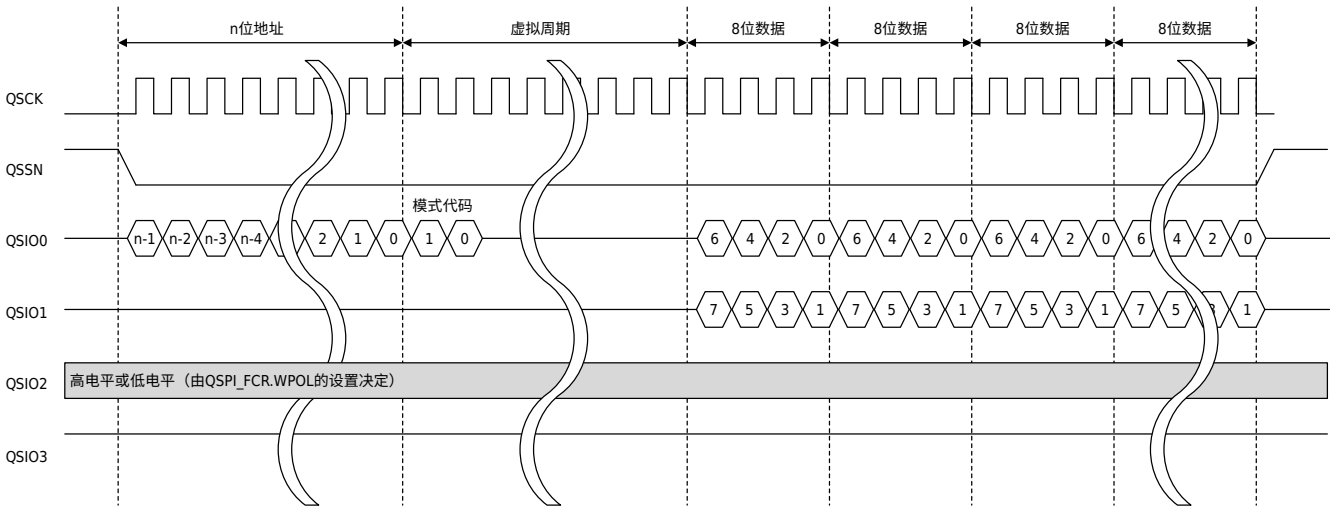


图 32-17 选择 XIP 模式的二线式输出快速读总线周期示意图

说明
要使用二线式输出快速读指令请确保使用支持该功能的串行闪存。

32.4.4.5 二线式输入输出快速读指令

二线式输入输出快速读是一种使用两根信号线进行地址发送和数据接收的读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSI00 管脚输出该指令的指令代码 (0xBB/0xBC)。在这之后 QSPI 从 QSI00 和 QSI01 两个管脚输出目标地址，地址宽度可以通过 QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。在这之后是一定数量的虚拟周期，其具体数量由 QSPI_FCR 寄存器中的 DMCYCN[3:0]决定。然后开始通过 QSI00 和 QSI01 两根管脚进行数据接收。偶数位的地址和虚拟周期（包括 XIP 模式选择信息）的传输及数据接收使用 QSI00 管脚，奇数位使用 QSI01 管脚。

虚拟周期最初的两个周期用于决定是否选择 XIP 模式。当选择 XIP 模式时，本次传输所使用的指令将被应用于下一个 QSPI 总线周期，在下一个 QSPI 总线周期时指令传输部分将会被省略。详细可参见“XIP 控制”。

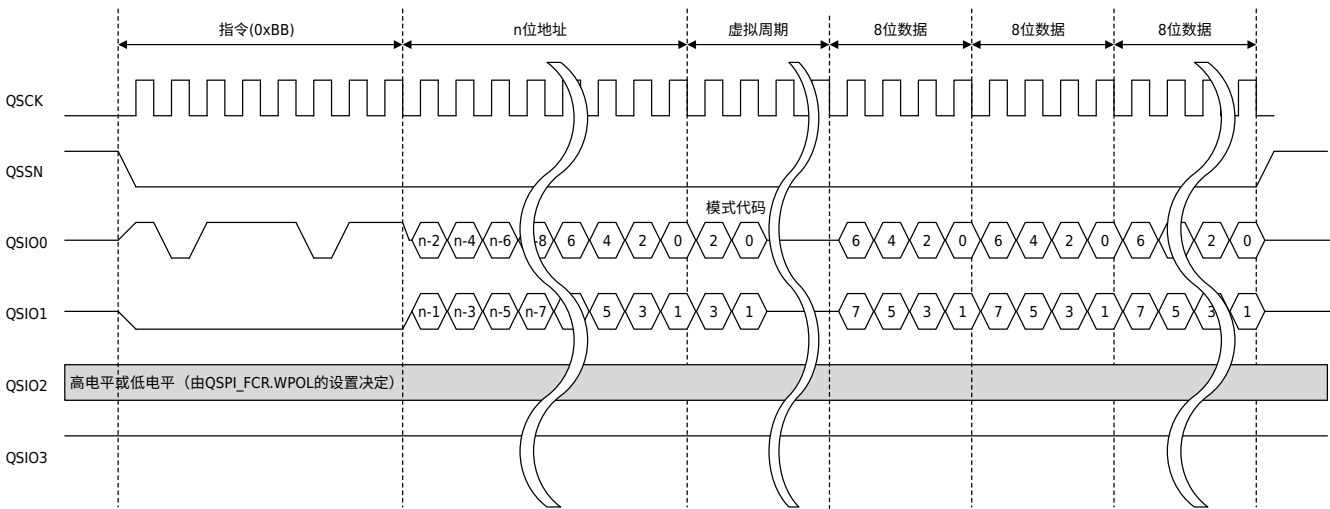


图 32-18 二线式输入输出快速读总线周期示意图

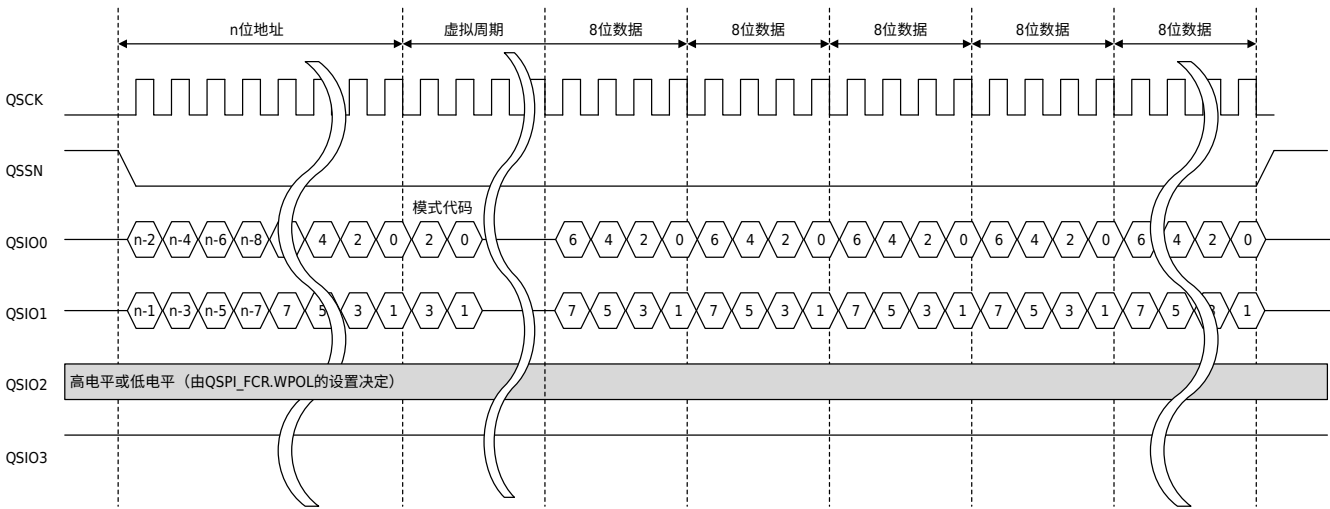


图 32-19 选择 XIP 模式的二线式输入输出快速读总线周期示意图

**说明**

要使用二线式输入输出快速读指令请确保使用支持该功能的串行闪存。

32.4.4.6 四线式输出快速读指令

四线式输出快速读是一种使用四根信号线进行数据接收的读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSIO0 管脚输出该指令的指令代码 (0x6B/0x6C) 和目标地址，地址宽度可以通过 QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。在这之后是 一定数量的虚拟周期，其具体数量由 QSPI_FCR 寄存器中的 DMCYCN[3:0]决定。然后开始通过 QSIO0, QSIO1, QSIO2 和 QSIO3 四根管脚进行数据接收。

虚拟周期最初的两个周期用于决定是否选择 XIP 模式。当选择 XIP 模式时，本次传输所使用的指令将被应用于下一个 SPI 总线周期，在下一个 SPI 总线周期时指令传输部分将会被省略。详细可参见“XIP 控制”。

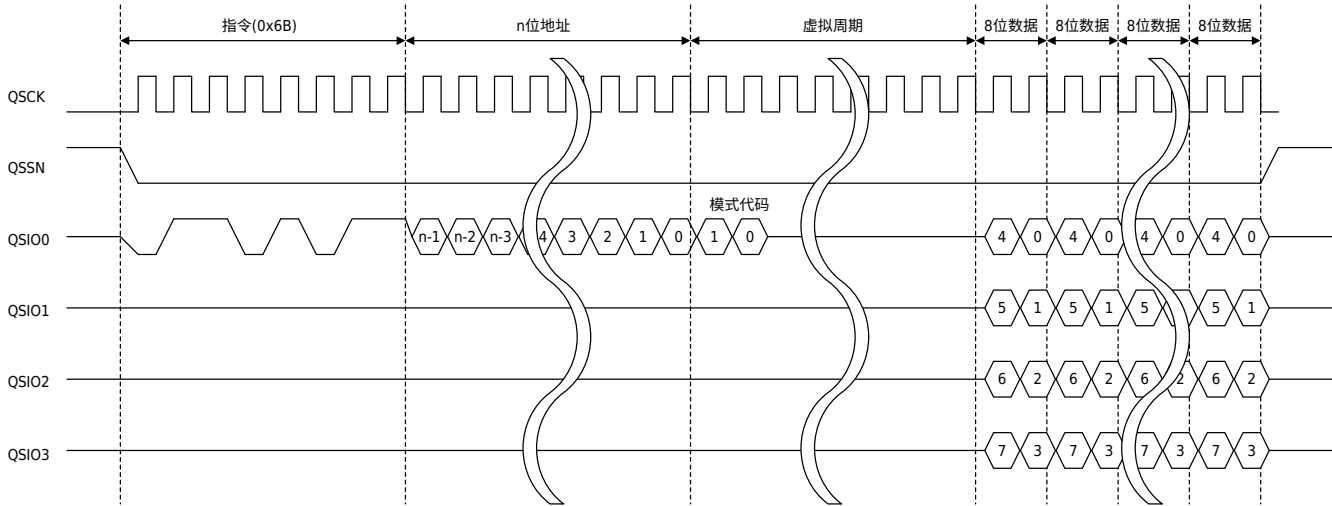


图 32-20 四线式输出快速读总线周期示意图

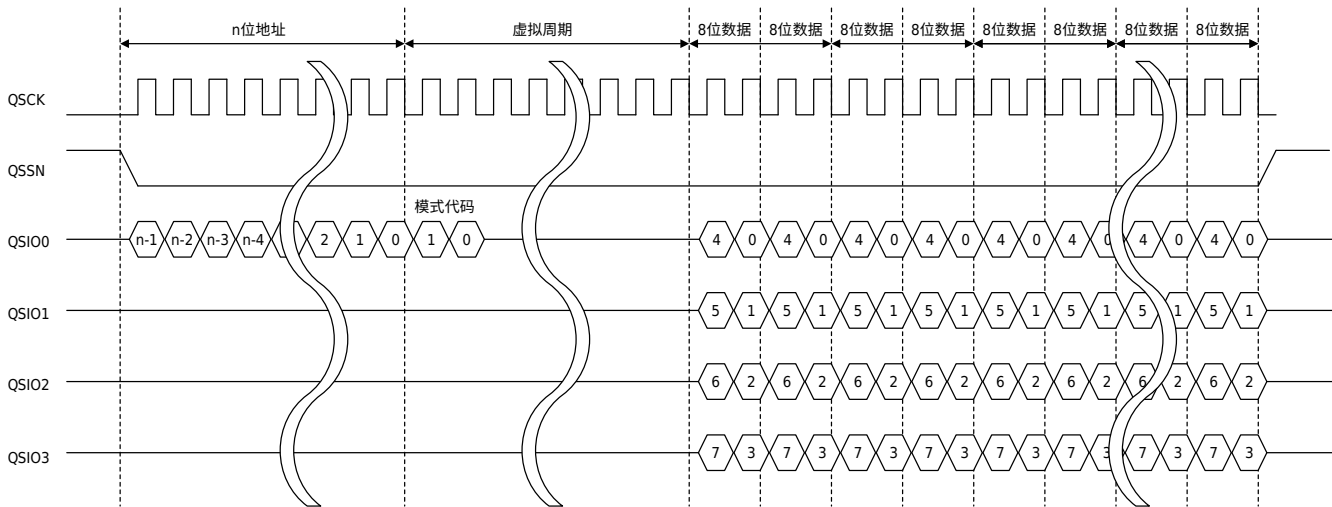


图 32-21 选择 XIP 模式的四线式输出快速读总线周期示意图

说明
要使用四线式输出快速读指令请确保使用支持该功能的串行闪存。

32.4.4.7 四线式输入输出快速读指令

四线式输入输出快速读是一种使用四根信号线进行地址发送和数据接收的读指令。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSI00 管脚输出该指令的指令代码 (0xEB/0xEC)。在这之后 QSPI 从 QSI00, QSI01, QSI02 和 QSI03 四个管脚输出目标地址，地址宽度可以通过 QSPI_FCR 寄存器中的 AWSL[1:0]位来设置。在这之后是一定数量的虚拟周期，其具体数量由 QSPI_FCR 寄存器中的 DMCYCN[3:0]决定。然后开始通过 QSI00, QSI01, QSI02 和 QSI03 四根管脚进行数据接收。虚拟周期最初的两个周期用于决定是否选择 XIP 模式。当选择 XIP 模式时，本次传输所使用的指令将被应用于下一个 QSPI 总线周期，在下一个 QSPI 总线周期时指令传输部分将会被省略。详细可参见“XIP 控制”。

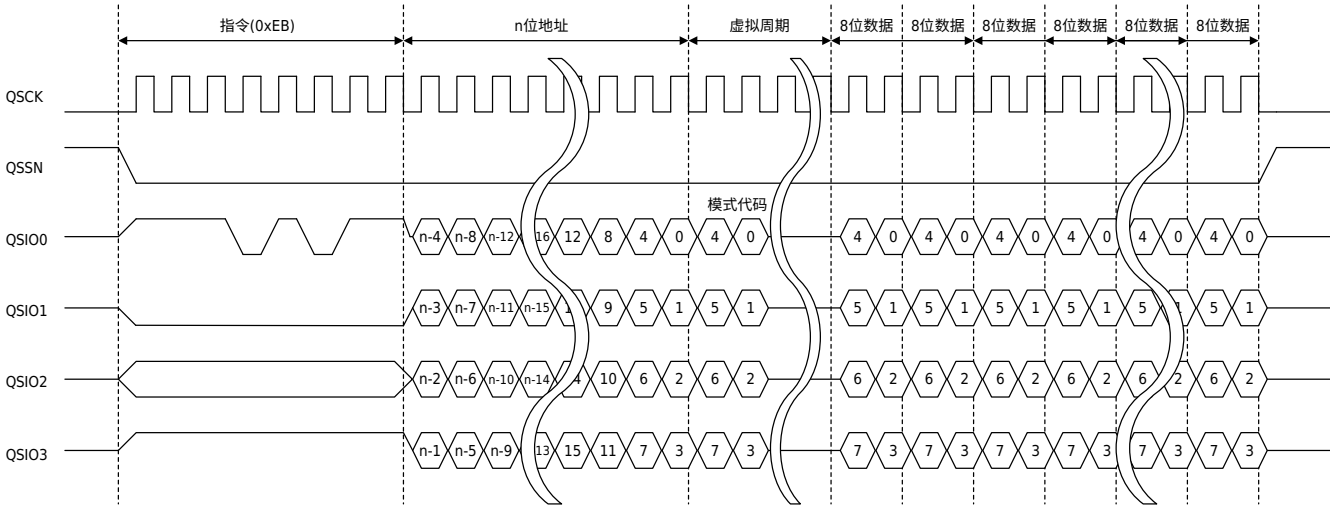


图 32-22 四线式输入输出快速读总线周期示意图

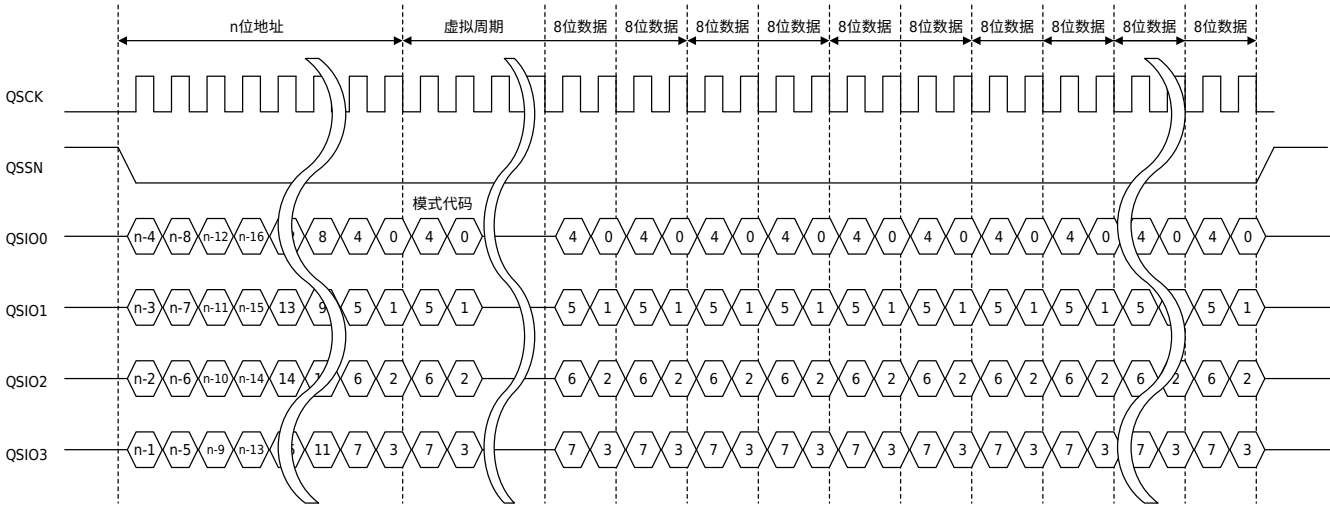


图 32-23 选择 XIP 模式的四线式输入输出快速读总线周期示意图

说明
要使用四线式输入输出快速读指令请确保使用支持该功能的串行闪存。

32.4.4.8 进入 4-Bytes 模式指令

进入 4-Bytes 模式指令可以将串行闪存地址宽度设置成 4 个字节。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSI00 管脚输出该指令的指令代码（0xB7）。

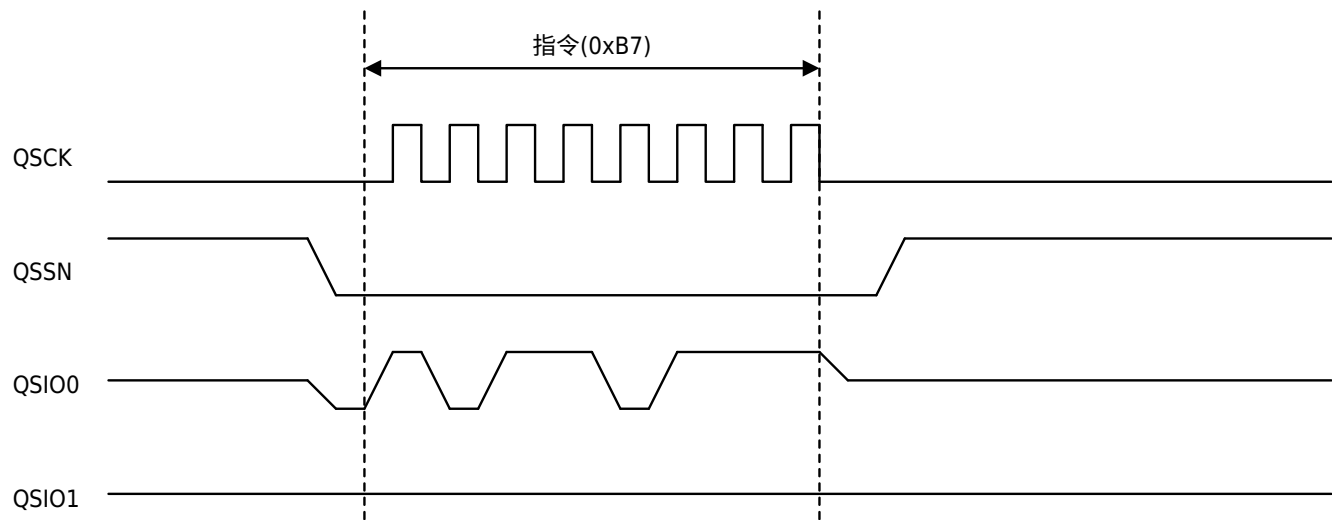


图 32-24 进入 4-Bytes 模式指令总线周期示意图



说明

无论串行闪存是否处于 4-Bytes 或是 3-Bytes 模式，该指令都可发布。

32.4.4.9 退出 4-Bytes 模式指令

退出 4-Bytes 模式指令可以将串行闪存地址宽度设置成 3 个字节。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSI00 管脚输出该指令的指令代码（0xE9）。

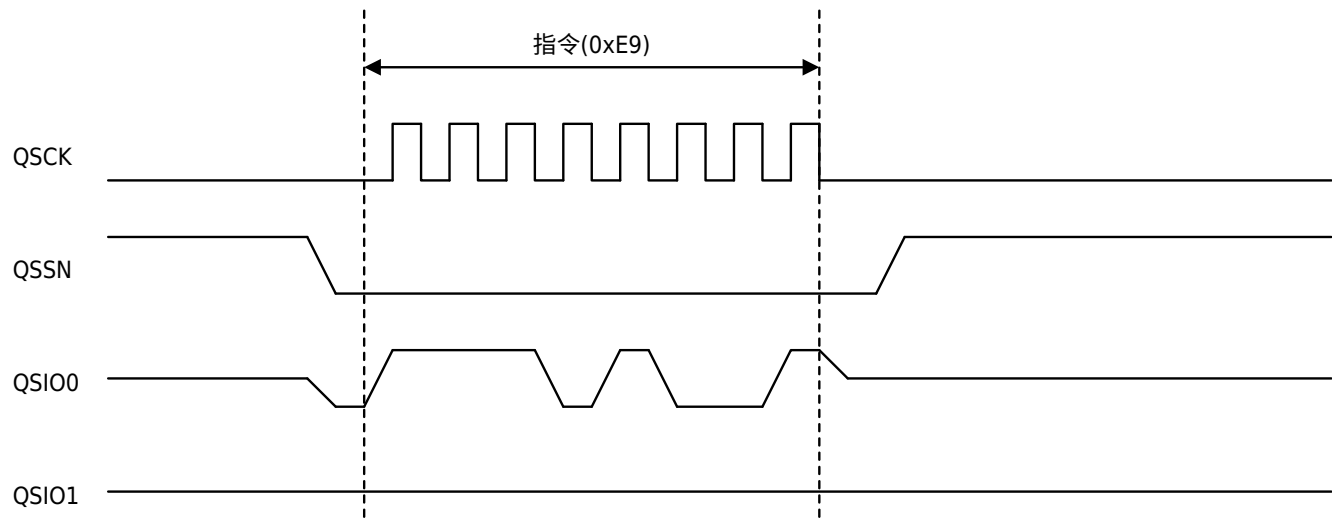


图 32-25 退出 4-Bytes 模式指令总线周期示意图



说明

无论串行闪存是否处于 4-Bytes 或是 3-Bytes 模式，该指令都可发布。

32.4.4.10 写许可指令

写许可指令允许串行闪存进行写操作。当一个串行总线周期开始的时候，串行闪存选择信号被置为有效状态，QSPI 开始从 QSI00 管脚输出该指令的指令代码（0x06）。

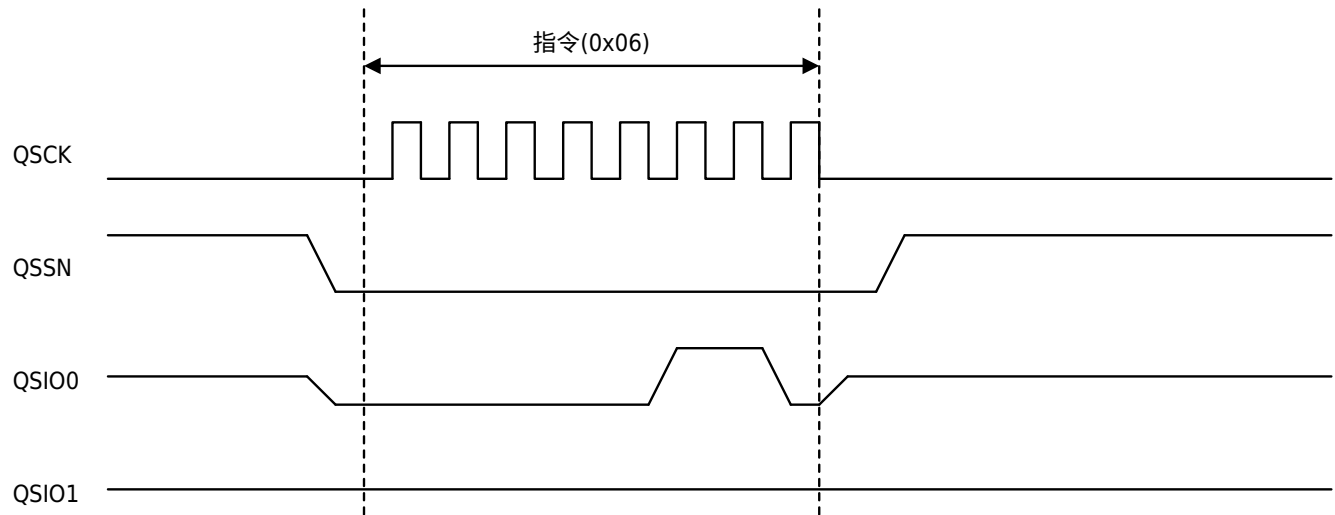


图 32-26 写许可指令总线周期示意图

32.4.5 QSPI 总线周期的安排

32.4.5.1 独立转换的单一闪存读取

对于 ROM 的单一读取指令会独立的从芯片内部总线周期一对一的转换为 QSPI 总线周期。当一个 ROM 的读取总线周期被检测到时，QSSN 信号会置为有效状态，从而启动一个 QSPI 总线周期。当接收完串行闪存的数据后，QSSN 信号变成无效状态，该 QSPI 总线周期宣告完成。

当另一个 ROM 的读取总线周期被检测到时，QSSN 信号会在确保无效保持时间已经超过最小无效保持宽度后再度置为有效状态，新一个 QSPI 总线周期开始。

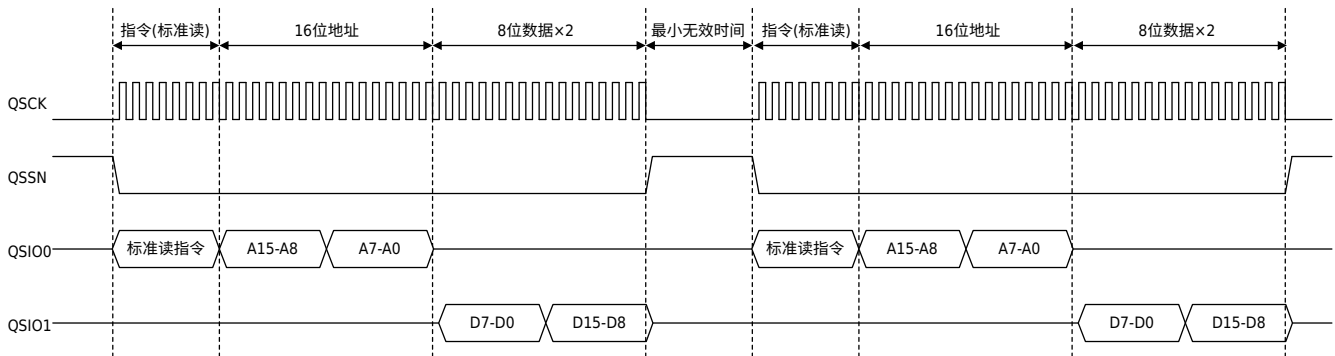


图 32-27 独立转换的单一闪存数据读取操作示意图

32.4.5.2 使用预读取功能进行闪存读取

对于诸如 CPU 指令或是数据块的传输，系统通常是以一个顺次递增的闪存地址顺序进行数据的读取。串行闪存具有连续数据传输能力而不需要再次发送指令代码和地址。但如果 MCU 发布的内部总线周期是独立转换的话，QSPI 总线周期也被分成了各自独立的个体，导致无法有效利用串行闪存的这种连续数据传输的优点。对此，QSPI 提供了预读取功能来进行连续的数据接收。

通过设置 QSPI_CR 寄存器中的 PFE 位为 1 激活预读取功能。当该功能有效后，数据会被连续接收并储存到缓冲区而不需要等待另一个闪存读取要求。当 MCU 发出一个闪存读取操作时，QSPI 将会对访问地址进行匹配。如果匹配成功，则将对对应位置的缓冲区的数据传给 MCU，若匹配失败，则缓冲区的数据将会被丢弃并重新发出一个新的 QSPI 总线周期。

预读取的缓冲区最大可存储 16 个字节的数据，除此以外，还有 2 个字节的数据接收缓冲区也可以存储预读取的数据，当所有的缓冲区数据取满后，QSPI 总线周期结束。当缓冲区数据被读取后产生新的缓冲空间，QSPI 会自动开始一个新的 QSPI 总线周期来恢复预读取动作。

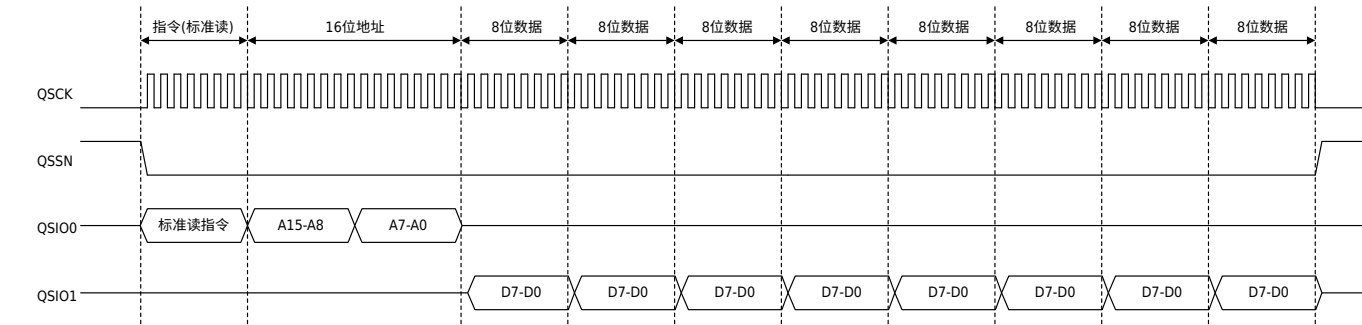


图 32-28 预读取功能有效时数据读取操作示意图

32.4.5.3 预读取的终止

在预读取传送过程中如果发生了对串行闪存其他地址的 ROM 读取总线周期，那么原预读取动作将会被终止并开始一个新的 QSPI 总线周期。通常情况下预读取动作会在当前字节传输完成之后终止，但如果将 QSPI_CR 寄存器中的 PFSAE 位设置成 1 的话，QSPI 会即时停止预读取动作而不会等到当前字节传输完成。使用这一功能需要串行闪存器件支持即时停止动作功能。

32.4.5.4 预读取状态监测

从一个低速串行闪存读取数据会增加系统的负荷，这是因为内部总线需要一直处在等待状态直到收信的 QSPI 总线周期完成。QSPI 提供了预读取状态监测功能来降低这种负荷。在预读取状态寄存器 QSPI_SR 中，PFAN 位显示了当前的预读取工作状态，PFFUL 位表示预读取数据缓冲区已经放满，而 PFNUM[4:0]则显示了目前已经读取到缓冲区的数据的字节数。通过这些状态位可以很方便的通过一条 CPU 指令来确定当前的预读取状况。

说明

当执行一段预读取状态监测程序时，请将该程序代码放置在对象串行闪存区域以外或者启用指令缓存。否则预读取对象将频繁的切换在对象数据区域和指令区域之间，失去了预读取的意义，而监测程序也会因为预读取始终无法完成而进入无限循环状态。

32.4.5.5 使用 QSPI 总线周期延长功能进行闪存读取

如果对 QSPI_CSCR 寄存器中的 SSNW[1:0]进行设置 0b00 以外的值，QSPI 总线周期将会在接收完数据后处于保持状态等待下一个数据的读取。这时 QSSN 信号将保持低电平的有效状态，而 QSCCK 则处于停止状态。如果下一条闪存读取指令到来时，如果读取对象地址是紧接着当前地址顺序递增的，则 QSPI 会将 QSCCK 重新启动直接进行数据接收。如果是非连续的地址的话，QSSN 信号会被置成高电平的无效状态将原先保持的 QSPI 总线周期结束，然后重新开始一个新的 SPI 总线周期。

这一功能可以避免系统在对连续递增地址进行不连续读取时反复发送指令代码和地址，从而提高读取效率。QSPI 总线周期的延长时间可以通过 SSNW[1:0]设置，当超过所设置的时间仍未发生下一次的读取时，QSSN 会自动置成高电平的无效状态，结束 QSPI 总线周期。如果 SSNW[1:0]设成 0b11 的话，QSSN 会被无限期延长，QSPI 总线周期将始终处在保持状态，但这样做会增加串行闪存的功耗。

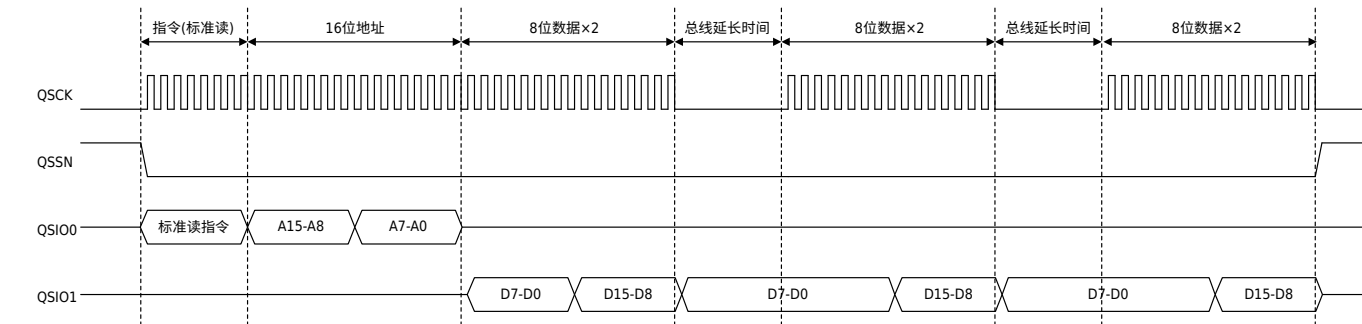


图 32-29 使用 QSPI 总线周期延长功能的数据读取操作示意图

32.4.6 XIP 控制

32.4.6.1 XIP 模式概述

一些串行闪存器件可以通过省略接收读取指令来降低延迟时间。该机能可通过虚拟周期期间发送的模式代码来选择。

在快速度指令时的虚拟周期期间，QSPI 在最初的两个周期通过发送 XIP 模式代码来控制串行闪存的 XIP 模式。不同的串行闪存其 XIP 模式代码也不相同，可通过寄存器 QSPI_XCMD 的 XIPMC[7:0]位进行针对性设置。具体参照下图。

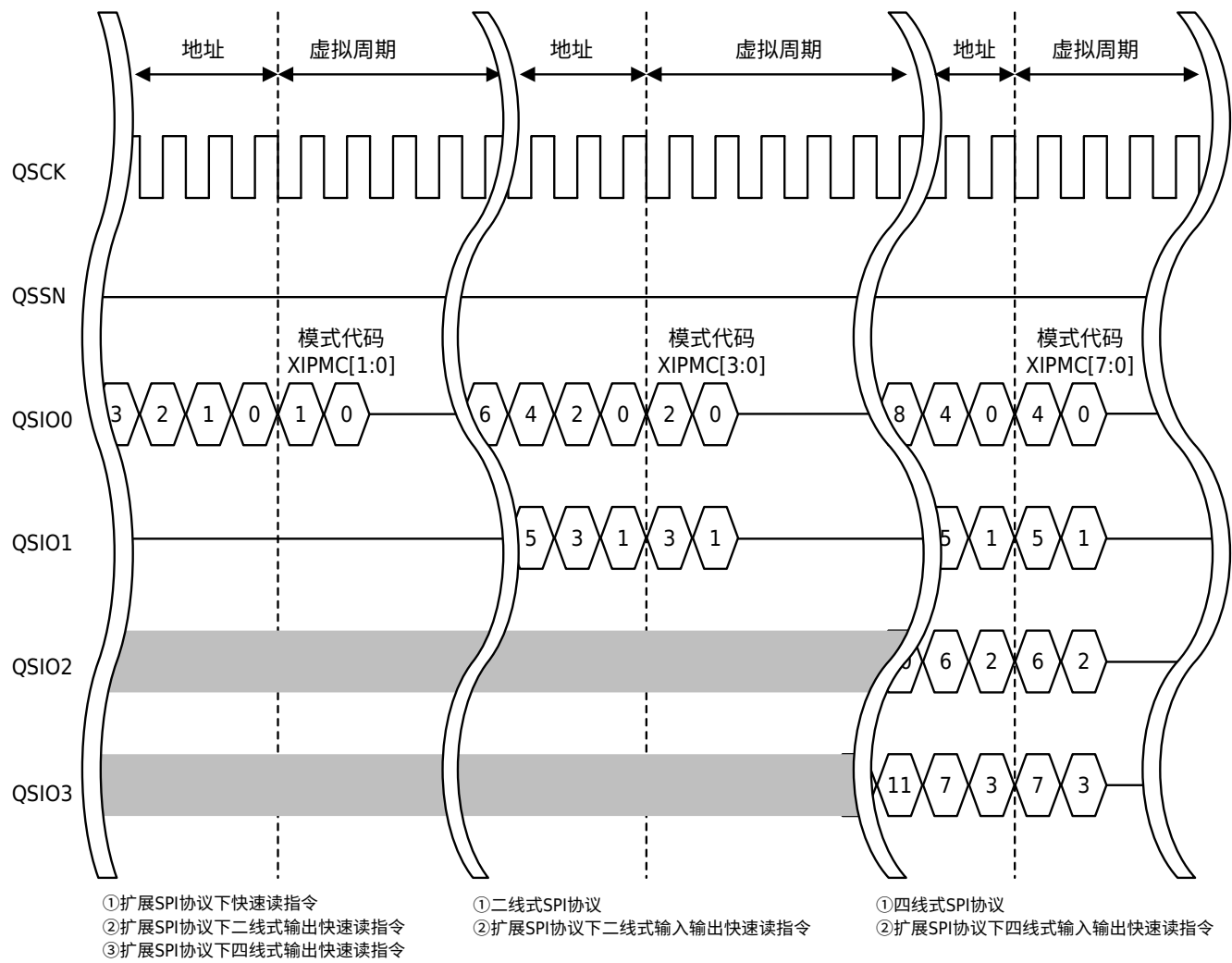


图 32-30 XIP 模式控制示意图

32.4.6.2 XIP 模式的设置

在将对应串行闪存的 XIP 模式代码写入寄存器 QSPI_XCMD 的 XIPMC[7:0]且寄存器 QSPI_CR 的 XIPE 位设置成 1 的情况下，当下一个快速读指令发生时，所设置的模式代码将在虚拟周期的前两个周期期间传送给对象串行闪存，在模式代码接收完成后，串行闪存及其控制部就启动 XIP 模式。可以通过访问 QSPI_SR 的 XIPF 位来确认是否已经进入 XIP 模式。

说明

启动串行闪存的 XIP 模式需要在 QSPI_XCMD[7:0]中设置相应的模式代码，控制部分的 XIP 模式只需要将 XIPE 位置成 1 就可以，而与 QSPI_XCMD[7:0]的值无关。

32.4.6.3 XIP 模式的退出

在将对应串行闪存的退出 XIP 模式的代码写入寄存器 QSPI_XCMD 的 XIPMC[7:0]且寄存器 QSPI_CR 的 XIPE 位设置成 0 的情况下，当下一个快速读指令发生时，所设置的退出模式代码将在虚拟周期的前两个周期期间传送给对象串行闪存，在退出模式代码接收完成后，串行闪存及其控制部的 XIP 模式被终止。可以通过访问 QSPI_SR 的 XIPF 位来确认是否已经退出 XIP 模式。

说明

退出串行闪存的 XIP 模式需要在 QSPI_XCMD[7:0]中设置相应的退出模式代码，控制部分的 XIP 模式只需要将 XIPE 位清零就可以，而与 QSPI_XCMD[7:0]的值无关。

32.4.7 QSI02 和 QSI03 的管脚状态

QSI02 和 QSI03 管脚状态取决于 QSPI_CR 寄存器中 MDSEL[2:0]位所设置的串行读取模式。

表 32-4 QSI02 和 QSI03 的管脚状态

QSPI_CR 寄存器 MDSEL[2:0]位	QSI02 状态	QSI03 状态	备注
0b000	输出状态，输出电平由 QSPI_FCR 寄存器的 WPOL 位决定，初始输出为低电平	输出高电平	标准读（初始状态）
0b001			快速读
0b010			二线式输出快速读
0b011			二线式输入输出快速读
0b100	作为第三根数据线进行输入或输出动作，空闲状态为 Hi-Z	作为第四根数据线进行输入或输出动作，空闲状态为 Hi-Z	四线式输出快速读
0b101			四线式输入输出快速读
0b110	参照各个阶段具体的协议设置	参照各个阶段具体的协议设置	自定义协议标准读
0b111			自定义协议快速读

说明

- QSI02 管脚也可用作串行闪存的 WP#功能。
- QSI03 管脚也可用作串行闪存 HOLD#或 RESET#功能。

32.4.8 直接通信模式

32.4.8.1 关于直接通信模式

QSPI 可以通过自动将 MCU 的外部 ROM 读取总线周期转换为 QSPI 总线周期来对串行闪存进行读取。但串行闪存还有很多不同的追加功能，诸如 ID 信息读取，擦除，写入及状态信息读取等。这些功能并没有一套标准的指令来进行设置，并且随着串行闪存新功能的迅速增加，硬件层面上的对应变得愈发的困难。

针对这种情况，QSPI 提供了直接通信模式，用户可通过软件直接对串行闪存进行控制。由此模式软件可以产生任意所需的 QSPI 总线周期。

32.4.8.2 直接通信模式的设置

将 QSPI_CR 寄存器的 DCOME 位设成 1 可以进入直接通信模式。一旦进入直接通信模式，将无法进行通常的闪存读取操作，如果要进行常规的闪存读取，需要将 DCOME 位清零退出直接通信模式。

**说明**

如果处于 XIP 模式，则需要先退出 XIP 模式再启动直接通信模式。

32.4.8.3 直接通信模式下 QSPI 总线周期的生成

直接通信模式下一个完整的 QSPI 总线周期从对寄存器 QSPI_DCOM 第一次操作开始直到对 QSPI_CR 寄存器的进行一次写操作后结束。在这期间可以对 DCOMPRSL[1:0]和 DCOM[7:0]进行多次操作。

直接通信模式支持多线式动作。

- 当 DCOMPRSL[1:0]设置为 0b00 时，DCOM[7:0]的写入转换为一次 QSPI 总线扩展式 SPI 协议的单字节的数据传送，而对 DCOM[7:0]的读则会转换成一次 QSPI 总线扩展式 SPI 协议的单字节的数据接收。
- 当 DCOMPRSL[1:0]设置为 0b01 时，DCOM[7:0]的写入转换为一次 QSPI 总线二线式 SPI 协议的单字节的数据传送，而对 DCOM[7:0]的读则会转换成一次 QSPI 总线二线式 SPI 协议的单字节的数据接收。
- 当 DCOMPRSL[1:0]设置为 0b10 时，DCOM[7:0]的写入转换为一次 QSPI 总线四线式 SPI 协议的单字节的数据传送，而对 DCOM[7:0]的读则会转换成一次 QSPI 总线四线式 SPI 协议的单字节的数据接收。

从对寄存器 QSPI_DCOM 的第一次操作开始到最后对 QSPI_CR 寄存器的写操作为止，这期间 QSSN 信号始终保持低电平的有效状态。

**说明**

- 在直接通信模式下对 QSPI_DCOM 以外的 QSPI 寄存器进行写操作，都会退出直接通信模式。使用“QSPI_CR 寄存器写操作”以外的方式退出直接通信模式可能会导致不可预料的情况出现，所以不推荐使用。
- 直接通信模式下可以通过 DCOMPRSL[1:0]的改写来自由切换一线式、二线式或四线式。如果是直接读操作之前需要切换线式，请注意 DCOMPRSL[1:0]的写入会带来一次 Dummy 数据传送。
- 对 QSPI_DCOM 寄存器进行写操作时，必须使用 32 位或 16 位操作，同时写入 DCOMPRSL[1:0]和 DCOM[7:0] 位。对 QSPI_DCOM 寄存器进行读操作时，可以使用 32 位、16 位或 8 位操作。

32.4.9 中断

当直接通信模式下检测到对 ROM 的读访问操作时，QSPI_SR 寄存器的 RAER 位被置成 1，这时候 QSPI 将产生总线硬件错误中断。该中断请求将会一直被保留直到 RAER 位被清零。详细信息请参见《中断控制器 (INTC) 》。

32.5 寄存器

32.5.1 寄存器总表

QSPI 基地址：0x9C000000

表 32-5 四线式串行外设接口（QSPI）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	QSPI_CR	QSPI 控制寄存器	32
0x0004	QSPI_CSCR	QSPI 片选控制寄存器	32
0x0008	QSPI_FCR	QSPI 格式控制寄存器	32
0x000C	QSPI_SR	QSPI 状态寄存器	32
0x0010	QSPI_DCOM	QSPI 直接通信指令寄存器	32
0x0014	QSPI_CCMD	QSPI 指令代码寄存器	32
0x0018	QSPI_XCMD	QSPI XIP 模式代码寄存器	32
0x0024	QSPI_CLR	QSPI 标志清除寄存器	32
0x0804	QSPI_EXAR	QSPI 外部扩展地址寄存器	32

32.5.2 QSPI 控制寄存器 (QSPI_CR)

Offset	Bit Position															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x003F0000															
Name	Reserved								DIV				Reserved	DPRSL	APRSL	IPRSL
Access									RW					RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21:16	DIV	基准时钟选择	串行接口基准时钟周期配置： ● 0b000000：2 个 HCLK 周期 ● 0b000001：2 个 HCLK 周期 ● 0b000010：3 个 HCLK 周期 ● ● 0b111110：63 个 HCLK 周期 ● 0b111111：64 个 HCLK 周期  说明 硬件实际使用值为编程值加 1。	RW
15:14	Reserved	保留	-	-
13:12	DPRSL	数据接收阶段 SPI 协议选择	● 0b00：扩展式 SPI 协议 ● 0b01：二线式 SPI 协议 ● 0b10：四线式 SPI 协议 ● 0b11：禁止设置	RW
11:10	APRSL	地址发送阶段 SPI 协议选择	● 0b00：扩展式 SPI 协议 ● 0b01：二线式 SPI 协议 ● 0b10：四线式 SPI 协议 ● 0b11：禁止设置	RW
9:8	IPRSL	指令发送阶段 SPI 协议选择	● 0b00：扩展式 SPI 协议 ● 0b01：二线式 SPI 协议 ● 0b10：四线式 SPI 协议 ● 0b11：禁止设置	RW
7	SPIMD3	SPI 模式选择	● 0b0：SPI 模式 0 ● 0b1：SPI 模式 3	RW
6	XIPE	XIP 模式许可	● 0b0：XIP 模式禁止 ● 0b1：XIP 模式许可	RW
5	DCOME	直接通信许可	QSPI 总线通信模式选择： ● 0b0：ROM 访问模式 ● 0b1：直接通信模式	RW
4	PFSAE	预读取即时停止许可	选择重置预读取动作的位置： ● 0b0：当前的预读取动作在字节边界中止 ● 0b1：当前的预读取动作即时中止	RW

位/位域	标记	位名	功能描述	读写
3	PFE	预读取许可	0b0: 预读取功能无效 0b1: 预读取功能有效	RW
2:0	MDSEL	QSPI 读取模式选择	0b000: 标准读 0b001: 快速读 0b010: 二线式输出快速读 0b011: 二线式输入输出快速读 0b100: 四线式输出快速读 0b101: 四线式输入输出快速读 0b110: 自定义标准读 0b111: 自定义快速读	RW

32.5.3 QSPI 片选控制寄存器 (QSPI_CSCR)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000000F																															
Name	Reserved																										SSNW		SSHW			
Access																											RW		RW			

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5:4	SSNW	QSSN 有效时间延长设置	QSPI 总线访问后 QSSN 有效时间延长功能选择： 0b00: 不延长 0b01: 延长 32 个 QSCK 周期 0b10: 延长 128 个 QSCK 周期 0b11: 无限延长  说明 直接模式下该位设定无效。	RW
3:0	SSHW	QSSN 最小无效时间设置	QSSN 信号最小无效时间选择： 0b0000: 1 个 QSCK 周期 0b0001: 2 个 QSCK 周期 0b1110: 15 个 QSCK 周期 0b1111: 16 个 QSCK 周期  说明 硬件实际使用值为编程值加 1。	RW

32.5.4 QSPI 格式控制寄存器 (QSPI_FCR)

Offset	Bit Position																																			
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00008033																																			
Name	Reserved																DUTY	Reserved				DMCYCN				Reserved	WPOL	SSNLD		SSNHD		Reserved	4BIC		AWSL	
Access																	RW					RW					RW	RW		RW			RW		RW	

32.5.5 QSPI 状态寄存器 (QSPI_SR)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008000																															
Name	Reserved																PFAN	PFFUL	Reserved	PFNUM				RAER	XIPF	Reserved				BUSY		
Access																	RO	RO		RO				RO	RO					RO		

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	PFAN	预读取动作状态	0b0: 预读取处于动作状态 0b1: 预读取处于停止状态	RO
14	PFFUL	预读取缓冲区状态	0b0: 预读取缓冲区有剩余空间 0b1: 预读取缓冲区数据已满	RO
13	Reserved	保留	-	-
12:8	PFNUM	预读取缓冲区已存储数据字节数	0b00000: 0 个字节 0b00001: 1 个字节 0b10001: 17 个字节 0b10010: 18 个字节 - 其他: Reserved	RO
7	RAER	ROM 访问错误标志	直接通信模式下发生 ROM 访问的错误标志位: 0b0: 未检测到发生 ROM 访问 0b1: 检测到发生 ROM 访问 清零条件: - QSPI_CLR 的 RAERCLR 位置 1	RO
6	XIPF	XIP 模式标志	0b0: 非 XIP 模式 0b1: XIP 模式	RO
5:1	Reserved	保留	-	-
0	BUSY	总线忙标志	直接通信模式下 QSPI 总线工作状态标志位: 0b0: 总线空闲, 无串行传输进程 0b1: 总线忙, 串行传输进程进行中	RO

32.5.6 QSPI 直接通信指令寄存器 (QSPI_DCOM)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																				DCOMPRSL		DCOM									
Access																					RW		RW									

位/位域	标记	位名	功能描述	读写
31:10	Reserved	保留	-	-
9:8	DCOMPRSL	直接通信模式下 SPI 协议选择	0b00: 扩展式 SPI 协议 0b01: 二线式 SPI 协议 0b10: 四线式 SPI 协议 0b11: 禁止设置	RW
7:0	DCOM	直接通信模式指令	直接通信模式时的接口，通过 QSPI 总线进行直接通信。对该接口的读写访问会被转化成一个相应的 QSPI 总线周期。该接口只在直接通信模式下有效，ROM 访问模式下该接口禁止访问。	RW

32.5.7 QSPI 指令代码寄存器 (QSPI_CCMD)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								RW R/C							
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	RIC	替换指令代码	用于替代默认指令的串行闪存指令代码	RW

32.5.8 QSPI XIP 模式代码寄存器 (QSPI_XCMD)

Offset	Bit Position																															
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000000FF																															
Name	Reserved																								XIPMC							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	XIPMC	XIP 模式代码	用于串行闪存的模式代码。(设置 XIP 模式)	RW

32.5.9 QSPI 标志清除寄存器 (QSPI_CLR)

Offset	Bit Position																															
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																								RAERCLR		Reserved					
Access																									WO							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	RAERCLR	RAER 标志清除	写 1 将 QSPI_SR 的 RAER 位清零, 写 0 无效	WO
6:0	Reserved	保留	-	-

32.5.10 QSPI 外部扩展地址寄存器 (QSPI_EXAR)

Offset	Bit Position																															
0x0804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RW EXADR						Reserved																									
Access																																

位/位域	标记	位名	功能描述	读写
31:26	EXADR	外部扩展地址代码	QSPI 外部地址高 6 位设置, 配合 QSPI 的 ROM 访问窗口地址最大可访问 64MB×64 块的外部 ROM 空间	RW
25:0	Reserved	保留	-	-

32.6 注意事项

32.6.1 QSPI 寄存器的设置

如果在 QSPI 通信过程中动态地对 QSPI 控制寄存器进行设置或更改，可能会导致寄存器还没有完全设置完成就开始新的通信，因此不建议在通信期间动态修改 QSPI 控制寄存器。

32.6.2 模块停止位的设置

QSPI 在系统复位后处于模块停止状态。需要 QSPI 模块工作时，请先将功能时钟控制寄存器 PWC_FCG 中的 QSPI 模块控制位清零。具体描述请参见《电源控制 (PWC)》章节“动作模式与低功耗模式”。

33 集成电路内置音频总线模块 (I2S)

33.1 简介

I2S (Inter_IC Sound Bus) ， 集成电路内置音频总线，该总线专责于音频设备之间的数据传输。

33.2 主要特性

表 33-1 I2S 主要特性

功能	主要特性
通信方式	● 支持全双工和半双工通信 ● 支持主模式或从模式操作
数据格式	● 可选通道长度：16/32 位 ● 可选传送数据长度：16/24/32 位 ● 数据移位顺序：MSB 开始
波特率	● 8 位可编程线性预分频器，可实现精确的音频采样频率 ● 支持采样频率 192kHz，96kHz，48kHz，44.1kHz，32kHz，22.05kHz，16kHz，8kHz ● 可输出驱动时钟以驱动外部音频元件，比率固定为 256*Fs（Fs 为音频采样频率）
支持 I2S 协议	● I2S Philips 标准 ● MSB 对齐标准 ● LSB 对齐标准 ● PCM 标准
数据缓冲	● 带有 4 字深，32 位宽的输入输出 FIFO 缓冲区域
时钟源	● 可使用内部 I2SCLK（PLLAR/PLLAQ/PLLAP/PLLHR/PLLHQ/PLLHP） ● 可由 I2S_EXCK 引脚上的外部时钟提供
中断	● 发送缓冲区有效空间达到报警阈值时产生中断 ● 接收缓冲区有效空间达到报警阈值时产生中断 ● 接收数据区域已满仍有写入数据请求，接收上溢 ● 发送数据区域已空仍有发送请求，发送下溢 ● 发送数据区域已满仍有写入数据请求，发送上溢

33.3 功能说明

33.3.1 系统框图

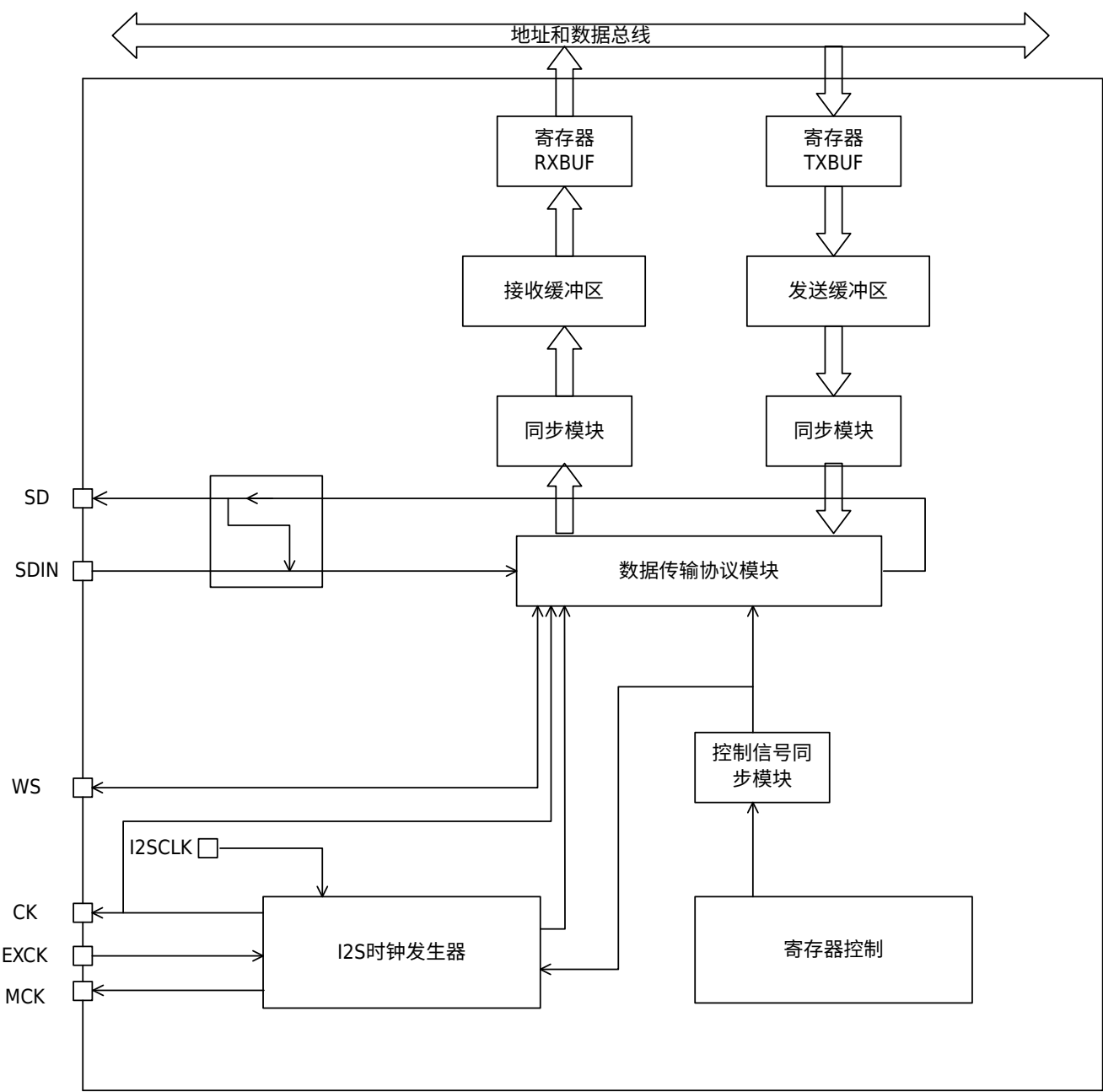


图 33-1 I2S 系统框图

33.3.2 管脚说明

表 33-2 I2S 管脚说明

管脚名	方向	功能描述
I2Sn_CK	输入输出	串行时钟，是主模式下的串行时钟输出以及从模式下的串行时钟输入。
I2Sn_WS	输入输出	字选择，是主模式下的数据控制信号输出以及从模式下的数据控制信号输入。
I2Sn_SD	输入输出	串行数据，用于半双工模式数据输入，或半/全双工模式数据输出。
I2Sn_SDIN	输入	全双工音频数据输入，I2S 全双工模式时用于串行数据输入的引脚。

管脚名	方向	功能描述
I2Sn_EXCK	输入	外部时钟源，是主模式下时钟发生器选择外部时钟作为分频时钟源。
I2Sn_MCK	输出	驱动时钟，当 I2S 配置为主模式（并且 MCKOE 位置 1）时，使用驱动时钟（单独映射）输出此附加时钟，该时钟输出频率 $256 \times F_s$ ，其中 F_s 为音频信号采样频率。I2S 在主模式下使用自身的时钟发生器生成通信时钟，此时钟发生器也是驱动时钟输出的源。

说明

n 取值为 1~4。

33.3.3 通信方式

支持半双工和全双工两种通信方式，可通过 I2S 控制寄存器 (I2S_CTRL) 的 DUPLEX 位进行选择。

- 当 I2S_CTRL.DUPLEX 位为 0 时，I2S 运行在半双工通信方式，只发送时使用 I2Sn_SD 引脚作为输出数据引脚，只接收时使用 I2Sn_SD 引脚作为输入数据引脚。
- 当 I2S_CTRL.DUPLEX 位为 1 时，I2S 运行在全双工通信方式，此时使用 I2Sn_SDIN 引脚作为输入数据引脚，I2Sn_SD 引脚作为输出数据引脚，以实现全双工通信。

33.3.4 支持的音频协议

33.3.4.1 音频协议概述

有四种数据和帧格式组合，可采用下列格式发送数据：

- 将 16 位数据封装在 16 位帧中
- 将 16 位数据封装在 32 位帧中
- 将 24 位数据封装在 32 位帧中
- 将 32 位数据封装在 32 位帧中

当使用 32 位数据包中的 16 位数据时，前 16 位 (MSB) 为有效位，16 位 LSB 被强制清零，无需任何软件操作（只需一个读/写操作）。

当使用 32 位数据包中的 24 位数据时，前 24 位 (MSB) 为有效位，8 位 LSB 被强制清零，无需任何软件操作（只需一个读/写操作）。

对于所有数据格式和通信标准而言，始终会先发送最高有效位 (MSB 优先)。

I2S 接口支持四种音频协议，可使用 I2S_CFGR 寄存器中的 I2SSTD[1:0]和 PCMSYNC 位对其进行配置。

33.3.4.2 I2S Philips 标准

使用 WS 信号来指示当前正在发送的数据所属的通道。该信号从当前通道数据的第一个位 (MSB) 之前的一个时钟开始有效。

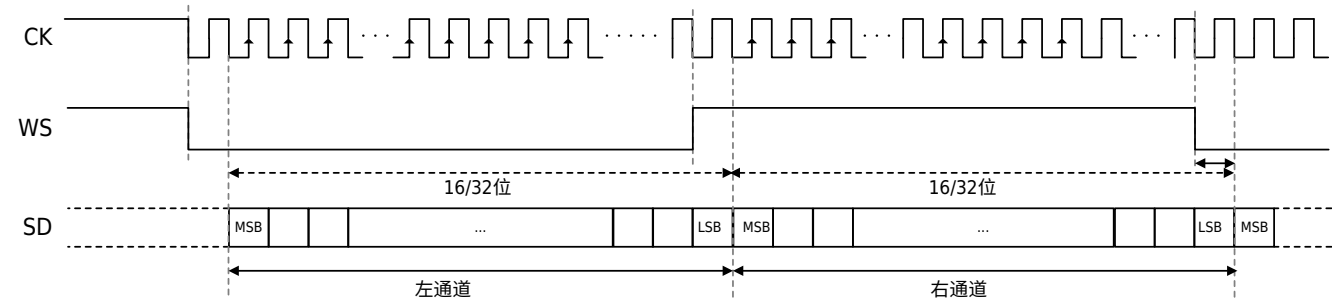


图 33-2 I2S Philips 协议波形 (16/32 位全精度)

16 位装载在 16 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x3344；接收模式下，SD 输入串行数据 0xEEDD，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

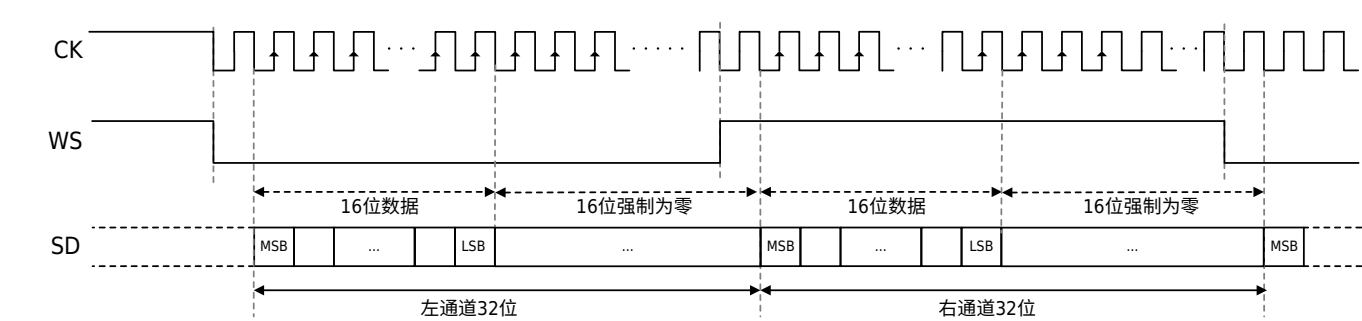


图 33-3 I2S Philips 协议波形（16 位数据封装在 32 位帧中）

16 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x33440000；接收模式下，SD 输入串行数据 0xEEDDXXXX，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

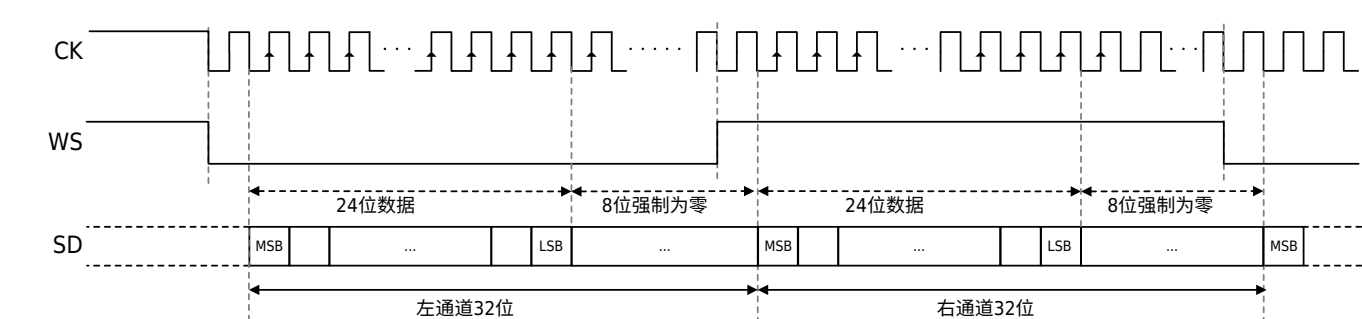


图 33-4 I2S Philips 协议波形（24 位数据封装在 32 位帧中）

24 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFF223344，SD 输出串行数据 0x22334400；接收模式下，SD 输入串行数据 0xEEDD11XX，I2S_RXBUF 寄存器读出数据 0x00EEDD11。

33.3.4.3 MSB 对齐标准

此标准同时生成 WS 信号和第一个数据位（即 MSBit）。

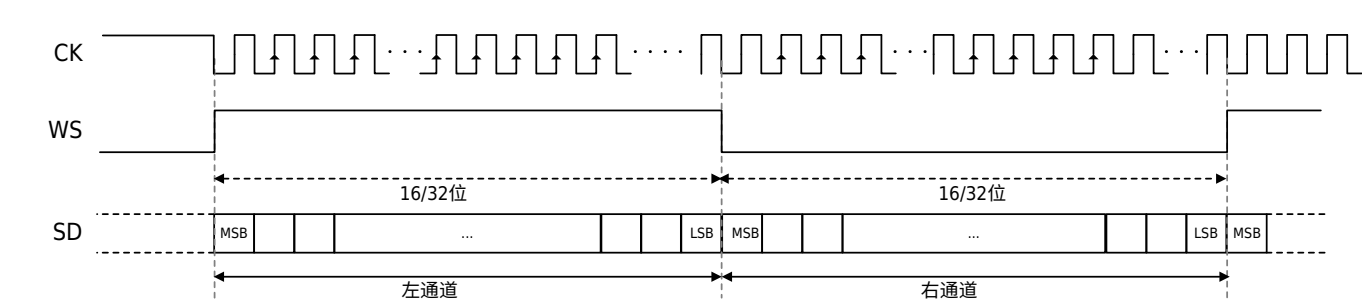


图 33-5 I2S MSB 协议波形（16/32 位全精度）

发送方在时钟信号的下降沿改变数据；接收方在上升沿读取数据。
16 位装载在 16 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x3344；接收模式下，SD 输入串行数据 0xEEDD，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

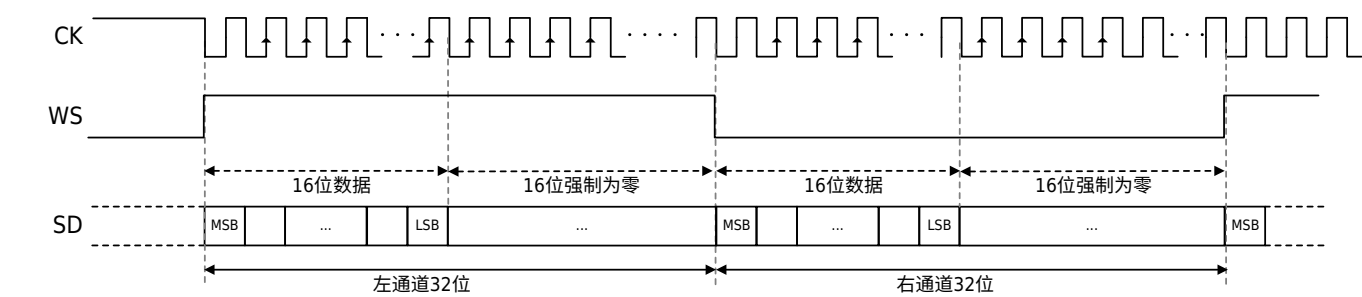


图 33-6 I2S MSB 协议波形 (16 位数据封装在 32 位帧中)

16 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x33440000；接收模式下，SD 输入串行数据 0xEEDDXXXX，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

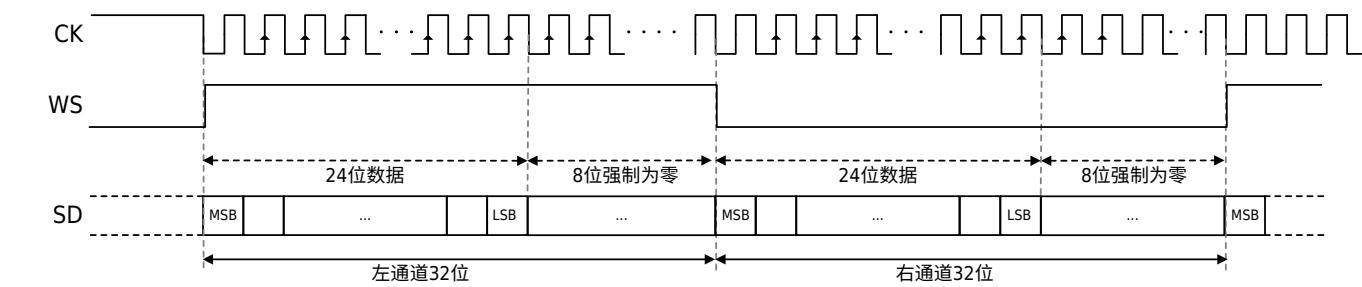


图 33-7 I2S MSB 协议波形 (24 位数据封装在 32 位帧中)

24 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFF223344，SD 输出串行数据 0x22334400；接收模式下，SD 输入串行数据 0xEEDD11XX，I2S_RXBUF 寄存器读出数据 0x00EEDD11。

33.3.4.4 LSB 对齐标准

该标准与 MSB 对齐标准类似（对于 16 位和 32 位全精度帧格式，没有任何不同）。

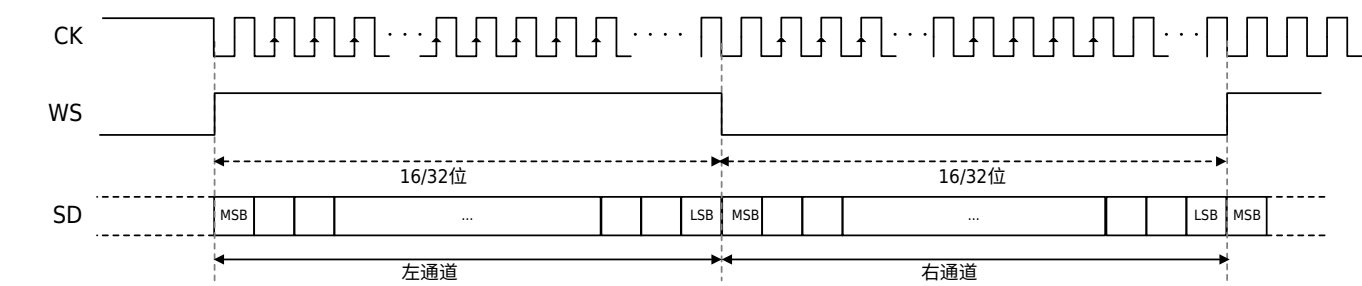


图 33-8 I2S LSB 协议波形 (16/32 位全精度)

16 位装载在 16 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x3344；接收模式下，SD 输入串行数据 0xEEDD，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

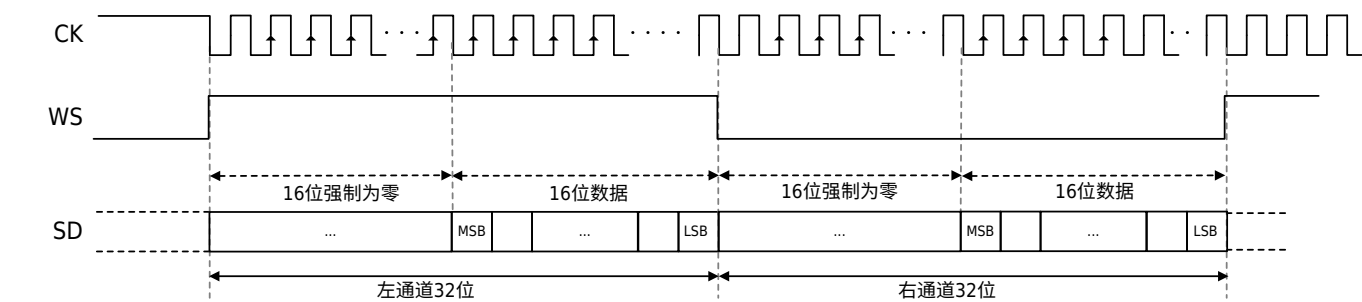


图 33-9 I2S LSB 协议波形 (16 位数据封装在 32 位帧中)

16 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x00003344；接收模式下，SD 输入串行数据 0xFFFF1122，I2S_RXBUF 寄存器读出数据 0x00001122。

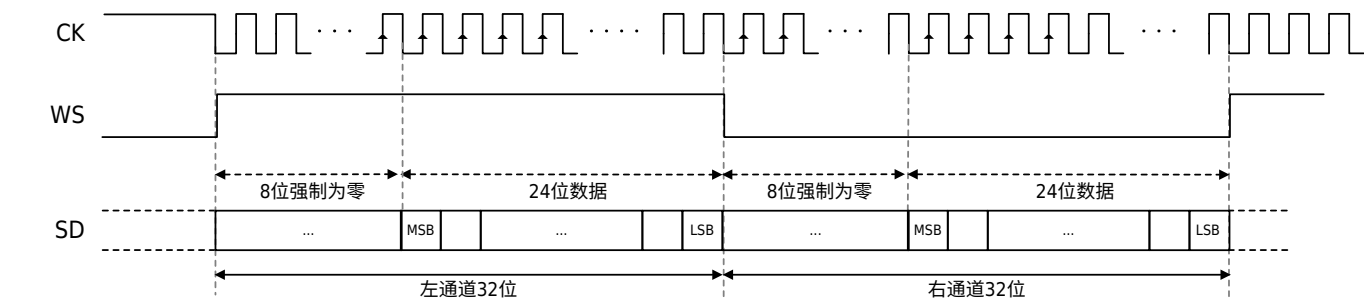


图 33-10 I2S LSB 协议波形（24 位数据封装在 32 位帧中）

24 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFF223344，SD 输出串行数据 0x00223344；接收模式下，SD 输入串行数据 0xFFdd1122，I2S_RXBUF 寄存器读出数据 0x00dd1122。

33.3.4.5 PCM 标准

对于 PCM 标准，无需使用通道信息。有两种 PCM 模式（短帧和长帧），并且可使用 I2S_CFGR 中的 PCMSYNC 位来配置。

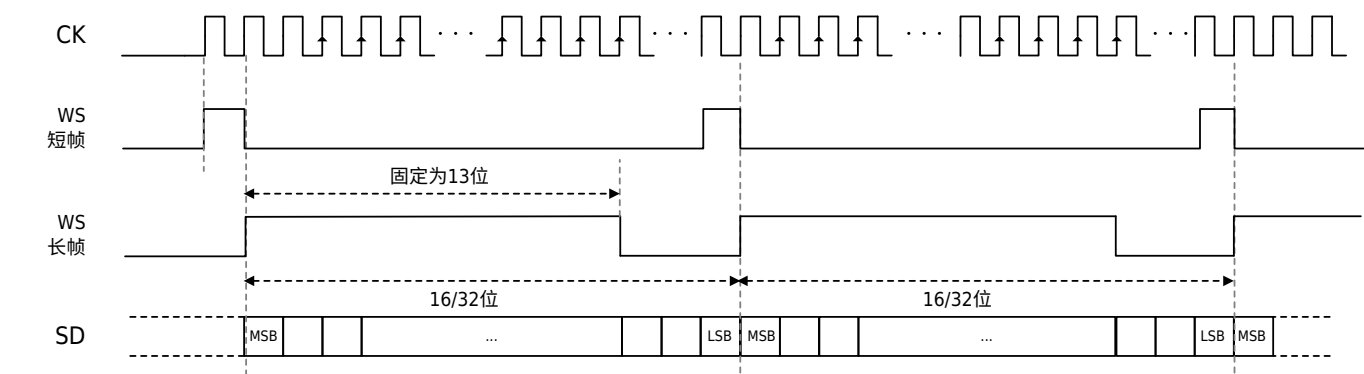


图 33-11 I2S PCM 协议波形（16/32 位全精度）

对于长帧同步，在主模式下会将 WS 信号持续 13 个周期。
对于短帧同步，WS 同步信号的持续时间仅为一个周期。
16 位装载在 16 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x3344；接收模式下，SD 输入串行数据 0xEEDD，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

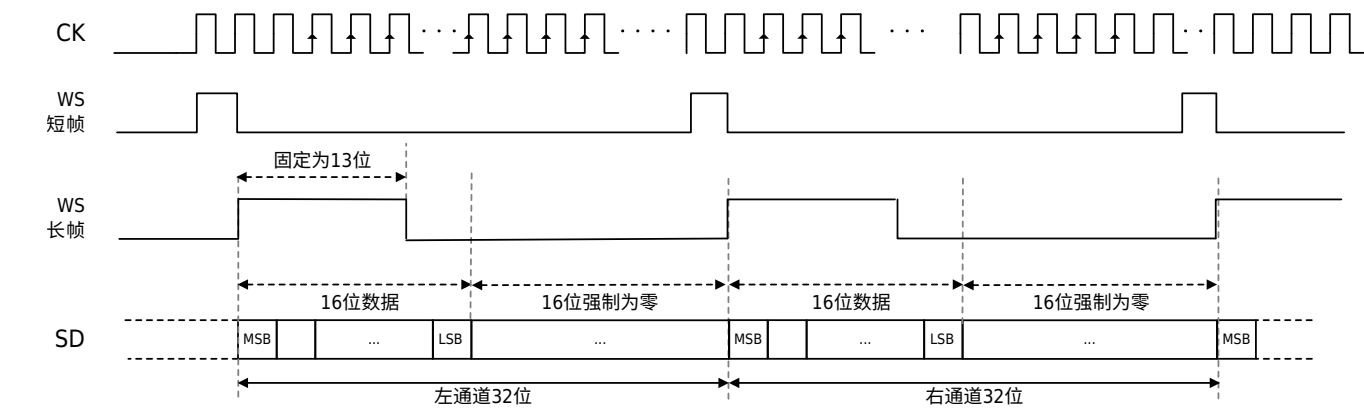


图 33-12 I2S PCM 协议波形（16 位数据封装在 32 位帧中）

16 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0xFFFF3344，SD 输出串行数据 0x00003344；接收模式下，SD 输入串行数据 0xEEDDXXXX，I2S_RXBUF 寄存器读出数据 0x0000EEDD。

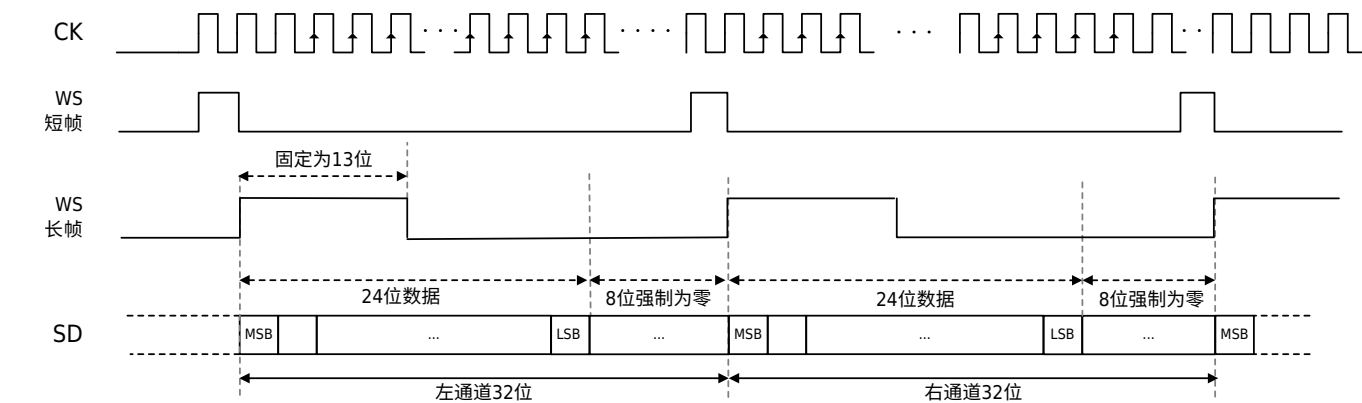


图 33-13 I2S PCM 协议波形（24 位数据封装在 32 位帧中）

24 位装载在 32 位帧，发送模式下，写入到 I2S_TXBUF 寄存器 0XX223344，SD 输出串行数据 0x00223344；接收模式下，SD 输入串行数据 0xEEDD11XX，I2S_RXBUF 寄存器读出数据 0x00EEDD11。

**说明**

对于两种模式（主/从模式）和两种同步（短/长同步），即使在从模式下，也需要指定两组连续数据（以及两个同步信号）之间位的个数（I2S_CFGR 寄存器中的 DATLEN 位和 CHLEN 位）。

33.3.5 时钟发生器

I2S 比特率用来确定 I2S 数据线上的数据流和 I2S 时钟信号频率。
I2S 比特率 = 每个通道的位数×通道数×音频采样频率
对于 16 位双通道音频，I2S 比特率的计算公式如下：I2S 比特率 = 16×2×Fs。如果数据包为 32 位宽，则 I2S 比特率 = 32×2×Fs。

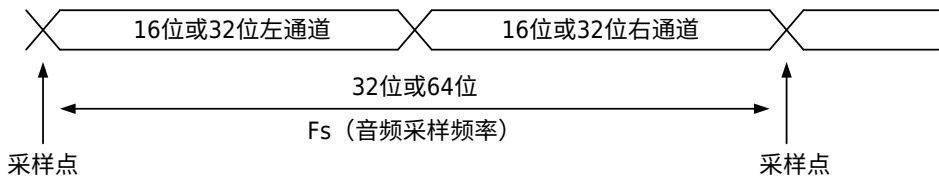


图 33-14 音频采样频率定义

配置主模式时，需要正确地对线性分频器进行设置，以便采用所需的音频频率进行通信。

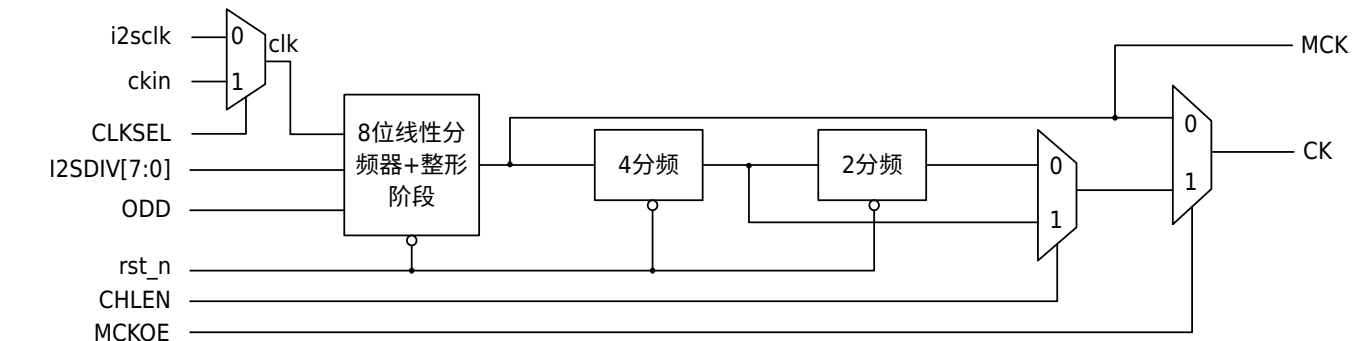


图 33-15 时钟发生器架构

clk 时钟源可为 I2SCLK 输出或外部时钟。

音频采样频率可能是 192kHz, 96kHz 或 48kHz 等。为达到所需频率, 需要根据以下公式对现行分频器进行编程:

● 输出驱动时钟 (mckoe 置 1) 时:

$F_s = \text{clk} / [(16 \times 2) \times ((2 \times \text{I2SDIV} + \text{ODD}) \times 8)]$ (通道帧宽度为 16 位时)

$F_s = \text{clk} / [(32 \times 2) \times ((2 \times \text{I2SDIV} + \text{ODD}) \times 4)]$ (通道帧宽度为 32 位时)

● 关闭驱动时钟 (mckoe 清零) 时:

$F_s = \text{clk} / [(16 \times 2) \times (2 \times \text{I2SDIV} + \text{ODD})]$ (通道帧宽度为 16 位时)

$F_s = \text{clk} / [(32 \times 2) \times (2 \times \text{I2SDIV} + \text{ODD})]$ (通道帧宽度为 32 位时)

下表提供了针对不同时钟配置的示例精度值。还有其他配置可达到更好的时钟精度。

表 33-3 音频频率精度 (针对 VCO 输入频率=1MHz)

驱动时钟	目标 Fs(Hz)	数据格式	倍频系数	输出分频比	I2SDIV	ODD	实时 Fs(Hz)	误差
输出关闭	8000	16 位	288	3	187	1	8000	0.0000%
		32 位	256	4	62	1	8000	0.0000%
	16000	16 位	256	4	62	1	16000	0.0000%
		32 位	256	2	62	1	16000	0.0000%
	32000	16 位	256	2	62	1	32000	0.0000%
		32 位	256	5	12	1	32000	0.0000%
	48000	16 位	384	10	12	1	48000	0.0000%
		32 位	384	5	12	1	48000	0.0000%
	96000	16 位	384	5	12	1	96000	0.0000%
		32 位	424	3	11	1	96014.49219	0.0151%
	22050	16 位	290	3	68	1	22049.87695	0.0006%
		32 位	302	2	53	1	22050.23438	0.0011%
	44100	16 位	302	2	53	1	44100.46875	0.0011%
		32 位	429	4	19	0	44099.50781	0.0011%
	192000	16 位	424	3	11	1	192028.9844	0.0151%
		32 位	258	3	3	1	191964.2813	0.0186%
输出使能	8000	-	256	5	12	1	8000	0.0000%
	16000	-	426	4	13	0	16000.60059	0.0038%
	32000	-	426	4	6	1	32001.20117	0.0038%
	48000	-	258	3	3	1	47991.07031	0.0186%
	96000	-	344	2	3	1	95982.14063	0.0186%
	22050	-	429	4	9	1	22049.75291	0.0011%
	44100	-	271	2	6	0	44108.07422	0.0183%

 说明

倍频系数可为 PLLAN/PLLHN, 倍频系数为 PLLAN 时输出分频比可为 PLLAP/PLLAQ/PLLAR, 倍频系数为 PLLHN 时输出分频比可为 PLLHP/PLLHQ/PLLHR。具体设置请参见《时钟控制器 (CMU) 》的 PLLA 配置寄存器 (CMU_PLLACFGR) 和 PLLH 配置寄存器 (CMU_PLLHCFGR)。

33.3.6 主从模式

33.3.6.1 I2S 主模式

I2S 可如下配置:

- 发送主器件或接收主器件 (使用 I2S 的半双工模式)。
- 同时收发的主器件 (使用 I2S 的全双工模式)。

I2S 工作在主模式，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 I2S_CTRL 的 MCKOE 位来选择输出或者不输出驱动时钟 (MCK)。

操作步骤

- 步骤 1. 设定 I2S 所需要使用的管脚。
- 步骤 2. 通过 I2S_CTRL.CLKSEL, I2S_CTRL.I2SPLLSEL 位，选择时钟源。
- 步骤 3. 设置 I2S_PR.I2SDIV[7:0]位和 I2S_CTRL.ODD 位，以定义串行数据波特率，从而达到适当的音频采样频率。
- 步骤 4. 设置 I2S 配置寄存器 (I2S_CFGR)，通过 I2SSTD[1:0]和 PCMSYNC 位选择 I2S 标准，通过 DATLEN[1:0]位选择数据长度并通过配置 CHLEN 位选择每个通道的位数。
- 步骤 5. 如需要使用中断，请设置系统的中断寄存器。
- 步骤 6. 如需使用 DMA，请设置 DMA 的相关寄存器。
- 步骤 7. 设定 I2S 控制寄存器 (I2S_CTRL)，包括有工作模式设定，通信方式设定，时钟输出许可设定，数据输出许可设定，FIFO 复位设定，发送/接收缓冲阈值设定等，工作模式 WMS 位选择 I2S 主模式。
- 步骤 8. 设置中断许可位。
- 步骤 9. 设置 I2S_CTRL.TXE 和 I2S_CTRL.RXE，动作开始。



说明

使用 TXIRQOUT 中断写通信数据到 TXBUF 中时，若每次中断写入两个数据时，在中断开始后首先关掉发送中断使能标志位 TXIE，写完数据后，再打开 TXIE。或者每次中断只写入一个数据。

发送序列

I2S_CTRL 寄存器中的 TXE 位置 1，允许发送。将数据写入发送缓冲区后，发送序列随即开始。一个完整帧表示先进行左通道数据发送再进行右通道数据发送。不存在仅发送左通道的部分帧。首位发送期间，数据并行加载到移位寄存器中，然后以串行方式移位并输出到 SD 引脚 (MSB 在前)。有关各种 I2S 标准模式中的写操作的更多详细信息，请参见[支持的音频协议](#)。

接收序列

该工作模式与发送模式基本相同，只有在 I2S_CTRL 寄存器的发送接收设置上不同，将 RXE 位置 1，允许接收。有关各种 I2S 标准模式中的读操作的更多详细信息，请参见[支持的音频协议](#)。如果在先前收到的数据尚未读取时又接收到新数据，将产生溢出错误并将 I2S_ER.RXERR 标志置 1。如果 I2S_CTRL.EIE 位置 1，将产生中断以指示该错误。

33.3.6.2 I2S 从模式

I2S 可如下配置：

- 发送从器件或接收从器件（使用 I2S 的半双工模式）。
- 同时收发的从器件（使用 I2S 的全双工模式）。

此工作模式所遵循的规则与 I2S 主模式基本相同。在从模式下，I2S 接口不产生时钟。时钟和 WS 信号从 I2S 接口所连接的外部主器件输入。这样，用户便不需要配置时钟。

操作步骤

- 步骤 1. 设定 I2S 所需要使用的管脚。
- 步骤 2. 设置 I2S 配置寄存器 (I2S_CFGR)，通过 I2SSTD[1:0]和 PCMSYNC 位选择 I2S 标准，通过 DATLEN[1:0]位选择数据长度并通过配置 CHLEN 位选择每个通道的位数。
- 步骤 3. 如需要使用中断，请设置系统的中断寄存器。
- 步骤 4. 如需使用 DMA，请设置 DMA 的相关寄存器。
- 步骤 5. 若要发送数据，应先向 I2S_TXBUF 中预先写入 1~4 个待发送数据。

- 步骤 6. 设定 I2S 控制寄存器 (I2S_CTRL)，包括有工作模式设定，通信方式设定，时钟输出许可设定，数据输出许可设定，FIFO 复位设定，发送/接收缓冲阈值设定等，工作模式 WMS 位选择 I2S 从模式。
- 步骤 7. 设置中断许可位。
- 步骤 8. 设置 I2S_CTRL.TXE 和 I2S_CTRL.RXE，动作开始。

发送序列

当 WMS 置 1，TXE 置 1，外部主器件发送时钟并且通过 WS 信号请求传输数据时，发送序列开始。通信开始时，数据从发送缓冲区传输到移位寄存器。首位发送期间，数据从内部总线并行加载到移位寄存器中，然后以串行方式移位并输出到 SD 引脚（MSB 在前）。每次发送缓冲 FIFO 空间大于设置的阈值时，如果 I2S_CTRL.TXIE 位置 1，将产生中断。有关各种 I2S 标准模式中的写操作的更多详细信息，请参见[支持的音频协议](#)。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送数据写入 TX FIFO。若在数据尚未写入 TX FIFO 时下一个数据通信的首个时钟边沿到来，将产生发送下溢，I2S_ER.TXERR 标志将置 1 并可能产生中断。如果 I2S_CTRL.EIE 位置 1，则当 I2S_ER.TXERR 标志变为 1 时，将产生中断。

接收序列

该工作模式与发送模式基本相同，只有在 I2S_CTRL 寄存器的发送接收设置上不同，将 RXE 位置 1，允许接收。有关各种 I2S 标准模式中的读操作的更多详细信息，请参见[支持的音频协议](#)。如果在先前收到的数据尚未读取时又接收到新数据，将产生接收溢出错误并将 RXERR 标志置 1。如果 I2S_CTRL.EIE 位置 1，将产生中断以指示该错误。

33.3.7 中断

I2S 的中断源有发送缓冲区有效空间大于报警阈值、接收缓冲区有效空间小于报警阈值、接收上溢、发送下溢和发送上溢。发送下溢和发送上溢被集成为 I2S 发送错误中断 TXERR，所以需要通​​过标志判断实际发生的中断源。I2S 中断源的具体说明如下表所示。一旦中断条件成立，就产生相应的中断请求。

表 33-4 I2S 中断请求

中断事件	事件标志	使能控制位
发送缓冲区有效空间大于报警阈值	TXBA	TXIE
接收缓冲区有效空间小于报警阈值	RXBA	RXIE
接收数据区域已满仍有写入数据请求，接收上溢	RXERR	EIE
发送数据区域已空仍有发送请求，发送下溢	TXERR	EIE
发送数据区域已满仍有写入数据请求，发送上溢	TXERR	EIE

用户可将上述事件触发源对应的向量写入不同的触发对象寄存器实现各种事件触发功能。上述事件触发源对应的向量请参见《中断控制器 (INTC) 》的“中断事件表”。

33.4 寄存器

33.4.1 寄存器总表

- I2S1 基地址：0x4001E000
- I2S2 基地址：0x4001E400
- I2S3 基地址：0x40022000
- I2S4 基地址：0x40022400

表 33-5 集成电路内置音频总线模块 (I2S) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x000	I2S_CTRL	I2S 控制寄存器	32
0x004	I2S_SR	I2S 状态寄存器	32
0x008	I2S_ER	I2S 错误状态寄存器	32
0x00C	I2S_CFGR	I2S 配置寄存器	32
0x010	I2S_TXBUF	I2S 发送缓冲 FIFO 数据寄存器	32
0x014	I2S_RXBUF	I2S 接收缓冲 FIFO 数据寄存器	32
0x018	I2S_PR	I2S 预分频寄存器	32

 说明

只支持 32 位写寄存器。

33.4.2 I2S 控制寄存器 (I2S_CTRL)

Offset	Bit Position																																																			
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
Reset	0x00004400																																																			
Name	Reserved		SYNCSEL		SRST		Reserved		CLKSEL		DUPLEX		CKOE		LRCKOE		SDOE		I2SPLLSEL		Reserved		FIFOR		Reserved		RXBIRQWL				Reserved		TXBIRQWL				MCKOE		ODD		WMS		EIE		RXIE		RXE		TXIE		TXE	
Access			RW		RW				RW		RW		RW		RW		RW		RW				RW				RW						RW				RW		RW		RW		RW		RW		RW		RW			

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28	SYNCSEL	4 路同步控制	0b0: 关闭 4 路同步控制 0b1: 打开 4 路同步控制  说明 仅单元 1 支持 SYNCSEL 功能位，其他单元的该位为保留位，请保持初始值 0。	RW
27	SRST	软件复位	0b0: 解除复位 0b1: 软件复位	RW
26:24	Reserved	保留	-	-
23	CLKSEL	时钟源选择	0b0: 选择 I2SPLL 0b1: 选择外部时钟	RW
22	DUPLEX	通信方式选择	0b0: 半双工 0b1: 全双工	RW
21	CKOE	通讯时钟输出许可	0b0: 输出禁止 0b1: 输出许可	RW
20	LRCKOE	声道时钟输出许可	0b0: 输出禁止 0b1: 输出许可	RW
19	SDOE	数据输出许可	0b0: 输出禁止 0b1: 输出许可	RW
18	I2SPLLSEL	I2SPLL 输入选择	0b0: 输入禁止 0b1: 输入许可	RW
17	Reserved	保留	-	-
16	FIFOR	FIFO 复位	0b0: 解除复位 0b1: 软件复位	RW
15	Reserved	保留	-	-
14:12	RXBIRQWL	接收缓冲中断请求水平	当可用空间少于设定值时中断请求触发： 只能设置为 0x0~0x4 (FIFO 空间为 4)	RW
11	Reserved	保留	-	-
10:8	TXBIRQWL	发送缓冲中断请求水平	当可用空间多于设定值时中断请求触发： 只能设置为 0x0~0x4 (FIFO 空间为 4)	RW

位/位域	标记	位名	功能描述	读写
7	MCKOE	驱动时钟输出使能	● 0b0: 禁止驱动时钟输出 ● 0b1: 使能驱动时钟输出 说明 只有 I2S 主模式支持此功能。	RW
6	ODD	预分频器奇数因子	● 0b0: 实际分频值=I2SDIV*2 ● 0b1: 实际分频值=I2SDIV*2+1 说明 只有 I2S 主模式支持此功能。 若要设置 ODD 为 1, 应先设置 I2S_PR 寄存器, 再设置 I2S_CTRL 寄存器。	RW
5	WMS	I2S 工作模式选择	● 0b0: I2S 主机模式 ● 0b1: I2S 从机模式	RW
4	EIE	通讯错误中断使能	● 0b0: 通讯错误中断无效 ● 0b1: 通讯错误中断有效	RW
3	RXIE	接收中断使能	● 0b0: 接收中断无效 ● 0b1: 接收中断有效	RW
2	RXE	接收使能	● 0b0: 禁止接收 ● 0b1: 允许接收	RW
1	TXIE	发送中断使能	● 0b0: 发送中断无效 ● 0b1: 发送中断有效	RW
0	TXE	发送使能	● 0b0: 禁止发送 ● 0b1: 允许发送	RW



说明

用 FIFOR 复位 FIFO 前需用 TXE 和 RXE 关闭 I2S 模块。

33.4.3 I2S 状态寄存器 (I2S_SR)

Offset	Bit Position															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x00000014															
Name	Reserved												RXID	RXBF	RXBE	TXBF
Access													RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6	RXID	接收数据 ID	● 0b0: 左声道数据 ● 0b1: 右声道数据	RO
5	RXBF	接收缓冲满	● 0b0: 接收缓冲区未满 ● 0b1: 接收缓冲区满	RO
4	RXBE	接收缓冲空	● 0b0: 接收缓冲区非空 ● 0b1: 接收缓冲区空	RO
3	TXBF	发送缓冲满	● 0b0: 发送缓冲区未满 ● 0b1: 发送缓冲区满	RO
2	TXBE	发送缓冲空	● 0b0: 发送缓冲区非空 ● 0b1: 发送缓冲区空	RO
1	RXBA	接收缓冲报警 (与 water level 有关)	● 0b0: 接收缓冲区未报警 ● 0b1: 接收缓冲区报警	RO
0	TXBA	发送缓冲报警 (与 water level 有关)	● 0b0: 发送缓冲区未报警 ● 0b1: 发送缓冲区报警	RO

说明

I2S 内部除了 FIFO 和移位寄存器以外，还有两个缓存单元。FIFO 标志位只表示 FIFO 内的数据状态，不表示 I2S 内部的全部存储数据。

33.4.4 I2S 错误状态寄存器 (I2S_ER)

Offset	Bit Position																															
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														RW	RXERR
Access																															RW	TXERR

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RXERR	接收错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未发生接收错误 ● 0b1：发生接收错误  说明 当发生接收上溢时，RXERR 置 1。	RW
0	TXERR	发送错误标志	写 1 清除标志，写 0 无效 读取： ● 0b0：未发生发送错误 ● 0b1：发生发送错误  说明 当发生发送上溢/下溢时，TXERR 置 1。	RW

33.4.5 I2S 配置寄存器 (I2S_CFGR)

Offset	Bit Position																															
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										PCMSYNC	CHLEN	DATLEN	I2SSTD		
Access																											RW				RW	RW

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	PCMSYNC	PCM 帧同步	0b0: 短帧同步 0b1: 长帧同步  说明 只有在 I2SSTD=0b11 (使用 PCM 标准) 时此位有效。	RW
4	CHLEN	单声道一帧数据长度选择	0b0: 16-bit 0b1: 32-bit	RW
3:2	DATLEN	传送数据长度选择	0b00: 16-bit 0b01: 24-bit 0b10/0b11: 32-bit	RW
1:0	I2SSTD	通讯协议选择	0b00: Philips 协议 0b01: MSB justified 协议 (左对齐) 0b10: LSB justified 协议 (右对齐) 0b11: PCM 协议	RW

33.4.6 I2S 发送缓冲 FIFO 数据寄存器 (I2S_TXBUF)

Offset	Bit Position																															
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TXBUF																															
Access	WO																															

位/位域	标记	位名	功能描述	读写
31:0	TXBUF	发送数据	存储发送数据: 16 位帧时, TXBUF[15:0]存储一帧左通道或一帧右通道发送数据。 32 位帧时, TXBUF[31:0]存储一帧左通道或一帧右通道发送数据。	WO

33.4.7 I2S 接收缓冲 FIFO 数据寄存器 (I2S_RXBUF)

Offset	Bit Position																															
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RXBUF																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	RXBUF	接收数据	存储接收数据： 16 位帧时，RXBUF[15:0]存储一帧左通道或一帧右通道接收数据。 32 位帧时，RXBUF[31:0]存储一帧左通道或一帧右通道接收数据。	RO

33.4.8 I2S 分频寄存器 (I2S_PR)

Offset	Bit Position																															
0x018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																								I2SDIV							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	I2SDIV	分频因子	实际分频值=I2SDIV*2+I2S_CTRL.ODD，详细请参见 时钟发生器 ： 0x0：禁止设置 0x1：禁止设置 0x2：2 分频 0x3：3 分频 0xFE：254 分频 0xFF：255 分频  说明 只有 I2S 主模式支持此功能。	RW

33.5 注意事项

33.5.1 作为主机使用时的注意事项

- 1. 当 I2S 作为主机进行数据只发送动作时，如果 I2S_TXBUF 中所有的数据都已发送完成而没有新的数据写入时，I2S 将在最后一个数据发送完成后暂停动作，此时 I2S 将不再产生通讯时钟，发送错误标志位 TXERR 将被置位成 1。此时用户可选择将 I2S_CTRL.TXE 位写成 0 来关闭 I2S，或者向 I2S_TXBUF 写入新的发送数据继续发送动作。继续发送时 WS 将重新从左声道开始（Philips、MSB/LSB 模式）。如果在发送动作过程中直接将 I2S_CTRL.TXE 位写成 0，则 I2S 将立刻被关闭，当前的数据发送被终止。这种做法会导致从机状态无法把握，在不复位从机的情况下重新开始通讯时造成从机接收数据混乱，因此建议用户在 I2S 处于暂停状态时将 I2S_CTRL.TXE 位写成 0 来关闭 I2S。
- 2. 当 I2S 作为主机进行数据只接收动作时，如果希望暂时停止接收动作，可预先写入两帧 dummy 数据，需要暂停时计数到相应位置将 I2S_CTRL.TXE 置 1，波特率为 8kHz~96kHz 时提前 4 个数据 TXE 置 1，波特率为 192kHz 时提前 5 个数据 TXE 置 1，当这两帧 dummy 数据发送后 I2S 会暂停动作，此时 I2S 将不再产生通讯时钟，发送错误标志位 TXERR 将被置位成 1。此时用户可选择将 I2S_CTRL.TXE 和 I2S_CTRL.RXE 清零来关闭 I2S，或者向 I2S_TXBUF 再写入一帧 dummy 数据重启通讯动作，在暂停后重新开始接收第一个数据时关掉 TXE，重启动作时 WS 将重新从左声道开始（Philips、MSB/LSB 模式）。当通讯时钟重新产生后将 I2S_CTRL.TXE 清零，即可重新回到只接收状态。详细请参照下图。

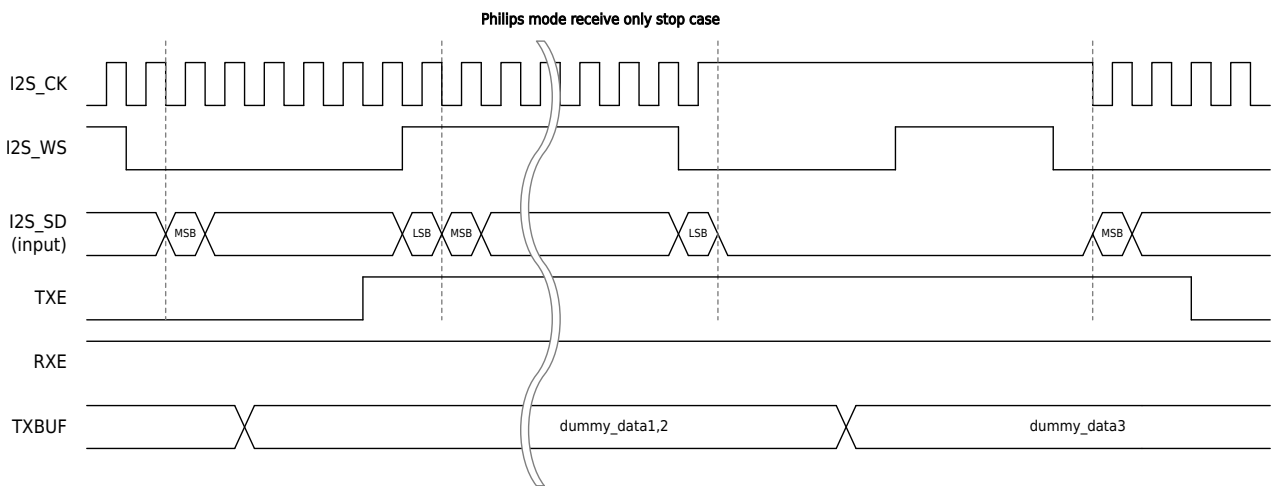


图 33-16 主机只接收暂时停止接收

- 如果在接收动作过程中直接将 I2S_CTRL.RXE 位写成 0，则 I2S 将立刻被关闭，当前的数据接收被终止。这种做法会导致从机状态无法把握，在不复位从机的情况下重新开始通讯时造成从机发送数据混乱，因此建议用户在 I2S 处于暂停状态时将 I2S_CTRL.TXE 和 I2S_CTRL.RXE 位清零来关闭 I2S。
- 3. 当 I2S 作为主机进行全双工动作时，如果 I2S_TXBUF 中所有的数据都已发送完成而没有新的数据写入时，I2S 将在最后一个数据发送完成后暂停动作，此时 I2S 将不再产生通讯时钟，发送错误标志位 TXERR 将被置位成 1。此时用户可选择将 I2S_CTRL.TXE 和 I2S_CTRL.RXE 清零来关闭 I2S，或者向 I2S_TXBUF 写入新的发送数据继续发送和接收动作。继续发送时 WS 将重新从左声道开始（Philips、MSB/LSB 模式）。如果在全双工动作过程中直接将 I2S_CTRL.TXE 和 I2S_CTRL.RXE 位写成 0，则 I2S 将立刻被关闭，当前的数据发送及接收被终止。这种做法会导致从机状态无法把握，在不复位从机的情况下重新开始通讯时造成通讯数据混乱，因此建议用户在 I2S 处于暂停状态时将 I2S_CTRL.TXE 和 I2S_CTRL.RXE 位清零来关闭 I2S。

4. 当 I2S 作为主机进行 PCM 短帧数据发送动作时，当 I2S 因为 I2S_TXBUF 没有新的数据写入动作暂停后，有两种设置方式可以重新开始发送，具体选择哪一种方式需要根据从机的数据接收规格来确定。
- 如果从机每次接收数据都要检测 WS 的状态，则 I2S 暂停后需要先将 I2S_CTRL.TXE 置 0，然后向 I2S_TXBUF 写入新的发送数据后再把 I2S_CTRL.TXE 置 1 来重新开始传送。具体动作如下图所示。

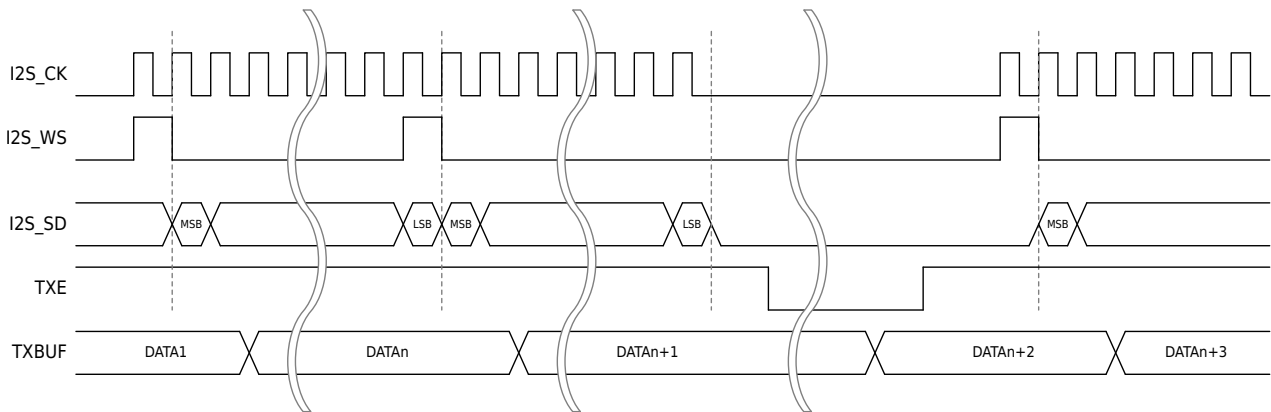


图 33-17 PCM 短帧主机发送暂停后重新发送方式一

- 如果从机只会在接收第一帧数据的时候才检测 WS 状态，则 I2S 暂停后可直接向 I2S_TXBUF 写入新的发送数据来重新开始传送。具体动作如下图所示。

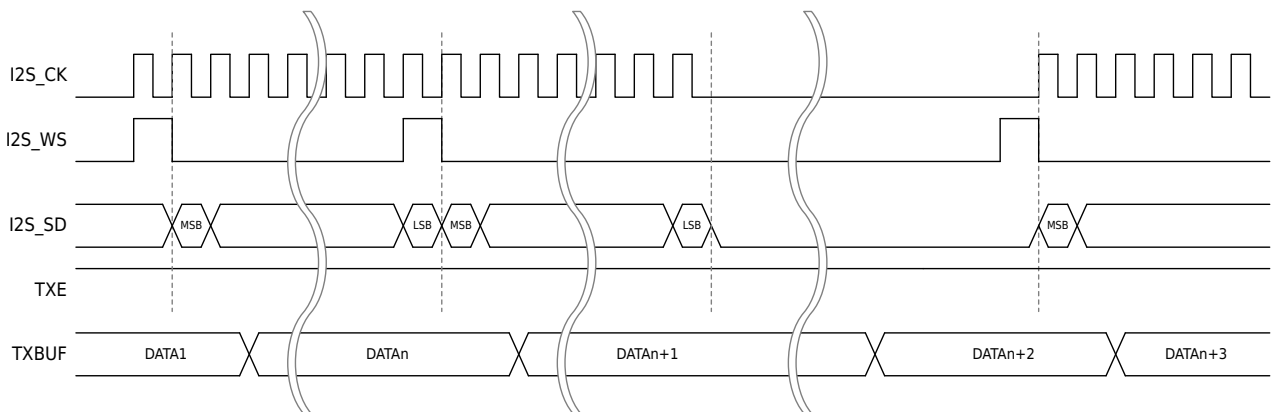


图 33-18 PCM 短帧主机发送暂停后重新发送方式二

33.5.2 作为从机使用时的注意事项

1. 当 I2S 作为从机动作时，需要确保在所有的寄存器配置都完成之后，最后再打开 I2S_CTRL.TXE 或者 I2S_CTRL.RXE 来启动从机动作。
2. 当 I2S 在 Philips、MSB/LSB 模式下启动从机动作时，需要保证启动时 WS 信号处于右声道电平状态，当 I2S 在 PCM 模式下启动从机动作时，需要保证启动时 WS 信号处于低电平状态。
3. 当 I2S 作为从机进行数据接收时，每次接收到的数据将会在下一帧数据接收开始后才能被读取，因此在通讯暂停或终止时，I2S 接收到的最后一帧数据将会在下次通讯开始时才能被读到。
4. 当 I2S 在 Philips、MSB/LSB 模式下作为从机进行数据接收动作时，每帧左声道数据接收时会检验 WS 是否为左声道电平。当 I2S 在 PCM 模式下作为从机进行数据接收时，每帧数据接收前都会检验 WS 是否按标准通讯协议产生了有效电平。
5. 当 I2S 作为从机进行全双工动作时，如果 I2S_TXBUF 中所有的数据都已发送完成而没有新的数据写入时，I2S 将在最后一个数据发送完成后内部收发电路将暂停工作（即也不能接收数据），发送错误标志位 TXERR 将被置位成 1。此时用户可选择使用 I2S_CTRL.TXE 和 I2S_CTRL.RXE 关闭 I2S，并复位 FIFO 及设置 SRST 进行软复位，然后清除错误标志，最后打开 I2S 进行新数据的收发。

33.5.3 抗干扰支持功能

1. 在从模式时，当接收到的 WS 信号指示与所配置的数据通道长度不一致时，按 WS 信号指示开始接收每个数据通道的数据，避免因传输干扰产生的积累误差。
2. 因传输干扰的原因，当 WS 信号小于 7 个数据通道长度时，会出现掉包情况。

33.5.4 数据通道标识功能

1. 在 Philips/MSB/LSB 从模式下，如需知道所接收数据属于左或者右声道，可以先读取 I2S_SR.RXID，得到数据通道标识，然后再读取 I2S_RXBUF 寄存器，得到该通道数据。
2. 每次读取 I2S_RXBUF 寄存器都会同时自动更新 I2S_SR.RXID。

33.5.5 四路同步控制功能

四路同步功能的使用流程如下：

- 步骤 1. 通过四路 I2S 对应的 I2S_CTRL.I2SPLLSEL, I2S_CTRL.TXE 和 I2S_CTRL.RXE 位禁止时钟并关闭 I2S。
- 步骤 2. 使四路 I2S 的配置一致并且 I2S_CTRL.I2SPLLSEL, I2S_CTRL.TXE 和 I2S_CTRL.RXE 保持关闭。
- 步骤 3. 置位 I2S1 的 I2S_CTRL.SYNCSEL。
- 步骤 4. 通过 I2S1 的 I2S_CTRL.I2SPLLSEL 使能四路 I2S 的时钟。
- 步骤 5. 通过 I2S1 的 I2S_CTRL.TXE、I2S_CTRL.RXE 使能四路 I2S 的 TX、RX。
- 步骤 6. 读写四路 I2S 对应的缓冲 FIFO 数据寄存器，进行数据收发。



说明

仅支持通过 I2S_CTRL.CLKSEL 选择内部时钟 (I2SPLL)。

33.5.6 其他注意事项

1. 推荐清 FIFO 流程：
 - 使用 I2S_CTRL.TXE 和 I2S_CTRL.RXE 关闭 I2S；
 - 通过 I2S_CTRL.FIFOR 复位 FIFO；
 - 通过 I2S_CTRL.SRST 进行软复位。
2. 推荐清 ERR 流程：
 - 使用 I2S_CTRL.TXE 和 I2S_CTRL.RXE 关闭 I2S；
 - 通过 I2S_CTRL.FIFOR 复位 FIFO；
 - 通过 I2S_CTRL.SRST 进行软复位；
 - 清除 ERR 标志。
3. 对控制寄存器中各复位 Bit 的说明：
 - I2S_CTRL.SRST 位用来复位工作电路及清空内部移位寄存器，不会影响寄存器配置；
 - I2S_CTRL.FIFOR 位用来复位 FIFO，不会影响其他电路和寄存器。
4. RX/TX 电路内部除了移位寄存器还由两个缓存会根据情况来获取 RX/TX FIFO 中的数据，该缓存用 FIFOR 清空。

34 USB2.0 高速模块 (USBHS)

34.1 简介

USB 高速 (USBHS) 控制器为便携式设备提供了一套 USB 通信解决方案。USBHS 控制器支持主机模式和设备模式，芯片内部集成全速 PHY 且支持 ULPI 接口外接芯片外高速 PHY。主机模式下，USBHS 控制器支持高速 (HS, 480Mbit/s)、全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 收发器，而设备模式下则仅支持高速 (HS, 480Mbit/s) 和全速 (FS, 12Mbit/s) 收发器。USBHS 控制器支持 USB 2.0 协议所定义的所有四种传输方式 (控制传输、批量传输、中断传输和同步传输)。该 USBHS 控制器支持 LPM (Link Power Management) 功能。

芯片内部集成全速 PHY 仅支持全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 传输，通过 ULPI 接口外接片外高速 PHY 则支持高速 (HS, 480Mbit/s)、全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 传输。

遵循协议如下：

- Universal Serial Bus Revision 2.0 Specification
- USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007
- Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007

34.2 主要特性

34.2.1 通用特性

- 支持两个 PHY 接口
 - ▶ 内置片上 USB2.0 全速 PHY (需要设置 USB_SYCTLREG.USBHS_FSPHYE=1 使能)
 - ▶ 连接片外高速 PHY 的 ULPI 接口
- 支持主机模式和设备模式
- 支持 HS/FS SOF 以及低速“Keep-alive”令牌并具有以下功能：
 - ▶ SOF 脉冲管脚输出功能
 - ▶ SOF 脉冲可作为芯片内部事件源去触发 TIMER，DMA 等模块工作
 - ▶ 可配置帧周期
 - ▶ 可配置的帧结束中断
- 模块内嵌 DMA，并可软件配置 AHB 突发传输类型
- 具备省电功能，例如 USB 挂起，停止 RAM 时钟，停止 PHY 域时钟
- 具有采用高级 FIFO 控制的 8KB 专用 RAM
 - ▶ 可以将 RAM 空间划分为不同的 FIFO，以便灵活有效的使用 RAM
 - ▶ 每个 FIFO 可存储多个数据包
 - ▶ 动态分配存储区
 - ▶ FIFO 的大小可配置成为非 2 的幂次方值，以便连续使用存储单元
- 一帧之内可以不需要应用程序干预，以达到最大 USB 带宽
- 可根据 ID 线的电平自动确定主机模式或者设备模式

34.2.2 主机模式特性

- 主机模式支持 USB2.0 高速 (HS, 480Mbit/s)、全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 传输
- 需要通过外部电源芯片生成 VBUS 电压
- 多达 16 主机通道 (管道)：每个通道都可以动态实现重新配置，可支持任何类型的 USB 传输

- 内置硬件调度器，可：
 - ▶ 在周期性硬件队列中存储多达 8 个中断加同步传输请求
 - ▶ 在非周期性硬件队列中存储多达 8 个控制加批量传输请求
- 管理一个共享 RX FIFO、一个周期性 TX FIFO 和一个非周期性 TX FIFO，以有效使用 USB 数据 RAM

34.2.3 设备模式特性

- 从机模式支持 USB2.0 高速 (HS, 480Mbit/s)、全速 (FS, 12Mbit/s) 传输
- 1 个双向控制端点 0
- 15 个 OUT 端点，可以配置为支持批量传输、中断传输或同步传输
- 15 个 IN 端点，可以配置为支持批量传输、中断传输或同步传输
- 包含 16 个发送 FIFOs (每个 IN 端点配备一个发送 FIFO) 和一个接收 FIFO (由所有的 OUT 端点共享)
- 支持远程唤醒功能
- 支持软断开功能
- VBUS PIN 支持 5V 耐压

34.3 功能说明

34.3.1 系统框图

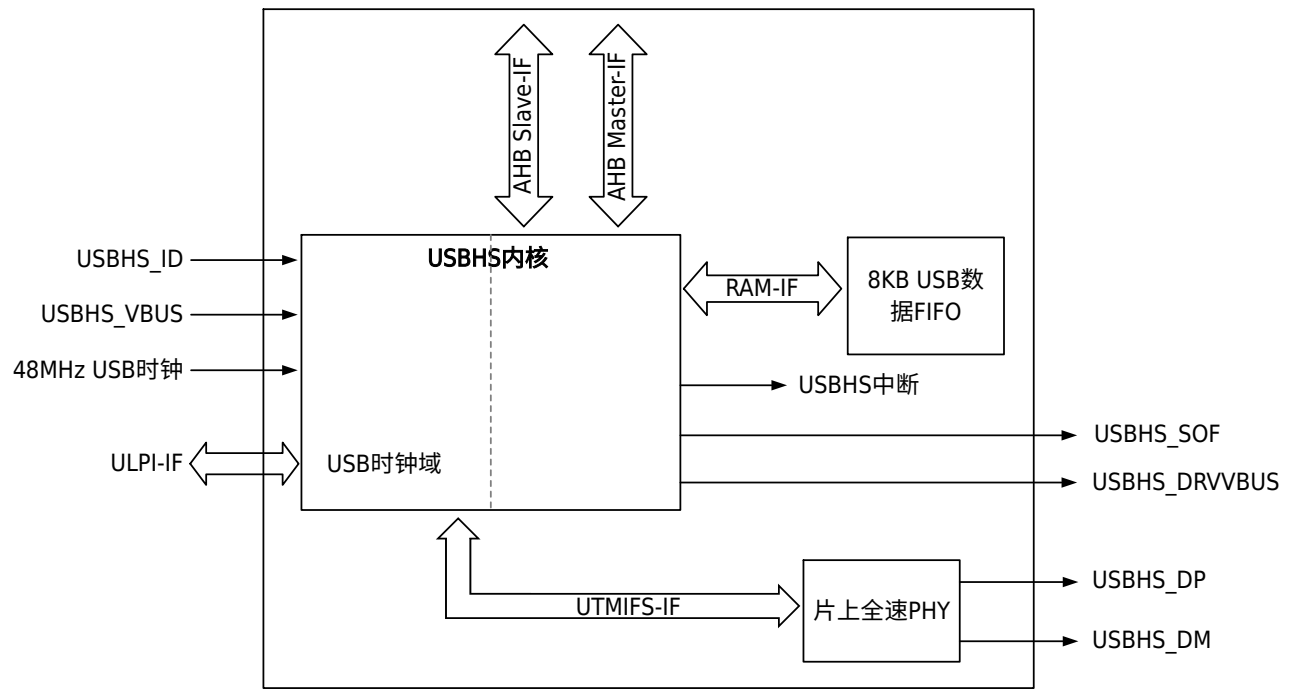


图 34-1 USBHS 系统框图

34.3.2 管脚说明

表 34-1 USBHS 管脚说明

管脚名	方向	适用模式	对应管脚	功能描述	适用 PHY
USBHS_VBUS	输入	设备	PB13	电源端口，5V 耐压	片上全速 PHY
USBHS_DP	输入/输出	主机、设备	PB15	差分数据 D+	片上全速 PHY
USBHS_DM	输入/输出	主机、设备	PB14	差分数据 D-	片上全速 PHY
USBHS_DRVVBUS	输出	主机	PD8	外部电源芯片使能	片上全速 PHY
USBHS_ID	输入	主机	PB12	USB A-B 器件识别	片上全速 PHY
USBHS_SOF	输出	主机、设备	PA4	SOF 输出脉冲	片上全速 PHY ULPI PHY
USBHS_ULPI_CLK	输入	主机、设备	PE12、PA5	ULPI Clock	ULPI PHY
USBHS_ULPI_DIR	输入	主机、设备	PC2、PI11	ULPI Direction	ULPI PHY
USBHS_ULPI_NXT	输入	主机、设备	PC3、PH4	ULPI Next	ULPI PHY
USBHS_ULPI_STP	输出	主机、设备	PC0	ULPI Stop	ULPI PHY
USBHS_ULPI_D0	输入/输出	主机、设备	PE13、PA3	ULPI Data 0	ULPI PHY
USBHS_ULPI_D1	输入/输出	主机、设备	PE14、PB0	ULPI Data 1	ULPI PHY

管脚名	方向	适用模式	对应管脚	功能描述	适用 PHY
USBHS_ULPI_D2	输入/输出	主机、设备	PE15、PB1	ULPI Data 2	ULPI PHY
USBHS_ULPI_D3	输入/输出	主机、设备	PB10	ULPI Data 3	ULPI PHY
USBHS_ULPI_D4	输入/输出	主机、设备	PB11、PD9	ULPI Data 4	ULPI PHY
USBHS_ULPI_D5	输入/输出	主机、设备	PB12	ULPI Data 5	ULPI PHY
USBHS_ULPI_D6	输入/输出	主机、设备	PB13	ULPI Data 6	ULPI PHY
USBHS_ULPI_D7	输入/输出	主机、设备	PB5、PE11	ULPI Data 7	ULPI PHY

由于 USBHS_DP 和 USBHS_DM 管脚和通用 GPIO 复用，在使用 USBHS 片上全速 PHY 时，建议关闭其对应管脚的数字功能，USBHS_DP 和 USBHS_DM 功能为模拟功能，与对应的 PFSR 寄存器设定无关。详细情况请参见《端口控制器 (GPIO) 》章节相关描述。另外 USBHS 功能不使用时，USBHS_DP 和 USBHS_DM 管脚对应的数字功能管脚翻转时，会产生额外的电流消耗。

34.3.3 时钟以及工作模式

USBHS 所使用的时钟有以下三种：

- 48MHz 时钟，该 48MHz 时钟由内部 PLL 电路产生，PLL 时钟源需选择外部高速振荡器，使用 USBHS 模块前，需要在 CMU 模块内配置好 USBHS 时钟。
- 60MHz 时钟，使用 UPLI 接口外接高速 PHY 的输入时钟，仅在外接 PHY 时使用。
- 模块时钟 PCLK1，需要在 CMU 模块内配置好该时钟，该时钟需要大于等于 60MHz。

USBHS 可以作为主机或者设备使用，并且包含一个片上全速 PHY 和一个 ULPI 接口用于外接高速 PHY。上拉和下拉电阻已经集成在片上全速 PHY 的内部，并且 USBHS 可以根据当前模式和连接状态自动选择。USBHS 使用片上全速 PHY 时，VCC 电压范围为 3.0~3.63V。

34.3.4 模式决定

USBHS 决定当前工作模式有以下两种方法：

- 方法 1：根据 USBHS_ID 线的状态自动识别，检测到 USBHS_ID 线为高电平时，模块工作在设备模式，检测到 USBHS_ID 线为低电平时，模块工作在主机状态。
- 方法 2：强制主机/设备模式，通过设定寄存器 USBHS_GUSBCFG 的 FDMOD 或者 FHMOD 位为 1，使模块忽略 USBHS_ID 线的电平而强制工作在设备或者主机模式。

34.3.5 主机功能

功能简介

当 USBHS 工作在主机模式下时，VBUS 为 USB 协议所规定的 5V 电源引脚。内部 PHY 不支持提供 5V 电源，所以需要外接 USB 电源芯片为设备供电。USBHS_DRVVBUS 用来使能外接 USB 电源芯片，外接电源芯片的过电流检出可以通过本 MCU 的外部中断 IRQ 实现。主机模式下 USBHS_VBUS 可以作为 GPIO 使用。典型的 USB 主机模式系统构建图如下：

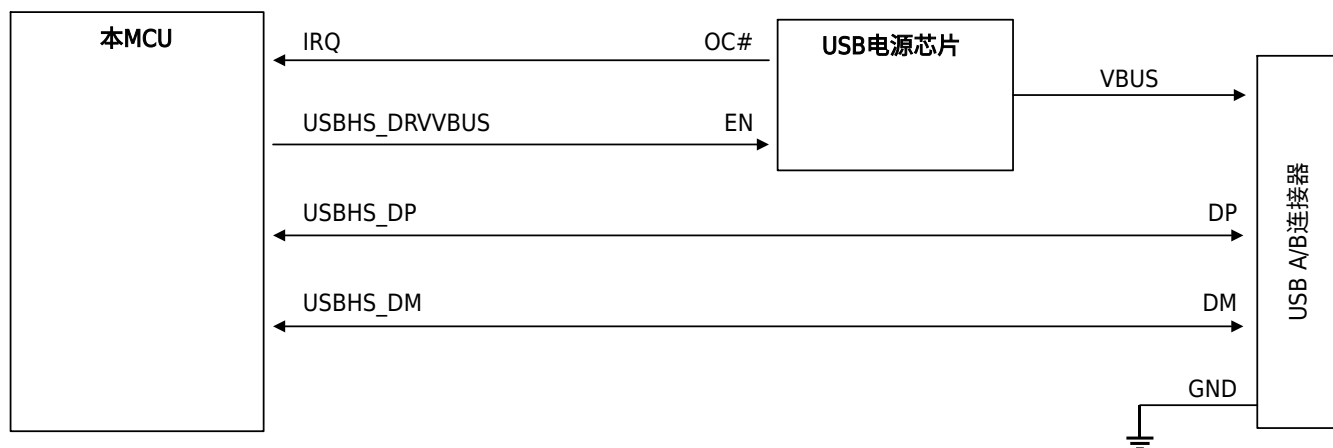


图 34-2 USBHS 主机模式系统构建图

主机端口供电

本 MCU 不能输出 5V 以提供 VBUS。为此，必须在微控制器以外添加 USB 电源芯片或基本电源开关（如应用电路板提供 5V 电源）来驱动 5V VBUS 线。外部电源芯片可通过任何 GPIO 输出或者 USBHS_DRVVBUS 驱动。当应用程序确定使用 GPIO 来控制外部器件提供 VBUS，仍须将主机端口控制和状态寄存器中的端口电源位（USBHS_HPRT 中的 PWPR 位）置 1。

主机检测设备连接和断开

USB 设备将在连接后立即被检测到。USBHS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（USBHS_HPRT 中的 PCDET 位）触发。

设备断开事件将触发断开连接检测中断（USBHS_GINTSTS 中的 DISCINT 位）。

主机枚举

检测到设备连接后，若又有新的设备连接进来，主机必须通过向新的设备发送 USB 复位和配置命令来启动枚举过程。

应用程序通过将主机端口控制和状态寄存器中的端口复位位（USBHS_HPRT 中的 PRST 位）置 1，使该过程最少持续 10ms、最多持续 20ms，以此通过 USB 驱动 USB 复位信号（单端零）。应用程序计算这个过程的持续时间，然后将端口复位位清零。

USB 复位序列完成后，端口使能/禁止更改位（USBHS_HPRT 中的 PENCHNG 位）立即触发主机端口中断，进而向应用程序发出通知，指示可从主机端口控制和状态寄存器中的端口速度字段（USBHS_HPRT 中的 PSPD）读取枚举的设备速度，以及主机已经开始驱动 SOF（HS/FS）或 Keep-alive 令牌（LS）。此时主机已就绪，可通过对设备发送命令来完成对设备的枚举。

主机挂起

应用程序通过将主机端口控制和状态寄存器中的端口挂起位（USBHS_HPRT 中的 PSUSP）置 1 来挂起 USB 总线。USBHS 模块停止发送 SOF 并进入挂起状态。

可由远程设备的自主活动（远程唤醒）使总线退出挂起状态。这种情况下，远程唤醒信号将触发远程唤醒中断（USBHS_GINTSTS 中的 WKUPINT 位），硬件把主机端口控制和状态寄存器中的端口恢复位

（USBHS_HPRT 中的 PRES 位）自行复位，并通过 USB 自动驱动恢复信号。应用程序必须为恢复窗口定时，然后将端口恢复位清零以退出挂起状态并重新启动 SOF。

如果由主机发起退出挂起状态，则应用程序必须将端口恢复位置 1 以启动主机端口上的恢复信号，为恢复窗口定时并最终将端口恢复位清零。

主机通道

USBHS 模块实现了 16 个主机通道。每个主机通道均可用于 USB 主机传输（USB 管道）。主机最多能同时处理 8 个传输请求。如果应用程序有 8 个以上的传输请求挂起，则在通道从之前任务释放后（即，接收到传输完成和通道停止中断后），主机控制器驱动器（HCD）必须为未处理的传输请求重新对通道进行分配。

每个主机通道都可配置为支持输入/输出以及周期性/非周期性事务。每个主机通道都使用专用控制 (HCCHARx) 寄存器、传输配置 (HCTSIZx) 寄存器/中断 (HCINTx) 寄存器以及和其相关的中断屏蔽寄存器 (HCINTMSKx)。

主机通道控制

应用程序可通过主机通道 x 特性寄存器 (HCCHARx) 对主机通道作以下控制：

- 通道使能/禁止
- 设置目标 USB 设备的速度：HS/FS/LS
- 设置目标 USB 设备的地址
- 设置与该通道通信的目标 USB 设备上的端点的编号
- 设置该通道上的传输方向：IN/OUT
- 设置该通道上的 USB 传输的类型：控制/批量/中断/同步
- 设置与该通道通信的设备端点的最大包长
- 设置要进行周期传输的帧：奇帧/偶帧

主机通道传输

主机通道传输大小寄存器 (HCTSIZx) 允许应用程序对传输大小参数进行编程并读取传输状态。必须在主机通道特性寄存器中的通道使能位置 1 之前完成对此寄存器的设置。使能端点后，数据包计数字段立即变为只读状态，同时 USBHS 模块根据当前传输状态对该字段进行更新。

可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输大小的数据包个数
- 初始数据 PID

主机通道状态/中断

主机通道 x 中断寄存器 (HCINTx) 指示端点在出现 USB 和 AHB 相关事件时的状态。当中断寄存器中的主机通道中断位 (USBHS_GINTSTS 中的 HCINT 位) 置 1 时，应用程序必须读取这些寄存器以获得详细信息。在读取这些寄存器之前，应用程序必须先读取主机全体通道中断 (HCAINT) 寄存器，以获取主机通道 x 中断寄存器的通道编号。应用程序必须将此寄存器中的相应位清零，才能将 HCAINT 和 GINTSTS 寄存器中的相应位清零。USBHS_HCINTMSKx 寄存器还提供每个通道各中断源的屏蔽位。

主机模块提供以下状态检查和中断产生功能：

- 传输完成中断，指示应用程序 (AHB) 和 USB 端均已完成数据传输
- 通道因传输完成、USB 事务错误或应用程序发出禁止命令而停止
- 相关的发送 FIFO 为半空或全空状态 (IN 端点)
- 接收到 ACK 响应
- 接收到 NAK 响应
- 接收到 STALL 响应
- 由于 CRC 校验失败、超时、位填充错误和错误的 EOP 导致 USB 事务错误
- 串扰错误
- 帧上溢
- 用于数据同步的翻转位出错

34.3.6 设备功能

功能简介

当 USBHS 工作在设备模式下时，VBUS 为 USB 协议所规定的 5V 电源引脚，为 5V 耐压管脚。本模块始终检测 VBUS 线的电平状态来连接或者断开设备。USBHS_VBUS 端口需先经过 $\geq 1k\Omega$ 的电阻才能接到 VBUS 电源。

典型的 USB 设备模式系统构建图如下：

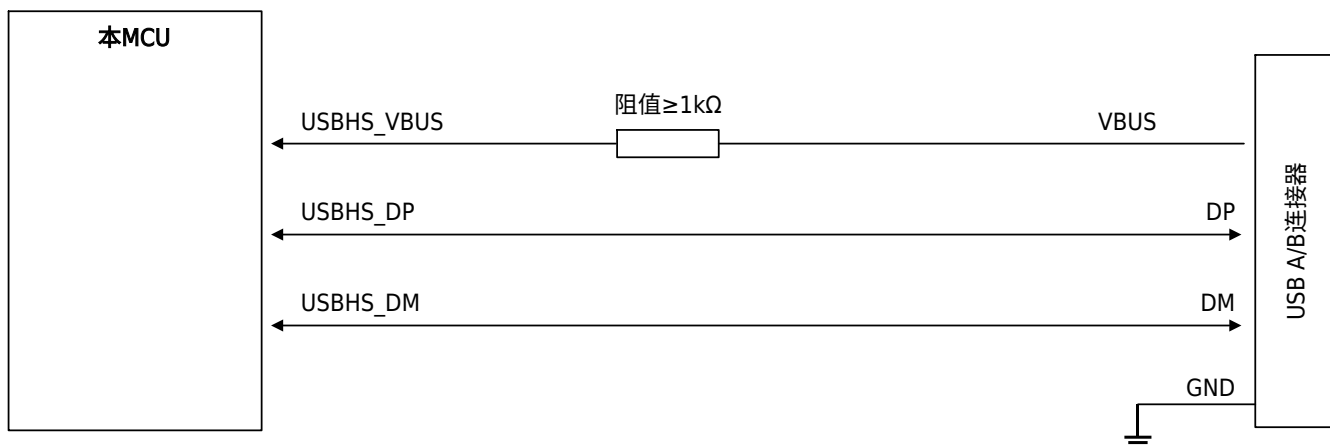


图 34-3 USBHS 设备模式系统构建图

设备供电状态

模块检测到 USBHS_VBUS 为高电平时，就会使 USB 设备进入供电状态。然后，USBHS 自动连接 DP 上拉电阻，发出全速设备与主机相连的信号并生成会话请求中断（USBHS_GINTSTS 中的 VBUSVINT 位），指示进入供电状态。

此外，USBHS_VBUS 输入还可确保主机在 USB 操作期间提供有效的 VBUS 电平。如果检测到 VBUS 电平为低电平（例如，因电源干扰或主机端口关闭引发），USBHS 将自动断开连接。

供电状态下，USBHS 期望收到来自主机的复位信号。其它 USB 操作则无法执行。收到复位信号后，立即生成检测到复位中断（USBHS_GINTSTS 中的 USBRST）。复位信号结束后，将生成枚举完成中断

（USBHS_GINTSTS 中的 ENUMDNE 位），USBHS 随即进入默认状态。

设备默认状态

默认状态下，USBHS 期望从主机收到 SET_ADDRESS 命令。其它 USB 操作则无法执行。当 USB 上解码出有效 SET_ADDRESS 命令时，应用程序会将相应的地址值写入设备配置寄存器中的设备地址字段

（USBHS_DCFG 中的 DAD 位）。USBHS 随即进入地址状态，并准备好以所配置的 USB 地址对主机事务进行应答。

设备挂起状态

USBHS 设备持续监视 USB 活动。在 USB 空闲时间达到 3ms 后，将发出早期挂起中断（USBHS_GINTSTS 中 ESUSP 位），并在 3ms 后由挂起中断（USBHS_GINTSTS 中的 USBSUSP 位）确认设备进入挂起状态。然后，设备状态寄存器中的设备挂起位（USBHS_DSTS 中的 SUSPSTS 位）自动置 1，USBHS 随即进入挂起状态。

可通过设备本身退出挂起状态。这种情况下，应用程序会将设备控制寄存器中的远程唤醒信号位

（USBHS_DCTL 中的 RWUSIG 位）置 1，并在 1ms 到 15ms 内将其清零。

但若设备检测到主机发出的恢复信号，将生成恢复中断（USBHS_GINTSTS 中的 WKUPINT 位），设备挂起位自动清零。

设备软断开

供电状态可借助软断开功能通过软件退出。将设备控制寄存器中的软断开位 (USBHS_DCTL 中的 SDIS 位) 置 1 即可移除 DP 上拉电阻, 此时尽管没有从主机端口实际拔出 USB 电缆, 但主机端仍会发生设备断开检测中断。

设备端点

端点类别

USBHS 模块实现了以下 USB 端点:

- 控制端点 0:
 - ▶ 双向且仅处理控制消息
 - ▶ 使用一组单独的寄存器来处理 IN 和 OUT 事务
 - ▶ 专用控制 (USBHS_DIEPCTL0/USBHS_DOEPCTL0) 寄存器、传输配置 (USBHS_DIEPTSIZ0/USBHS_DOEPTSIZ0) 寄存器和状态中断 (USBHS_DIEPINTx/USBHS_DOEPINT0) 寄存器。控制和传输大小寄存器中可用的位组与其它端点中稍有不同
- 15 个 IN 端点:
 - ▶ 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - ▶ 每个端点都有专用控制 (USBHS_DIEPCTLx) 寄存器、传输配置 (USBHS_DIEPTSIZx) 寄存器和状态中断 (USBHS_DIEPINTx) 寄存器
 - ▶ 设备 IN 端点通用中断屏蔽寄存器 (USBHS_DIEPMSK) 可用于使能/禁止所有 IN 端点 (包括 EP0) 上的同一类端点中断源
 - ▶ 支持未完成的同步 IN 传输中断 (USBHS_GINTSTS 中的 IISOIXFR 位), 该中断将在当前帧中至少有一个同步 IN 端点上的传输未完成时触发。该中断和周期性帧中断 (USBHS_GINTSTS/EOPF) 一起触发
- 15 个 OUT 端点:
 - ▶ 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - ▶ 每个端点都有专用控制 (USBHS_DOEPCTLx) 寄存器、传输配置 (USBHS_DOEPTSIZx) 寄存器和状态中断 (USBHS_DOEPINTx) 寄存器
 - ▶ 设备 OUT 端点通用中断屏蔽寄存器 (USBHS_DOEPMSK) 可用于使能/禁止所有 OUT 端点 (包括 EP0) 上的同一类端点中断源
 - ▶ 支持未完成的同步 OUT 传输中断 (USBHS_GINTSTS 中的 INCOMPISOOOUT 位), 该中断将在当前帧中至少有一个同步 OUT 端点上的传输未完成时触发。该中断和周期性帧中断 (USBHS_GINTSTS/EOPF) 一起触发

端点控制

应用程序可通过设备端点 x IN/OUT 控制寄存器 (DIEPCTLx/DOEPCTLx) 对端点采取以下控制:

- 端点使能/禁止
- 在当前配置下激活端点
- 设置 USB 传输类型 (同步、批量和中断)
- 设置支持的数据包大小
- 设置与 IN 端点相关的 Tx-FIFO 编号
- 设置希望收到的或发送时要使用到的 data0/data1 PID (仅限批量/中断传输)
- 设置接收或发送事务时所对应的奇/偶帧 (仅限同步传输)
- 可以设置 NAK 位, 从而不论此时 FIFO 的状态如何, 都对主机的请求回复 NAK
- 可以设置 STALL 位, 使得主机对该端点的令牌都被硬件回复 STALL
- 可以将 OUT 端点设置为侦听模式, 即对接收到的数据不进行 CRC 检查

端点传输

设备端点 x 传输尺寸寄存器 (DIEPTSIZE/DOEPTSIZE) 允许应用程序对传输尺寸参数进行编程并读取传输状态。必须在端点控制寄存器中的端点使能位置 1 之前完成对此寄存器的设置。使能端点后, 这些字段立即变为只读状态, 同时 USBHS 模块根据当前传输状态对这些字段进行更新。

可对以下传输参数进行编程:

- 以字节为单位的传输大小
- 构成整个传输的数据包个数

端点状态/状态

设备端点 x 中断寄存器 (DIEPINT/DOEPINT) 指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 OUT 端点中断位或 IN 端点中断位 (分别为 USBHS_GINTSTS 中 OEPINT 位或 USBHS_GINTSTS 中的 IEPINT 位) 置 1 时, 应用程序必须读取这些寄存器以获得详细信息。在应用程序读取这些寄存器之前, 必须先读取设备全体端点中断 (USBHS_DAINTE) 寄存器, 以获取设备端点 x 中断寄存器的端点编号。应用程序必须将此寄存器中的相应位清零, 才能将 DAINTE 和 GINTSTS 寄存器中的相应位清零。

模块提供以下状态检查和中断产生功能:

- 传输完成中断, 指示应用程序 AHB 和 USB 端均已完成数据传输
- Setup 阶段已完成 (仅针对控制传输类型的 OUT 端点)
- 相关的发送 FIFO 为半空或全空状态 (IN 端点)
- NAK 应答已发送到主机 (仅针对同步传输的 IN 端点)
- TxFIFO 为空时接收到 IN 令牌 (仅针对批量和中断传输类型的 IN 端点)
- 尚未使能端点时接收到 OUT 令牌
- 检测到 babble 错误
- 应用程序关闭端点生效
- 应用程序对端点设置 NAK 生效 (仅针对同步传输类型的 IN 端点)
- 接收到 3 个以上连续 setup 数据包 (仅针对控制类型的 OUT 端点)
- 检测到超时状况 (仅针对控制传输类型的 IN 端点)

34.3.7 SOF 脉冲管脚输出功能

USBHS 在主机和设备模式下都可以监视、跟踪和配置 SOF 帧并且还具备 SOF 脉冲输出功能。SOF 脉冲通过 USBHS_SOF 管脚输出, 输出宽度为 16 个 HCLK 时钟周期。

主机 SOF

主机模式下, 可以在主机帧间隔寄存器 (HFIR) 中对所产生的两个连续 SOF (HS/FS) 或 keep-alive (LS) 令牌期间所出现的 PHY 时钟数进行编程, 进而应用程序可对 SOF 帧周期进行控制。帧开始

(USBHS_GINTSTS 中的 SOF 位) 时都将生成中断。当前帧编号和出现下一个 SOF 前剩余的时间应用程序在主机帧编号寄存器 (HFNUM) 中能够进行跟踪。

使用 USBHS 系统控制寄存器 USBHS_SYCTLREG 中的 SOFEN 位, 可以使任何 SOF 令牌发出的同时产生的、宽度为 16 个 HCLK 时钟周期的 SOF 脉冲信号从 USBHS_SOF 引脚输出。

此外, SOF 脉冲还可以作为内部事件触发 DMA 传输、TIMER 计数等外部模块工作。

设备 SOF

在设备模式下, USB 每次接收到 SOF 令牌时, 都将触发帧开始中断 (USBHS_GINTSTS 中的 SOF 位)。相应的帧编号可从设备状态寄存器 (USBHS_DSTS 中的 FNSOF 位) 读取。使用 USBHS 系统控制寄存器 USBHS_SYCTLREG 中的 SOFEN 位, 还可以生成宽度为 16 个 HCLK 时钟周期的 SOF 脉冲信号, 并使该信号在 USBHS_SOF 引脚输出, 以实现外部可用。

此外, SOF 脉冲还可以作为内部事件触发 DMA 传输、TIMER 计数等外部模块工作。

周期性帧结束中断 (GINTSTS/EOPF) 用于在经过了 80%、85%、90%或 95%的帧间隔时间时通知应用程序，具体取决于设备配置寄存器中的周期性帧间隔字段 (USBHS_DCFG 中的 PFIVL 位)。此功能可用于确定该帧的所有同步通信是否完成。

34.3.8 功耗控制

不使用 USBHS 模块时，可以通过 CMU 模块停止 USBHS 模块的 HCLK 和 PHY 时钟，从而降低功耗。使用 USB 模块，但设备 USB 会话未开始或设备未连接时，可以在 USB 挂起状态下使用功率降低技术。

- 停止 PHY 时钟 (USBHS_GCCTL 中的 STPPCLK 位)
将时钟门控控制寄存器中的停止 PHY 时钟位置 1 时，USBHS 全速模块的大多数 48MHz 内部时钟域均由时钟门控关闭。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗还会关掉收发器的大部分单元，只有负责检测异步恢复事件或远程唤醒事件的部分还保持工作状态。
- HCLK 门控 (USBHS_GCCTL 中的 GATEHCLK 位)
将时钟门控控制寄存器中的 GATEHCLK 位置 1 时，USBHS 模块内部的大多数 HCLK 时钟域均由时钟门控关闭。只有寄存器读取和写入接口保持活动状态。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗。

为了节省动态功耗，只在 USB 数据 FIFO 被 USBHS 模块访问时为其提供时钟。

34.3.9 动态更新 USBHS_HFIR 寄存器

主机模式下，USB 模块具有对帧周期进行动态微调的功能，能够将外部设备与 SOF 帧进行同步。如果 USBHS_HFIR 寄存器在当前 SOF 帧内发生更改，则将在下一个帧中对 SOF 周期进行相应修正，具体说明详见下图。

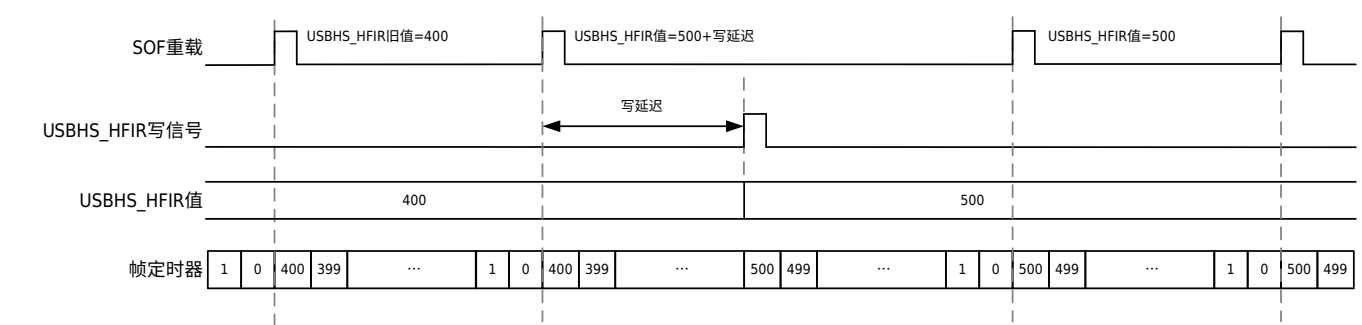


图 34-4 USBHS 动态更新 USBHS_HFIR 寄存器示意图

34.3.10 数据 FIFO

USBHS 系统具有 8KB 专用 RAM，采用高效的 FIFO 控制机制。USBHS 模块中的数据包 FIFO 控制器模块将 RAM 空间划分为多个 TxFIFO (USB 传输前，应用程序将数据压入其中进行短暂存储) 和单个 Rx FIFO (从 USB 接收到的数据被应用程序读取之前，在其中进行短暂存储)。

RAM 中所构建的 FIFO 的数量与组织方式取决于设备的角色。设备模式下，为每个激活的 IN 端点配置一个 Tx FIFO。FIFO 的大小均由软件配置，以更好地满足应用要求。

34.3.11 主机 FIFO 架构

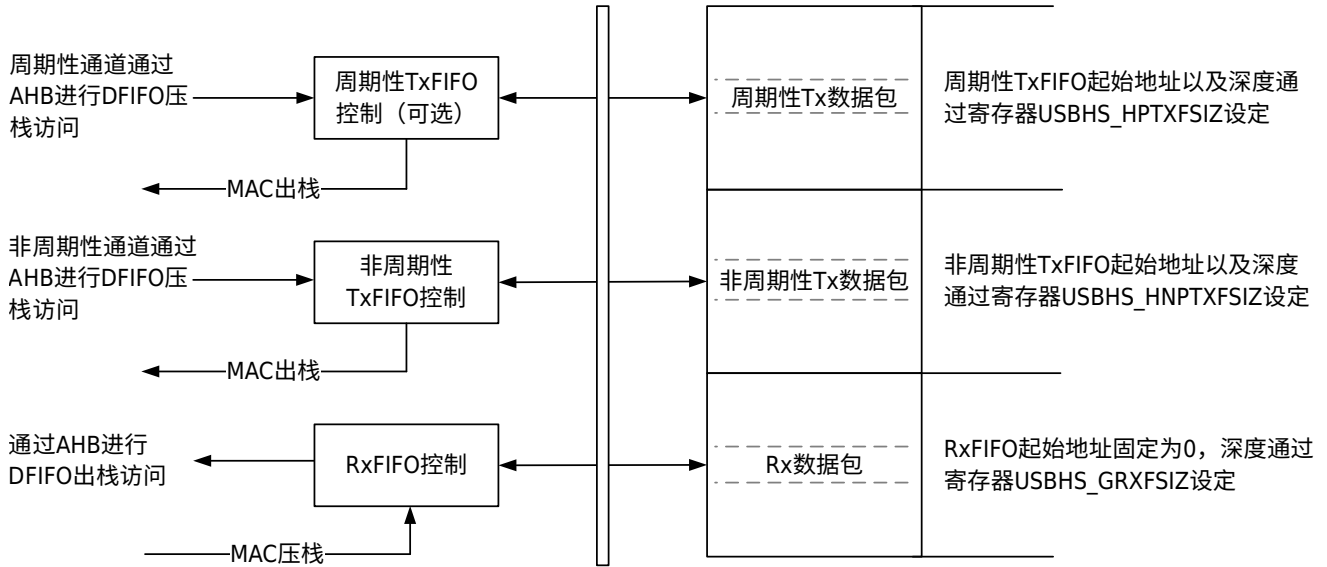


图 34-5 主机模式下 FIFO 架构示意图

主机 Rx FIFO

主机使用一个接收 FIFO 处理所有周期和非周期事务。FIFO 用作接收缓冲区以保存从 USB 接收到的数据（接收到的数据包的数据部分），直至这些数据传输到系统存储器。只要 FIFO 中有空间，来自设备 IN 端点的数据包就接收进来并挨个存储。接收到的每个数据包的状态（包含主机目标通道、字节数、数据 PID 和对所接收数据的校验）也存储在 FIFO 中。接收 FIFO 的大小在接收 FIFO 大小寄存器（GRXFSIZ）中配置。

单个接收 FIFO 架构使得 USB 主机高效地填充接收数据缓冲区：

- 所有 IN 配置主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- 对于主机软件驱动的任意序列 IN 令牌，USBHS 模块可将接收 FIFO 填充至限值

只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会接收 Rx FIFO 非空中断。应用程序从接收状态读取和出栈寄存器中读取数据包信息，最后从 Rx FIFO 中读出数据。

主机 Tx FIFO

主机使用一个发送 FIFO 处理所有非周期（控制和批量）OUT 事务，使用另一个发送 FIFO 处理所有周期（同步和中断 OUT 事务）。FIFO 用作发送缓冲区以保存要通过 USB 发送的数据（发送数据包）。周期（非周期）Tx FIFO 的大小在主机周期（非周期）发送 FIFO 大小（HPTXFSIZ/HNPTXFSIZ）寄存器中配置。

两个 Tx FIFO 按优先级实施操作，周期性通信的优先级较高，因此在 USB 一帧的时间内首先进行周期性通信。帧起始时，内置的主机调度器先处理周期请求队列，再处理非周期请求队列。

两个发送 FIFO 的架构使得 USB 主机能够对周期和非周期发送数据缓冲区分别进行优化管理：

- 配置为支持周期（非周期）OUT 事务的所有主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- 对于主机软件驱动的任意序列 OUT 令牌，USBHS 模块可将周期性（非周期性）发送 FIFO 填充至限值

只要周期性 Tx FIFO 为半空或全空，USBHS 模块就会发出周期性 Tx FIFO 空中断（USBHS_GINTSTS 中的 PTXFE 位），具体取决于 AHB 配置寄存器中的周期性 Tx-FIFO 空等级位（USBHS_GAHBCFG 中的 PTXFELVL 位）的值。只要周期性 Tx FIFO 和周期性请求队列中均存在空闲空间，应用程序便可提前写入发送数据。可通过读取主机周期性发送 FIFO 和队列状态寄存器（HPTXSTS）来了解二者的可用空间。

只要非周期性 Tx FIFO 为半空或全空，USBHS 模块就会发出非周期性 Tx FIFO 空中断（USBHS_GINTSTS 中的 NPTXFE 位），具体取决于 AHB 配置寄存器中的非周期性 Tx FIFO 空等级位（USBHS_GAHBCFG 中的 TXFELVL 位）。只要非周期性 Tx FIFO 和非周期性请求队列中均存在空闲空间，应用程序便可写入发送数据。可通过读取主机非周期性发送 FIFO 和队列状态寄存器（HNPTXSTS）来了解二者的可用空间。

34.3.12 设备 FIFO 架构

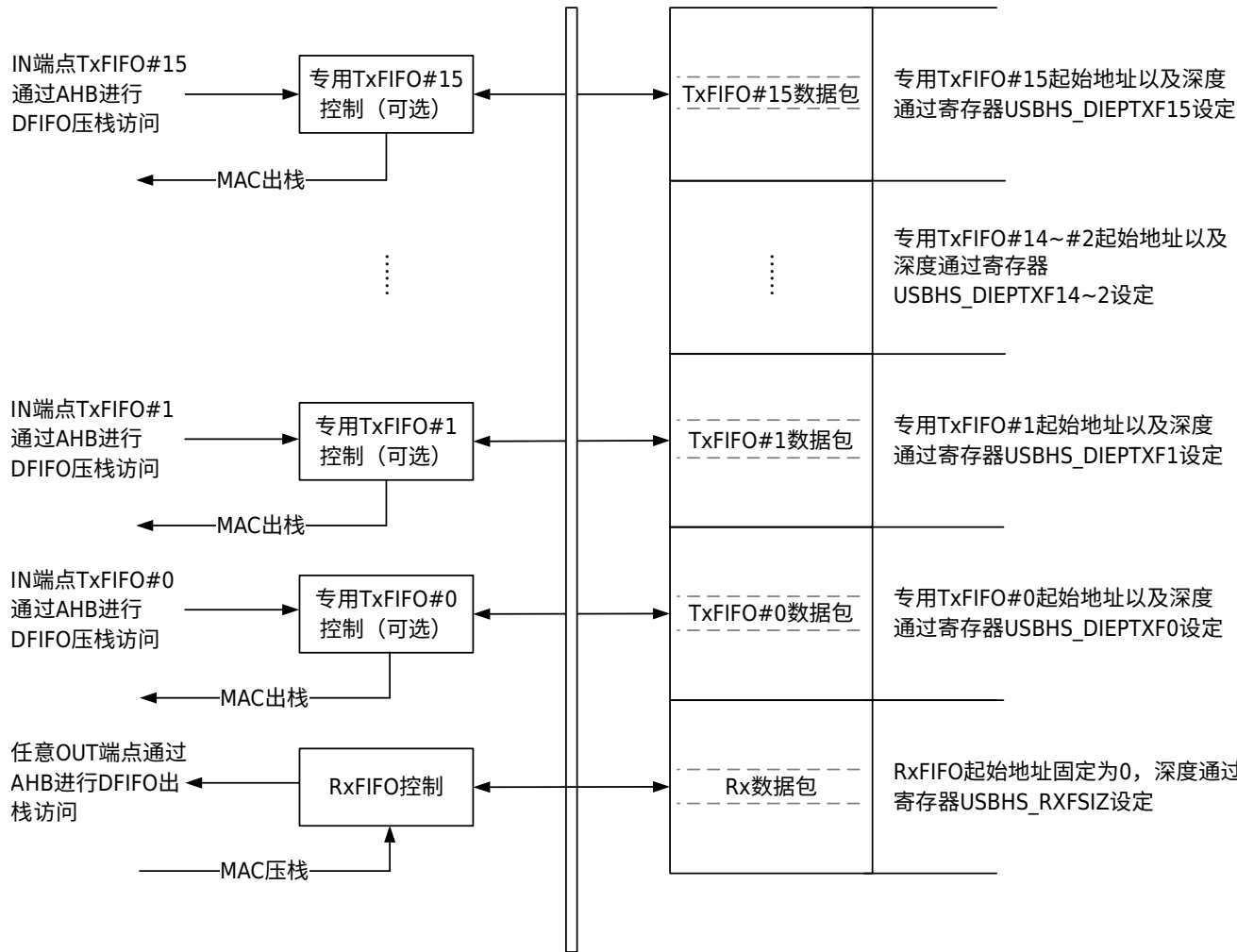


图 34-6 USBHS 设备模式下 FIFO 架构示意图

设备 Rx FIFO

USBHS 设备使用单个接收 FIFO 接收发送到所有 OUT 端点的数据。只要 Rx FIFO 中有空余空间，收到的数据包就挨个填入 Rx FIFO。除了有效数据外，接收到的数据包状态（包含 OUT 端点目标编号、字节数、数据 PID 和对所接收数据的验证）也由模块进行存储。没有可用空间时，设备会回复主机事务 NAK 应答并在被寻址的端点上触发中断。接收 FIFO 的大小在接收 FIFO 大小寄存器（GRXFSIZ）中配置。

单个接收 FIFO 架构使得 USB 设备更高效地填充接收 RAM 缓冲区：

- 所有 OUT 端点共享同一个 RAM 缓冲区（共享 FIFO）
- 对于任意主机序列 OUT 令牌，USBHS 模块可将接收 FIFO 填充至限值

只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会一直接收 Rx FIFO 非空中断（USBHS_GINTSTS 中的 RXFNE 位）。应用程序从接收状态读取和出栈寄存器（GRXSTSP）中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

设备 Tx FIFO

模块为各个 IN 端点提供了专用的 FIFO。应用程序通过非周期发送 FIFO 大小寄存器（USBHS_DIEPTSIZ0）为 IN 端点 0 配置 FIFO 大小；通过设备 IN 端点发送 FIFOx 寄存器（DIEPTSIZx）为 IN 端点 x 配置 FIFO 大小。

34.3.13 FIFO RAM 分配

主机模式

接收 FIFO RAM 分配

状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+2 的空间。如果使能了多个同步通道，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+2。通常，推荐的空间为（最大数据包/4+2）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被写入 FIFO。所以必须为此分配一个位置。

以 DMA 模式运行时，每个主机通道的 DMA 地址寄存器将存储在 FIFO 中，因此需要在 FIFO 中为每个通道保留一个位置存放其地址寄存器。

发送 FIFO RAM 分配

主机非周期性发送 FIFO 所需的最小 RAM 为所支持的所有非周期性 OUT 通道上传输的最大数据包的大小。通常，推荐的空间为最大数据包大小的两倍，这样当 USB 正在发送当前数据包的同时，AHB 可以往发送 FIFO 填入下一个数据包。

主机周期性发送 FIFO 所需的最小 RAM 为所支持的所有周期性 OUT 通道上传输的最大数据包的大小。如果至少有一个同步 OUT 端点，则空间必须至少为该通道中最大数据包大小的两倍。

以 DMA 模式运行时，每个主机通道的 DMA 地址寄存器将存储在 FIFO 中，因此需要在 FIFO 中为每个通道保留一个位置存放其地址寄存器。

设备模式

接收 FIFO RAM 分配

应用程序应为 SETUP 数据包分配 RAM：接收 FIFO 中必须保留 13 个位置以在控制端点上接收 SETUP 数据包。USBHS 模块不会向这些为 SETUP 数据包保留的位置写入任何其它数据。将会为全局 OUT NAK 分配一个位置。状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+1 的空间。如果使能了多个同步端点，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+1。通常，推荐的空间为（最大数据包/4+1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被推入 FIFO。通常情况下，推荐为每个 OUT 端点分配一个位置。

发送 FIFO RAM 分配

各个 IN 端点发送 FIFO 所需的最小 RAM 空间为该特定 IN 端点的最大数据包大小。

34.3.14 系统性能

凭借大容量 RAM 缓冲区、高度可配置的 FIFO 大小、通过 AHB 压栈/出栈寄存器进行 32 位 FIFO 快速访问，尤其是高级 FIFO 控制机制可获得最佳 USB 和系统性能。实际上，无论当前 USB 序列如何，USBHS 均可通过该机制高效填充可用的 RAM 空间。借助这些特性：

- 应用程序有足够的裕量来计算并校正 CPU 的负载，从而优化 CPU 带宽利用率：
 - ▶ 应用程序可先积累大量发送数据，再通过 USB 发送出去
 - ▶ 可带来足够的时间裕量，以从接收 FIFO 读取数据
- USB 模块能够保持全速工作状态，也就是提供最大的全速带宽（尽量多的硬件自动运行，尽量少的软件参与）：
 - ▶ USB 模块可提前积累大量发送数据供其支配，从而可对 USB 数据发送进行自主管理
 - ▶ 接收缓冲区中有大量空白空间，可通过 USB 中的数据自动填满

由于 USBHS 模块能够高效填充 8KB RAM 缓冲区且 8KB 发送/接收数据足以满足一个全速帧所能容纳的数据量，因此 USB 系统在一帧之内可以无需应用程序干预达到最大 USB 带宽。

34.3.15 中断和事件

USBHS 中断有以下四类，SOTP 模式唤醒中断 USBHS_WKUP、端点 1 OUT 中断 USBHS_EP1_OUT、端点 1 IN 中断 USBHS_EP1_IN、USBHS 全局中断 USBHS_GLB。

● USBHS_WKUP 中断

USBHS_WKUP 中断用于 STOP 模式下，通过 USBHS_DP 或者 USBHS_DM 唤醒系统的 STOP 模式，该中断使能位为 INT_WUPEN.USH_WUEN。

使用 USBHS 唤醒系统 STOP 前，需要确保 USBHS 控制器处于挂起状态，并设定寄存器 USBHS_SYCTLREG 中的对应的滤波范围和使能滤波功能。

● USBHS_EP1_OUT 中断

USBHS_EP1_OUT 中断是软件需要处理设备模式端点 1 的 OUT 中断，中断标志可以在 USBHS_DEACHINT 中查询。

● USBHS_EP1_IN 中断

USBHS_EP1_IN 中断是软件需要处理设备模式端点 1 的 IN 中断，中断标志可以在 USBHS_DEACHINT 中查询

● USBHS_GLB 中断

USBHS_GLB 中断是软件需要处理的主要中断，全局中断的标志位可在 USBHS_GINTSTS 寄存器读取。

表 34-2 USBHS_GLB 中断事件表

中断标志	描述	运行模式	内部事件源
WKUPINT	恢复/远程唤醒中断	主机或者设备	-
VBUSVINT	VBUS 有效中断	设备	-
DISCINT	断开连接中断	主机	-
CIDSCHG	连接器 ID 线状态变化中断	主机或者设备	-
PTXFE	周期性 Tx FIFO 空中断	主机	-
LPMINT	LPM 中断	主机或者设备	-
HCINT	主机通道中断	主机	-
HPRTINT	主机端口中断	主机	-
DATAFSUSP	数据获取挂起	设备	-
IPXFR/INCOMPISOOOUT	未完成周期性传输/未完成 OUT 同步传输	设备	-
IISOIXFR	未完成 IN 同步传输	设备	-
OEPINT	OUT 端点中断	设备	-
IEPINT	IN 端点中断	设备	-
EOPF	周期性帧结束中断	设备	-
ISOODRP	丢弃同步 OUT 数据包中断	设备	-
ENUMDNE	枚举完成	设备	-

中断标志	描述	运行模式	内部事件源
USBRST	USB 复位中断	设备	-
USBSUSP	USB 挂起中断	设备	-
ESUSP	早期挂起中断	设备	-
GONAKEFF	全局 OUT NAK 有效中断	设备	-
GINAKEFF	全局非周期性 IN NAK 有效中断	设备	-
NPTXFE	非周期性 TxFIFO 空中断	主机	-
RXFNE	RxFIFO 非空中断	主机或者设备	-
SOF	帧起始中断	主机或者设备	是
MMIS	模式不匹配中断	主机或者设备	-

34.4 编程模型

34.4.1 模块初始化

背景信息

应用程序必须执行模块初始化序列。模式决定方法请参见[模式决定](#)。

本节介绍了 USBHS 控制器上电后的初始化过程。无论是以主机模式还是设备模式工作，应用程序都必须遵循初始化序列。根据模块配置对所有模块全局寄存器进行初始化：

操作步骤

- 步骤 1.** 在 USBHS_GAHBCFG 寄存器中编程以下字段：
- 全局中断屏蔽位 GINTMSK = 1
 - RxFIFO 非空 (USBHS_GINTSTS 中的 RXFNE 位)
 - 周期性 TxFIFO 空门限
- 步骤 2.** 在 USBHS_GUSBCFG 寄存器中编程以下字段：
- FS 超时校准字段
 - USB 周转时间字段
- 步骤 3.** 软件必须取消对 USBHS_GINTMSK 寄存器中以下位的屏蔽：
- 模式不匹配中断屏蔽
- 步骤 4.** 通过读取 USBHS_GINTSTS 中的 CMOD 位，软件可确定 USBHS 控制器是在主机模式还是设备模式下工作。

34.4.2 主机初始化

背景信息

要将模块作为主机进行初始化，应用程序必须执行以下步骤：

操作步骤

- 步骤 1.** 编程 USBHS_GINTMSK 寄存器中的 HPRTINT 以将对取消屏蔽。
- 步骤 2.** 编程 USBHS_HCFG 寄存器以选择全速主机。
- 步骤 3.** 将 USBHS_HPRT 中的 PWPR 位编程为 1，给 USB 总线提供 VBUS。
- 步骤 4.** 等待 USBHS_HPRT 中的 PCDET 中断。这表示某设备已连接到主机端口。
- 步骤 5.** 将 USBHS_HPRT 中的 PRST 位编程为 1，在 USB 总线上发出复位信号。

- 步骤 6. 至少等待 10ms，以便完成复位过程。
- 步骤 7. 将 USBHS_HPRT 中的 PRST 位编程为 0。
- 步骤 8. 等待 USBHS_HPRT 中的 PENCHNG 中断。
- 步骤 9. 读取 USBHS_HPRT 中的 PSPD 位以获取枚举速度。
- 步骤 10. 使用所选 PHY 时钟，相应地设置 HFIR 寄存器。
- 步骤 11. 根据步骤 9 中检测到的设备速度编程 USBHS_HCFG 寄存器中的 FSLSPCS 字段。如果 FSLSPCS 发生更改，则必须执行端口复位。
- 步骤 12. 编程 USBHS_GRXFSIZ 寄存器以选择接收 FIFO 的大小。
- 步骤 13. 编程 USBHS_HNPTXFSIZ 寄存器，以选择用于非周期性通信事务的非周期性发送 FIFO 的大小和起始地址。
- 步骤 14. 编程 USBHS_HPTXFSIZ 寄存器，以选择用于周期性事务的周期性通信发送 FIFO 的大小和起始地址。

后续任务

要与设备通信，系统软件必须初始化并使能至少一个通道。

34.4.3 设备初始化

背景信息

上电期间或者从主机模式切换为设备模式后，应用程序必须执行下列步骤来将模块作为设备进行初始化。

操作步骤

- 步骤 1. 在 USBHS_DCFG 寄存器中编程以下字段：
 - 设备速度
 - 非零长度状态 OUT 握手信号
- 步骤 2. 编程 USBHS_GINTMSK 寄存器以取消屏蔽以下中断：
 - USB 复位
 - 枚举完成
 - 早期挂起
 - USB 挂起
 - SOF
- 步骤 3. 等待 USBHS_GINTSTS 中的 VBUSVINT 中断，表示进入供电状态。
- 步骤 4. 等待 USBHS_GINTSTS 中的 USBRST 中断。这表示已在 USB 上检测到复位信号，复位过程自接收到此中断后约持续 10ms。
- 步骤 5. 等待 USBHS_GINTSTS 中的 ENUMDNE 中断。此中断指示 USB 上复位过程结束。接收到此中断时，应用程序必须读取 USBHS_DSTS 寄存器以确定枚举速度并执行枚举完成时的端点初始化中所列的步骤。

操作结果

此时，设备已准备好接受 SOF 数据包并在控制端点 0 上执行控制传输。

34.4.4 DMA 模式

USB 使用 AHB 主接口来获取发送数据包数据 (AHB 到 USB) 和接收数据更新 (USB 到 AHB)。AHB 主接口使用经过编程的 DMA 地址 (主机模式下的 HCDMAx 寄存器和设备模式下的 DIEPDMAx/DOEPDMAx 寄存器) 来访问数据缓冲区。

34.4.5 主机编程模型

34.4.5.1 通道初始化

背景信息

应用程序必须初始化一个或多个通道，之后才能与所连接的设备通信。

要初始化和使能通道，应用程序必须执行以下步骤：

操作步骤

步骤 1. 编程 USBHS_GINTMSK 寄存器以取消对以下位的中断屏蔽：

- 用于 OUT 事务的非周期性发送 FIFO 为空（在流水线事务级别工作且数据包计数字段编程值大 1 时适用）。
- 用于 OUT 事务的非周期性发送 FIFO 为半空（在流水线事务级别工作且数据包计数字段编程值大 1 时适用）。

步骤 2. 编程 USBHS_HAINTMSK 寄存器以使能所选通道中断。

步骤 3. 编程 USBHS_HCINTMSK 寄存器，以使能主机通道中断寄存器中反映的和通信事务有关的中断。

步骤 4. 编程所选通道的 USBHS_HCTSIZx 寄存器，指定以字节为单位的总传输大小和包括短数据包在内的预期数据包个数。应用程序必须使用初始数据 PID（用于第一个 OUT 事务或预期从第一个 IN 事务获取）编程 PID 字段。

步骤 5. 编程所选通道的 USBHS_HCCHARx 寄存器，指定设备的端点特性，例如类型、速度、方向等。（仅当应用程序准备好发送或接收数据包时，才能通过将通道使能位置 1 来使能通道）。

34.4.5.2 通道停止

应用程序可以通过编程 USBHS_HCCHARx 寄存器将 CHDIS 和 CHENA 位置 1 来禁止任何通道。这会使 USBHS 主机清空之前在该通道上发出的请求（如果有）并生成通道停止中断。应用程序在将通道重新分配给其它通信事务之前，必须等待 USBHS_HCINTx 中的 CHH 中断。USBHS 主机不会中断已在 USB 上启动的通信事务。

禁止通道前，应用程序必须确保非周期性请求队列（禁止非周期性通道时）或周期性请求队列（禁止周期性通道时）中至少有一个空闲空间。应用程序可以在请求队列已满时（禁止通道之前），通过编程 USBHS_HCCHARx 寄存器将 CHDIS 位置 1 和将 CHENA 位清零，清空请求队列。出现以下任一情况时，应用程序将禁止通道：

- IN 或 OUT 通道的 USBHS_HCINTx 中接收到 STALL、TXERR、BBERR 或 DTERR 中断。应用程序在接收到通道停止信号之前，必须能够接收相同通道的其它中断（DTERR、Nak、Data、TXERR）。
- 接收到 USBHS_GINTSTS 中的 DISCINT（断开设备连接）中断。（应用程序将禁止所有已使能的通道）。
- 应用程序在传输正常完成之前将其中止。

在 DMA 模式时，应用程序不能同过写寄存器停止不可分割的周期性传输。

34.4.5.3 Ping 协议

当 USBHS 主机工作在高速模式时，如果应用程序要与高速批量或控制（数据和状态阶段）OUT 端点进行通信，则必须使用 ping 协议。

当应用程序接收到 NAK/NYET/TXER 中断时，必须使用 pin 协议。当 USBHS 主机接收到上述其中一个响应时，它不会在该端点上继续执行任何通信事务，而是丢弃所有已发出或已获取 OUT 请求（从请求队列中），然后清空发送 FIFO 中的相应数据。

这仅在从模式下有效。在从模式下，应用程序可通过以下两种方式发送 ping 令牌：在使能通道之前，将 HCTSIZx 中的 DOPIN 位置 1，或者在通道已使能之后，对 HCTSIZx 寄存器执行写操作时将 DOPING 位置 1。这将使能 USBHS 主机将 ping 请求写入到请求队列中。应用程序必须等待设备对 ping 令牌的响应（NAK、ACK 或 TXERR 中断），然后才能继续执行通信事务或发送另一个 ping 令牌。仅当应用程序从设备的 OUT 端点接收到对 ping 令牌的 ACK 响应后，才能继续执行数据通信事务。在 DMA 模式下工作时，对于

批量/控制 OUT 事务，应用程序不需要在收到 NAK/NYET 回复时将 HCTSIZx 中的 DOPING 位置 1。USBHS 主机自动将 HCTSIZx 中的 DOPING 位置 1，然后在批量/控制 OUT 传输时发出 ping 令牌。USBHS 主机持续发送 ping 令牌，直至收到 ACK 为止，随后自动切换到数据通信事务。

34.4.6 设备编程模型

34.4.6.1 USB 复位时端点初始化

操作步骤

- 步骤 1. 为所有 OUT 端点将 NAK 位置 1
 - USBHS_DOEPTLx 中，SNAK = 1（对于所有 OUT 端点）
- 步骤 2. 取消对以下中断位的屏蔽
 - USBHS_DAINMSK 中，INEP0=1（控制 0 IN 端点）
 - USBHS_DAINMSK 中，OUTEP0=1（控制 0 OUT 端点）
 - DOEPMK 中，STUP=1
 - DOEPMK 中，XFRC=1
 - DIEPMK 中，XFRC=1
 - DIEPMK 中，TOC=1
- 步骤 3. 为每个 FIFO 设置数据 FIFO RAM
 - 对 USBHS_GRXFSIZ 寄存器进行编程，以能够接收控制传输的 OUT 数据和设置数据。该寄存器必须至少等于控制端点 0 的 1 个最大数据包大小+2 个字（用于控制 OUT 数据包的状态）+10 个字（用于 SETUP 数据包）。
 - 对 USBHS_TX0FSIZ 寄存器进行编程（取决于所选的 FIFO 编号），以能够发送控制 IN 数据。该寄存器至少必须等于控制端点 0 的 1 个最大数据包大小。
- 步骤 4. 对端点相关寄存器中的以下字段进行编程，以使控制 OUT 端点 0 接收 SETUP 数据包
 - USBHS_DOEPTSIZE 中的 STUPCNT=3（接收最多 3 个连续的 SETUP 数据包）

操作结果

此时，接收 SETUP 数据包所需的所有初始化工作便已完成。

34.4.6.2 USB 枚举完成时端点初始化

操作步骤

- 步骤 1. 在枚举完成中断（USBHS_GINTSTS 中的 ENUMDNE）中，读取 USBHS_DSTS 寄存器以确定设备的枚举速度。
- 步骤 2. 对 USBHS_DIEPTL0 中的 MPSIZ 字段进行编程以设置最大数据包大小。该步骤配置控制端点 0。控制端点的最大数据包大小取决于枚举速度。

操作结果

此时，设备已准备好接收 SOF 数据包并配置为在控制端点 0 执行控制传输。

34.4.6.3 收到 SetAddress 命令时的端点初始化

背景信息

本节介绍了应用程序在 SETUP 数据包中接收到 SetAddress 命令时必须执行的操作。

操作步骤

- 步骤 1. 使用在 SetAddress 命令中接收到的设备地址来对 USBHS_DCFG 寄存器进行编程
- 步骤 2. 对模块进行编程以发出状态阶段的 IN 数据包

34.4.6.4 收到 SetConfiguration/SetInterface 命令时的端点初始化

背景信息

本节介绍了应用程序在 SETUP 包中接收 SetConfiguration 或 SetInterface 命令时必须执行的操作。

操作步骤

- 步骤 1. 接收到 SetConfiguration 命令时，应用程序必须对端点寄存器进行编程，以使用新配置中有效端点的特性来配置这些端点寄存器。
- 步骤 2. 接收到 SetInterface 命令时，应用程序必须对命令指定的端点的端点寄存器进行编程。
- 步骤 3. 在先前配置或其它设置中有效的端点在新的配置或其它设置中无效。必须停用这些无效端点。
- 步骤 4. 使用 USBHS_DAINMSK 寄存器使能有效端点的中断，屏蔽无效端点的中断。
- 步骤 5. 为每个 FIFO 设置数据 FIFO RAM。
- 步骤 6. 配置完所有必需的端点后，应用程序必须对模块进行编程以发送状态阶段的 IN 数据包。

操作结果

此时，设备模块已可以接收和发送任何类型的数据包。

34.4.6.5 端点激活

背景信息

本节介绍激活设备端点或者将现有设备端点配置为新类型所需的步骤。

操作步骤

- 步骤 1. 在 USBHS_DIEPCTLx 寄存器（对于 IN 或双向端点）或 USBHS_DOEPCTLx 寄存器（对于 OUT 或双向端点）的以下字段中，对所需端点的特性进行编程。
 - 最大数据包大小
 - USB 活动端点位置 1
 - 端点初始数据同步位（对于中断和批量端点）
 - 端点类型
 - TxFIFO 编号
- 步骤 2. 激活端点后，模块便开始解码发送到该端点的令牌，并在收到的令牌有效的情况下回复有效握手信号。

34.4.6.6 端点停用

背景信息

本节介绍停用现有端点所需的步骤。

操作步骤

- 步骤 1. 在要停用的端点中，将 USBHS_DIEPCTLx 寄存器（对于 IN 或双向端点）或 USBHS_DOEPCTLx 寄存器（对于 OUT 或双向端点）中的 USB 活动端点位清零。
- 步骤 2. 停用端点后，模块便会忽略发送到该端点的令牌，从而导致 USB 超时。

34.5 操作模型

34.5.1 SETUP 和 OUT 数据传输

本节介绍了数据 OUT 传输和 SETUP 事务期间的内部数据流和应用程序操作步骤。

数据包读取

介绍如何从接收 FIFO 读取数据包 (OUT 数据和 SETUP 数据包)。

1. 捕获到 RXFNE 中断 (USBHS_GINTSTS 寄存器) 时, 应用程序必须读取接收状态弹出寄存器 (USBHS_GRXSTSP)。
2. 应用程序可以通过写入 RXFNE=0 (在 USBHS_GINTMSK 中) 来屏蔽 RXFNE 中断 (在 USBHS_GINTSTS 中), 直到它把数据包从接收 FIFO 中读取出来。
3. 如果已接收数据包的字节计数不是 0, 则从接收数据 FIFO 中弹出这些数据并存储在存储器中。如果接收到的数据包字节计数为 0, 则不会从接收数据 FIFO 中弹出任何数据。
4. 从接收 FIFO 读出的数据包状态有以下几种状态:
 - 全局 OUT NAK:
PKTSTS=全局 OUT NAK, BCNT=0x000, EPNUM 和 DPID 的值无关紧要。
这些数据表示全局 OUT NAK 位已生效。
 - SETUP 数据包:
PKTSTS=SETUP, BCNT=0x008, EPNUM=控制 EP 编号, DPID=D0。这些数据表示指定端点上收到的 SETUP 数据包现在可从接收 FIFO 中读取。
 - 建立阶段完成:
PKTSTS=建立阶段完成, BCNT=0x0, EPNUM=控制 EP 编号, DPID 值无关紧要。
这些数据表示指定端点的建立阶段完成并且数据阶段已启动。在此状态条目从接收 FIFO 中弹出后, 模块将在该控制 OUT 端点上产生建立中断。
 - OUT 数据包:
PKTSTS=DataOUT, BCNT=接收的 OUT 数据包的大小 (BCNT:0~1024), EPNUM=收到数据包的端点编号, DPID=实际数据 PID。
 - 数据传输完成:
PKTSTS=OUT 数据传输完成, BCNT=0x0, EPNUM=完成数据传输的 OUT EP 编号, DPID 值无关紧要。
这些数据表示指定 OUT 端点的 OUT 数据传输完成。在此状态条目从接收 FIFO 中弹出后, 模块将在指定的 OUT 端点上引发“传输完成”中断。
5. 从接收 FIFO 中弹出数据后, 必须取消对 RXFNE 中断的屏蔽 (USBHS_GINTSTS)。
6. 每次应用程序检测到 USBHS_GINTSTS 中的 RXFNE 中断时, 都将重复步骤 1 到 5。读取空的接收 FIFO 可能导致未定义的模块行为。

SETUP 事务

介绍模块处理 SETUP 数据包的方式以及应用程序处理 SETUP 事务的顺序。

应用程序要求:

1. 要接收 SETUP 数据包, 必须将控制 OUT 端点 STUPCNT 字段 (USBHS_DOEPTSIZE) 编程为非零值。如果应用程序将 STUPCNT 字段编程为非零值, 模块会接收 SETUP 数据包并将其写入接收 FIFO, 而不考虑 NAK 状态和 USBHS_DOEPCTLx 中的 EPENA 位设置。控制端点每收到一个 SETUP 数据包后, STUPCNT 字段都会递减。如果在接收 SETUP 数据包之前, 未将 STUPCNT 字段编程为适当值, 模块仍能接收 SETUP 数据包并使 STUPCNT 字段递减, 但应用程序可能无法确定在控制传输的建立阶段中接收的 SETUP 数据包正确数量。
 - 在 USBHS_DOEPTSIZE 中, STUPCNT=3

2. 应用程序必须始终在接收数据 FIFO 中分配一些额外空间，以便能够在控制端点上接收连续的最多三个 SETUP 数据包。
 - 预留空间 10 个字。第一个 SETUP 数据包需要 3 个字，“建立阶段完成”状态双字需要 1 个字，还需要 6 个字以存储两个额外的 SETUP 数据包。
 - 每个 SETUP 数据包需要 3 个字以存储 8 个字节的 SETUP 数据和 4 个字节的 SETUP 状态。模块将在接收 FIFO 中保留这些空间。
 - 这段 FIFO 仅用于存储 SETUP 包，绝对不会将该空间用于数据包。
3. 应用程序必须从接收 FIFO 中读取 SETUP 数据包的 2 个字。
4. 应用程序必须从接收 FIFO 中读取并丢弃“建立阶段完成”状态字。

内部数据流：

1. 接收到 SETUP 数据包时，模块会将接收到的数据写入接收 FIFO，而不会检查接收 FIFO 中的可用空间，且不考虑端点的 NAK 和 STALL 位设置。
 - 模块会在内部将接收到 SETUP 数据包的控制 IN/OUT 端点的 IN NAK 和 OUTNAK 位置 1。
2. USB 上接收到的每个 SETUP 数据包，模块会将 3 个字的数据写入接收 FIFO，并且将 STUPCNT 字段递减 1。
 - 第一个字包含由模块所使用的内部控制信息
 - 第二个字包含 SETUP 命令的前 4 个字节
 - 第三个字包含 SETUP 命令的最后 4 个字节
3. 当建立阶段结束，数据 IN/OUT 阶段开始时，模块会将一个状态条目（“建立阶段完成”字）写入接收 FIFO，指示建立阶段完成。
4. 在 AHB 端，SETUP 数据包被应用程序读取。
5. 当应用程序从接收 FIFO 中弹出“建立阶段完成”字时，模块将使用 STUP 中断 (USBHS_DOEPINTx) 来中断应用程序，指示其可以处理接收到的 SETUP 数据包。
 - 模块会将控制 OUT 端点的端点使能位清零。

应用程序编程顺序：

1. 对 USBHS_DOEPTSIZEx 寄存器进行编程。
 - STUPCNT=3
2. 等待 RXFNE 中断 (USBHS_GINTSTS) 并且从接收 FIFO 中读取数据包。
3. STUP 中断的触发 (USBHS_DOEPINTx) 表示 SETUP 数据传输成功完成。
 - 发生该中断时，应用程序必须读取 USBHS_DOEPTSIZEx 寄存器以确定接收的 SETUP 数据包数量并处理最后接收的 SETUP 数据包。

处理三个以上连续的 SETUP 数据包：

根据 USB2.0 规范，在 SETUP 数据包错误中，主机通常不会向同一个端点发送 3 个以上连续的 SETUP 数据包。但是，USB2.0 规范并未限制主机可以向同一个端点发送的连续 SETUP 数据包数量。发生这种情况时，USBHS 控制器将生成中断 (USBHS_DOEPINTx 中的 B2BSTUP)。

将全局 OUT NAK 置 1

内部数据流：

1. 如果应用程序将全局 OUT NAK (USBHS_DCTL 中的 SGONAK 位) 置 1，模块将停止向接收 FIFO 中写入 SETUP 数据包以外的数据。无论接收 FIFO 中可用空间大小如何，设备都会对主机发送的非同步 OUT 令牌回复 NAK，而对同步 OUT 数据包直接予以忽略。
2. 模块将全局 OUT NAK 写入接收 FIFO。应用程序必须为此留出足够空间。
3. 当应用程序从接收 FIFO 中弹出全局 OUT NAK 字时，模块会将 GONAKEFF 中断 (USBHS_GINTSTS) 置 1。

- 应用程序检测到该中断后，会认为模块处于全局 OUT NAK 模式。应用程序可以通过将 USBHS_DCTL 中的 SGONAK 位清零来清除该中断。

应用程序编程顺序：

- 要停止接收任何类型的数据到接收 FIFO 中，应用程序必须通过编程以下字段以将全局 OUT NAK 位置 1。
 - 在 USBHS_DCTL 中，SGONAK=1
- 等待 USBHS_GINTST 中的 GONAKEFF 中断。一旦被触发，该中断表示模块已停止接收 SETUP 数据包以外的任何类型数据。
 - 如果应用程序已将 USBHS_DCTL 中的 SGONAK 位置 1，则在模块引发 GONAKEFF 中断 (USBHS_GINTSTS) 之前，应用程序可以接收有效 OUT 数据包。
- 应用程序可通过对 USBHS_GINTMSK 寄存器中的 GINAKEFFM 位执行写操作来暂时屏蔽此中断。
 - 在 USBHS_GINTMSK 寄存器中，GINAKEFFM=0
- 当应用程序准备退出全局 OUT NAK 模式时，必须将 USBHS_DCTL 中的 SGONAK 位清零。此操作还会清除 GONAKEFF 中断 (USBHS_GINTSTS)。
 - 在 CGONAK 中，USBHS_DCTL=1
- 如果应用程序在之前已屏蔽此中断，则必须按以下方式取消对该中断的屏蔽：
 - 在 GINTMSK 中，GINAKEFFM=1

将全局 OUT NAK 置 1

应用程序必须使用以下顺序禁止已使能的 OUT 端点。

应用程序编程顺序：

- 禁止任何 OUT 端点前，应用程序必须在模块中使能全局 OUT NAK 模式。
 - 在 USBHS_DCTL 中，SGONAK=1
- 等待 GONAKEFF 中断 (USBHS_GINTSTS)
- 通过编程以下字段来禁止 OUT 端点：
 - 在 USBHS_DOEPCTLx 中，EPDIS=1
 - 在 USBHS_DOEPCTLx 中，SNAK=1
- 等待 EPDISD 中断 (USBHS_DOEPINTx)，该中断表示已完全禁止 OUT 端点。引发 EPDISD 中断时，模块还会将以下位清零：
 - 在 USBHS_DOEPCTLx 中，EPDIS=0
 - 在 USBHS_DOEPCTLx 中，EPENA=0
- 应用程序必须将全局 OUT NAK 位清零，以开始从其它未禁止的 OUT 端点接收数据。
 - 在 USBHS_DCTL 中，SGONAK=0

通用非同步 OUT 数据传输

本节介绍一种常规非同步 OUT 数据传输（控制、批量或中断）。

应用程序要求：

- 建立 OUT 传输前，应用程序必须在存储器中分配一个缓冲区，以容纳要作为 OUT 传输的一部分而接收的所有数据。
- 对于 OUT 传输，端点的传输大小寄存器中的传输大小字段必须是端点的最大数据包大小的倍数（且以字对齐）。
 - 传输大小[EPNUM] = $n \times (\text{MPSIZ}[\text{EPNUM}] + 4 - (\text{MPSIZ}[\text{EPNUM}] \bmod 4))$
 - 数据包计数[EPNUM] = n
 - $n > 0$
- 发生 OUT 端点中断时，应用程序必须读取端点的传输大小寄存器以计算存储器中有效数据量。接收的有效数据量可能小于编程的传输大小。

- 存储器中的有效数据量 = 应用程序设置的初始传输量 - 模块更新后的剩余传输量
- 接收到 USB 数据包数 = 应用程序设置的初始数据包数 - 模块更新后的剩余数据包数

内部数据流：

1. 应用程序必须在端点相关寄存器中设置传输大小和数据包计数字段，将 NAK 位清零，并使能端点来接收数据。
2. NAK 位清零后，模块便开始接收数据并将数据写入接收 FIFO（只要接收 FIFO 中有空间）。对于 USB 上接收的每个数据包，数据包及其状态都会写入接收 FIFO。写入接收 FIFO 的每个数据包（数据量达到最大数据包大小的数据包或短数据包）都会使该端点的数据包计数字段递减 1。
 - 收到的数据包若 CRC 无效，则自动被从接收 FIFO 中清除。
 - 在 USB 上为数据包回复 ACK 后，模块将丢弃主机因无法检测到 ACK 而重新发送的非同步 OUT 数据包。应用程序不会在具有相同数据 PID 的相同端点上检测到多个连续的 OUT 数据包。在这种情况下，数据包计数不会递减。
 - 如果接收 FIFO 中没有空间，则会忽略同步或非同步数据包并且不会将它们写入接收 FIFO。此外，非同步 OUT 令牌将会收到 NAK 握手应答。
 - 在上述所有三种情况中，数据包计数都不会递减，因为没有任何数据写入接收 FIFO。
3. 当数据包计数变为 0 或者在端点上接收到短数据包时，该端点的 NAK 位将置 1。NAK 位置 1 后，将忽略同步或非同步数据包并且不会将它们写入接收 FIFO，同时非同步 OUT 令牌会收到 NAK 握手应答。
4. 在数据写入接收 FIFO 后，应用程序将从接收 FIFO 中读取数据并将数据写入外部存储器，一次一个数据包，逐个端点过来。
5. 在 AHB 上向外部存储器写入完每个数据包后，端点的传输大小都会自动减去该数据包的大小。
6. 在以下情况时，OUT 端点的 OUT 数据传输完成状态将写入接收 FIFO：
 - 传输大小为 0 并且数据包计数为 0
 - 写入接收 FIFO 的最后一个 OUT 数据包是短数据包
(数据包大小:0~最大数据包大小-1)
7. 当应用程序弹出此状态条目（OUT 数据传输完成），并生成该端点的传输完成中断，同时清零端点使能位。

应用程序编程顺序：

1. 使用传输大小和相应数据包个数对 USBHS_DOEPTSTZx 寄存器进行编程。
2. 使用端点特性对 USBHS_DOEPCTLx 寄存器进行编程，并将 EPENA 和 CNAK 位置 1。
 - 在 USBHS_DOEPCTLx 中，EPENA=1
 - 在 USBHS_DOEPCTLx 中，CNAK=1
3. 等待 RXFNE 中断（在 USBHS_GINTSTS 中）并且从接收 FIFO 中读走数据包。
 - 此步骤可重复多次，具体取决于传输大小。
4. 触发 XFRC 中断（USBHS_DOEPINTx），以表示非同步 OUT 数据传输成功完成。
5. 读取 USBHS_DOEPTSTZx 寄存器，以确定有效数据量。

通用同步 OUT 数据传输

本节介绍常规的同步 OUT 数据传输。

应用程序要求：

1. 非同步 OUT 数据传输的所有应用程序要求均适用于同步 OUT 数据传输。
2. 对于同步 OUT 数据传输中的传输大小和数据包计数字段，必须始终将其设置为单个帧中可接收的最大数据包大小的数据包数目。同步类型的 OUT 数据传输事务必须在一个帧内完成。
3. 在周期性帧结束（USBHS_GINTSTS 中的 EOPF 中断）之前，应用程序必须从接收 FIFO 中读取所有同步 OUT 数据包（数据条目和状态条目）。

4. 要接收下一帧中的数据，必须在 EOPF (USBHS_GINTSTS) 之后 SOF (USBHS_GINTSTS) 之前使能一个同步 OUT 端点。

内部数据流：

1. 同步 OUT 端点的内部数据流与非同步 OUT 端点的基本相同，但稍有差异。
2. 同步 OUT 端点通过将端点使能位置 1 并将 NAK 位清零来使能时，必须相应地将偶数/奇数帧位置 1。仅当符合以下条件时，模块才会在同步 OUT 端点上接收特定帧中的数据：
 - EONUM (在 USBHS_DOEPCTLx 中) = FNSOF[0] (在 USBHS_DSTS 中)
3. 当应用程序从接收 FIFO 中完整地读取一个同步 OUT 数据包 (数据和状态) 时，模块会根据从接收 FIFO 中读取的最后一个同步 OUT 数据包的数据 PID 更新 USBHS_DOEPSIZx 中的 RXDPID 字段。

应用程序编程顺序：

1. 使用传输大小和相应数据包计数对 USBHS_DOEPSIZx 寄存器进行编程。
2. 使用端点特性对 USBHS_DOEPCTLx 寄存器进行编程，并将端点使能位、清除 NAK 位和奇数/偶数帧位置 1。
 - EPENA1
 - CNAK=1
 - EONUM= (0: 偶数/1: 奇数)
3. 等待 RXFNE 中断 (在 USBHS_GINTSTS 中) 并且从接收 FIFO 中读走数据包。
 - 此步骤可重复多次，具体取决于传输大小。
4. XFRC 中断 (在 USBHS_DOEPINTx 中) 表示同步 OUT 数据传输完成。该中断不一定意味着存储器中的数据是有效的。
5. 对于同步 OUT 传输，应用程序可能并不总会检测到该中断。相反，应用程序可能检测到 USBHS_GINTSTS 中的 IISOXFRM 中断。
6. 读取 USBHS_DOEPSIZx 寄存器以确定接收的传输大小以及确定帧中接收的数据的有效性。仅当符合以下条件之一时，应用程序才必须将存储器中接收的数据视为有效数据：
 - RXDPID=D0 (在 USBHS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=1
 - RXDPID=D1 (在 USBHS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=2
 - RXDPID=D2 (在 USBHS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=3

接收该有效数据的 USB 数据包数量=应用程序编程的初始数据包个数-模块更新后的剩余数据包个数。

应用程序可将无效数据包丢弃。

不完整的同步 OUT 数据传输

本节介绍了同步 OUT 数据包出现丢包时应用程序编程顺序。

内部数据流：

1. 对于同步 OUT 端点，可能不会始终引发 XFRC 中断 (在 USBHS_DOEPINTx 中)。如果模块丢弃同步 OUT 数据包，则在以下情况下，应用程序可能无法检测到 XFRC 中断 (USBHS_DOEPINTx)：
 - 在接收 FIFO 无法容纳完整的 ISO OUT 数据包时，模块将丢弃接收到的 ISO OUT 数据
 - 接收到的同步 OUT 数据包存在 CRC 错误
 - 模块接收到的同步 OUT 令牌损坏
 - 应用程序从接收 FIFO 中读取数据的速度非常缓慢
2. 如果模块在所有同步 OUT 端点的传输完成前检测到周期性帧结束，将触发未完成同步 OUT 数据中断 (USBHS_GINTSTS 中的 IISOXFRM)，指示至少有一个同步 OUT 端点上未触发 XFRC 中断 (在 USBHS_DOEPINTx 中)。此时，未完成传输的端点仍保持使能，但在 USB 的该端点上，没有进行中的有效传输。

应用程序编程顺序:

1. 硬件触发 IISOXFRM 中断 (USBHS_GINTSTS) 表示当前帧中至少有一个同步 OUT 端点具有未完成的传输。
2. 如果因未从端点完全读取同步 OUT 数据而发生这种情况, 应用程序必须确保首先从接收 FIFO 读取走所有同步 OUT 数据 (包括数据条目和状态条目), 然后再继续处理。
 - 从接收 FIFO 读取所有数据后, 应用程序即可检测到 XFRC 中断 (USBHS_DOEPINTx)。在此情况下, 应用程序必须重新使能端点以接收下一个帧中的同步 OUT 数据。
3. 当应用程序接收到 IISOXFRM 中断 (在 USBHS_GINTSTS 中) 时, 应用程序必须读取所有同步 OUT 端点的控制寄存器 (USBHS_DOEPCTLx), 以确定哪些端点在当前帧中具有不完整的传输。同时满足以下两个条件时, 表示端点传输未完成:
 - EONUM 位 (在 USBHS_DOEPCTLx 中) = FNSOF[0] (在 USBHS_DSTS 中)
 - EPENA=1 (在 USBHS_DOEPCTLx 中)
4. 在检测到 SOF 中断 (在 USBHS_GINTSTS 中) 前, 必须执行完成上一步操作, 以确保当前帧编号未发生变更。
5. 对于具有不完整传输的同步 OUT 端点, 应用程序必须丢弃存储器中的数据, 并通过将 USBHS_DOEPCTLx 中的 EPDIS 位置 1 来禁止端点。
6. 等待 EPDIS 中断 (在 USBHS_DOEPINTx 中), 并且使能端点以在下一帧中接收新数据。
 - 由于模块可能需要一些时间才能禁止端点, 因此应用程序在接收到无效同步数据后, 可能无法接收下一个帧中的数据。

停止非同步 OUT 端点

本节介绍应用程序如何才能停止非同步端点。

1. 将模块置于全局 OUT NAK 模式。
2. 禁止所需的端点
 - 禁止端点时, 请设置 STALL=1 (在 USBHS_DOEPCTL 中), 而不是将 USBHS_DOEPCTL 中的 SNAK 位置 1。STALL 位的优先级始终高于 NAK 位。
3. 当应用程序不再需要端点回复 STALL 握手信号时, 必须将 STALL 位 (在 USBHS_DOEPCTLx 中) 清零。
4. 如果应用程序由于收到主机的 SetFeature.Endpoint Halt 或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态, 则必须在该控制端点上的状态阶段传输前, 将 STALL 位置 1 或清零。

34.5.2 IN 数据传输

数据包写入

本节介绍在已使能专用发送 FIFO 的情况下应用程序如何将数据包写入端点 FIFO。

1. 应用程序可以选择轮询模式或中断模式。
 - 在轮询模式下, 应用程序通过读取 USBHS_DTXFSTSx 寄存器来监视端点发送数据 FIFO 的状态, 从而确定数据 FIFO 中是否有足够空间。
 - 在中断模式下, 应用程序等待 TXFE 中断 (在 USBHS_DIEPINTx 中), 然后读取 USBHS_DTXFSTSx 寄存器以确定数据 FIFO 中是否有足够空间。
 - 要写入单个非零长度的数据包, 数据 FIFO 中必须有足够的空间来容纳整个数据包。
 - 要写入零长度的数据包, 应用程序不能查看 FIFO 空间。
2. 如果使用上述方法之一, 当应用程序确定有足够空间来写入发送数据包时, 应用程序必须首先对端点控制寄存器进行相应写操作, 然后再将数据写入数据 FIFO。通常, 应用程序必须对 USBHS_DIEPCTLx 寄存器执行读-修改-写操作, 以避免在将端点使能位置 1 的同时, 修改寄存器中的其它内容。

如果有足够空间，应用程序可将同一端点的多个数据包写入发送 FIFO。对于周期性 IN 端点，应用程序只能一次写入一个帧内的多个数据包。只有先前一个帧的通信事务传输完成之后，应用程序才会写入下一个帧内要发送的所有数据包。

将 IN 端点 NAK 置 1

内部数据流：

1. 当应用程序将特定端点的 IN NAK 置 1 时，模块将停止端点上的数据发送，而不考虑端点发送 FIFO 中的数据是否可用。
2. 非同步端点收到 IN 令牌，回复 NAK 握手应答。
 - 同步端点收到 IN 令牌，返回零长度数据包
3. 模块在 USBHS_DIEPINTx 中触发 INEPNE (IN 端点 NAK 有效) 中断以响应 USBHS_DIEPCTLx 中的 SNAK 位。
4. 应用程序检测到该中断后，便会认为端点处于 IN NAK 模式。应用程序可通过将 USBHS_DIEPCTLx 中的 CNAK 位置 1 来清除该中断。

应用程序编程顺序：

1. 要在特定 IN 端点上停止发送任何数据，应用程序必须将 IN NAK 位置 1。要将该位置 1，必须编程以下字段。
 - USBHS_DIEPCTLx 中的 SNAK=1
2. 等待 USBHS_DIEPINTx 中的 INEPNE 中断触发。该中断表示模块已在端点上停止发送数据。
3. 在应用程序将 NAK 位置 1 但“NAK 有效”中断尚未触发时，模块可以在端点上发送有效 IN 数据。
4. 应用程序可通过写入 DIEPMSK 中的 INEPNEM 位来临时屏蔽该中断。
 - 在 DIEPMSK 中，INEPNEM = 0
5. 要退出端点 NAK 模式，应用程序必须将 USBHS_DIEPCTLx 中的 NAK 状态位 (NAKSTS) 清零。此操作还会清除 INEPNE 中断 (在 USBHS_DIEPINTx 中)。
 - 在 USBHS_DIEPCTLx 中，CNAK=1
6. 如果应用程序已将该中断屏蔽，则必须按以下方式取消屏蔽：
 - 在 DIEPMSK 中，INEPNEM=1

禁止 IN 端点

使用以下顺序来禁止先前已使能的特定 IN 端点。

应用程序编程顺序：

1. 应用程序必须先停止在 AHB 上写入数据，之后才能禁止 IN 端点。
2. 应用程序必须将端点设置为 NAK 模式。
 - USBHS_DIEPCTLx 中的 SNAK=1
3. 等待 USBHS_DIEPINTx 中的 INEPNE 中断。
4. 将必须禁止的端点的 USBHS_DIEPCTLx 寄存器中的以下位置 1。
 - USBHS_DIEPCTLx 中的 EPDIS=1
 - USBHS_DIEPCTLx 中的 SNAK=1
5. USBHS_DIEPINTx 中的 EPDISD 中断的触发表示模块已完全禁止指定的端点。在触发中断的同时，模块还会将以下位清零：
 - 在 USBHS_DIEPCTLx 中，EPENA=0
 - 在 USBHS_DIEPCTLx 中，EPDIS=0
6. 应用程序必须为周期性 IN EP 读取 USBHS_DIEPTSIZx 寄存器，以计算端点上有多少数据是在 USB 上发送的。
7. 应用程序必须通过将 USBHS_GRSTCTL 寄存器中的以下字段置 1，来清空端点发送 FIFO 中的数据：

- TXFNUM (在 USBHS_GRSTCTL 中) = 端点发送 FIFO 编号
- TXFFLSH (在 USBHS_GRSTCTL 中) = 1

应用程序必须轮询 USBHS_GRSTCTL 寄存器, 直至模块将 TXFFLSH 位清零, 这表示 FIFO 清空操作结束。要在该端点上发送新数据, 应用程序可以在稍后重新使能该端点。

通用非周期性 IN 数据传输

应用程序要求:

1. 建立 IN 传输前, 应用程序必须确保组成一次 IN 传输的每个数据包都可以容纳在单个缓冲区中。
2. 对于 IN 传输, 端点传输大小寄存器中的传输大小字段表示本次传输的有效数据量, 它由多个最大数据包大小和单个短数据包组成。该短数据包在传输结束时发送。
 - 要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包:
 传输大小[EPNUM] = $x \times \text{MPSIZ}[\text{EPNUM}] + \text{sp}$
 如果 ($\text{sp} > 0$), 数据包计数[EPNUM] = $x + 1$ 。
 否则, 数据包计数[EPNUM] = x
 - 要发送单个零长度数据包:
 传输大小[EPNUM] = 0
 数据包计数[EPNUM] = 1
 - 要发送多个最大数据包大小的数据包并在传输结束时外加一个零长度数据包, 应用程序必须将传输拆分为两个部分。
 第一部分发送最大数据包大小的数据包, 第二部分仅发送零长度数据包。
 第一次传输: 传输大小[EPNUM] = $x \times \text{MPSIZ}[\text{epnum}]$; 数据包计数 = n ;
 第二次传输: 传输大小[EPNUM] = 0; 数据包计数 = 1;
3. 使能某个端点进行数据传输后, 模块会更新传输大小寄存器。在 IN 传输结束时, 应用程序必须读取传输大小寄存器, 以确定送入发送 FIFO 中的数据已有多少通过 USB 发送出去。
4. 送入发送 FIFO 中的数据量 = 应用程序编程的初始传输大小 - 模块更新后的最终传输大小
 - 通过 USB 已经发送的数据量 = (应用程序编程的初始数据包计数 - 模块更新后的最终数据包计数) $\times$ $\text{MPSIZ}[\text{EPNUM}]$
 - 要通过 USB 发送的剩余数据量 = (应用程序编程的初始传输大小 - 已通过 USB 发送的数据量)

内部数据流:

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段, 并使能该端点来发送数据。
2. 应用程序还必须向该端点的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包, 该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO, 直到该端点的传输大小变为 0。向 FIFO 写入数据后, “FIFO 中的数据包数”计数会递增 (这是一个 3 位计数, 由模块在内部进行维护, 每个 IN 端点发送 FIFO 对应一个。在 IN 端点 FIFO 中, 模块所维护的最大数据包数始终为八个)。对于零长度数据包, 每个 FIFO 均另有一个单独的标志, FIFO 中没有任何数据。
4. 当数据写入发送 FIFO 后, 模块会在接收到 IN 令牌时将这些数据送出。每个数据包发送出去并收到回复的 ACK 握手信号后, 该端点的数据包计数都会递减 1, 直到数据包计数变 0 为止。发生超时, 数据包计数不会递减。
5. 对于零长度数据包 (由内部零长度标志指示), 模块会针对 IN 令牌发出一个零长度数据包, 并递减数据包计数字段的值。
6. 如果接收到 IN 令牌的端点对应的 FIFO 中无数据, 且该端点的数据包计数字段为零, 则模块会针对该端点生成一个“TxFIFO 为空时接收到 IN 令牌” (ITTXFE) 中断 (前提是该端点的 NAK 位未置 1)。模块在该非同步端点上回复 NAK 握手信号。
7. 模块会在内部使 FIFO 指针重新返回到开头, 并且不会生成超时中断。

8. 当传输大小为 0 且数据包计数为 0 时，将生成该端点的传输完成 (XFRC) 中断，同时将端点使能清零。

应用程序编程顺序：

1. 使用传输大小和相应数据包计数对 USBHS_DIEPTSIZE_x 寄存器进行编程。
2. 使用端点特性对 USBHS_DIEPTCTL_x 寄存器进行编程，并将 CNAK 和 EPENA（端点使能）位置 1。
3. 发送非零长度数据包时，应用程序必须轮询 USBHS_DTXFSTS_x 寄存器（其中 x 为与该端点相关联的 FIFO 编号）以确定数据 FIFO 中是否有足够的空间。写入数据前，应用程序也可选用 TXFE 位（在 USBHS_DIEPTINT_x 中）。

通用周期性 IN 数据传输

本节介绍典型的周期性 IN 数据传输。

应用程序要求：

1. 通用非周期性 IN 数据传输的应用程序要求 1、2、3、4 对周期性 IN 数据传输同样适用（只是对要求 2 稍加修改）。
 - 应用程序只能发送若干个最大数据包大小的数据包或若干个最大数据包大小的包，外加传输结束时的一个短数据包。
要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包，必须满足以下条件：
传输大小[EPNUM] = $x \times \text{MPSIZE}[\text{EPNUM}] + \text{sp}$
（其中 x 是大于 0 的整数，且 sp 范围是 0~MPSIZE[EPNUM]-1）
如果 (sp>0)，数据包计数[EPNUM] = $x + 1$
否则，数据包计数[EPNUM] = x；
MCNT[EPNUM] = 数据包计数[EPNUM]
 - 应用程序无法在传输结束时发送零长度数据包。应用程序可以单独发送一个零长度数据包。
 - 要发送单个零长度数据包：
传输大小[EPNUM]=0
数据包计数[EPNUM]=1
MCNT[EPNUM]=数据包计数[EPNUM]
2. 应用程序一次只能安排一帧的数据传输。
 - $(\text{MCNT} - 1) \times \text{MPSIZE} \leq \text{XFERSIZE} \leq \text{MCNT} \times \text{MPSIZE}$
 - PKTCNT = MCNT（在 USBHS_DIEPTSIZE_x 中）
 - 如果 XFERSIZE < MCNT × MPSIZE，则传输的最后一个数据包为短数据包
 - 请注意：MCNT 位于 USBHS_DIEPTSIZE_x 中、MPSIZE 位于 USBHS_DIEPTCTL_x 中、PKTCNT 位于 USBHS_DIEPTSIZE_x 中、XFERSIZE 位于 USBHS_DIEPTSIZE_x 中
3. 接收到 IN 令牌前，应用程序必须将要在帧中发送的完整数据写入到发送 FIFO 中。在接收到 IN 令牌时，即使发送 FIFO 中该帧要发送的数据只差 1 个双字未写进来，模块也会执行 FIFO 为空时的操作。当发送 FIFO 为空时：
 - 同步端点上将回复零长度数据包
 - 中断端点上将回复 NAK 握手信号

内部数据流：

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。
2. 应用程序还必须向与该端点相关联的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。
4. 当周期性端点接收到 IN 令牌时，模块将开始发送 FIFO 中的数据（如果 FIFO 中有数据）。如果 FIFO 中没有该帧要发送的数据的完整数据包，则模块将为该端点生成一个“TxFIFO 为空时接收到 IN 令牌”中断。

- 同步端点上将回复零长度数据包
 - 中断端点上将回复 NAK 握手信号
5. 端点的数据包计数会在下列情况下递减 1:
- 对于同步端点, 发送一个零长度或非零长度的数据包时
 - 对于中断端点, 在发送 ACK 握手信号时递减
 - 当传输大小和数据包计数均为 0 时, 将生成该端点的传输完成中断, 同时将端点使能位清零。
6. 在“周期性帧间隔”(由 USBHS_DCFG 中的 PFIVL 位控制)内, 当模块发现任何在当前帧内应为空的同步 IN 端点 FIFO 中的数据还未发送完成时, 都会在 USBHS_GINTSTS 中生成一个 IISOIXFR 中断。

应用程序编程顺序:

1. 使用端点特性对 USBHS_DIEPCTLx 寄存器进行编程, 并将 CNAK 和 EPENA 位置 1。
2. 将需要在下一帧中发送的数据写入发送 FIFO。
3. 硬件触发 ITTXFE 中断 (在 USBHS_DIEPINTx 中) 表示应用程序尚未将需要发送的全部数据写入发送 FIFO。
4. 如果在检测到中断前已使能中断端点, 则将忽略该中断。如果中断端点未使能, 则使能此端点, 以便数据能够在收到下一次 IN 令牌时发送出去。
5. 硬件触发 XFRC 中断 (在 USBHS_DIEPINTx 中) 时如果 USBHS_DIEPINTx 中未产生 ITTXFE 中断, 则表示成功完成同步 IN 传输。读取 USBHS_DIEPTSIZx 寄存器时始终得到传输大小=0 且数据包计数=0, 则表示所有数据都已通过 USB 发送完毕。
6. 置位 XFRC 中断 (在 USBHS_DIEPINTx 中) 时无论是否产生 ITTXFE 中断 (在 USBHS_DIEPINTx 中), 都表示成功完成同步 IN 传输。读取 USBHS_DIEPTSIZx 寄存器时始终得到传输大小=0 且数据包计数=0, 则表示所有数据都已通过 USB 发送完毕。
7. 在 USBHS_GINTSTS 中置位未完成的同步 IN 传输 (IISOIXFR) 中断时如果未产生任何前述中断, 则表示在当前帧中模块至少未收到 1 个周期性的 IN 令牌。

未完成同步 IN 数据传输

本节介绍应用程序针对未完成同步 IN 数据传输必须执行的操作。

内部数据流:

1. 符合下列条件之一时, 即认为同步 IN 传输未完成:
 - 模块在至少一个同步 IN 端点上接收到损坏的同步 IN 令牌。此时, 应用程序检测到未完成同步 IN 传输中断 (USBHS_GINTSTS 中的 IISOIXFR 位)。
 - 应用程序向发送 FIFO 写入数据的速度过慢, 在将完整数据写入 FIFO 之前便接收到 IN 令牌。此时, 应用程序在 USBHS_DIEPINTx 中检测到“Tx FIFO 为空时接收到 IN 令牌”中断。应用程序可忽略此中断, 因为最终这将在周期性帧结束时产生一个未完成同步 IN 传输中断 (USBHS_GINTSTS 中的 IISOIXFR 位)。模块会通过 USB 发送一个零长度数据包来响应接收到的 IN 令牌。
2. 应用程序必须尽快停止向发送 FIFO 写入数据。
3. 应用程序必须将端点的 NAK 位和禁止位置 1。
4. 模块会禁止该端点, 将禁止位清零并触发端点的“端点禁止”中断。

应用程序编程顺序:

1. 应用程序可以在任何同步 IN 端点上忽略 USBHS_DIEPINTx 中的“Tx FIFO 为空时接收到 IN 令牌”中断, 因为最终这将产生一个未完成同步 IN 传输中断 (在 USBHS_GINTSTS 中)。
2. 硬件触发未完成同步 IN 传输中断 (在 USBHS_GINTSTS 中) 表示在至少一个同步 IN 端点上存在未完成的同步 IN 传输。
3. 应用程序必须读取所有同步 IN 端点的“端点控制”寄存器来检测存在未完成 IN 数据传输的端点。
4. 应用程序必须停止向与这些端点相关联的“周期性发送 FIFO”写入数据。
5. 对 USBHS_DIEPCTLx 寄存器中的下列字段进行编程以禁止端点:

- USBHS_DIEPCTLx 中的 SNAK=1
 - USBHS_DIEPCTLx 中的 EPDIS=1
6. 硬件触发 USBHS_DIEPINTx 中的“端点禁止”中断表示模块已禁止该端点。
- 此时，应用程序必须清空相关联的发送 FIFO 中的数据，或者通过在下一帧中使能新传输的端点来覆盖 FIFO 中的现有数据。要刷新数据，应用程序必须使用 USBHS_GRSTCTL 寄存器。

停止非同步 IN 端点

本节介绍应用程序如何才能停止非同步端点。

应用程序编程顺序：

1. 禁止要停止的 IN 端点。同时将 STALL 位置 1。
2. USBHS_DIEPCTLx 中的 EPDIS=1（当端点已使能时）
 - USBHS_DIEPCTLx 中的 STALL=1
 - STALL 位的优先级始终高于 NAK 位
3. 硬件触发“端点禁止”中断（在 USBHS_DIEPINTx 中）可以让应用程序知道模块已禁止指定端点。
4. 应用程序必须根据端点类型清空非周期性或周期性发送 FIFO。对于非周期性端点，应用程序必须重新使能另一个无需停止的非周期性端点来发送数据。
5. 当应用程序准备好结束该端点的 STALL 握手信号时，必须将 USBHS_DIEPCTLx 的 STALL 位清零。
6. 如果应用程序因收到来自主机的 SetFeature.Endpoint Halt 命令或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点的状态阶段传输之前将 STALL 位置 1 或清零。

特例：停止控制 OUT 端点

如果在控制传输的数据阶段，主机发送的 IN/OUT 令牌数超过 SETUP 数据包指定的值，则模块必须对这些多余的 IN/OUT 令牌回复 STALL。在这种情况下，应用程序必须在控制传输的数据阶段使能 USBHS_DIEPINTx 的 ITTXFE 中断和 USBHS_DOEPINTx 的 OTEPDIS 中断（当模块已完成传输 SETUP 数据包指定的数据量后）。随后，当应用程序收到此中断时，必须将相应端点控制寄存器中的 STALL 位置 1 并清除此中断。

34.6 寄存器

应用程序通过 AHB 从接口对控制和状态寄存器进行读写操作，以此来控制 USBHS 模块。USBHS 模块所有寄存器为 32 位寄存器，其地址按 32 位对齐，因此只能以 32 位的方式访问。

控制和状态寄存器分为以下几类：

- USBHS 系统控制寄存器
- 模块全局寄存器
- 主机模式寄存器
- 设备模式寄存器
- 电源和时钟门控控制寄存器
- 数据 FIFO(DFIFO)访问寄存器

其中 USBHS 系统控制寄存器与其它寄存器基地址不同，该寄存器独立于 USBHS 模块以控制 USBHS 模块的相关设定。

只有模块全局寄存器、电源和时钟门控控制寄存器和数据 FIFO 访问寄存器可以在主机和设备模式下进行访问。当 USBHS 模块处以一种模式（主机或者设备）下，应用程序不得以另外一种角色模式访问寄存器，比如主机模式时，访问设备模式的寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器 USBHS.GINTSTS.NMIS 位反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。

控制状态寄存器存储器映射

主机和设备模式寄存器占用不同的地址。所有寄存器均在 AHB 时钟域内实现。

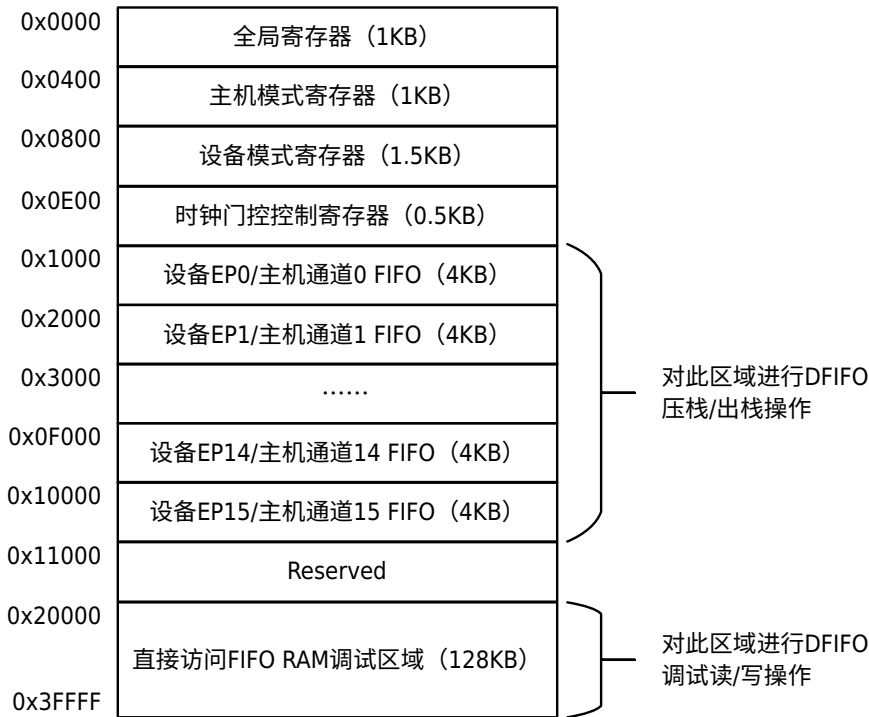


图 34-7 USBHS 控制状态寄存器存储器映射

34.6.1 寄存器总表

USB 系统控制寄存器基地址：0x40055400

表 34-3 USB 系统控制寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	USB_SYCTLREG	USB 系统控制寄存器	32

USBHS 模块寄存器基地址：0x400C0000

表 34-4 USB2.0 高速模块 USBHS 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	USBHS_GVBUSCFG	VBUS 控制寄存器	32
0x08	USBHS_GAHBCFG	AHB 控制寄存器	32
0x0C	USBHS_GUSBCFG	USB 配置寄存器	32
0x10	USBHS_GRSTCTL	复位寄存器	32
0x14	USBHS_GINTSTS	模块中断寄存器	32
0x18	USBHS_GINTMSK	中断屏蔽寄存器	32
0x1C	USBHS_GRXSTSR	接收状态调试读取/状态读取和出栈寄存器	32
0x20	USBHS_GRXSTSP		32
0x24	USBHS_GRXFSIZ	接收 FIFO 大小寄存器	32
0x28	USBHS_HNPTXFSIZ	主机非周期性发送 FIFO 大小寄存器	32
0x28	USBHS_DIEPRXF0	设备端点 0 发送 FIFO 大小寄存器	32
0x2C	USBHS_HNPTXSTS	非周期性发送 FIFO/队列状态寄存器	32
0x3C	USBHS_CID	模块 ID 寄存器	32
0x54	USBHS_GLPMCFG	LPM 配置寄存器	32
0x100	USBHS_HPTXFSIZ	主机周期性发送 FIFO 大小寄存器	32
0x100+0x4*x	USBHS_DIEPTXFx	设备 IN 端点 n 发送 FIFO 大小寄存器 (x=1~15)	32
0x400	USBHS_HCFG	主机配置寄存器	32
0x404	USBHS_HFIR	主机帧时间间隔寄存器	32
0x408	USBHS_HFNUM	主机帧编号/帧剩余时间间隔寄存器	32
0x410	USBHS_HPTXSTS	主机周期性发送 FIFO/队列状态寄存器	32
0x414	USBHS_HAINT	主机全体通道中断寄存器	32
0x418	USBHS_HAINTMSK	主机全体通道中断屏蔽寄存器	32

偏移地址	寄存器	描述	位宽
0x440	USBHS_HPRT	USBHS 主机端口控制和状态寄存器	32
0x500+0x20*x	USBHS_HCCHARx	主机通道 x 特性寄存器	32
0x504+0x20*x	USBHS_HCSPLTx	主机通道 x 分离控制寄存器	32
0x508+0x20*x	USBHS_HCINTx	主机通道 x 中断寄存器	32
0x50C+0x20*x	USBHS_HCINTMSKx	主机通道 x 中断屏蔽寄存器	32
0x510+0x20*x	USBHS_HCTSIZx	主机通道 x 传输大小寄存器	32
0x514+0x20*x	USBHS_HCDMAX	主机通道 x DMA 地址寄存器	32
0x800	USBHS_DCFG	设备配置寄存器	32
0x804	USBHS_DCTL	设备控制寄存器	32
0x808	USBHS_DSTS	设备状态寄存器	32
0x810	USBHS_DIEPMSK	设备 IN 端点通用中断屏蔽寄存器	32
0x814	USBHS_DOEPMSK	设备 OUT 端点通用中断屏蔽寄存器	32
0x818	USBHS_DAINTh	设备全体端点中断寄存器	32
0x81C	USBHS_DAINThMSK	设备全体端点中断屏蔽寄存器	32
0x830	USBHS_DTHRCTL	设备阈值控制寄存器	32
0x834	USBHS_DIEPEMPMSK	设备 IN 端点 FIFO 空中断屏蔽寄存器	32
0x838	USBHS_DEACHINT	设备单个端点中断寄存器	32
0x83C	USBHS_DEACHINTMSK	设备个单端点中断屏蔽寄存器	32
0x844	USBHS_DIEPEACHMSK1	设备 IN 端点 1 中断屏蔽寄存器	32
0x884	USBHS_DOEPEACHMSK1	设备 OUT 端点 1 中断屏蔽寄存器	32
0x900	USBHS_DIEPCTL0	设备 IN 端点 0 控制寄存器	32
0x900+0x20*x	USBHS_DIEPCTLx	设备 IN 端点 x 控制寄存器 (x=1~15)	32
0x908+0x20*x	USBHS_DIEPINTx	设备 IN 端点 x 中断寄存器	32
0x910	USBHS_DIEPTSIZE0	设备 IN 端点 0 传输大小寄存器	32
0x910+0x20*x	USBHS_DIEPTSIZEx	设备 IN 端点 x 传输大小寄存器 (x=1~15)	32
0x914+0x20*x	USBHS_DIEPDMAx	设备 IN 端点 x DMA 地址寄存器	32
0x918+0x20*x	USBHS_DTXFSTSx	设备 IN 端点 x 发送 FIFO 状态寄存器	32
0xB00	USBHS_DOEPCTLx	设备 OUT 端点 0 控制寄存器	32
0xB00+0x20*x	USBHS_DOEPCTLx	设备 OUT 端点 x 控制寄存器 (x=1~15)	32
0xB08+0x20*x	USBHS_DOEPINTx	设备 OUT 端点 x 中断寄存器	32

偏移地址	寄存器	描述	位宽
0xB10	USBHS_DOEPSIZ0	设备 OUT 端点 0 传输大小寄存器	32
0xB10+0x20*x	USBHS_DOEPSIZx	设备 OUT 端点 x 传输大小寄存器 (x=1~15)	32
0xB14+0x20*x	USBHS_DOEPDMAX	设备 OUT 端点 x DMA 地址寄存器	32
0xE00	USBHS_GCCTL	时钟门控控制寄存器	32

 **说明**

表中的 x 代表存在多个同类型寄存器，取值为 0~15。特殊标识的以寄存器行描述为准。

34.6.2 USB 系统控制寄存器

34.6.2.1 USB 系统控制寄存器 (USB_SYCTLREG)

Offset	Bit Position																																									
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
Reset	0x00000000																																									
Name	Reserved					USBHS_NFE		USBHS_NFS		Reserved					USBFS_NFE		USBFS_NFS		Reserved					USBHS_FSPHYE		USBHS_SOFEN		USBHS_DFB		Reserved					USBFS_SOFEN		USBFS_DFB					
Access						RW		RW							RW		RW							RW		USBHS_FSPHYE	RW		USBHS_SOFEN	RW							RW		USBFS_SOFEN	RW		USBFS_DFB

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26	USBHS_NFE	USBHS 滤波使能寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的开关 0b0：模拟滤波器关 0b1：模拟滤波器开，滤波范围请参考 USBHS_NFS 的设定	RW
25:24	USBHS_NFS	USBHS 滤波选择寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的滤波范围 0b00：滤波宽度档位 1 0b01：滤波宽度档位 2 0b10：滤波宽度档位 3 0b11：滤波宽度档位 4 各档位具体值请参考《数据手册》的“电气特性”章节中的“USB 片上全速 PHY STOP 模式下滤波特性”。	RW
23:19	Reserved	保留	-	-
18	USBFS_NFE	USBFS 滤波使能寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的开关 0b0：模拟滤波器关 0b1：模拟滤波器开，滤波范围请参考 USBFS_NFS 的设定	RW
17:16	USBFS_NFS	USBFS 滤波选择寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的滤波范围 0b00：滤波宽度档位 1 0b01：滤波宽度档位 2 0b10：滤波宽度档位 3 0b11：滤波宽度档位 4 各档位具体值请参考《数据手册》的“电气特性”章节中的“USB 片上全速 PHY STOP 模式下滤波特性”章节。	RW
15:11	Reserved	保留	-	-
10	USBHS_FSPHYE	USBHS 片上全速 PHY 使能位	0b0：USBHS 片上全速 PHY 挂起 0b1：USBHS 片上全速 PHY 使能	RW

位/位域	标记	位名	功能描述	读写
9	USBHS_SOFEN	USBHS SOF 脉冲输出使能位	USBHS 主机发出 SOF 或者设备成功接收到 SOF 时, 16 个 HCLK 时钟周期宽度的 SOF 脉冲从 PAD 输出使能 ● 0b0: SOF 脉冲不输出 ● 0b1: SOF 脉冲输出	RW
8	USBHS_DFB	USBHS VBUS/ID 管脚内部去抖滤波器旁路使能位	● 0b0: 模块内部去抖动滤波器有效 ● 0b1: 旁路模块内部去抖动滤波器  说明 设备模式和主机模式均有效。	RW
7:2	<i>Reserved</i>	<i>保留</i>	-	-
1	USBFS_SOFEN	USBFS SOF 脉冲输出使能位	USBFS 主机发出 SOF 或者设备成功接收到 SOF 时, 16 个 HCLK 时钟周期宽度的 SOF 脉冲从 PAD 输出使能 ● 0b0: SOF 脉冲不输出 ● 0b1: SOF 脉冲输出  说明 设备模式和主机模式均有效。	RW
0	USBFS_DFB	USBFS VBUS/ID 管脚内部去抖滤波器旁路使能位	● 0b0: 模块内部去抖动滤波器有效 ● 0b1: 旁路模块内部去抖动滤波器  说明 设备模式和主机模式均有效。	RW

34.6.3 USBHS 全局寄存器

这些寄存器在主机模式和设备模式下都可用，且在这两个模式间切换时无需对其进行重新编程。

34.6.3.1 VBUS 控制寄存器 (USBHS_GVBUSCFG)

该寄存器可用于设定 VBUS 值从而忽略 VBUS 管脚的状态。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved															VBUSVAL	VBUSOVEN	Reserved														
Access																RW	RW															

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	VBUSVAL	VBUS 值 (VBUS Value)	用于设定 USBHS 的 VBUS 值，当设定为 1，且 VBUSOVEN 设定 1 后对 USBHS 完成上电。  说明 仅设备模式有效。	RW
6	VBUSOVEN	VBUS Override 使能	用于将 VBUSVAL 设定的值反映到 USBHS CORE 的状态。仅当该位设置为 1，VBUSVAL 的值才有效。  说明 仅设备模式有效。	RW
5:0	Reserved	保留	-	-

34.6.3.2 AHB 控制寄存器 (USBHS_GAHBCFG)

该寄存器可用于在上电后或更改角色模式时对模块进行配置，主要包含 AHB 系统相关的配置参数。
应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程，请勿在初始编程后更改该寄存器。

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								PTXFELVL	TXFELVL	Reserved	DMAEN	HBSTLEN		GINTMSK	
Access																									RW	RW		RW	RW			

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8	PTXFELVL	周期性 Tx FIFO 空门限 (Periodic Tx FIFO empty level)	指示何时触发模块中断寄存器中的周期性 Tx FIFO 空中断位 (USBHS_GINTSTS 中的 PTXFE 位)。 0b0: PTXFE (位于 USBHS_GINTSTS) 中断指示周期性 Tx FIFO 为半空状态 0b1: PTXFE (位于 USBHS_GINTSTS) 中断指示周期性 Tx FIFO 为全空状态  说明 仅可主机模式下访问。	RW
7	TXFELVL	设备 Tx FIFO 空门限 (Tx FIFO empty level)	在设备模式下，该位指示何时触发 IN 端点发送 FIFO 空中断 (USBHS_DIEPINTx 中的 TXFE)。 0b0: TXFE (位于 USBHS_DIEPINTx) 中断指示 IN 端点 Tx FIFO 为半空状态 0b1: TXFE (位于 USBHS_DIEPINTx) 中断指示 IN 端点 Tx FIFO 为全空状态  说明 仅设备模式有效。	RW
6	Reserved	保留	-	-
5	DMAEN	DMA 使能 (DMA enable)	0b0: 模块以从模式运行 0b1: 模块以 DMA 模式运行  说明 设备模式和主机模式均有效。	RW
4:1	HBSTLEN	批量长度/类型 (Burst length/type)	0b0000: 单次 0b0001: INCR 0b0011: INCR4 0b0101: INCR8 0b0111: INCR16 - 其他: 禁止设置  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	GINTMSK	全局中断屏蔽 (Global interrupt mask)	该位用于屏蔽全局中断或对全局中断取消屏蔽。中断状态寄存器由模块进行更新，与此位的设置无关。 ● 0b0：屏蔽应用程序触发的中断 ● 0b1：取消对应用程序触发的中断的屏蔽  说明 设备模式和主机模式均有效。	RW

34.6.3.3 USB 配置寄存器 (USBHS_GUSBCFG)

该寄存器可用于在上电或更改角色模式后对模块进行配置。其中包含与 USB 和 USB-PHY 相关的配置参数。应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。请勿在初始编程后更改该寄存器。

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00001410																															
Name	Reserved	FDMOD	FHMOD	Reserved			ULPIPD	PTCI	PCCI	Reserved	ULPIEBUSI	ULPIEBUSD	ULPICSM	ULPIAR	ULFSL	Reserved	PHYLPSC	Reserved	TRDT			Reserved			PHYSEL	Reserved			TOTAL			
Access		RW	RW			RW	RW	RW			RW	RW	RW	RW	RW		RW			RW						RW					RW	

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30	FDMOD	强制设备模式 (Force device mode)	向该位写入 1 时, 可将模块强制为设备模式, 忽略 USBHS_ID 输入引脚的状态。 0b0: 正常模式, 取决 USBHS_ID 管脚的输入状态 0b1: 强制设备模式 将强制位置 1 后, 应用程序必须等待至少 25ms 后更改方可生效。  说明 设备模式和主机模式均有效。	RW
29	FHMOD	强制主机模式 (Force host mode)	向该位写入 1 时, 可将模块强制为主机模式, 忽略 USBHS_ID 输入引脚的状态。 0b0: 正常模式, 取决 USBHS_ID 管脚的输入状态 0b1: 强制主机模式 将强制位置 1 后, 应用程序必须等待至少 25ms 后更改方可生效。  说明 设备模式和主机模式均有效。	RW
28:26	Reserved	保留	-	-
25	ULPIPD	ULPI 接口保护禁止 (ULPI Interface protect disable)	该位控制 PHY 中内置的电路, 以保护 UPLI 接口。该功能所采用的任何上拉或下拉电阻均可禁止。详细信息请参考 ULPI 规范。 0b0: 使能接口保护电路 0b1: 禁止接口保护电路	RW
24	PTCI	通过指示符 (Indicator pass through)	该位控制互补输出是否需要通过内部 VBUS 有效电平比较器功能, 之后方可用于 RX CMD 中的 VBUS 状态。详细信息请参考 ULPI 规范。 0b0: 互补输出信号通过内部 VBUS 有效电平比较器的验证 0b1: 互补输出信号不通过内部 VBUS 有效电平比较器的验证	RW

位/位域	标记	位名	功能描述	读写
23	PCCI	互补指示符 (Indicator complement)	该位控制 PHY 将 ExternalVbusIndicator 输入信号反转并生成互补输出。有关详细信息, 请参见 ULPI 规范。 ● 0b0: PHY 不将 ExternalVbusIndicator 信号反转 ● 0b1: PHY 将 ExternalVbusIndicator 信号反转	RW
22	Reserved	保留	-	-
21	ULPIEVBUSI	ULPI 外部 VBUS 指示符 (ULPI external VBUS indicator)	该位指示 ULPI PHY 使用外部 VBUS 过流指示器。 ● 0b0: PHY 使用内部 VBUS 有效比较器 ● 0b1: PHY 使用外部 VBUS 有效比较器	RW
20	ULPIEVBUSD	ULPI 外部 VBUS 驱动器 (ULPI External VBUS Drive)	该位选择 ULPI PHY 使用内部或外部电源来驱动 VBUS 上的 5V 电压。 ● 0b0: PHY 使用内部电荷泵驱动 VBUS (默认) ● 0b1: PHY 使用外部电源驱动 VBUS	RW
19	ULPICSM	ULPI 时钟挂起 (ULPI Clock SuspendM)	该位置位 ULPI PHY 接口控制寄存器中的 ClockSuspendM 位。 ● 0b0: PHY 在挂起期间断开内部时钟的电源 ● 0b1: PHY 不断开内部时钟的电源  说明 该位仅适用于串行模式。	RW
18	ULPIAR	ULPI 时钟自动恢复 (ULPI Auto-resume)	该位置位 ULPI PHY 接口控制寄存器中的 AutoResume 位。 ● 0b0: PHY 不支持自动恢复功能 ● 0b1: PHY 使用自动恢复功能	RW
17	ULFSLS	ULPI FS/LS 选择 (ULPI FS/LS select)	应用程序使用该位为 ULPI PHY 选择 FS/LS 串行接口。 该位仅在 ULPI PHY 上选择了 FS 串行收发器的情况下有效。 ● 0b0: ULPI 接口 ● 0b1: ULPI FS/LS 串行接口	RW
16	Reserved	保留	-	-
15	PHYLPCS	PHY 低功耗时钟选择 (PHY Low-power clock select)	该位用于选择 480MHz 或 48MHz (低功耗) PHY 模式。在 FS 和 LS 模式下, PHY 通常可使用 48MHz 时钟运行, 从而节约功耗。 ● 0b0: 480MHz 内部 PLL 时钟 ● 0b1: 48MHz 外部时钟在 480MHz 模式下, UTMI 接口以 60MHz 或 30MHz 运行, 具体取决于选择的是 8 位还是 16 位数据宽度。在 48MHz 模式下, UTMI 接口在 FS 和 LS 模式下以 48MHz 运行。	RW
14	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
13:10	TRDT	USB 周转时间 (USB turnaround time)	以 PHY 时钟数为单位设置周转时间。要计算 TRDT 的值，请使用如下公式： $TRDT = 4 \times \text{AHB 时钟} + 1 \text{ 个 PHY 时钟}$ 例如： ● 如果 AHB 时钟频率=84MHz (PHY 时钟频率=48MHz)，则 TRDT 设置为 9。 ● 如果 AHB 时钟频率=48MHz (PHY 时钟频率=48MHz)，则 TRDT 设置为 5。  说明 仅设备模式有效。	RW
9:7	Reserved	保留	-	-
6	PHYSEL	USB2.0 高速 ULPI PHY 或 USB 片上全速串行收发器选择 (USB 2.0 high-speed ULPI PHY or USB 1.1 full-speed serial transceiver select)	● 0b0: USB2.0 高速 ULPI PHY ● 0b1: SB 片上全速串行收发器	RW
5:3	Reserved	保留	-	-
2:0	TOTAL	FS 超时校准 (FS timeout calibration)	PHY 引入的额外延迟包括应用程序在该字段中设置的 PHY 时钟数，以及模块的全速数据包间超时间隔。不同 PHY 引入的延迟对数据线状态的影响是不同的。全速操作的 USB 标准超时值为 16 到 18 (含) 个位时间。应用程序必须根据枚举速度编程该字段。每个 PHY 时钟增加的位时间数为 0.25 个位时间。  说明 设备模式和主机模式均有效。	RW

34.6.3.4 复位寄存器 (USBHS_GRSTCTL)

应用程序通过此寄存器复位模块中的各项硬件特性。

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80000000																															
Name	AHBIDL	DMAREQ	Reserved																			TXFNUM			TXFFLSH	RXFFLSH	Reserved	FCRST	HSRST	CSRST		
Access	RO	RO																				RW			RW	RW		RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31	AHBIDL	AHB 主器件空闲 (AHB master idle)	指示 AHB 主器件状态机处于空闲情况。  说明 设备模式和主机模式均有效。	RO
30	DMAREQ	DMA 请求信号 (DMA request signal)	该位指示 DMA 请求正在进行中。用于调试。  说明 设备模式和主机模式均有效。	RO
29:11	Reserved	保留	-	-
10:6	TXFNUM	TxFIFO 编号 (TxFIFO number)	使用 TxFIFO 刷新位进行 FIFO 刷新的 FIFO 编号。只有在模块将 TxFIFO 刷新位清零后，方可更改此字段。 0b00000：主机模式下刷新非周期性 TxFIFO、设备模式下刷新 TxFIFO0 0b00001：主机模式下刷新周期性 TxFIFO、设备模式下刷新 TxFIFO1 0b00010：设备模式下刷新 TxFIFO2 0b00011：设备模式下刷新 TxFIFO3 0b00101：设备模式下刷新 TxFIFO15 0b10000：在设备模式或主机模式下刷新所有的发送 FIFO - 其他：禁止设置  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
5	TXFFLSH	TxFIFO 刷新 (TxFIFO flush)	此位选择性地刷新一个或所有的发送 FIFO，但当模块处理通信事务时无法执行该操作。 只有在确认模块当前未对 TxFIFO 执行读写操作后，应用程序方可对此位执行写操作。使用以下寄存器进行确认： ● 读：NAK 有效中断可确保模块当前未对 FIFO 执行读操作 ● 写：USBHS_GRSTCTL 中的 AHBIDL 位可确保模块当前未对 FIFO 执行任何写操作  说明 设备模式和主机模式均有效。	RW
4	RXFFLSH	RxFIFO 刷新 (RxFIFO flush)	应用程序可使用此位刷新整个 RxFIFO，但必须首先确保模块当前未在处理通信事务。只有在确认模块当前未对 RxFIFO 执行读写操作后，应用程序方可对此位执行写操作。 应用程序必须等到此位清零后，方可执行其它操作。通常需要等待 8 个时钟周期（以 PHY 或 AHB 时钟中最慢的为准）。  说明 设备模式和主机模式均有效。	RW
3	Reserved	保留	-	-
2	FCRST	主机帧计数器复位 (Host frame counter reset)	应用程序对该位执行写操作时，模块中的帧数计数器复位。帧计数器复位后，由模块发送的下一个 SOF 的帧号为 0。  说明 设备模式和主机模式均有效。	RW
1	HSRST	HCLK 域逻辑软复位 (HCLK soft reset)	应用程序使用此位来刷新 AHB 时钟域中的控制逻辑。仅复位 AHB 时钟域流水线。 FIFO 不通过此位来刷新。 遵照协议终止 AHB 上的事务后，AHB 时钟域中的所有状态机均复位至空闲状态。 AHB 时钟域状态机所使用的 CSR 控制位清零。 要清零该中断，需要将由 AHB 时钟域状态机生成并用于控制中断状态的状态屏蔽位清零。 由于中断状态位并未清零，因此应用程序可以获取在该位置 1 后所发生的所有模块事件的状态。 此位为自清零位，模块将在其中所有必要逻辑复位后将该位清零。该过程需要若干个时钟的时间，具体取决于模块的当前状态。  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	CSRST	模块软复位 (Core soft reset)	按如下所述将 HCLK 和 PCLK 域复位： 除以下各位外，将各个中断和所有 CSR 寄存器位清零： ● USBHS_GCCTL 中的 GATEHCLK 位 ● USBHS_GCCTL 中的 STPPCLK 位 ● USBHS_HCFG 中的 FSLSPCS 位 ● USBHS_DCFG 中的 DSPD 位 将所有模块状态机（AHB 从器件除外）复位至空闲状态，并清空所有发送 FIFO 和接收 FIFO。 在 AHB 传输的最后数据阶段结束后，尽快终止 AHB 主器件上的所有事务。立即终止 USB 上的所有事务。 应用程序可在需要复位模块时随时对该位执行写操作。该位为自清零位，模块将在其中所有必要逻辑复位后将该位清零，该过程需要若干个时钟的时间，具体取决于模块的当前状态。该位一旦清零，软件必须等待至少 3 个 PHY 时钟后才可以访问 PHY 域（同步延迟）。此外，软件还必须在确定该寄存器中的位 31 置 1（AHB 主器件空闲）后方可开始运行。 软件复位通常在两种情况下使用，一是软件开发期间，二是用户动态更改以上所列 USB 配置寄存器中的 PHY 选择位后。用户更改 PHY 时，将为 PHY 选择相应的时钟并用于 PHY 域中。一旦选择了新的时钟，则必须复位 PHY 域，才能保证正常运行。  说明 设备模式和主机模式均有效。	RW

34.6.3.5 全局中断状态寄存器 (USBHS_GINTSTS)

该寄存器用于在当前模式（设备模式或主机模式）下借助系统级别的事件来中断应用程序。该寄存器可指示当前模式。

FIFO 状态中断为只读；如果软件在处理这些中断期间对 FIFO 执行读写操作，则 FIFO 中断标志将自动清零。使能中断位前，应用程序必须在初始化时将 USBHS_GINTSTS 寄存器清零，才可以避免在初始化前产生任何中断。

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x14000020																															
Name	WKUINT	VBUSVINT	DISCINT	CIDSCHG	LPMINT	PTXFE	HCINT	HPRTINT	Reserved	DATAFSUSP	IPXFR/INCOMPI	ISOIXFR	OEPI	IEPI	Reserved	EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Reserved	GONAKEFF	GINAKEFF	NPTXFE	RXFNE	SOF	Reserved	MMIS	CMOD		
Access	RW	RW	RW	RW	RW	RO	RO	RO		RW	RW	RW	RO	RO		RW	RW	RW	RW	RW	RW		RO	RO	RO	RO	RW		RW	RO	RO	

位/位域	标记	位名	功能描述	读写
31	WKUINT	检测到恢复/远程唤醒中断 (Resume/remote wakeup detected interrupt)	在设备模式下，当 USB 总线上检测到恢复信号时，将触发该中断。 在主机模式下，当 USB 上检测到远程唤醒时，将触发该中断。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
30	VBUSVINT	VBUS 有效中断 (VBUS valid interrupt)	设备模式下，当检测到 USBHS_VBUS 管脚由低变高时，将触发该中断。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
29	DISCINT	检测到断开连接中断 (Disconnect detected interrupt)	当检测到设备断开连接时触发该中断。 通过软件对该位写 1 清零。  说明 仅主机模式有效。	RW
28	CIDSCHG	连接器 ID 线状态更改 (Connector ID status change)	当连接器 ID 线状态发生更改时，模块将该位置 1。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
27	LPMINT	LPM 中断	- 设备模式：设备收到正确 LPM 传输且有效应答时该位置 1。 - 主机模式：主机发送 LPM 后收到有效应答，或者超过设定的重试次数后该位置 1。	RW

位/位域	标记	位名	功能描述	读写
26	PTXFE	周期性 TxFIFO 空中断(Periodic TxFIFO empty interrupt)	当周期性发送 FIFO 为半空或全空状态，且周期性请求队列中存在可写入至少一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 USBHS_GAHBCFG 寄存器中的周期性 TxFIFO 空级别位 (USBHS_GAHBCFG 中的 PTXFELVL 位) 决定。  说明 仅主机模式有效。	RO
25	HCINT	主机通道中断 (Host channels interrupt)	模块将该位置 1 时，指示模块中一个通道上存在挂起的中断（在主机模式下）。应用程序必须读取主机 USBHS_HAINT 寄存器，以确定发生中断的通道的准确编号，然后读取相应的 USBHS_HCINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将 USBHS_HCINTx 寄存器的相应状态位清零，之后才能将该位清零。  说明 仅主机模式有效。	RO
24	HPRTINT	主机端口中断 (Host port interrupt)	模块将该位置 1 时，指示主机模式下 USBHS 控制器端口的状态发生变化。应用程序必须读取 USBHS_HPRT 寄存器，以确定引发此中断的确切事件。应用程序必须先将 USBHS_HPRT 寄存器的相应状态位清零，之后才能将该位清零。  说明 仅主机模式有效。	RO
23	Reserved	保留	-	-
22	DATAFSUSP	数据获取挂起 (Data fetch suspended)	该中断仅在 DMA 模式下有效。该中断指示，模块因 TxFIFO 空间或请求队列空间不可用而停止为 IN 端点获取数据。应用程序将该中断用于端点不匹配算法中。例如，在检测到端点不匹配后，应用程序将执行以下操作： ● 将全局非周期性 IN NAK 握手信号置 1 ● 禁止 IN 端点 ● 清空 FIFO ● 根据 IN 令牌序列学习队列确定令牌序列 ● 重新使能端点 ● 清零全局非周期性 IN NAK 握手信号 如果全局非周期性 IN NAK 已清零但模块尚未从 IN 端点获取数据，同时又接收到 IN 令牌，模块将产生“FIFO 为空时接收到 IN 令牌”中断。然后 USBHS 将 NAK 响应发送到主机。为避免上述情况，应用程序可以检查 USBHS_GINTSTS 中的 DATAFSUSP 中断，该中断可确保在 FIFO 存满后再将全局 NAK 握手信号清零。另外，应用程序可以在将全局 IN NAK 握手信号清零时屏蔽“当 FIFO 为空时接收到 IN 令牌中断”。通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
21	IPXFR/ INCOMPISOO UT	IPXFR: 未完成 周期性传输 (Incomplete periodic transfer) INCOMPISOO OUT: 未完成 OUT 同步传输 (Incomplete isochronous OUT transfer)	● IPXFR: 在主机模式下, 如果存在仍处于挂起状态的未完成周期性事务, 而这些事务计划在当前帧期间完成, 则模块会将该中断位置 1。 通过软件对该位写 1 清零。  说明 仅主机模式有效。 ● INCOMPISOOOUT: 在设备模式下, 模块将该中断置 1 时, 指示当前帧中至少有一个同步 OUT 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
20	IISOIXFR	未完成 IN 同步 传输 (Incomplete isochronous IN transfer)	模块将该中断置 1 时, 指示当前帧中至少有一个同步 IN 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。 通过软件对该位写 1 清零。	RW
19	OEPINT	OUT 端点中断 (OUT endpoint interrupt)	模块将该位置 1 时, 指示模块中一个 OUT 端点上存在挂起的中断 (在设备模式下)。应用程序必须读取主机 USBHS_DAIN_T 寄存器, 以确定发生中断的 OUT 端点的准确编号, 然后读取相应的 USBHS_DOEPINTx 寄存器, 以确定引发中断的确切原因。应用程序必须先将相应 USBHS_DOEPINTx 寄存器的相应状态位清零, 之后才能将该位清零。	RO
18	IEPINT	IN 端点中断(IN endpoint interrupt)	模块将该位置 1 时, 指示模块中一个 IN 端点上存在挂起的中断 (在设备模式下)。应用程序必须读取主机 USBHS_DAIN_T 寄存器, 以确定发生中断的 IN 端点的准确编号, 然后读取相应的 USBHS_DIEPINTx 寄存器, 以确定引发中断的确切原因。应用程序必须先将相应 USBHS_DIEPINTx 寄存器的相应状态位清零, 之后才能将该位清零。	RO
17:16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	EOPF	周期性帧结束中断(End of periodic frame interrupt)	指示当前帧已达到 USBHS_DCFG 寄存器中周期性帧间隔字段 (USBHS_DCFG 中的 PFIVL 位) 所指定的周期。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
14	ISOODRP	丢弃同步 OUT 数据包中断 (Isochronous OUT packet dropped interrupt)	如果由于 Rx FIFO 空间不足, 无法容纳同步 OUT 端点的最大数据包, 从而导致模块无法向 Rx FIFO 写入同步 OUT 数据包, 模块会将该位置 1。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
13	ENUMDNE	枚举完成中断 (Enumeration done interrupt)	模块将该位置 1 时, 指示速度枚举已完成。应用程序必须读取 USBHS_DSTS 寄存器来获取枚举速度。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
12	USBRST	USB 复位中断 (USB reset interrupt)	模块将该位置 1 时, 指示在 USB 上检测到复位信号。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
11	USBSUSP	USB 挂起中断 (USB suspend interrupt)	模块将该位置 1 时, 指示在 USB 上检测到挂起状态。当 USB 总线上的空闲状态保持 3ms, 模块便会进入挂起状态。通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
10	ESUSP	早期挂起中断 (Early suspend interrupt)	模块将该位置 1 时, 指示已检测到 USB 处于空闲状态的时间达到 3ms。  说明 仅设备模式有效。	RW
9:8	Reserved	保留	-	-
7	GONAKEFF	全局 OUT NAK 有效中断(Global OUT NAK effective interrupt)	指示 USBHS_DCTL 寄存器中由应用程序设置的“将全局 OUT NAK 置 1”位 (USBHS_DCTL 中的 SGONAK 位) 已在模块中生效。通过写入 USBHS_DCTL 寄存器中的“将全局 OUT NAK 清零”位 (USBHS_DCTL 中的 CGONAK 位), 可将该位清零。  说明 仅设备模式有效。	RO

位/位域	标记	位名	功能描述	读写
6	GINAKEFF	全局非周期性 IN NAK 有效中断 (Global IN nonperiodic NAK effective interrupt)	指示 USBHS_DCTL 寄存器中由应用程序设置的“将全局非周期性 IN NAK 置 1”位 (USBHS_DCTL 中的 SGINAK 位) 已在模块中生效。也就是说, 模块已对应用程序设置的全局 IN NAK 位进行采样, 结果已生效。通过清零 USBHS_DCTL 寄存器中的“将全局非周期性 IN NAK 清零”位 (USBHS_DCTL 中的 CGINAK 位), 可将该位清零。此中断不一定表示 USB 上已发送了一个 NAK 握手信号。STALL 位优先级高于 NAK 位。  说明 仅设备模式有效。	RO
5	NPTXFE	非周期性 Tx FIFO 空中断 (Non-periodic Tx FIFO empty interrupt)	当非周期性 Tx FIFO 为半空或全空状态, 且非周期性发送请求队列中至少存在可写入一个条目的空间时, 将触发该中断。该 FIFO 为半空状态还是全空状态由 USBHS_GAHBCFG 寄存器中的非周期性 Tx FIFO 空级别位 (USBHS_GAHBCFG 中的 TXFELVL 位) 决定。  说明 仅主机模式有效。	RO
4	RXFNE	Rx FIFO 非空中断 (Rx FIFO non-empty interrupt)	指示 Rx FIFO 中至少有一个数据包等待读取。  说明 设备模式和主机模式均有效。	RO
3	SOF	帧起始中断 (Start of frame interrupt)	在主机模式下, 模块将该位置 1 时, 指示 USB 上已发送一个 SOF (FS) 或 Keep-Alive (LS) 信号。应用程序必须将此位置 1 才可清除该中断。 在设备模式下, 模块将该位置 1 时, 指示 USB 上已接收到一个 SOF 令牌。应用程序可通过读取设备状态寄存器来获得当前的帧编号。只有在模块以 FS 模式运行时, 才会出现此中断。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
2	Reserved	保留	-	-
1	MMIS	模式不匹配中断 (Mode mismatch interrupt)	当应用程序尝试访问以下寄存器时, 模块将该位置 1: ● 模块运行在设备模式下访问主机模式寄存器 ● 模块运行在主机模式下访问设备模式寄存器 寄存器访问在 AHB 上以 OKAY 响应结束, 但该访问在内部被模块忽略并且不会影响模块运行。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	CMOD	当前工作模式 (Current mode of operation)	指示当前模式。 ● 0b0：设备模式 ● 0b1：主机模式  说明 设备模式和主机模式均有效。	RO

34.6.3.6 全局中断屏蔽寄存器 (USBHS_GINTMSK)

该寄存器与全局中断状态寄存器 (USBHS_GINTSTS) 结合使用，以中断应用程序。如果将某个中断位屏蔽，则不会产生与该位相关的中断，但是与该中断相对应的寄存器位仍会置 1。

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	WKUIM	VBUSVIM	DISCIM	CIDSCHGM	LPMINTM	PTXFEM	HCIM	HPRTIM	Reserved	DATAFSUSPM	IPXFRM/INCOMPISOOUTM	IISOXFRM	OEPIM	IEPIM	Reserved	EOPFM	ISOODRPM	ENUMDNEM	USBRSTM	USBSUSPM	ESUSPM	Reserved	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFNEM	SOFM	Reserved	MMISM	Reserved		
Access	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW		RW			

位/位域	标记	位名	功能描述	读写
31	WКУIM	检测到恢复/远程唤醒中断屏蔽 (Resume/remote wakeup detected interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
30	VBUSVIM	VBUS 有效中断屏蔽 (VBUS valid interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 仅设备模式有效。	RW
29	DISCIM	检测到断开连接中断屏蔽 (Disconnect detected interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 仅主机模式有效。	RW
28	CIDSCHGM	连接器 ID 线状态更改中断屏蔽 (Connector ID status change interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
27	LPMINTM	LPM 中断屏蔽 (LPM interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
26	PTXFEM	周期性 TxFIFO 空中断屏蔽 (Periodic TxFIFO empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
25	HCIM	主机通道中断屏蔽 (Host channels interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
24	HPRTIM	主机端口中断屏蔽 (Host port interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
23	Reserved	保留	-	-
22	DATAFSUSPM	数据获取挂起中断屏蔽 (Data fetch suspended interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
21	IPXFRM/ INCOMPISOO UTM	IPXFRM: 未完成周期性传输中断屏蔽 (Incomplete periodic transfer interrupt mask) INCOMPISOOOUTM: 未完成 OUT 同步传输中断屏蔽 (Incomplete isochronous OUT transfer interrupt mask)	● IPXFRM: ▶ 0b0: 屏蔽中断 ▶ 0b1: 使能中断  说明 仅主机模式有效。 ● INCOMPISOOOUTM: ▶ 0b0: 屏蔽中断 ▶ 0b1: 使能中断  说明 仅设备模式有效。	RW
20	IISOIXFRM	未完成 IN 同步传输中断屏蔽 (Incomplete isochronous IN transfer interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
19	OEPIIM	OUT 端点中断屏蔽 (OUT endpoint interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
18	IEPIM	IN 端点中断屏蔽 (IN endpoint interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
17:16	Reserved	保留	-	-
15	EOPFM	周期性帧结束中断屏蔽(End of periodic frame interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
14	ISOODRPM	丢弃同步 OUT 数据包中断屏蔽(Isochronous OUT packet dropped interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
13	ENUMDNEM	枚举完成中断屏蔽(Enumeration done interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
12	USBRSTM	USB 复位中断屏蔽(USB reset interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
11	USBSUSPM	USB 挂起中断屏蔽(USB suspend interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
10	ESUSPM	早期挂起中断屏蔽(Early suspend interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
9:8	Reserved	保留	-	-
7	GONAKEFFM	全局 OUT NAK 有效中断屏蔽(Global OUT NAK effective interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
6	GINAKEFFM	全局非周期性 IN NAK 有效中断屏蔽(Global IN nonperiodic NAK effective interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
5	NPTXFEM	非周期性 TxFIFO 空中断屏蔽 (Non-periodic TxFIFO empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
4	RXFNEM	RxFIFO 非空中断屏蔽(RxFIFO non-empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
3	SOFM	帧起始中断屏蔽 (Start of frame interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
2	Reserved	保留	-	-
1	MMISM	模式不匹配中断屏蔽 (Mode mismatch interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
0	Reserved	保留	-	-

34.6.3.7 接收状态调试读取/状态读取和出栈寄存器 (USBHS_GRXSTSR/USBHS_GRXSTSP)

读取接收状态调试读取寄存器将返回接收 FIFO 顶部的内容。读取接收状态读取和出栈寄存器将额外弹出 RxFIFO 顶部的数据条目。接收状态内容在主机模式和设备模式下的解释不同。

当接收 FIFO 为空时，模块会忽略对该寄存器的读取或出栈操作，并返回值 0x0000 0000。当模块中断寄存器的接收 FIFO 非空位 (USBHS_GINTSTS 中的 RXFNE 位) 置位时，应用程序必须仅弹出接收状态 FIFO。

读取的偏移地址 (Offset) : 0x1C

出栈的偏移地址 (Offset) : 0x20

主机模式:

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												PKTSTS				DPID		BCNT										CHNUM			
Access													RO				RO		RO										RO			

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:17	PKTSTS	数据包状态 (Packet status)	指示接收的数据包的状态 0b0010: 接收到 IN 数据包 0b0011: IN 传输完成 (触发中断) 0b0101: 数据同步错误 (触发中断) 0b0111: 暂停通道 (触发中断) - 其他: Reserved	RO
16:15	DPID	数据 PID (Data PID)	指示接收的数据包的数据 PID 0b00: DATA0 0b01: DATA2 0b10: DATA1 0b11: MDATA	RO
14:4	BCNT	字节计数(Byte count)	指示接收的 IN 数据包的字节数	RO
3:0	CHNUM	通道编号 (Channel number)	指示当前接收的数据包所属的通道编号	RO

设备模式：

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved											PKTSTS		DPID	BCNT										EPNUM							
Access												RO	RO	RO										RO								

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:17	PKTSTS	数据包状态 (Packet status)	指示接收的数据包的状态 0b0001：全局 OUT NAK（触发中断） 0b0010：接收到 OUT 数据包 0b0011：OUT 传输完成（触发中断） 0b0100：SETUP 事务完成（触发中断） 0b0110：接收到 SETUP 数据包 - 其他：Reserved	RO
16:15	DPID	数据 PID (Data PID)	指示接收的 OUT 数据包的数据 PID 0b00：DATA0 0b01：DATA2 0b10：DATA1 0b11：MDATA	RO
14:4	BCNT	字节计数(Byte count)	指示接收的数据包的字节数	RO
3:0	EPNUM	通道编号 (Channel number)	指示当前接收的数据包所属的端点编号	RO

34.6.3.8 接收 FIFO 大小寄存器 (USBHS_GRXFSIZ)

此应用程序可以对必须分配给 Rx FIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000800																															
Name	Reserved																RXFD															
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	RXFD	Rx FIFO 深度 (Rx FIFO depth)	以 32 位字为单位。取值范围是 16~2048 上电复位值为最大 Rx 数据 FIFO 深度	RW

34.6.3.9 主机非周期性发送 FIFO 大小寄存器 (USBHS_HNPTXFSIZ) / 端点 0 发送 FIFO 大小 (USBHS_DIEPTXF0)

此应用程序可以对必须分配给 Tx FIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x08000800																															
Name	NPTXFD/TX0FD																NPTXFSA/TX0FSA															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	NPTXFD/ TX0FD	NPTXFD: 非周期性 Tx FIFO 深度 TX0FD: 端点 0 Tx FIFO 深度	- 主机模式: NPTXFD, 以 32 位字为单位。取值范围是 16~2048 - 设备模式: TX0FD, 以 32 位字为单位。取值范围是 16~2048	RW
15:0	NPTXFSA/ TX0FSA	NPTXFSA: 非周期性发送 RAM 起始地址 TX0FSA: 端点 0 发送 RAM 起始地址	- 主机模式: NPTXFSA, 此字段包含非周期性发送 FIFO RAM 的存储器起始地址 - 设备模式: TX0FSA, 此字段包含端点 0 发送 FIFO RAM 的存储器起始地址	RW

34.6.3.10 非周期性发送 FIFO/队列状态寄存器 (USBHS_HNPTXSTS)

此只读寄存器包含非周期性 TxFIFO 和非周期性发送请求队列的自由空间信息。

此寄存器仅在主机模式有效，设备模式无效。

Offset	Bit Position																																
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00080800																																
Name	Reserved	NPTXQTOP								NPTQXSAV								NPTXFSAV															
Access		RO								RO								RO															

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:24	NPTXQTOP	非周期性发送请求队列顶部(Top of the non-periodic transmit request queue)	非周期性发送请求队列中 MAC 目前正在处理的条目。 [30:27]: 通道/端点编号 [26:25]: 0b00: IN/OUT 令牌 0b01: 长度为零的发送数据包 0b10: Reserved 0b11: 通道停止命令 [24]: 结束 (所选通道/端点的最后一个条目)	RO
23:16	NPTQXSAV	非周期性发送请求队列可用空间 (Non-periodic transmit request queue space available)	指示非周期性发送请求队列中的可用空闲空间大小，取值范围是 0~8。 在主机模式下，此队列保存 IN 和 OUT 请求。 0: 非周期性发送请求队列已满 1: 1 个位置可用 2: 2 个位置可用 8: 8 个位置可用	RO
15:0	NPTXFSAV	非周期性 TxFIFO 可用空间 (Non-periodic TxFIFO space available)	指示非周期性 TxFIFO 中的可用空闲空间大小，以 32 位字为单位，取值范围是 0~2048。 0: 非周期性 TxFIFO 已满 1: 1 个字可用 2: 2 个字可用 2048: 2048 个字可用	RO

34.6.3.11 模块 ID 寄存器 (USBHS_CID)

该寄存器为可编程用户配置 ID 寄存器。

Offset	Bit Position																															
0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x12345678																															
Name	PRODUCT_ID																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PRODUCT_ID	产品 ID 字段 (Product ID field)	可通过应用程序编程的 ID 字段。	RW

34.6.3.12 LPM 配置寄存器 (USBHS_GLPMCFG)

该寄存器为可编程 LPM 配置寄存器。

Offset	Bit Position																																								
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
Reset	0x00000000																																								
Name	Reserved	ENBSEL		LPMRCNTSTS				SENDLPM		LPMRCNT				LPMCHIDX				L1RSMOK		SLPSTS		LPMRSP		L1DSEN		BSELTHRS				L1SSEN		REMWAKE		BSEL				LPMACK		LPMEN	
Access		RW		RO				RW		RW				RW				RO		RO		RO		RW		RW				RW		RW		RW				RW		RW	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28	ENBSEL	BSEL 使能位	0b0: 遵循 USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007 0b1: 遵循 Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007	RW
27:25	LPMRCNTSTS	LPM 重试状态	记录 LPM 剩余的重试次数  说明 仅主机模式有效。	RO
24	SENDLPM	发送 LPM 传输	0b0: 不发送 LPM 传输 0b1: 发送 LPM 传输 (EXT 和 LPM 令牌) 当收到设备返回的应答 (ACK, STALL 或 NYET) 后该位自动清零。  说明 仅主机模式有效。	RW
23:21	LPMRCNT	LPM 重试次数	当收到 ERROR 应答时, 主机发送 LPM 传输的重试次数, 直到收到有效的应答 (ACK, STALL 或 NYET)  说明 仅主机模式有效。	RW
20:17	LPMCHIDX	LPM 发送通道索引	LPM 传输所用主机通道索引, 硬件自动插入该通道设定的设备地址以及端点信息。  说明 仅主机模式有效。	RW
16	L1RSMOK	L1 状态位	0b0: 当前状态不能从 L1 恢复 0b1: 当前状态可以从 L1 恢复	RO
15	SLPSTS	sleep 状态位	设备模式: 当设备发送 ACK 响应 LPM 传输并经过协议规定的时间后, 该位置 1, 表示进入 SLEEP 模式。 当总线状态变化, 或者断连, 或者发送远程唤醒信号时, 该位自动清 0, 退出 SLEEP 模式。 主机模式:	RO

位/位域	标记	位名	功能描述	读写
15	SLPSTS	sleep 状态位	主机发送 LPM 传输, 接收到设备响应的 ACK 应答后, 该位置 1, 表示进入 SLEEP 模式。 当主机接收到远程唤醒信号, 或者主机发起唤醒, 或者主机发起复位时, 该位自动清 0, 退出 SLEEP 模式。 ● 0b0: Not in L1 ● 0b1: In L1	RO
14:13	LPMRSP	LPM 应答	主机接收或者从机发送的 LPM 应答 ● 0b00: ERROR(无应答) ● 0b01: STALL ● 0b10: NYET ● 0b11: ACK	RO
12	L1DSEN	L1 deep sleep 使能位	最大限度的降低功耗, 应用程序需要将此位设定为 1	RW
11:8	BSELTHRS	BSEL 阈值设定	设备模式: 当设备接收到 BSEL 值大于等于该寄存器设定时, 进入 L1。 主机模式: 主机发送 resume 信号的时间。	RW
7	L1SSEN	L1 shallow sleep 使能位	最大限度的降低功耗, 应用程序需要将此位设定为 1	RW
6	REMWAKE	bRemoteWake 值	主机发送或者设备接收到的 bRemoteWake 值	RW
5:2	BSEL	BSEL 设置	设备模式: BSEL bmAttribute 值, 有效应答 LPM 传输后自动更新。 主机模式: 主机发送 LPM 传输的 BSEL 值, 或者 resume 信号长度值。 ● 0b0000: 125us ● 0b0001: 150us ● 0b0010: 200us ● 0b0100: 400us ● 0b0101: 500us ● 0b0110: 1000us ● 0b0111: 2000us ● 0b1000: 3000us ● 0b1001: 4000us ● 0b1010: 5000us ● 0b1011: 6000us ● 0b1100: 7000us ● 0b1101: 8000us ● 0b1110: 9000us ● 0b1111: 10000us ● 其他: 禁止设置	RW
1	LPMACK	LPM 应答	● 0b0: NYET ● 0b1: ACK 即使该位写 1, 但在以下场景 (LPM 传输有错误) 中不会发送 ACK 应答: ● 存在 PID/CRC 错误, 该场景发送 ERROR 应答 ● bLinkState 不等于 0001b, 该场景发送 STALL 应答	RW

位/位域	标记	位名	功能描述	读写
1	LPMACK	LPM 应答	存在挂起的数据传输，该场景发送 NYET 应答  说明 仅设备模式有效。	RW
0	LPMEN	LPM 使能位	0b0: 禁止 0b1: 使能	RW

34.6.3.13 主机周期性发送 FIFO 大小寄存器 (USBHS_HPTXFSIZ)

此应用程序可以对必须分配给周期 Tx FIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x08001000																															
Name	PTXFD																PTXSA															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	PTXFD	主机周期性 Tx FIFO 深度	以 32 位字为单位，最小值为 16	RW
15:0	PTXSA	主机周期性 Tx FIFO 起始地址	上电复位值是最大 Rx FIFO 深度与最大非周期性 Tx FIFO 深度之和	RW

34.6.3.14 设备 IN 端点发送 FIFO 大小寄存器 (USBHS_DIEPTXFx) (x=1~15)

此应用程序可以对必须分配给设备 Tx FIFO 的大小进行编程。

偏移地址 (Offset) : 0x100+0x4*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x08001000+0x800*(x-1)																															
Name	INEPTXFD																INEPTXSA															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	INEPTXFD	设备 IN 端点 Tx FIFO 深度	以 32 位字为单位，最小值为 16	RW
15:0	INEPTXSA	IN 端点 Tx FIFOx RAM 起始地址	此字段包含 IN 端点发送 FIFOx 的存储器起始地址。该地址必须与 32 位存储器位置对齐。	RW

34.6.4 USBHS 主机模式寄存器

主机模式寄存器会影响主机模式下的模块操作。在设备模式下不得访问主机模式寄存器，因为产生的结果不明确。

34.6.4.1 主机配置寄存器 (USBHS_HCFG)

此寄存器将在上电后对模块进行配置，请勿在初始化主机后更改此寄存器。

Offset	Bit Position																															
0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000200																															
Name	Reserved																												FSLSS			
Access																													RW	RW		

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	FSLSS	仅支持 FS 和 LS 功能控制位	应用程序使用此位控制模块的枚举速度。使用此位，应用程序可使模块工作为 FS 主机，即使所连接的设备支持 HS 通信也是如此。请勿在初始编程后更改此字段。 0b0：HS/FS/LS，取决于所连接设备支持的最大速度 0b1：仅限 FS/LS，即使所连接设备可支持 HS	RW
1:0	FSLSPCS	FS/LS PHY 时钟选择(FS/LS PHY clock select)	当模块处于 FS 主机模式时： 0b01：PHY 时钟以 48MHz 运行 - 其他：禁止设置 当模块处于 LS 主机模式时： 0b01：PHY 时钟以 48MHz 运行 0b10：PHY 时钟以 6MHz 运行 - 其他：禁止设置  注意 当设备连上主机时，必须依照所连接设备的速度设置 FSLSPCS（更改此位后，必须进行软件复位）。	RW

34.6.4.2 主机帧时间间隔寄存器 (USBHS_HFIR)

此寄存器用于存储 USBHS 控制器对已连接设备当前速度所设定的帧间隔信息。

Offset	Bit Position																															
0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000EA60																															
Name	Reserved																FRIVL															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	FRIVL	帧间隔 (Frame interval)	应用程序在此字段编程的值用于指定两个连续 SOF(FS)或 Keep-Alive 令牌(LS)之间的时间间隔。此字段包含构成所需帧间隔的 PHY 时钟数。只有将主机端口控制和状态寄存器的端口使能位 (USBHS_HPRT 的 PENA 位) 置 1 后, 应用程序才能向此寄存器中写入值。如果未对值进行编程, 模块将根据在主机配置寄存器的 FS/LS PHY 时钟选择字段 (USBHS_HCFG 中的 FSLSPCS) 中指定的 PHY 时钟来计算。请勿在初始配置后更改此字段的值。 设定值=帧间隔 (ms) × (PHY 时钟频率) -1  注意 当应用程序需要更改帧间隔时间时, 可对 FRIVL 位进行修改。	RW

34.6.4.3 主机帧编号/帧剩余时间寄存器 (USBHS_HFNUM)

Offset	Bit Position																															
0x408	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00003FFF																															
Name	FTREM																FRNUM															
Access	RO																RO															

位/位域	标记	位名	功能描述	读写
31:16	FTREM	帧剩余时间 (Frame time remaining)	指示当前帧的剩余时间 (以 PHY 时钟数为单位)。每过去 1 个 PHY 时钟, 此字段递减 1。当值达到零时, 此字段将重新装载帧间隔寄存器中的值, 并由模块在 USB 上发送一个新 SOF。	RO
15:0	FRNUM	帧编号 (Frame number)	当在 USB 上发送 1 个新 SOF 时此字段的值将递增 1, 当达到 0x3FFF 时会清零。	RO

34.6.4.4 主机周期性发送 FIFO/队列状态寄存器 (USBHS_HPTXSTS)

此只读寄存器包含周期性 TxFIFO 和周期性发送请求队列的空闲空间信息。

Offset	Bit Position																															
0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00080800																															
Name	PTXQTOP								PTXQSAV								PTXFSAVL															
Access	RO								RO								RO															

位/位域	标记	位名	功能描述	读写
31:24	PTXQTOP	周期性发送请求队列顶部(Top of the periodic transmit request queue)	指示周期性 Tx 请求队列中 MAC 当前正在处理的项。该寄存器用于调试。 [31]: 奇数/偶数帧 ● 0b0: 以偶数帧发送 ● 0b1: 以奇数帧发送 [30:27]: 通道/端点编号 [26:25]: 类型 ● 0b00: 输入/输出 ● 0b01: 零长度数据包 ● 0b10: Reserved ● 0b11: 禁止通道命令 [24]: 结束 (所选通道的最后一个条目)	RO
23:16	PTXQSAV	周期性发送请求队列可用空间 (Periodic transmit request queue space available)	指示可供写入的周期性发送请求队列的空闲位置的数量, 取值范围是 0~8。该队列既包含 IN 请求, 又包含 OUT 请求。 ● 0: 周期性发送请求队列已满 ● 1: 1 个位置可用 ● ● 8: 8 个位置可用	RO
15:0	PTXFSAVL	周期性发送数据 FIFO 可用空间 (Periodic transmit data FIFO space available)	指示可供写入的周期性 TxFIFO 的空闲位置的数量。以 32 位字为单位。 ● 0: 周期性 TxFIFO 已满 ● 1: 1 个字可用 ● ● n: n 个字可用 (n 取值为寄存器 USBHS_HPTXFSIZ.PTXFD 的值)	RO

34.6.4.5 主机全体通道中断寄存器 (USBHS_HAINT)

当通道上有事件发生时，主机全体通道中断寄存器会使用模块中断寄存器中的主机通道中断位 (USBHS_GINTSTS 中的 HCINT 位) 中断应用程序。每个通道对应 1 个中断位，最多有 16 个位。
当应用程序通过相应主机通道 x 中断寄存器清零中断时，该寄存器中的位也会清零。

Offset	Bit Position																															
0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																HAINT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	HAINT	通道中断 (Channel interrupt)	对应通道 15~通道 0： ● 0b0：该通道没有产生中断 ● 0b1：该通道产生了中断  说明 每个 Bit 对应一个通道。	RO

34.6.4.6 主机全体通道中断屏蔽寄存器 (USBHS_HAINTMSK)

主机全体通道中断屏蔽寄存器与主机全体通道中断寄存器结合使用，进而在通道上发生事件时中断应用程序。
每个通道对应 1 个中断屏蔽位，最多有 16 个位。

Offset	Bit Position																															
0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																HAINTM															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	HAINTM	通道中断屏蔽 (Channel interrupt mask)	对应通道 15~通道 0： ● 0b0：屏蔽中断 ● 0b1：使能中断  说明 每个 Bit 对应一个通道。	RW

34.6.4.7 主机端口控制和状态寄存器 (USBHS_HPRT)

该寄存器仅在主机模式下可用。当前，USBHS 主机仅支持一个端口。

该寄存器包含 USB 端口相关的信息，如 USB 复位、使能、挂起、恢复、连接状态。该寄存器中的 PENCHNG/PCDET 位可通过模块中断寄存器中的主机端口中断位 (USBHS_GINTST 中 HPRTINT 位) 触发应用程序中断。发生端口中断时，应用程序必须读取该寄存器，并将引起中断的位清零。应用程序必须向该位写入 1 以清除该中断。

Offset	Bit Position																																
0x440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved													PSPD	PTCTL				PWPR		PLSTS		Reserved	PRST	PSUSP		PRES	Reserved		PENCHNG	PENA	PCDET	PCSTS
Access														RO	RW				RW	RO			RW	RW	RW				RW	RW	RW	RO	

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18:17	PSPD	端口速度(Port speed)	指示连接到该端口的设备的速度。 0b00: 高速 0b01: 全速 0b10: 低速 0b11: Reserved	RO
16:13	PTCTL	端口测试控制(Port test control)	应用程序向该字段写入一个非零值，以将端口置于测试模式，同时端口上会产生对应模式的信号。 0b0000: 测试模式禁止 0b0001: Test_J 模式 0b0010: Test_K 模式 0b0011: Test_SE0_NAK 模式 0b0100: Test_Packet 模式 0b0101: Test_Force_Enable - 其他: 禁止设置	RW
12	PWPR	端口电源(Port power)	应用程序使用该字段控制该端口的电源。由于本 USBHS 内置 PHY 不具备供电能力，所以此为设置为 1 时，通过 USBHS_DRVVBUS 使能外部 USB 电源芯片供电。 0b0: 掉电 0b1: 通电	RW
11:10	PLSTS	端口线状态	指示 USB 数据线的当前逻辑电平。 [11]: USBHS_DM 的逻辑电平 [10]: USBHS_DP 的逻辑电平	RO
9	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
8	PRST	端口复位(Port reset)	应用程序将该位置 1 时, 会在该端口上启动复位序列。应用程序必须为复位周期定时, 并在复位序列完成后将该位清零。 ● 0b0: 端口未处于复位状态 ● 0b1: 端口处于复位状态 应用程序必须将该位置 1 并最少保持 10ms, 以在端口上启动复位。	RW
7	PSUSP	端口挂起(Port suspend)	应用程序将此位置 1 以将此端口置于挂起模式。只有此位置 1 时, 模块才会停止发送 SOF。要停止 PHY 时钟, 应用程序必须将端口时钟停止位置 1, 这会使能 PHY 的挂起输入引脚。 此位的读取值反映该端口的当前挂起状态。检测到远程唤醒信号, 或者应用程序将此寄存器中的端口复位位或端口恢复位置 1 后, 模块可将此位清零; 或应用程序将模块中断寄存器中的恢复/远程唤醒检测中断位或断开连接检测中断位 (分别为 USBHS_GINTSTS 中的 WKUINT 或 DISCINT) 置 1, 模块也可将此位清零。 ● 0b0: 端口未处于挂起模式 ● 0b1: 端口处于挂起模式	RW
6	PRES	端口恢复 (Port resume)	应用程序将此位置 1 以在该端口上驱动恢复信号。模块会持续驱动恢复信号直到应用程序将此位清零。 如模块中断寄存器中的端口恢复/远程唤醒检测中断位 (USBHS_GINTSTS 中 WKUINT 位) 指示, 如果模块检测到 USB 远程唤醒序列, 则开始驱动恢复信号, 而无需应用程序进行干预; 如果模块检测到断开连接的情况, 则将此位清零。 此位的读取值指示当前模块是否正在驱动恢复信号。 ● 0b0: 不驱动恢复信号 ● 0b1: 驱动恢复信号	RW
5:4	Reserved	保留	-	-
3	PENCHNG	端口使能/禁止变化 (Port enable/disable change)	该寄存器中的端口使能位 2 的状态发生变化时, 模块将此位置 1。 通过软件对该位写 1 清零。	RW
2	PENA	端口使能(Port enable)	端口执行复位序列后, 只能由模块使能, 并且可以由过流状况、断开连接状况或应用程序将此位清零来禁止。应用程序无法通过对寄存器执行写操作将此位置 1。只能将此位清零来禁止端口。对此位的操作不会触发应用程序的任何中断。 ● 0b0: 禁止端口 ● 0b1: 使能端口	RW
1	PCDET	检测到端口连接 (Port connect detected)	当检测到设备连接时, 模块将此位置 1, 以使用模块中断寄存器中的主机端口中断位 (USBHS_GINTSTS 中的 HPRTINT 位) 触发应用程序的中断。 应用程序必须将此位置 1 才可清除该中断。	RW
0	PCSTS	端口连接状态 (Port connect status)	● 0b0: 端口未连接设备 ● 0b1: 端口已连接设备	RO

34.6.4.8 主机通道 x 特性寄存器 (USBHS_HCCHARx) (x=0~15)

该寄存用于设定主机通道特性。

偏移地址 (Offset) : 0x500+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHENA	CHDIS	ODDFRM	DAD								MC	EPTYP	LSDEV	Reserved	EPDIR	EPNUM								MPSIZ							
Access	RW	RW	RW	RW								RW	RW	RW		RW	RW								RW							

位/位域	标记	位名	功能描述	读写
31	CHENA	通道使能 (Channel enable)	此字段由应用程序软件置 1，并由 USBHS 主机硬件清零。 ● 0b0：禁止通道 ● 0b1：使能通道	RW
30	CHDIS	通道禁止 (Channel disable)	应用程序将此位置 1 以停止通过通道发送/接收数据，即使通过该通道的传输还未完成，停止操作仍然生效。 应用程序必须等待禁止通道的中断以确认通道已经被禁止。	RW
29	ODDFRM	奇数帧 (Odd frame)	此字段由应用程序置位或复位，以分别指示 USBHS 主机必须传输奇数帧或偶数帧。此字段只适用于周期性（同步和中断）事务。 ● 0b0：偶数（微）帧 ● 0b1：奇数（微）帧	RW
28:22	DAD	设备地址 (Device address)	此字段用于指定要与该主机通信的特定设备。	RW
21:20	MC	多重计数 (MC)/ 错误计数 (EC) (Multi Count/ Error Count)	当 USBHS_HCSPLTx 的分离使能位 SPLITEN 复位 0 时：此字段向主机指示此周期性端点每微帧必须执行的事务数。对于非周期性传输，此字段指定内部 DMA 引擎更改仲裁之前此通道要获取的数据包数。 ● 0b00：禁止设置 ● 0b01：此端点每微帧需要发出 1 个事务 ● 0b10：此端点每微帧需要发出 2 个事务 ● 0b11：此端点每微帧需要发出 3 个事务 当 USBHS_HCSPLTx 的分离使能位 SPLITEN 置 1 时：此字段指示周期性分离事务发生错误时要对该事务执行的立即重试次数。此字段至少须置为 0b01。	RW
19:18	EPTYP	端点类型 (Endpoint type)	指示选择的传输类型。 ● 0b00：控制 ● 0b01：同步 ● 0b10：批量 ● 0b11：中断	RW
17	LSDEV	低速设备 (Low-speed device)	此字段由应用程序置 1，表示此通道正在与一个低速设备进行通信。	RW

位/位域	标记	位名	功能描述	读写
16	Reserved	保留	-	-
15	EPDIR	端点方向 (Endpoint direction)	指示通信事务的方向是输入还是输出。 ● 0b0: 输出 ● 0b1: 输入	RW
14:11	EPNUM	端点编号 (Endpoint number)	指示要与该主机通道通信的 USB 设备的端点号。	RW
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	指示与该主机通道通信的设备端点的最大数据包大小。	RW

34.6.4.9 主机通道 x 分离控制寄存器 (USBHS_HCSPLTx) (x=0~15)

偏移地址 (Offset) : 0x504+0x20*x

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SPLITEN	Reserved														COMPLSPLT	XACTPOS	HUBADDR						PRTADDR								
Access	RW															RW	RW	RW						RW								

位/位域	标记	位名	功能描述	读写
31	SPLITEN	分离使能 (Split enable)	应用程序将此位置 1 时, 指示允许该通道执行分离通信事务。	RW
30:17	Reserved	保留	-	-
16	COMPLSPLT	执行完全分离 (Do complete split)	应用程序将此位置 1 时, 可请求主机执行完全分离通信事务。	RW
15:14	XACTPOS	事务位置 (Transaction position)	此字段用于决定随各 OUT 事务发送全部、第一个、中间还是最后一个有效负载。 ● 0b00: 中间。指此事务的中间有效负载 (大于 188 字节) ● 0b01: 结尾。指此事务的最后一个有效负载 (大于 188 字节) ● 0b10: 起始。指此事务的第一个数据有效负载 (大于 188 字节) ● 0b11: 全部。指此事务的全部数据有效负载 (小于或等于 188 字节)	RW
13:7	HUBADDR	集线器地址 (Hub address)	此字段存储事务转发器的集线器设备地址。	RW
6:0	PRTADDR	端口地址 (Port address)	此字段是接收方事务转发器的端口编号。	RW

34.6.4.10 主机通道 x 中断寄存器 (USBHS_HCINTx) (x=0~15)

该寄存器指示在出现 USB 和 AHB 相关事件时通道的状态。当模块中断寄存器中的主机通道中断位 (USBHS_GINTSTS 中的 HCINT 位) 置 1 时, 应用程序必须读取该寄存器。在对寄存器执行读操作之前, 应用程序必须先读取主机全体通道中断(USBHS_HAINT)寄存器, 以获取主机通道 x 中断寄存器的准确通道编号。应用程序必须将该寄存器中的相应位清零, 才能将 USBHS_HAINT 和 USBHS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0x508+0x20*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																						DTERR	FRMOR	BBERR	TXERR	NYET	ACK	NAK	STALL	AHBERR	CHH	XFRC
Access																							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10	DTERR	数据同步错误	应用程序需通过写 1 清除该位。	RW
9	FRMOR	帧溢出错误	应用程序需通过写 1 清除该位。	RW
8	BBERR	串扰错误	产生串扰事件的典型原因是端点发送了一个数据包, 但是数据包长度超过了端点的最大包长。 应用程序需通过写 1 清除该位。	RW
7	TXERR	通信事务错误	指示 USB 上发生下列错误之一: ● CRC 校验失败 ● 超时 ● 位填充错误 ● 错误的 EOP 应用程序需通过写 1 清除该位。	RW
6	NYET	收到 NYET 响应	应用程序需通过写 1 清除该位。	RW
5	ACK	收到/发出 ACK 响应	应用程序需通过写 1 清除该位。	RW
4	NAK	收到 NAK 响应	应用程序需通过写 1 清除该位。	RW
3	STALL	收到 STALL 响应	应用程序需通过写 1 清除该位。	RW
2	AHBERR	AHB 错误	仅当处于内部 DMA 模式下且 AHB 读/写操作期间发生 AHB 错误时才生成此错误。应用程序可通过读取相应的 DMA 通道地址寄存器来获取错误地址。 应用程序需通过写 1 清除该位。	RW
1	CHH	通道停止	因任意 USB 事务错误或为响应应用程序的禁止请求而导致传输非正常结束。 应用程序需通过写 1 清除该位。	RW
0	XFRC	传输完成	未出现任何错误, 正常完成传输。 应用程序需通过写 1 清除该位。	RW

34.6.4.11 主机通道 x 中断屏蔽寄存器 (USBHS_HCINTMSKx) (x=0~15)

该寄存器用于选择屏蔽主机通道中断。

偏移地址 (Offset) : 0x50C+0x20*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																						DTERM	FRMORM	BBERRM	TXERRM	NYETM	ACKM	NAKM	STALLM	AHBERRM	CHHM	XFCRM
Access																							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10	DTERM	数据同步错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
9	FRMORM	帧溢出屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
8	BBERRM	串扰错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	TXERRM	通信事务错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
6	NYETM	NYET 响应接收中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
5	ACKM	ACK 响应接收/发送中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
4	NAKM	NAK 响应接收中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
3	STALLM	收到 STALL 响应中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
2	AHBERRM	AHBERR 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
1	CHHM	通道停止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFCRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

34.6.4.12 主机通道 x 传输大小寄存器 (USBHS_HCTSIZx) (x=0~15)

该寄存器用于设定主机通道传输大小以及数据 PID。

偏移地址 (Offset) : 0x510+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DOPING	DPID	PKTCNT											XFRSIZ																		
Access	RW	RW	RW											RW																		

位/位域	标记	位名	功能描述	读写
31	DOPING	执行 PING 操作 (Do ping)	此位仅用于 OUT 传输。 将此位置 1 可指示主机执行 PING 协议。  注意 请勿针对 IN 传输将此位置 1。如果针对 IN 传输将此位置 1，将禁止通道。	RW
30:29	DPID	数据 PID(Data PID)	应用程序在此字段设置数据通信的初始同步 PID。 主机在此次传输事务过程中保留该字段的设置。 ● 0b00: DATA0 ● 0b01: DATA2 ● 0b10: DATA1 ● 0b11: MDATA/SETUP	RW
28:19	PKTCNT	数据包计数 (Packet count)	应用程序在此字段中设置将要发送或接收的数据包数。 主机每成功发送或接收一个数据包便递减一次计数值。此值达到 0 后，将中断应用程序来指示操作正常完成。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	对于 OUT 操作，此字段为传输期间主机发送的数据字节数。 对于 IN 操作，此字段为应用程序保留给传输的缓冲区大小。对于 IN 事务（周期性和非周期性），应用程序会将此字段编程为最大数据包大小的整数倍。	RW

34.6.4.13 主机通道 x DMA 地址寄存器 (USBHS_HCDMAx) (x=0~15)

该寄存器用于设定主机 DMA 模式时 DMA 地址。

偏移地址 (Offset) : 0x514+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	此字段存储主机从设备端点获取数据或往设备端点发送数据所用 DMA 传输的存储器的地址。每次 AHB 传输结束，该寄存器都会递增。	RW

34.6.5 USBHS 设备模式寄存器

设备模式寄存器会影响设备模式下的模块操作。在主机模式下不得访问设备模式寄存器，因为产生的结果不明确。

34.6.5.1 设备配置寄存器 (USBHS_DCFG)

此寄存器在上电、执行某些控制命令或枚举后，会将模块配置为设备模式。请勿在初始编程后更改该寄存器。

Offset	Bit Position																																	
0x800	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x08220000																																	
Name	Reserved																PFIVL		DAD						Reserved		NZLSOHSK				DSPD			
Access																	RW		RW								RW				RW			

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:11	PFIVL	周期性帧间隔 (Periodic frame interval)	指示一帧内必须使用周期性帧中断通知应用程序的时间点。此功能可用于确定该帧的所有同步通信是否完成。 0b00: 80% (微) 帧间隔 0b01: 85% (微) 帧间隔 0b10: 90% (微) 帧间隔 0b11: 95% (微) 帧间隔	RW
10:4	DAD	设备地址 (Device address)	应用程序必须在执行每个 SetAddress 控制命令后根据命令参数对该字段进行设置。	RW
3	Reserved	保留	-	-
2	NZLSOHSK	非零长度状态 OUT 握手信号 (Non-zero-length status OUT handshake)	在控制传输状态阶段的 OUT 事务期间，当模块收到非零长度数据包后，应用程序可以使用此字段选择要发送的握手信号。 0b0: 将收到的 OUT 数据包（零长度或非零长度）发送给应用程序，并基于设备端点控制寄存器中端点的 NAK 和 STALL 位回复握手信号。 0b1: 收到非零长度状态 OUT 事务时，回复 STALL 握手信号，收到的 OUT 数据包不发送给应用程序。	RW
1:0	DSPD	设备速度 (Device speed)	指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成 chirp 序列后才能确定，同时此速度基于与模块连接的 USB 主机的速度。 0b00: 高速 0b01: 禁止设置 0b10: 禁止设置 0b11: 全速 (USB1.1 收发器时钟为 48MHz)	RW

34.6.5.2 设备控制寄存器 (USBHS_DCTL)

Offset	Bit Position																																
0x804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000002																																
Name	Reserved																				POPGRDNE	CGONAK	SGONAK	CGINAK	SGINAK	TCTL			GONSTS	GINSTS	SDIS	RWUSIG	
Access																					RW	RW	RW	RW	RW	RW	RW			RO	RO	RW	RW

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	POPGRDNE	上电编程完成 (Power-on programming done)	应用程序使用此位指示寄存器从掉电模式唤醒后已完成编程。	RW
10	CGONAK	清零全局 OUT NAK(Clear global OUT NAK)	对此位执行写操作会将全局 OUT NAK 清零。	RW
9	SGONAK	置位全局 OUT NAK(Set global OUT NAK)	对此位执行写操作会将全局 OUT NAK 置 1。 应用程序使用此位在所有 OUT 端点发送 NAK 握手信号。 应用程序只有确定模块中断寄存器中全局 OUT NAK 有效位 (USBHS_GINTSTS 中 GONAKEFF 位) 已清零时, 才可以将此位置 1。	RW
8	CGINAK	清零全局 IN NAK(Clear global IN NAK)	对此位执行写操作会将全局 IN NAK 清零。	RW
7	SGINAK	置位全局 IN NAK(Set global IN NAK)	对此字段执行写操作会将全局非周期性 IN NAK 置 1。应用程序使用此位使所有非周期性 IN 端点发送 NAK 握手信号。 应用程序只有确定模块中断寄存器中全局 IN NAK 有效位 (USBHS_GINTSTS 中 GINAKEFF 位) 已清零时, 才可以将此位置 1。	RW
6:4	TCTL	测试控制(Test control)	0b000: 测试模式禁止 0b001: Test_J 模式 0b010: Test_K 模式 0b011: Test_SE0_NAK 模式 0b100: Test_Packet 模式 0b101: Test_Force_Enable - 其他: 禁止设置	RW
3	GONSTS	全局 OUT NAK 状态(Global OUT NAK status)	0b0: 将根据 FIFO 状态和 NAK 和 STALL 位设置发送握手信号。 0b1: 无论 RxFIFO 中是否还有空闲空间都不接收数据。除 SETUP 事务之外, 对所有收到的数据包回复 NAK 握手信号。所有同步类型的 OUT 数据包都将被丢弃。	RO

位/位域	标记	位名	功能描述	读写
2	GINSTS	全局 IN NAK 状态(Global IN NAK status)	● 0b0: 将根据发送 FIFO 中的数据可用性回复握手信号。 ● 0b1: 使所有非周期性 IN 端点回复 NAK 握手信号, 无需考虑发送 FIFO 中的数据可用性。	RO
1	SDIS	软断连(Soft disconnect)	应用程序使用该位向 USBHS 模块发出执行软断开的信号。该位置 1 时, 主机不会看到设备已连接, 且该设备也不会接收 USB 上的信号。在应用程序将此位清零之前, 模块会保持断开状态。 ● 0b0: 正常工作。此位在软断连之后清零, 会使主机收到设备已连接的事件。重新连接设备之后, USB 主机重新启动设备枚举。 ● 0b1: 使主机收到设备断开连接的事件。 全速时, 软断连的最小时间规定如下: ● 挂起状态: 最小时间为 1ms+2.5μs ● 空闲状态: 2.5μs ● 非空闲或挂起状态: 2.5μs	RW
0	RWUSIG	发送远程唤醒信号(Remote wakeup signaling)	应用程序将此位置 1 时, 模块会启动远程唤醒信号, 以唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出挂起状态。根据 USB2.0 规范, 应用程序必须在将此位置 1 之后的 1ms 到 15ms 内将其清零。	RW

34.6.5.3 设备状态寄存器 (USBHS_DSTS)

此寄存器指示模块在出现 USB 相关事件时的状态。发生中断时，必须从设备全体段端点中断寄存器 (USBHS_DAINTE) 读取发生中断的端点信息。

Offset	Bit Position																																	
0x808	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000002																																	
Name	Reserved								LNSTS				FNSOF												Reserved				EERR		ENUMSPD		SUSPSTS	
Access									RO				RO																RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:22	LNSTS	USB 总线状态	[23]: D+的逻辑电平 [22]: D-的逻辑电平	RO
21:8	FNSOF	接收 SOF 帧编号	-	RO
7:4	Reserved	保留	-	-
3	EERR	不定错误(Erratic error)	模块将该位置 1 以报告任何不定错误。 由于不定错误，USBHS 控制器会进入挂起状态，并且会 USBHS_GINTSTS 寄存器的早期挂起位 (USBHS_GINTSTS 中的 ESUSP 位) 生成一个中断。 如果早期挂起中断是由不定错误触发，则应用程序只能执行软断开来恢复通信。	RO
2:1	ENUMSPD	枚举速度 (Enumerated speed)	指示 USBHS 控制器通过 chirp 序列检测速度后被枚举成的速度。 ● 0b00: 高速 ● 0b01: Reserved ● 0b10: Reserved ● 0b11: 全速 (PHY 时钟运行频率为 48MHz)	RO
0	SUSPSTS	挂起状态 (Suspend status)	在设备模式下，只要在 USB 上检测到挂起状态，该位就会置 1。当 USB 总线上的空闲状态保持 3ms，模块便会进入挂起状态。出现以下情况时，模块会退出挂起状态： ● USB 数据线上有活动 ● 应用程序对 USBHS_DCTL 寄存器的远程唤醒信号位 (USBHS_DCTL 中的 RWUSIG 位) 执行写操作。	RO

34.6.5.4 设备 IN 端点通用中断屏蔽寄存器 (USBHS_DIEPMSK)

此寄存器与全体端点的各个 USBHS_DIEPINTx 寄存器配合使用，以便在每个 IN 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBHS_DIEPINTx 寄存器中的 IN 端点中断。默认情况下，状态中断都被屏蔽。

Offset	Bit Position																																			
0x810	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																			NAKM	Reserved							TXFURM	Reserved	INEPNEM	INEPNMM	TTXFEMSK	TOM	Reserved	EPDM	XFCRM
Access																				RW								RW		RW	RW	RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	NAKM	NAK 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
12:9	Reserved	保留	-	-
8	TXFURM	发送 FIFO 下溢中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	Reserved	保留	-	-
6	INEPNEM	IN 端点 NAK 有效中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
5	INEPNMM	EP 不匹配时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
4	TTXFEMSK	TxFIFO 为空时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
3	TOM	超时中断屏蔽 (非同步端点)	0b0: 屏蔽中断 0b1: 使能中断	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFCRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

34.6.5.5 设备 OUT 端点通用中断屏蔽寄存器 (USBHS_DOEPMASK)

此寄存器与全体端点的各个 USBHS_DOEPINTx 寄存器配合使用，以便在每个 OUT 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBHS_DOEPINTx 寄存器中的 OUT 端点中断。默认情况下，状态中断都被屏蔽。

Offset	Bit Position																																	
0x814	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																NYETM	Reserved								OPEM	Reserved	B2BSTUP	Reserved	OTEPDM	STUPM	Reserved	EPDM	XFCRM
Access																	RW									RW		RW		RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14	NYETM	NYET 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
13:9	Reserved	保留	-	-
8	OPEM	OUT 数据包错误中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	Reserved	保留	-	-
6	B2BSTUP	接收到连续的 SETUP 数据包中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断  说明 仅适用于控制 OUT 端点。	RW
5	Reserved	保留	-	-
4	OTEPDM	端点禁止时接收到 OUT 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断  说明 仅适用于控制 OUT 端点。	RW
3	STUPM	SETUP 阶段完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断  说明 仅适用于控制端点。	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFCRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

34.6.5.6 设备全体端点中断寄存器 (USBHS_DAIN_T)

当端点上发生有效事件时，USBHS_DAIN_T 寄存器将通过 USBHS_GINTSTS 寄存器中的设备 OUT 端点中断位或设备 IN 端点中断位（分别为 USBHS_GINTSTS 中的 OEPINT 或 IEPINT 位）来中断应用程序。每个端点对应一个中断位，OUT 端点和 IN 端点均最多有 16 个中断位。双向端点将使用相应的 IN 和 OUT 中断位。当应用程序将相应设备端点 x 中断寄存器(USBHS_DIEPINT_x/USBHS_DOEPINT_x)中的位置 1 和清零时，此寄存器中的相应位也将置 1 和清零。

Offset	Bit Position																															
0x818	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OEPINT																IEPINT															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	OEPINT	OUT 端点中断位 (OUT endpoint interrupt bits)	对应 OUT 端点 15~OUT 端点 0： ● 0b0：该端点未发生中断 ● 0b1：该端点发生中断  说明 每个 Bit 对应一个 OUT 端点。	RW
15:0	IEPINT	IN 端点中断位 (IN endpoint interrupt bits)	对应 IN 端点 15~IN 端点 0： ● 0b0：该端点未发生中断 ● 0b1：该端点发生中断  说明 每个 Bit 对应一个 IN 端点。	RW

34.6.5.7 设备全体端点中断屏蔽寄存器 (USBHS_DAINMSK)

USBHS_DAINMSK 寄存器与设备端点中断寄存器结合使用，在设备端点上发生事件时中断应用程序。但是，与该中断相对应的 USBHS_DAIN 寄存器位仍会置 1。

Offset	Bit Position																															
0x81C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OEPIINTM																IEPIINTM															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	OEPIINTM	OUT 端点中断屏蔽位(OUT endpoint interrupt mask bits)	对应 OUT 端点 15~OUT 端点 0: ● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 每个 Bit 对应一个 OUT 端点。	RW
15:0	IEPIINTM	IN 端点中断屏蔽位(IN endpoint interrupt mask bits)	对应 IN 端点 15~IN 端点 0: ● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 每个 Bit 对应一个 IN 端点。	RW

34.6.5.8 设备阈值控制寄存器 (USBHS_DTHRCTL)

Offset	Bit Position																															
0x830	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0C100020																															
Name	Reserved				ARPEN	Reserved	RXTHRLEN								RXTHREN	Reserved				TXTHRLEN								ISOTHREN	NONISOTHREN			
Access					RW		RW								RW					RW								RW	RW			

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27	ARPEN	仲裁器驻留使能	该位用于控制内部 DMA 仲裁对 IN 端点的驻留。当阈值使能且该位置 1 后，DMA 会对收到令牌的 IN 端点驻留其仲裁。采用这种方式可以避免出现下溢情况。默认情况下使能驻留。	RW
26	Reserved	保留	-	-
25:17	RXTHRLEN	接收阈值长度	该字段以双字为单位指定接收阈值的大小。该字段指定模块在 AHB 上启动传输之前在 USB 上接收的数据量。阈值长度最小值为八个双字。推荐 RXTHRLEN 和设定的 AHB 突发传输长度相同。	RW
16	RXTHREN	接收阈值使能	该位置 1 时，模块会使能接收方向的阈值。	RW
15:11	Reserved	保留	-	-
10:2	TXTHRLEN	发送阈值长度	该字段以双字为单位指定发送阈值的大小。该字段指定模块在 USB 上启动传输之前相应端点发送 FIFO 中的数据量。阈值长度最小值为八个双字。该字段控制同步和非同步 IN 端点阈值。推荐 TXTHRLEN 和设定的 AHB 突发传输长度相同。	RW
1	ISOTHREN	ISO IN 端点阈值使能	该位置 1 时，模块会使能同步 IN 端点的阈值。	RW
0	NONISOTHREN	非同步 IN 端点阈值使能	该位置 1 时，模块会使能非同步 IN 端点的阈值。	RW

34.6.5.9 设备 IN 端点 FIFO 空中断屏蔽寄存器 (USBHS_DIEPMPMSK)

此寄存器用于控制 IN 端点 FIFO 空中断的生成。

Offset	Bit Position																															
0x834	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																INEPTXFEM															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	INEPTXFEM	IN EP Tx FIFO 空中断屏蔽位(IN EP Tx FIFO empty interrupt mask bits)	这些位用作 USBHS_DIEPINTx 的屏蔽位。 对应 IN 端点 15~IN 端点 0: ● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 每个 Bit 对应一个 IN 端点的 TXFE 中断。	RW

34.6.5.10 设备单个端点中断寄存器 (USBHS_DEACHINT)

Offset	Bit Position																																
0x838	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved															OEPI1INT	Reserved															IEPI1INT	Reserved
Access																RW																RW	

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	OEPI1INT	OUT 端点 1 中断位	0b0: OUT 端点 1 未发生中断 0b1: OUT 端点 1 发生中断	RW
16:2	Reserved	保留	-	-
1	IEPI1INT	IN 端点 1 中断位	0b0: IN 端点 1 未发生中断 0b1: IN 端点 1 发生中断	RW
0	Reserved	保留	-	-

34.6.5.11 设备单个端点中断屏蔽寄存器 (USBHS_DEACHINTMASK)

Offset	Bit Position																																
0x83C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved															OEPI1INTM	Reserved															IEPI1INTM	Reserved
Access																RW																RW	

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17	OEPI1INTM	OUT 端点 1 中断屏蔽位	0b0: 屏蔽中断 0b1: 使能中断	RW
16:2	Reserved	保留	-	-
1	IEPI1INTM	IN 端点 1 中断屏蔽位	0b0: 屏蔽中断 0b1: 使能中断	RW
0	Reserved	保留	-	-

34.6.5.12 设备 IN 端点 1 中断屏蔽寄存器 (USBHS_DIEPEACHMSK1)

Offset	Bit Position																																								
0x844	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
Reset	0x00000000																																								
Name	Reserved																			NAKM	Reserved											TXFURM	Reserved	INEPNEM	INEPNMM	TTXFEMSK	TOM	Reserved	EPDM	XFCRM	
Access																				RW												RW		RW	RW	RW	RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	NAKM	NAK 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
12:9	Reserved	保留	-	-
8	TXFURM	发送 FIFO 下溢中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	Reserved	保留	-	-
6	INEPNEM	IN 端点 NAK 有效中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
5	INEPNMM	EP 不匹配时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
4	TTXFEMSK	TxFIFO 为空时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
3	TOM	超时中断屏蔽 (非同步端点)	0b0: 屏蔽中断 0b1: 使能中断	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

34.6.5.13 设备 OUT 端点 1 中断屏蔽寄存器 (USBHS_DOEPEACHMSK1)

Offset	Bit Position																																											
0x884	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
Reset	0x00000000																																											
Name	Reserved																		NYETM		Reserved						OPEM		Reserved		B2BSTUP		Reserved		OTEPDM		STUPM		Reserved		EPDM		XFCM	
Access																			RW								RW				RW				RW		RW				RW		RW	

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14	NYETM	NYET 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
13:9	Reserved	保留	-	-
8	OPEM	OUT 数据包错误中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	Reserved	保留	-	-
6	B2BSTUP	接收到连续的 SETUP 数据包中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断 仅适用于控制 OUT 端点。	RW
5	Reserved	保留	-	-
4	OPEPDM	端点禁止时接收到 OUT 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断 仅适用于控制 OUT 端点。	RW
3	STUPM	SETUP 阶段完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断 仅适用于控制端点。	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFCM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

34.6.5.14 设备控制 IN 端点 0 控制寄存器 (USBHS_DIEPCTL0)

此寄存器用于控制控制传输端点 0。

Offset	Bit Position																																
0x900	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00008000																																
Name	EPENA	EPDIS	Reserved		SNAK	CNAK		TXFNUM			STALL	Reserved		EPTYP	NAKSTS	Reserved	USBAEP	Reserved											MPSIZ				
Access	RW	RW			RW	RW		RW			RW			RO	RO		RO																RW

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点 0 上启动数据发送。在此端点上触发以下任一中断之前，模块会将此位清零： ● SETUP 阶段完成 ● 端点禁止 ● 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传输完成之前，应用程序也可将此位置 1，以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置 1 前，模块会将此位清零。只有在该端点的端点使能位置 1 后，应用程序才可将该位置 1。	RW
29:28	Reserved	保留	-	-
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位，应用程序可以控制端点上 NAK 握手信号的发送。模块也可在端点接收到 SETUP 数据包后将该端点的此位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	TXFNUM	TxFIFO 编号 (TxFIFO number)	该值设置为分配给 IN 端点 0 的 FIFO 编号。只能使用 TX-FIFO0。	RW
21	STALL	STALL 握手 (STALL handshake)	应用程序只能将此位置 1，端点接收到 SETUP 令牌时，模块会将此位清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位均置 1，则 STALL 位优先。	RW
20	Reserved	保留	-	-
19:18	EPTYP	端点类型 (Endpoint type)	硬件设置为'0b00'，表示控制类型的端点。	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0：模块根据 FIFO 状态回复非 NAK 握手 ● 0b1：模块在此端点上回复 NAK 握手 当此位置 1 时（无论是被应用程序还是被模块），即使 TxFIFO 中仍有数据可用，模块也会停止发送数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	USBAEP	USB 活动端点 (USB active endpoint)	此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。	RO
14:2	Reserved	保留	-	-
1:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。 ● 0b00: 64 字节 ● 0b01: 32 字节 ● 0b10: 16 字节 ● 0b11: 8 字节	RW

34.6.5.15 设备 IN 端点 x 控制寄存器 (USBHS_DIEPCTLx) (x=1~15)

应用程序使用此寄存器控制各个逻辑端点 (端点 0 除外) 的行为。

偏移地址 (Offset) : 0x900+0x20*x

Offset	Bit Position																																		
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	EPENA	EPDIS	SODDFRM	SD0PID/SEVNFIRM	SNACK	CNACK	TXFNUM					STALL	Reserved	EPTYP	NAKSTS	EONUM/DPID	USBAEP	Reserved					MPSIZ												
Access	RW	RW	RW	RW	RW	RW	RW					RW		RO	RO	RO	RO							RW											

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点上启动数据发送。在此端点上触发以下任一中断之前, 模块会将此位清零: - 端点禁止 - 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传输完成之前, 应用程序也可将此位置 1, 以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后, 才能将端点视为禁止端点。在端点禁止中断位置 1 前, 模块会将此位清零。只有在该端点的端点使能位置 1 后, 应用程序才可将该位置 1。	RW
29	SODDFRM	设置奇数帧 (Set odd frame)	仅适用于同步 IN 和 OUT 端点。对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为奇数帧。	RW
28	SD0PID/SEVNFRM	设置 DATA0 PID/设置偶数帧	SD0PID: 仅适用于中断/批量 IN 端点。对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。 SEVNFRM: 仅适用于同步 IN 端点。对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。	RW
27	SNAK	置位 NAK 位 (Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位, 应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后, 模块也可以将 IN 端点的这个位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	TXFNUM	TxFIFO 编号 (TxFIFO number)	这些位用于指定与此端点相关联的 FIFO 编号。必须为每个有效的 IN 端点设置单独的 FIFO 编号。此字段仅针对 IN 端点有效。	RW
21	STALL	STALL 握手 (STALL handshake)	应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1, 则 STALL 位优先。只有应用程序能够将此位清零, 而模块则不能。	RW
20	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
19:18	EPTYP	端点类型 (Endpoint type)	● 0b00: 控制 ● 0b01: 同步 ● 0b10: 批量 ● 0b11: 中断	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0: 模块根据 FIFO 状态回复非 NAK 握手 ● 0b1: 模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时: ● 对于非同步 IN 端点: 即使 TxFIFO 中存在可用数据, 模块也会停止通过 IN 端点发送任何数据。 ● 对于同步 IN 端点: 即使 TxFIFO 中存在可用数据, 模块也会发送长度为零的数据包。 无论此位如何设置, 模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	EONUM/DPID	偶数/奇数帧/端点数据 PID	EONUM: 仅适用于同步 IN 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程, 以便此端点发送/接收同步数据。 ● 0b0: 偶数帧 ● 0b1: 奇数帧 DPID: 仅适用于中断/批量 IN 端点。 包含此端点上将要接收或发送的数据包的 PID。端点激活后, 应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SDOPID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。 ● 0b0: DATA0 ● 0b1: DATA1	RO
15	USBAEP	USB 活动端点 (USB active endpoint)	指示此端点在当前配置和接口中是否激活。检测到 USB 复位后, 模块会为所有端点 (端点 0 除外) 将此位清零。接收到 SetConfiguration 和 SetInterface 命令后, 应用程序必须相应地对端点寄存器进行编程并将此位置 1。	RO
14:11	Reserved	保留	-	-
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。	RW

34.6.5.16 设备 IN 端点 x 中断寄存器 (USBHS_DIEPINTx) (x=0~15)

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 IN 端点中断位 (USBHS_GINTSTS 中的 IEPINT 位) 置 1 时, 应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前, 必须先读取设备全体端点中断(USBHS_DAINTEP)寄存器, 以获取设备端点 x 中断寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零, 才能将 USBHS_DAINTEP 和 USBHS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0x908+0x20*x

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
Reset	0x00000080																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
Name	Reserved																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													</

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	NAK	NAK 中断	当设备发出或收到 NAK 时, 模块将生成该中断。如果是同步 IN 端点, 由于 Tx FIFO 中无数据可发而发送长度为零的数据包时也会生成该中断。通过软件写 1 清零。	RW
12	BERR	Babble 错误中断	通过软件写 1 清零。	RW
11	PKTDRPSTS	数据包丢弃状态	该位用于向应用程序指示有 ISOC OUT 数据包被丢弃。该位没有相应的中断屏蔽位, 也不会生成中断。通过软件写 1 清零。	RW
10:9	Reserved	保留	-	-
8	TXFIFOUDRN	发送 FIFO 下溢	当模块检测到该端点发送 FIFO 下溢时, 将生成该中断。相关性: 该中断仅在使能了阈值时有效。通过软件写 1 清零。	RW
7	TXFE	发送 FIFO 为空	当此端点的 Tx FIFO 为半空或全空时, 此中断被置位。Tx FIFO 为半空还是全空状态由 USBHS_GAHBCFG 寄存器中的 Tx FIFO 空白别位 (USBHS_GAHBCFG 中的 TXFELVL 位) 决定。	RO
6	INEPNE	IN 端点 NAK 有效	当应用程序通过向 USBHS_DIEPCTLx 中的 CNAK 位写入数据来将 IN 端点 NAK 清零时, 此位可被清零。该中断指示模块已对 (由应用程序或模块) 置 1 的 NAK 采样, 结果已生效。该中断指示由应用程序置 1 的 IN 端点 NAK 位已在模块中起作用。此中断不保证在 USB 上发送了 NAK 握手信号。STALL 位的优先级高于 NAK 位。软件写 1 也可将此位清零。	RW
5	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
4	TTXFE	TxFIFO 为空时接收到 IN 令牌	仅适用于非周期性 IN 端点。 当和该端点相对应的 TxFIFO（周期性/非周期性）为空时，接收到 IN 令牌，从而产生中断。 通过软件写 1 清零。	RW
3	TOC	超时条件	仅适用于控制 IN 端点。 指示该端点对最近收到的 IN 令牌响应超时。 通过软件写 1 清零。	RW
2	Reserved	保留	-	-
1	EPDISD	端点禁止中断	此位指示该端点已经由应用程序禁止掉。 通过软件写 1 清零。	RW
0	XFRC	传输完成中断	此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。 通过软件写 1 清零。	RW

34.6.5.17 设备 IN 端点 0 传输大小寄存器 (USBHS_DIEPTSZ0)

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位 (USBHS_DIEPCTL0 中的 EPENA) 使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。
非零端点使用端点 1~15 的寄存器。

Offset	Bit Position																																
0x910	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved												PKTCNT	Reserved										XFRSIZ									
Access													RW											RW									

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:19	PKTCNT	数据包计数 (Packet count)	指示端点 0 的一次数据传输包含的数据包个数。 每次从 TxFIFO 读取数据包（最大大小或短数据包）时，此字段将递减。	RW
18:7	Reserved	保留	-	-
6:0	XFRSIZ	传输大小 (Transfer size)	指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次向 TxFIFO 写入来自外部存储器的数据包时，模块会使此字段递减。	RW

34.6.5.18 设备 IN 端点 x 传输大小寄存器 (USBHS_DIEPTSIZE) (x=1~15)

在使能该端点之前，应用程序必须修改此寄存器。通过 USBHS_DIEPTLx 寄存器中的端点使能位 (USBHS_DIEPTLx 中的 EPENA 位) 使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

偏移地址 (Offset) : 0x910+0x20*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved	MCNT		PKTCNT										XFRSIZ																			
Access		RW		RW										RW																			

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	MCNT	多重计数(Multi count)	对于周期性 IN 端点，此字段指示在 USB 上每帧必须发送的数据包数。模块使用此字段计算同步 IN 端点的数据 PID。 ● 0b00：禁止设置 ● 0b01：1 个数据包 ● 0b10：2 个数据包 ● 0b11：3 个数据包	RW
28:19	PKTCNT	数据包计数 (Packet count)	指示该端点上的一次数据传输包含的数据包个数。每次从 TxFIFO 读取数据包（最大大小或短数据包）时，此字段将递减。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。每次向 TxFIFO 写入来自外部存储器的数据包时，模块会使此字段递减。	RW

34.6.5.19 设备 IN 端点 x DMA 地址寄存器 (USBHS_DIEPDMAx) (x=0~15)

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。

偏移地址 (Offset) : 0x914+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	该位包含使用 DMA 进行端点上数据存储时的外部存储区起始地址。  注意 对于控制端点，该字段所指向的存储区也用于存储控制 OUT 数据包以及 SETUP 事务数据包。连续接收到三个以上的 SETUP 数据包时，存储器中的 SETUP 数据包将被覆盖。每次进行 AHB 传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	RW

34.6.5.20 设备 IN 端点发送 FIFO 状态寄存器 (USBHS_DTXFSTx) (x=0~15)

偏移地址 (Offset) : 0x918+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000800																															
Name	Reserved																INEPTFSAV															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	INEPTFSAV	IN 端点 Tx FIFO 可用空间(IN endpoint Tx FIFO space available)	指示端点 Tx FIFO 中的可用空闲空间大小。以 32 位字为单位： ● 0: 端点 Tx FIFO 已满 ● 1: 1 个字可用 ● 2: 2 个字可用 ● ● n: n 个字可用 (n 取值为寄存器 USBHS_DIEPTFx.INEPTXFD 的值)	RO

34.6.5.21 设备控制 OUT 端点 0 控制寄存器 (USBHS_DOEPTL0)

此寄存器用于控制控制传输端点 0。

Offset	Bit Position																															
0xB00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008000																															
Name	EPENA	EPDIS	Reserved		SNAK	CNAK	Reserved				STALL	SNPM	EPTYP	NAKSTS	Reserved	USBAEP	Reserved										MPSIZ					
Access	RW	RW			RW	RW					RW	RW	RO	RO		RO											RW					

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点 0 上启动数据发送。在此端点上触发以下任一中断之前，模块会将此位清零： ● SETUP 阶段完成 ● 端点禁止 ● 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	应用程序无法禁止控制 OUT 端点 0。	RW
29:28	Reserved	保留	-	-
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位，应用程序可以控制端点上 NAK 握手信号的发送。模块也可在端点接收到 SETUP 数据包后将该端点的此位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	Reserved	保留	-	-
21	STALL	STALL 握手 (STALL handshake)	此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RW
20	SNPM	监听模式 (Snoop mode)	此位用于将端点配置为监听模式。在监听模式下，模块不会在将 OUT 数据包传输到应用存储区前检查其是否正确。	RW
19:18	EPTYP	端点类型 (Endpoint type)	硬件设置为‘0b00’，表示控制类型的端点。	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0：模块根据 FIFO 状态回复非 NAK 握手 ● 0b1：模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时，即使 RxFIFO 中存在空间可继续容纳收到的数据包，模块也会停止接收数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	USBAEP	USB 活动端点 (USB active endpoint)	此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。	RO
14:2	Reserved	保留	-	-
1:0	MPSIZ	最大数据包大小 (Maximum packet size)	控制 OUT 端点 0 的最大数据包大小与在控制 IN 端点 0 中进行编程的值相同。 ● 0b00: 64 字节 ● 0b01: 32 字节 ● 0b10: 16 字节 ● 0b11: 8 字节	RW

34.6.5.22 设备 OUT 端点 x 控制寄存器 (USBHS_DOEPTCTLx) (x=1~15)

应用程序使用此寄存器控制各个逻辑端点 (端点 0 除外) 的行为。

偏移地址 (Offset) : 0xB00+0x20*x

Offset	Bit Position																																	
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	EPENA	EPDIS	SD1PID/SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Reserved					STALL	SNPM	EPTYP	NAKSTS	EONUM/DPID	USBAEP	Reserved					MPSIZ											
Access	RW	RW	RW	RW	RW	RW						RW	RW	RO	RO	RO	RO																	

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点上启动数据发送。在此端点上触发以下任一中断之前, 模块会将此位清零: - 端点禁止 - 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传送完成之前, 应用程序也可将此位置 1, 以停止端点上的数据发送/接收。应用程序必须等到发生端点禁止中断后, 才能将端点视为禁止端点。在端点禁止中断位置 1 前, 模块会将此位清零。只有在该端点的端点使能位置 1 后, 应用程序才可将该位置 1。	RW
29	SD1PID/SODDFRM	设置 DATA1 PID/设置奇数帧	SD1PID : 仅适用于中断/批量 OUT 端点。对此字段进行写操作会将此寄存器中的端点数据 PID(DPID)字段设置为 DATA1。 SODDFRM : 仅适用于同步 OUT 端点。对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为奇数帧。	RW
28	SD0PID/SEVNFRM	设置 DATA0 PID/设置偶数帧	SD0PID : 仅适用于中断/批量 OUT 端点。对此字段进行写操作会将此寄存器中的端点数据 PID(DPID)字段设置为 DATA0。 SEVNFRM : 仅适用于同步 OUT 端点。对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为偶数帧。	RW
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位, 应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后, 模块也可以将 OUT 端点的这个位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	Reserved	保留	-	-
21	STALL	STALL 握手 (STALL handshake)	此端点接收到 SETUP 令牌时, 应用程序只能将此位置 1, 而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1, 则 STALL 位优先。只有应用程序能够将此位清零, 而模块则不能。	RW

位/位域	标记	位名	功能描述	读写
20	SNPM	监听模式(Snoop mode)	此位用于将端点配置为监听模式。在监听模式下，模块不会再检查接收数据的正确性。	RW
19:18	EPTYP	端点类型 (Endpoint type)	● 0b00: 控制 ● 0b01: 同步 ● 0b10: 批量 ● 0b11: 中断	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0: 模块根据 FIFO 状态回复非 NAK 握手 ● 0b1: 模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时，即使 RxFIFO 存在空间可容纳传入数据包，模块也会停止在 OUT 端点上接收任何数据。 无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	EONUM/DPID	偶数/奇数帧/端点数据 PID	EONUM: 仅适用于同步 OUT 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。 ● 0b0: 偶数帧 ● 0b1: 奇数帧 DPID: 仅适用于中断/批量 OUT 端点。 包含此端点上将要接收或发送的数据包的 PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SD0PID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。 ● 0b0: DATA0 ● 0b1: DATA1	RO
15	USBAEP	USB 活动端点 (USB active endpoint)	指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。	RO
14:11	Reserved	保留	-	-
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。	RW

34.6.5.23 设备 OUT 端点 x 中断寄存器 (USBHS_DOEPINTx) (x=0~15)

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当 USBHS_GINTSTS 寄存器中的 OUT 端点中断位 (USBHS_GINTSTS 中的 OEPINT 位) 置 1 时, 应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前, 必须先读取 USBHS_DAINTE 寄存器, 以获取 USBHS_DOEPINTx 寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零, 才能将 USBHS_DAINTE 和 USBHS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0xB08+0x20*x

Offset	Bit Position																																			
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																		NYET	Reserved										B2BSTUP	Reserved	OTEPDIS	STUP	Reserved	EPDISD	XFRC
Access																			RW											RW		RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14	NYET	NYET 中断	当非同步 OUT 端点回复 NYET 握手信号时, 模块将产生此中断。软件写 1 也可将此位清零。	RW
13:7	Reserved	保留	-	-
6	B2BSTUP	接收到连续的 SETUP 数据包	仅适用于控制 OUT 端点。此位指示该端点已接收到三个以上的连续 SETUP 数据包。软件写 1 也可将此位清零。	RW
5	Reserved	保留	-	-
4	OTEPDIS	端点禁止时接收到 OUT 令牌	仅适用于控制 OUT 端点。指示在尚未使能端点时接收到 OUT 令牌, 从而产生中断。通过软件写 1 清零。	RW
3	STUP	SETUP 阶段完成	仅适用于控制 OUT 端点。指示控制端点的 SETUP 阶段已完成, 当前控制传输中不再接收到连续的 SETUP 数据包。在此中断上, 应用程序可以对接收到的 SETUP 数据包进行解码。通过软件写 1 清零。	RW
2	Reserved	保留	-	-
1	EPDISD	端点禁止中断	此位指示该端点已经由应用程序禁止掉。通过软件写 1 清零。	RW
0	XFRC	传输完成中断	此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。通过软件写 1 清零。	RW

34.6.5.24 设备 OUT 端点 0 传输大小寄存器 (USBHS_DOEPTSIZE0)

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位 (USBHS_DIEPCTL0 中的 EPENA) 使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1~15 的寄存器。

Offset	Bit Position																															
0xB10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	STUPCNT		Reserved								PKTCNT	Reserved												XFRSIZ							
Access		RW										RW													RW							

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	STUPCNT	SETUP 数据包计数 (SETUP packet count)	此字段指定端点能连续接收的 SETUP 数据包数量。 0b00: 禁止设置 0b01: 1 个数据包 0b10: 2 个数据包 0b11: 3 个数据包	RW
28:20	Reserved	保留	-	-
19	PKTCNT	数据包计数 (Packet count)	一次传输中应该接收到数据包数量。 在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。	RW
18:7	Reserved	保留	-	-
6:0	XFRSIZ	传输大小 (Transfer size)	指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从 RxFIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。	RW

34.6.5.25 设备 OUT 端点 x 传输大小寄存器 (USBHS_DOEPTSIZE_x) (x=1~15)

在使能该端点之前，应用程序必须修改此寄存器。通过 USBHS_DOEPTL_x 寄存器中的端点使能位 (USBHS_DOEPTL_x 中的 EPENA 位) 使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

偏移地址 (Offset) : 0xB10+0x20*x

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	RXDPID/STUPCNT				PKTCNT								XFRSIZ																		
Access		RW					RW								RW																	

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	RXDPID/ STUPCNT	接收到的数据 PID/SETUP 数据 包计数	RXDPID : 仅适用于同步 OUT 端点。这是此端点收到的上一个数据包的 PID。 ● 0b00: DATA0 ● 0b01: DATA2 ● 0b10: DATA1 ● 0b11: MDATA STUPCNT : 仅适用于控制 OUT 端点。此字段指定端点能连续接收的 SETUP 数据包数量。 ● 0b00: 禁止设置 ● 0b01: 1 个数据包 ● 0b10: 2 个数据包 ● 0b11: 3 个数据包	RW
28:19	PKTCNT	数据包计数 (Packet count)	指示该端点上的一次数据传输包含的数据包个数。每次向 RxFIFO 写入数据包 (最大大小或短数据包) 后, 此字段将递减。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	此字段包含当前端点的一次数据传输包含的数据量, 以字节为单位。仅当应用程序传输完这些数据后, 模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小, 以在每个数据包结束时中断。每次从 RxFIFO 读取数据包并将其写入外部存储器时, 模块会使此字段递减。	RW

34.6.5.26 设备 OUT 端点 x DMA 地址寄存器 (USBHS_DOEPDMAx) (x=0~15)

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。。

偏移地址 (Offset) : 0xB14+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	该位包含使用 DMA 进行端点上数据发送时的外部存储区起始地址。 ! 注意 对于控制端点，该字段所指向的存储区也用于存储控制 OUT 数据包以及 SETUP 事务数据包。连续接收到三个以上的 SETUP 数据包时，存储器中的 SETUP 数据包将被覆盖。每次进行 AHB 传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	RW

34.6.6 USBHS 时钟门控控制寄存器

通过门控时钟控制寄存器控制 HCLK 和 PHY 时钟从而降低功耗。

34.6.6.1 时钟门控控制寄存器 (USBHS_GCCTL)

此寄存器在主机模式和设备模式下均可用。

Offset	Bit Position																																			
0xE00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																								SUSP		PHYSLEEP		ENL1GTG		Reserved		GATEHCLK		STPPCLK	
Access																									RO		RO		RW				RW		RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SUSP	Deep Sleep (L1 suspended) 状态位	0b0: 未处于 Deep Sleep 状态 0b1: 处于 Deep Sleep 状态	RO
6	PHYSLEEP	PHY SLEEP 模式状态位 (PHY in sleep)	0b0: PHY 未处于 Sleep 状态 0b1: PHY 处于 Sleep 状态	RO
5	ENL1GTG	L1 时钟门控 (Enable Sleep Clock Gating)	该位用于 L1 模式时时钟门控控制。	RW
4:2	Reserved	保留	-	-
1	GATEHCLK	门控 HCLK (Gate HCLK)	当 USB 通信挂起或会话无效时，应用程序会将此位置 1，以停止对除 AHB 总线从接口、主接口和唤醒逻辑之外的模块提供时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。	RW
0	STPPCLK	停止 PHY 时钟 (Stop PHY clock)	当 USB 通信挂起、会话无效或设备断开连接时，应用程序将此位置 1 以停止 PH 时钟。当 USB 恢复通信时，应用程序将此位清零。	RW

35 USB2.0 全速模块 (USBFS)

35.1 简介

USB 全速 (USBFS) 控制器为便携式设备提供了一套 USB 通信解决方案。USBFS 控制器支持主机模式和设备模式，且芯片内部集成全速 PHY。主机模式下，USBFS 控制器支持全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 收发器，而设备模式下则仅支持全速 (FS, 12Mbit/s) 收发器。USBFS 控制器支持 USB 2.0 协议所定义的所有四种传输方式 (控制传输、批量传输、中断传输和同步传输)。该 USBFS 控制器支持 LPM(Link Power Management)功能。

遵循协议如下：

- Universal Serial Bus Revision 2.0 Specification
- USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007
- Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007

35.2 主要特性

35.2.1 通用特性

- 内置片上 USB2.0 全速 PHY
- 支持主机模式和设备模式
- 支持 FS SOF 以及低速“Keep-alive”令牌并具有以下功能：
 - ▶ SOF 脉冲管脚输出功能
 - ▶ SOF 脉冲可作为芯片内部事件源去触发 TIMER, DMA 等模块工作
 - ▶ 可配置帧周期
 - ▶ 可配置的帧结束中断
- 模块内嵌 DMA，并可软件配置 AHB 突发传输类型
- 具备省电功能，例如 USB 挂起，停止 RAM 时钟，停止 PHY 域时钟
- 具有采用高级 FIFO 控制的 2.5KB 专用 RAM
 - ▶ 可以将 RAM 空间划分为不同的 FIFO，以便灵活有效的使用 RAM
 - ▶ 每个 FIFO 可存储多个数据包
 - ▶ 动态分配存储区
 - ▶ FIFO 的大小可配置成为非 2 的幂次方值，以便连续使用存储单元
- 一帧之内可以不需要应用程序干预，以达到最大 USB 带宽
- 可根据 ID 线的电平自动确定主机模式或者设备模式

35.2.2 主机模式特性

- 主机模式支持 USB2.0 全速 (FS, 12Mbit/s) 和低速 (LS, 1.5Mbit/s) 传输
- 需要通过外部电源芯片生成 VBUS 电压
- 多达 16 主机通道 (管道)：每个通道都可以动态实现重新配置，可支持任何类型的 USB 传输
- 内置硬件调度器，可：
 - ▶ 在周期性硬件队列中存储多达 8 个中断加同步传输请求
 - ▶ 在非周期性硬件队列中存储多达 8 个控制加批量传输请求
- 管理一个共享 RX FIFO、一个周期性 TX FIFO 和一个非周期性 TX FIFO，以有效使用 USB 数据 RAM

35.2.3 设备模式特性

- 从机模式支持 USB2.0 全速 (FS, 12Mbit/s) 传输
- 1 个双向控制端点 0
- 15 个 OUT 端点, 可以配置为支持批量传输、中断传输或同步传输
- 15 个 IN 端点, 可以配置为支持批量传输、中断传输或同步传输
- 包含 16 个发送 FIFOs (每个 IN 端点配备一个发送 FIFO) 和一个接收 FIFO (由所有的 OUT 端点共享)
- 支持远程唤醒功能
- 支持软断开功能
- VBUS PIN 支持 5V 耐压

35.3 功能说明

35.3.1 系统框图

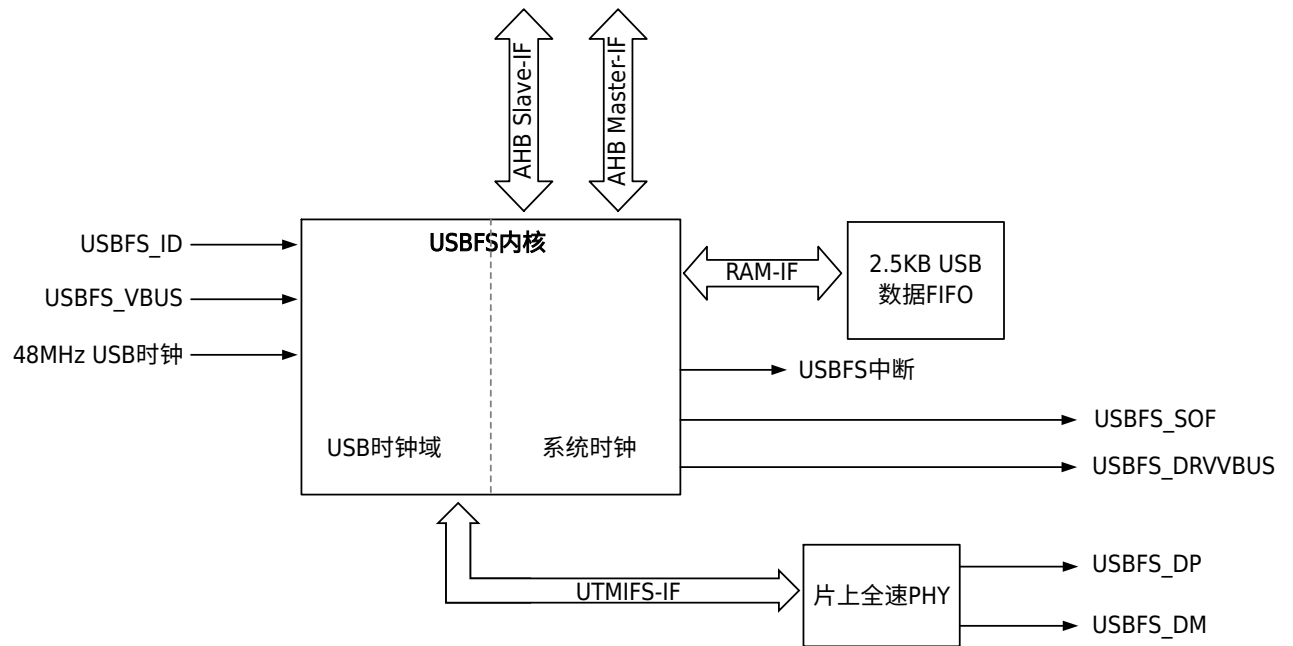


图 35-1 USBFS 系统框图

35.3.2 管脚说明

表 35-1 USBFS 管脚说明

管脚名	方向	适用模式	对应管脚	功能描述
USBFS_VBUS	输入	设备	PA9	电源端口，5V 耐压
USBFS_DP	输入/输出	主机、设备	PA12	差分数据 D+信号
USBFS_DM	输入/输出	主机、设备	PA11	差分数据 D-信号
USBFS_DRVVBUS	输出	主机	PC9、PB8	外部电源芯片使能信号
USBFS_ID	输入	主机	PA10	USB A-B 器件识别信号
USBFS_SOF	输出	主机、设备	PA8	SOF 输出脉冲信号

由于 USBFS_DP 和 USBFS_DM 管脚和通用 GPIO 复用，在使用 USBFS 时，建议关闭其对应管脚的数字功能，USBFS_DP 和 USBFS_DM 功能为模拟功能，与对应的 PFSR 寄存器设定无关。详细情况请参见《端口控制器（GPIO）》章节相关描述。另外 USBFS 功能不使用时，USBFS_DP 和 USBFS_DM 管脚对应的数字功能管脚翻转时，会产生额外的电流消耗。

35.3.3 时钟以及工作模式

USBFS 所使用的时钟需要配置为 48MHz，该 48MHz 时钟由内部 PLL 电路产生，PLL 时钟源需选择外部高速振荡器，使用 USBFS 模块前，需要在 CMU 模块内配置好 USBFS 时钟。

USBFS 可以作为主机或者设备使用，并且包含一个片上全速 PHY。

上拉和下拉电阻已经集成在片上全速 PHY 的内部，并且 USBFS 可以根据当前模式和连接状态自动选择。

USBFS 工作时，VCC 电压范围为 3.0~3.63V。

35.3.4 模式决定

USBFS 决定当前工作模式有以下两种方法：

- 方法 1：根据 USBFS_ID 线的状态自动识别，检测到 USBFS_ID 线为高电平时，模块工作在设备模式，检测到 USBFS_ID 线为低电平时，模块工作在主机状态。
- 方法 2：强制主机/设备模式，通过设定寄存器 USBFS_GUSBCFG 的 FDMOD 或者 FHMOD 位为 1，使模块忽略 USBFS_ID 线的电平而强制工作在设备或者主机模式。

35.3.5 主机功能

功能简介

当 USBFS 工作在主机模式下时，VBUS 为 USB 协议所规定的 5V 电源引脚。内部 PHY 不支持提供 5V 电源，所以需要外接 USB 电源芯片为设备供电。USBFS_DRVVBUS 用来使能外接 USB 电源芯片，外接电源芯片的过电流检出可以通过本 MCU 的外部中断 IRQ 实现。主机模式下 USBFS_VBUS 可以作为 GPIO 使用。

典型的 USB 主机模式系统构建图如下：

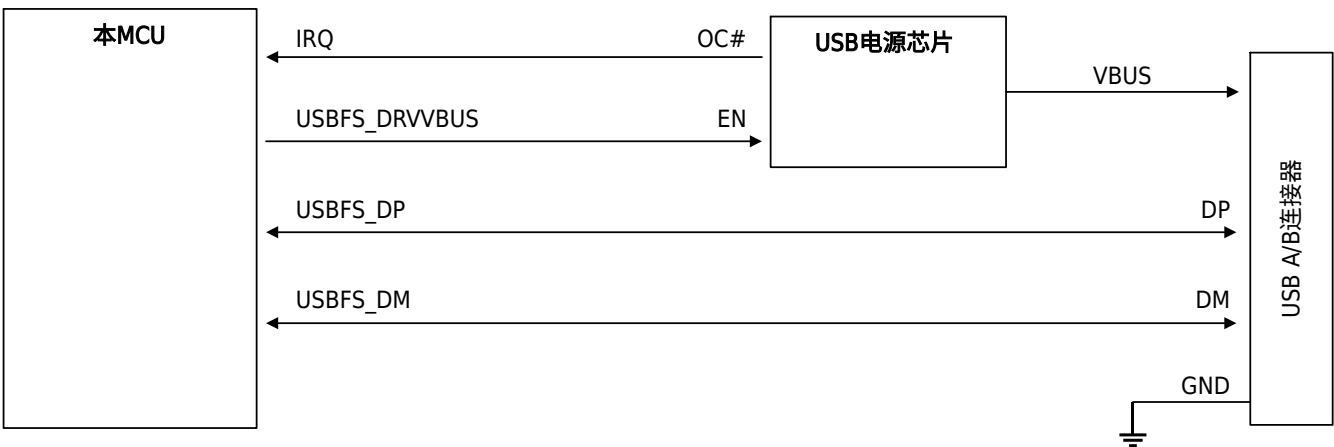


图 35-2 USBFS 主机模式系统构建图

主机端口供电

本 MCU 不能输出 5V 以提供 VBUS。为此，必须在微控制器以外添加 USB 电源芯片或基本电源开关（如应用电路板提供 5V 电源）来驱动 5V VBUS 线。外部电源芯片可通过任何 GPIO 输出或者 USBFS_DRVVBUS 驱动。当应用程序确定使用 GPIO 来控制外部器件提供 VBUS，仍须将主机端口控制和状态寄存器中的端口电源位（USBFS_HPRT 中的 PWPR 位）置 1。

主机检测设备连接和断开

USB 设备将在连接后立即被检测到。USBFS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（USBFS_HPRT 中的 PCDET 位）触发。

设备断开事件将触发断开连接检测中断（USBFS_GINTSTS 中的 DISCINT 位）。

主机枚举

检测到设备连接后，若又有新的设备连接进来，主机必须通过向新的设备发送 USB 复位和配置命令来启动枚举过程。

应用程序通过将主机端口控制和状态寄存器中的端口复位位（USBFS_HPRT 中的 PRST 位）置 1，使该过程最少持续 10ms、最多持续 20ms，以此通过 USB 驱动 USB 复位信号（单端零）。应用程序计算这个过程的持续时间，然后将端口复位位清零。

USB 复位序列完成后，端口使能/禁止更改位（USBFS_HPRT 中的 PENCHNG 位）立即触发主机端口中断，进而向应用程序发出通知，指示可从主机端口控制和状态寄存器中的端口速度字段（USBFS_HPRT 中的

PSPD) 读取枚举的设备速度, 以及主机已经开始驱动 SOF (FS) 或 Keep-alive 令牌 (LS)。此时主机已就绪, 可通过对设备发送命令来完成对设备的枚举。

主机挂起

应用程序通过将主机端口控制和状态寄存器中的端口挂起位 (USBFS_HPRT 中的 PSUSP) 置 1 来挂起 USB 总线。USBFS 模块停止发送 SOF 并进入挂起状态。

可由远程设备的自主活动 (远程唤醒) 使总线退出挂起状态。这种情况下, 远程唤醒信号将触发远程唤醒中断 (USBFS_GINTSTS 中的 WKUPINT 位), 硬件把主机端口控制和状态寄存器中的端口恢复位

(USBFS_HPRT 中的 PRES 位) 自行复位, 并通过 USB 自动驱动恢复信号。应用程序必须为恢复窗口定时, 然后将端口恢复位清零以退出挂起状态并重新启动 SOF。

如果由主机发起退出挂起状态, 则应用程序必须将端口恢复位置 1 以启动主机端口上的恢复信号, 为恢复窗口定时并最终将端口恢复位清零。

主机通道

USBFS 模块实现了 16 个主机通道。每个主机通道均可用于 USB 主机传输 (USB 管道)。主机最多能同时处理 8 个传输请求。如果应用程序有 8 个以上的传输请求挂起, 则在通道从之前任务释放后 (即, 接收到传输完成和通道停止中断后), 主机控制器驱动器 (HCD) 必须为未处理的传输请求重新对通道进行分配。

每个主机通道都可配置为支持输入/输出以及周期性/非周期性事务。每个主机通道都使用专用控制

(HCCHARx) 寄存器、传输配置 (HCTSIZx) 寄存器/中断 (HCINTx) 寄存器以及和其相关的中断屏蔽寄存器 (HCINTMSKx)。

主机通道控制

应用程序可通过主机通道 x 特性寄存器 (HCCHARx) 对主机通道作以下控制:

- 通道使能/禁止
- 设置目标 USB 设备的速度: FS/LS
- 设置目标 USB 设备的地址
- 设置与该通道通信的目标 USB 设备上的端点的编号
- 设置该通道上的传输方向: IN/OUT
- 设置该通道上的 USB 传输的类型: 控制/批量/中断/同步
- 设置与该通道通信的设备端点的最大包长
- 设置要进行周期传输的帧: 奇帧/偶帧

主机通道传输

主机通道传输大小寄存器 (HCTSIZx) 允许应用程序对传输大小参数进行编程并读取传输状态。必须在主机通道特性寄存器中的通道使能位置 1 之前完成对此寄存器的设置。使能端点后, 数据包计数字段立即变为只读状态, 同时 USBFS 模块根据当前传输状态对该字段进行更新。

可对以下传输参数进行编程:

- 以字节为单位的传输大小
- 构成整个传输大小的数据包个数
- 初始数据 PID

主机通道状态/中断

主机通道 x 中断寄存器 (HCINTx) 指示端点在出现 USB 和 AHB 相关事件时的状态。当中断寄存器中的主机通道中断位 (USBFS_GINTSTS 中的 HCINT 位) 置 1 时, 应用程序必须读取这些寄存器以获得详细信息。在读取这些寄存器之前, 应用程序必须先读取主机全体通道中断 (HCAINT) 寄存器, 以获取主机通道 x 中断寄存器的通道编号。应用程序必须将此寄存器中的相应位清零, 才能将 HCAINT 和 GINTSTS 寄存器中的相应位清零。USBFS_HCINTMSKx 寄存器还提供每个通道各中断源的屏蔽位。

主机模块提供以下状态检查和中断产生功能:

- 传输完成中断，指示应用程序 (AHB) 和 USB 端均已完成数据传输
- 通道因传输完成、USB 事务错误或应用程序发出禁止命令而停止
- 相关的发送 FIFO 为半空或全空状态 (IN 端点)
- 接收到 ACK 响应
- 接收到 NAK 响应
- 接收到 STALL 响应
- 由于 CRC 校验失败、超时、位填充错误和错误的 EOP 导致 USB 事务错误
- 串扰错误
- 帧上溢
- 用于数据同步的翻转位出错

主机调度器

主机模块内置硬件调度器，可自主对应用程序发出的 USB 事务请求重新排序和管理。每一帧开始时，主机都先执行周期性（同步和中断）事务，然后执行非周期性（控制和批量）事务，以符合 USB 规范对同步和中断传输高优先级的保证。

主机通过请求队列（一个周期性请求队列和一个非周期请求队列）处理 USB 事务。每个请求队列最多可存储 8 个条目。每个条目代表一个应用程序发起但还未得到响应的 USB 事务请求，并存储了执行该 USB 事务所用到的 IN 或 OUT 通道的编号，以及其它相关信息。USB 事务请求在队列中的写入顺序决定了事务在 USB 接口上的执行顺序。

每一帧开始时，主机都先处理周期性请求队列，然后处理非周期性请求队列。如果当前帧结束时，计划在当前帧执行的同步或中断类型的 USB 传输事务请求仍处于挂起状态，则主机将发出未完成周期性传输中断 (USBFS_GINTSTS 中的 IPXFR 位)。USBFS 模块负责对周期性和非周期性请求队列的管理。周期性发送 FIFO 和队列状态寄存器 (HPTXSTS) 与非周期性发送 FIFO 和队列状态寄存器 (HNPTXSTS) 都为只读寄存器，应用程序可使用它们来读取各请求队列的状态。其中包括：

- 周期性（非周期性）请求队列中当前可用的空闲条目数（最多 8 个）
- 周期性（非周期性）Tx FIFO (OUT 事务) 中当前可用的空闲空间
- IN/OUT 令牌、主机通道编号和其它状态信息

由于每个请求队列最多可存储 8 个 USB 事务请求，因此应用程序可以把主机 USB 事务请求提前发送给调度器；实际的通信最晚会在调度器处理完已挂起的 8 个周期事务和 8 个非周期事务完成之后出现在 USB 总线上。

要向主机调度器（队列）发出事务请求，应用程序必须读取 USBFS_HPTXSTS 寄存器中的 PTXQSAV 位或 USBFS_HNPTXSTS 寄存器中的 NPTQXSAV 位，确保周期性（非周期性）请求队列中至少有一个可用空间来存储当前请求。

35.3.6 设备功能

功能简介

当 USBFS 工作在设备模式下时，VBUS 为 USB 协议所规定的 5V 电源引脚，为 5V 耐压管脚。本模块始终检测 VBUS 线的电平状态来连接或者断开设备。USBFS_VBUS 端口需先经过 $\geq 1k\Omega$ 的电阻才能接到 VBUS 电源。

典型的 USB 设备模式系统构建图如下：

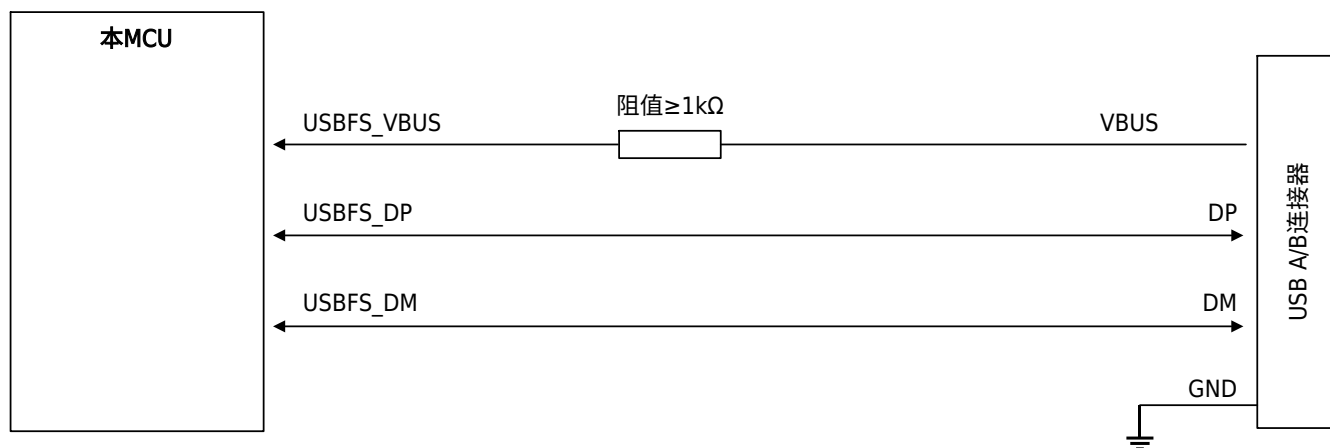


图 35-3 USBFS 设备模式系统构建图

设备供电状态

模块检测到 USBFS_VBUS 为高电平时，就会使 USB 设备进入供电状态。然后，USBFS 自动连接 DP 上拉电阻，发出全速设备与主机相连的信号并生成会话请求中断（USBFS_GINTSTS 中的 VBUSVINT 位），指示进入供电状态。

此外，USBFS_VBUS 输入还可确保主机在 USB 操作期间提供有效的 VBUS 电平。如果检测到 VBUS 电平为低电平（例如，因电源干扰或主机端口关闭引发），USBFS 将自动断开连接。

供电状态下，USBFS 期望收到来自主机的复位信号。其它 USB 操作则无法执行。收到复位信号后，立即生成检测到复位中断（USBFS_GINTSTS 中的 USBRST）。复位信号结束后，将生成枚举完成中断（USBFS_GINTSTS 中的 ENUMDNE 位），USBFS 随即进入默认状态。

设备默认状态

默认状态下，USBFS 期望从主机收到 SET_ADDRESS 命令。其它 USB 操作则无法执行。当 USB 上解码出有效 SET_ADDRESS 命令时，应用程序会将相应的地址值写入设备配置寄存器中的设备地址字段

（USBFS_DCFG 中的 DAD 位）。USBFS 随即进入地址状态，并准备好以所配置的 USB 地址对主机事务进行应答。

设备挂起状态

USBFS 设备持续监视 USB 活动。在 USB 空闲时间达到 3ms 后，将发出早期挂起中断（USBFS_GINTSTS 中 ESUSP 位），并在 3ms 后由挂起中断（USBFS_GINTSTS 中的 USBSUSP 位）确认设备进入挂起状态。然后，设备状态寄存器中的设备挂起位（USBFS_DSTS 中的 SUSPSTS 位）自动置 1，USBFS 随即进入挂起状态。

可通过设备本身退出挂起状态。这种情况下，应用程序会将设备控制寄存器中的远程唤醒信号位（USBFS_DCTL 中的 RWUSIG 位）置 1，并在 1ms 到 15ms 内将其清零。

但若设备检测到主机发出的恢复信号，将生成恢复中断（USBFS_GINTSTS 中的 WKUPINT 位），设备挂起位自动清零。

设备软断开

供电状态可借助软断开功能通过软件退出。将设备控制寄存器中的软断开位（USBFS_DCTL 中的 SDIS 位）置 1 即可移除 DP 上拉电阻，此时尽管没有从主机端口实际拔出 USB 电缆，但主机端仍会发生设备断开检测中断。

设备端点

端点类别

USBFS 模块实现了以下 USB 端点：

- 控制端点 0：
 - ▶ 双向且仅处理控制消息

- ▶ 使用一组单独的寄存器来处理 IN 和 OUT 事务
- ▶ 专用控制 (USBFS_DIECTL0/USBFS_DOECTL0) 寄存器、传输配置 (USBFS_DIEPTSIZE0/USBFS_DOEPTSIZE0) 寄存器和状态中断 (USBFS_DIEPINTx/USBFS_DOEPINT0) 寄存器。控制和传输大小寄存器中可用的位组与其它端点中稍有不同
- 15 个 IN 端点
 - ▶ 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - ▶ 每个端点都有专用控制 (USBFS_DIECTLx) 寄存器、传输配置 (USBFS_DIEPTSIZEx) 寄存器和状态中断 (USBFS_DIEPINTx) 寄存器
 - ▶ 设备 IN 端点通用中断屏蔽寄存器 (USBFS_DIEPMASK) 可用于使能/禁止所有 IN 端点 (包括 EP0) 上的同一类端点中断源
 - ▶ 支持未完成的同步 IN 传输中断 (USBFS_GINTSTS 中的 IISOIXFR 位)，该中断将在当前帧中至少有一个同步 IN 端点上的传输未完成时触发。该中断和周期性帧中断 (USBFS_GINTSTS/EOPF) 一起触发
- 15 个 OUT 端点
 - ▶ 每个端点都可配置为支持同步传输、批量传输或中断传输类型
 - ▶ 每个端点都有专用控制 (USBFS_DOECTLx) 寄存器、传输配置 (USBFS_DOEPTSIZEx) 寄存器和状态中断 (USBFS_DOEPINTx) 寄存器
 - ▶ 设备 OUT 端点通用中断屏蔽寄存器 (USBFS_DOEPMASK) 可用于使能/禁止所有 OUT 端点 (包括 EP0) 上的同一类端点中断源
 - ▶ 支持未完成的同步 OUT 传输中断 (USBFS_GINTSTS 中的 INCOMPIISOOUT 位)，该中断将在当前帧中至少有一个同步 OUT 端点上的传输未完成时触发。该中断和周期性帧中断 (USBFS_GINTSTS/EOPF) 一起触发

端点控制

应用程序可通过设备端点 x IN/OUT 控制寄存器 (DIECTLx/DOECTLx) 对端点采取以下控制：

- 端点使能/禁止
- 在当前配置下激活端点
- 设置 USB 传输类型 (同步、批量和中断)
- 设置支持的数据包大小
- 设置与 IN 端点相关的 Tx-FIFO 编号
- 设置希望收到的或发送时要使用到的 data0/data1 PID (仅限批量/中断传输)
- 设置接收或发送事务时所对应的奇/偶帧 (仅限同步传输)
- 可以设置 NAK 位，从而不论此时 FIFO 的状态如何，都对主机的请求回复 NAK
- 可以设置 STALL 位，使得主机对该端点的令牌都被硬件回复 STALL
- 可以将 OUT 端点设置为侦听模式，即对接收到的数据不进行 CRC 检查

端点传输

设备端点 x 传输尺寸寄存器 (DIEPTSIZEx/DOEPTSIZEx) 允许应用程序对传输尺寸参数进行编程并读取传输状态。必须在端点控制寄存器中的端点使能位置 1 之前完成对此寄存器的设置。使能端点后，这些字段立即变为只读状态，同时 USBFS 模块根据当前传输状态对这些字段进行更新。

可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输的数据包个数

端点状态/状态

设备端点 x 中断寄存器 (DIEPINTx/DOEPINTx) 指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 OUT 端点中断位或 IN 端点中断位 (分别为 USBFS_GINTSTS 中 OEPINT 位或 USBFS_GINTSTS

中的 IEPINT 位) 置 1 时, 应用程序必须读取这些寄存器以获得详细信息。在应用程序读取这些寄存器之前, 必须先读取设备全体端点中断 (USBFS_DAINTE) 寄存器, 以获取设备端点 x 中断寄存器的端点编号。应用程序必须将此寄存器中的相应位清零, 才能将 DAINTE 和 GINTSTS 寄存器中的相应位清零。

模块提供以下状态检查和中断产生功能:

- 传输完成中断, 指示应用程序 AHB 和 USB 端均已完成数据传输
- Setup 阶段已完成 (仅针对控制传输类型的 OUT 端点)
- 相关的发送 FIFO 为半空或全空状态 (IN 端点)
- NAK 应答已发送到主机 (仅针对同步传输的 IN 端点)
- TxFIFO 为空时接收到 IN 令牌 (仅针对批量和中断传输类型的 IN 端点)
- 尚未使能端点时接收到 OUT 令牌
- 检测到 babble 错误
- 应用程序关闭端点生效
- 应用程序对端点设置 NAK 生效 (仅针对同步传输类型的 IN 端点)
- 接收到 3 个以上连续 setup 数据包 (仅针对控制类型的 OUT 端点)
- 检测到超时状况 (仅针对控制传输类型的 IN 端点)

35.3.7 SOF 脉冲管脚输出功能

USBFS 在主机和设备模式下都可以监视、跟踪和配置 SOF 帧并且还具备 SOF 脉冲输出功能。SOF 脉冲通过 USBFS_SOF 管脚输出, 输出宽度为 16 个 HCLK 时钟周期。

主机 SOF

主机模式下, 可以在主机帧间隔寄存器 (HFIR) 中对所产生的两个连续 SOF (FS) 或 keep-alive (LS) 令牌期间所出现的 PHY 时钟数进行编程, 进而应用程序可对 SOF 帧周期进行控制。帧开始 (USBFS_GINTSTS 中的 SOF 位) 时都将生成中断。当前帧编号和出现下一个 SOF 前剩余的时间应用程序在主机帧编号寄存器 (HFNUM) 中能够进行跟踪。

使用 USBFS 系统控制寄存器 USBFS_SYCTLREG 中的 SOFEN 位, 可以使任何 SOF 令牌发出的同时产生的、宽度为 16 个 HCLK 系统时钟周期的 SOF 脉冲信号从 USBFS_SOF 引脚输出。

此外, SOF 脉冲还可以作为内部事件触发 DMA 传输、TIMER 计数等外部模块工作。

设备 SOF

在设备模式下, USB 每次接收到 SOF 令牌时, 都将触发帧开始中断 (USBFS_GINTSTS 中的 SOF 位)。相应的帧编号可从设备状态寄存器 (USBFS_DSTS 中的 FNSOF 位) 读取。使用 USBFS 系统控制寄存器 USBFS_SYCTLREG 中的 SOFEN 位, 还可以生成宽度为 16 个 HCLK 系统时钟周期的 SOF 脉冲信号, 并使该信号在 USBFS_SOF 引脚输出, 以实现外部可用。

此外, SOF 脉冲还可以作为内部事件触发 DMA 传输、TIMER 计数等外部模块工作。

周期性帧结束中断 (GINTSTS/EOPF) 用于在经过了 80%、85%、90% 或 95% 的帧间隔时间时通知应用程序, 具体取决于设备配置寄存器中的周期性帧间隔字段 (USBFS_DCFG 中的 PFIVL 位)。此功能可用于确定该帧的所有同步通信是否完成。

35.3.8 功耗控制

不使用 USBFS 模块时, 可以通过 CMU 模块停止 USBFS 模块的 HCLK 和 PHY 时钟, 从而降低功耗。

使用 USB 模块, 但设备 USB 会话未开始或设备未连接时, 可以在 USB 挂起状态下使用功率降低技术。

- 停止 PHY 时钟 (USBFS_GCCTL 中的 STPPCLK 位)

将时钟门控控制寄存器中的停止 PHY 时钟位置 1 时, USBFS 全速模块的大多数 48MHz 内部时钟域均由时钟门控关闭。即使应用程序仍提供时钟输入, 也会节省掉模块由于时钟信号翻转带来的动态功耗还会关掉收发器的大部分单元, 只有负责检测异步恢复事件或远程唤醒事件的部分还保持工作状态。

- HCLK 门控 (USBFS_GCCTL 中的 GATEHCLK 位)
将时钟门控控制寄存器中的 GATEHCLK 位置 1 时, USBFS 模块内部的大多数 HCLK 系统时钟域均由时钟门控关闭。只有寄存器读取和写入接口保持活动状态。即使应用程序仍提供时钟输入, 也会节省掉模块由于时钟信号翻转带来的动态功耗。
为了节省动态功耗, 只在 USB 数据 FIFO 被 USBFS 模块访问时为其提供时钟。

35.3.9 动态更新 USBFS_HFIR 寄存器

主机模式下, USB 模块具有对帧周期进行动态微调的功能, 能够将外部设备与 SOF 帧进行同步。如果 USBFS_HFIR 寄存器在当前 SOF 帧内发生更改, 则将在下一个帧中对 SOF 周期进行相应修正, 具体说明详见下图。

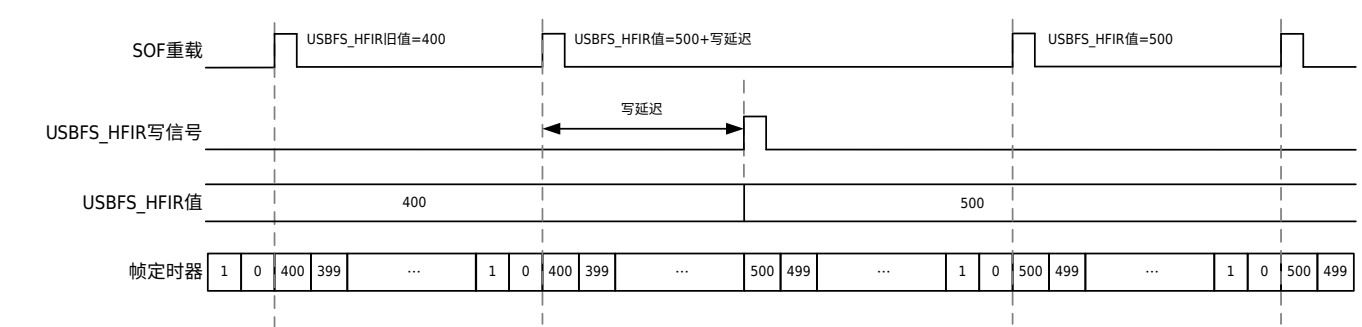


图 35-4 USBFS 动态更新 USBFS_HFIR 寄存器示意图

35.3.10 数据 FIFO

USBFS 系统具有 2.5KB 专用 RAM, 采用高效的 FIFO 控制机制。USBFS 模块中的数据包 FIFO 控制器模块将 RAM 空间划分为多个 Tx FIFO (USB 传输前, 应用程序将数据压入其中进行短暂存储) 和单个 Rx FIFO (从 USB 接收到的数据被应用程序读取之前, 在其中进行短暂存储)。
RAM 中所构建的 FIFO 的数量与组织方式取决于设备的角色。设备模式下, 为每个激活的 IN 端点配置一个 Tx FIFO。FIFO 的大小均由软件配置, 以更好地满足应用要求。

35.3.11 主机 FIFO 架构

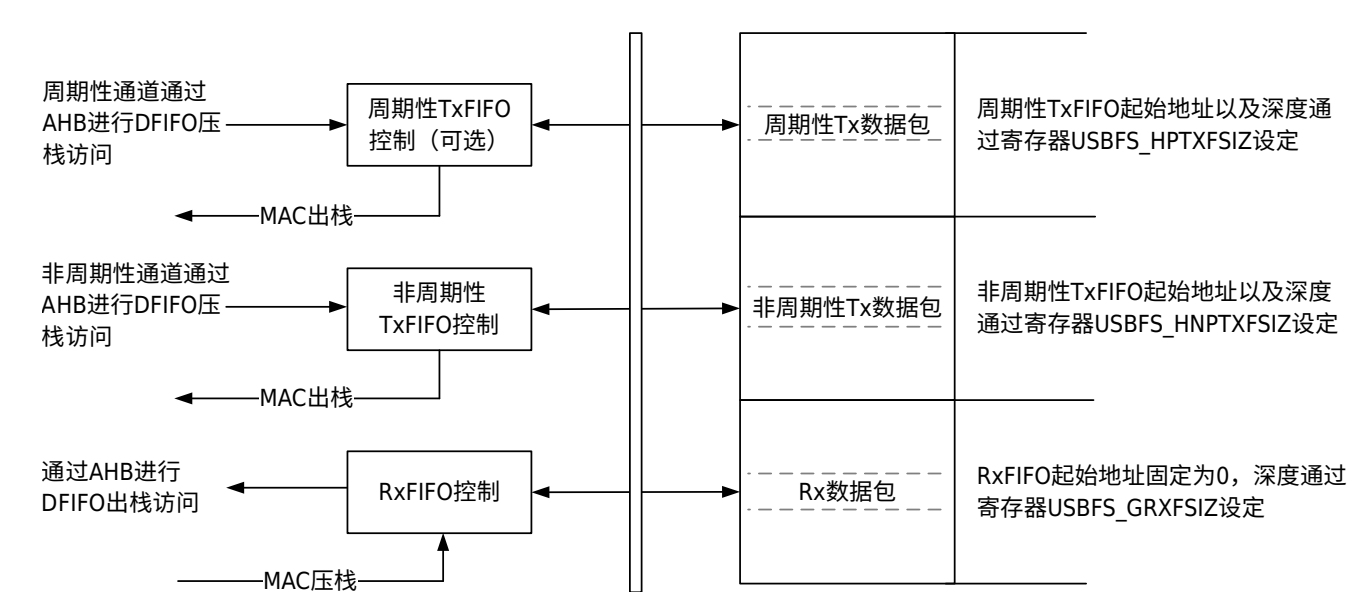


图 35-5 主机模式下 FIFO 架构示意图

主机 RxFIFO

主机使用一个接收 FIFO 处理所有周期和非周期事务。FIFO 用作接收缓冲区以保存从 USB 接收到的数据（接收到的数据包的数据部分），直至这些数据传输到系统存储器。只要 FIFO 中有空间，来自设备 IN 端点的数据包就接收进来并挨个存储。接收到的每个数据包的状态（包含主机目标通道、字节数、数据 PID 和对所接收数据的校验）也存储在 FIFO 中。接收 FIFO 的大小在接收 FIFO 大小寄存器（GRXFSIZ）中配置。

单个接收 FIFO 架构使得 USB 主机高效地填充接收数据缓冲区：

- 所有 IN 配置主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- 对于主机软件驱动的任何序列 IN 令牌，USBFS 模块可将接收 FIFO 填充至限值

只要至少有一个数据包在 RxFIFO 中可供读取，应用程序就会接收 RxFIFO 非空中断。应用程序从接收状态读取和出栈寄存器中读取数据包信息，最后从 RxFIFO 中读出数据。

主机 TxFIFO

主机使用一个发送 FIFO 处理所有非周期（控制和批量）OUT 事务，使用另一个发送 FIFO 处理所有周期（同步和中断 OUT 事务。FIFO 用作发送缓冲区以保存要通过 USB 发送的数据（发送数据包）。周期（非周期）TxFIFO 的大小在主机周期（非周期）发送 FIFO 大小（HPTXFSIZ/HNPTXFSIZ）寄存器中配置。

两个 TxFIFO 按优先级实施操作，周期性通信的优先级较高，因此在 USB 一帧的时间内首先进行周期性通信。帧起始时，内置的主机调度器先处理周期请求队列，再处理非周期请求队列。

两个发送 FIFO 的架构使得 USB 主机能够对周期和非周期发送数据缓冲区分别进行优化管理：

- 配置为支持周期（非周期）OUT 事务的所有主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- 对于主机软件驱动的任何序列 OUT 令牌，USBFS 模块可将周期性（非周期性）发送 FIFO 填充至限值

只要周期性 TxFIFO 为半空或全空，USBFS 模块就会发出周期性 TxFIFO 空中断（USBFS_GINTSTS 中的 PTXFE 位），具体取决于 AHB 配置寄存器中的周期性 Tx-FIFO 空等级位（USBFS_GAHBCFG 中的 PTXFELVL 位）的值。只要周期性 TxFIFO 和周期性请求队列中均存在空闲空间，应用程序便可提前写入发送数据。可通过读取主机周期性发送 FIFO 和队列状态寄存器（HPTXSTS）来了解二者的可用空间。

只要非周期性 TxFIFO 为半空或全空，USBFS 模块就会发出非周期性 TxFIFO 空中断（USBFS_GINTSTS 中的 NPTXFE 位），具体取决于 AHB 配置寄存器中的非周期性 TxFIFO 空等级位（USBFS_GAHBCFG 中的 TXFELVL 位）。只要非周期性 TxFIFO 和非周期性请求队列中均存在空闲空间，应用程序便可写入发送数据。可通过读取主机非周期性发送 FIFO 和队列状态寄存器（HNPTXSTS）来了解二者的可用空间。

35.3.12 设备 FIFO 架构

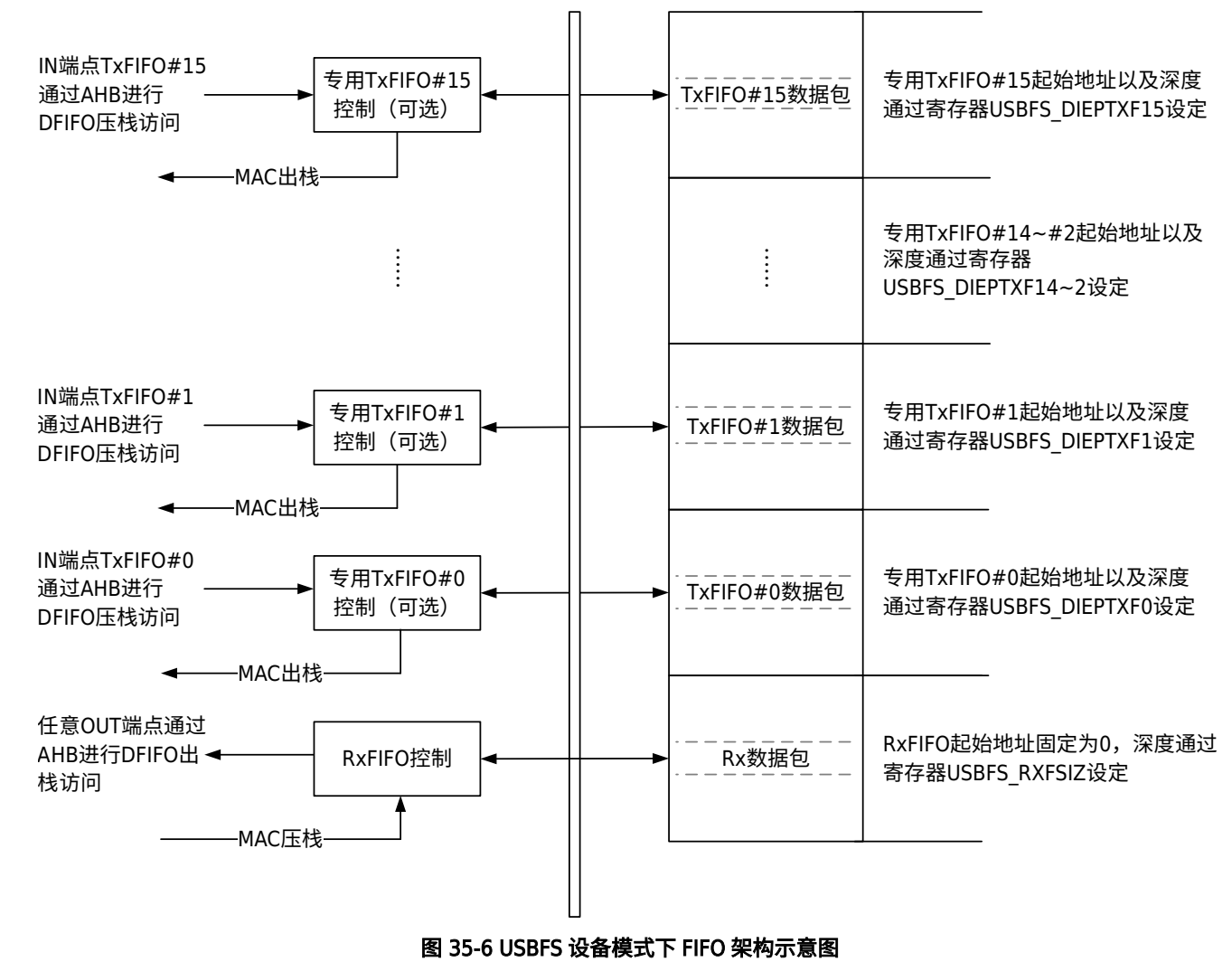


图 35-6 USBFS 设备模式下 FIFO 架构示意图

设备 RxFIFO

SBFS 设备使用单个接收 FIFO 接收发送到所有 OUT 端点的数据。只要 RxFIFO 中有空余空间，收到的数据包就挨个填入 RxFIFO。除了有效数据外，接收到的数据包状态（包含 OUT 端点目标编号、字节数、数据 PID 和对所接收数据的验证）也由模块进行存储。没有可用空间时，设备会回复主机事务 NAK 应答并在被寻址的端点上触发中断。接收 FIFO 的大小在接收 FIFO 大小寄存器（GRXFSIZ）中配置。

单个接收 FIFO 架构使得 USB 设备更高效地填充接收 RAM 缓冲区：

- 所有 OUT 端点共享同一个 RAM 缓冲区（共享 FIFO）
- 对于任意主机序列 OUT 令牌，USBFS 模块可将接收 FIFO 填充至限值

只要至少有一个数据包在 RxFIFO 中可供读取，应用程序就会一直接收 RxFIFO 非空中断（USBFS_GINTSTS 中的 RXFNE 位）。应用程序从接收状态读取和出栈寄存器（GRXSTSP）中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

设备 TxFIFO

模块为各个 IN 端点提供了专用的 FIFO。应用程序通过非周期发送 FIFO 大小寄存器（USBFS_DIEPTSIZE0）为 IN 端点 0 配置 FIFO 大小；通过设备 IN 端点发送 FIFOx 寄存器（DIEPTSIZEx）为 IN 端点 x 配置 FIFO 大小。

35.3.13 FIFO RAM 分配

主机模式

接收 FIFO RAM 分配

状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+1 的空间。如果使能了多个同步通道，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+1。通常，推荐的空间为（最大数据包/4+1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被写入 FIFO。所以必须为此分配一个位置。

以 DMA 模式运行时，每个主机通道的 DMA 地址寄存器将存储在 FIFO 中，因此需要在 FIFO 中为每个通道保留一个位置存放其地址寄存器。

发送 FIFO RAM 分配

主机非周期性发送 FIFO 所需的最小 RAM 为所支持的所有非周期性 OUT 通道上传输的最大数据包的大小。通常，推荐的空间为最大数据包大小的两倍，这样当 USB 正在发送当前数据包的同时，AHB 可以往发送 FIFO 填入下一个数据包。

主机周期性发送 FIFO 所需的最小 RAM 为所支持的所有周期性 OUT 通道上传输的最大数据包的大小。如果至少有一个同步 OUT 端点，则空间必须至少为该通道中最大数据包大小的两倍。

以 DMA 模式运行时，每个主机通道的 DMA 地址寄存器将存储在 FIFO 中，因此需要在 FIFO 中为每个通道保留一个位置存放其地址寄存器。

设备模式

接收 FIFO RAM 分配

应用程序应为 SETUP 数据包分配 RAM：接收 FIFO 中必须保留 11 个位置以在控制端点上接收 SETUP 数据包。USBFS 模块不会向这些为 SETUP 数据包保留的位置写入任何其它数据。将会为全局 OUT NAK 分配一个位置。状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+1 的空间。如果使能了多个同步端点，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+1。通常，推荐的空间为（最大数据包/4+1）的两倍，这样当上一个数据包向 CPU 传送时，USB 可同时接收后续的数据包。

传输完成状态信息和该端点收到的最后一个数据包会一起被推入 FIFO。通常情况下，推荐为每个 OUT 端点分配一个位置。

发送 FIFO RAM 分配

各个 IN 端点发送 FIFO 所需的最小 RAM 空间为该特定 IN 端点的最大数据包大小。

35.3.14 系统性能

凭借大容量 RAM 缓冲区、高度可配置的 FIFO 大小、通过 AHB 压栈/出栈寄存器进行 32 位 FIFO 快速访问，尤其是高级 FIFO 控制机制可获得最佳 USB 和系统性能。实际上，无论当前 USB 序列如何，USBFS 均可通过该机制高效填充可用的 RAM 空间。借助这些特性：

- 应用程序有足够的裕量来计算并校正 CPU 的负载，从而优化 CPU 带宽利用率：
 - ▶ 应用程序可先积累大量发送数据，再通过 USB 发送出去
 - ▶ 可带来足够的时间裕量，以从接收 FIFO 读取数据
- USB 模块能够保持全速工作状态，也就是提供最大的全速带宽（尽量多的硬件自动运行，尽量少的软件参与）
 - ▶ USB 模块可提前积累大量发送数据供其支配，从而可对 USB 数据发送进行自主管理
 - ▶ 接收缓冲区中有大量空白空间，可通过 USB 中的数据自动填满

由于 USBFS 模块能够高效填充 2.5KB RAM 缓冲区且 2.5KB 发送/接收数据足以满足一个全速帧所能容纳的数据量，因此 USB 系统在一帧之内可以无需应用程序干预达到最大 USB 带宽。

35.3.15 中断和事件

USBFS 中断有以下两类，SOTP 模式唤醒中断 USBFS_WKUP 和 USBFS 全局中断 USBFS_GLB。

● USBFS_WKUP 中断

USBFS_WKUP 中断用于 STOP 模式下，通过 USBFS_DP 或者 USBFS_DM 唤醒系统的 STOP 模式，该中断使能位为 INT_WUPEN.USF_WUEN。

使用 USBFS 唤醒系统 STOP 前，需要确保 USBFS 控制器处于挂起状态，并设定寄存器 USBFS_SYCTLREG 中的对应的滤波范围和使能滤波功能。

● USBFS_GLB 中断

USBFS_GLB 中断是软件需要处理的主要中断，全局中断的标志位可在 USBFS_GINTSTS 寄存器读取。

表 35-2 USBFS_GLB 中断事件表

中断标志	描述	运行模式	内部事件源
WKUPINT	恢复/远程唤醒中断	主机或者设备	-
VBUSVINT	VBUS 有效中断	设备	-
DISCINT	断开连接中断	主机	-
CIDSCHG	连接器 ID 线状态变化中断	主机或者设备	-
PTXFE	周期性 Tx FIFO 空中断	主机	-
LPMINT	LPM 中断	主机或者设备	-
HCINT	主机通道中断	主机	-
HPRTINT	主机端口中断	主机	-
DATAFSUSP	数据获取挂起	设备	-
IPXFR/INCOMPISOOOUT	未完成周期性传输/未完成 OUT 同步传输	设备	-
IISOIXFR	未完成 IN 同步传输	设备	-
OEPINT	OUT 端点中断	设备	-
IEPINT	IN 端点中断	设备	-
EOPF	周期性帧结束中断	设备	-
ISOODRP	丢弃同步 OUT 数据包中断	设备	-
ENUMDNE	枚举完成	设备	-
USBRST	USB 复位中断	设备	-
USBSUSP	USB 挂起中断	设备	-
ESUSP	早期挂起中断	设备	-
GONAKEFF	全局 OUT NAK 有效中断	设备	-
GINAKEFF	全局非周期性 IN NAK 有效中断	设备	-

中断标志	描述	运行模式	内部事件源
NPTXFE	非周期性 TxFIFO 空中断	主机	-
RXFNE	RxFIFO 非空中断	主机或者设备	-
SOF	帧起始中断	主机或者设备	是
MMIS	模式不匹配中断	主机或者设备	-

35.4 编程模型

35.4.1 模块初始化

背景信息

应用程序必须执行模块初始化序列。模式决定方法请参见[模式决定](#)。

本节介绍了 USBFS 控制器上电后的初始化过程。无论是以主机模式还是设备模式工作，应用程序都必须遵循初始化序列。根据模块配置对所有模块全局寄存器进行初始化：

操作步骤

- 步骤 1. 在 USBFS_GAHBCFG 寄存器中编程以下字段：
 - 全局中断屏蔽位 GINTMSK=1
 - RxFIFO 非空（USBFS_GINTSTS 中的 RXFNE 位）
 - 周期性 TxFIFO 空门限
- 步骤 2. 在 USBFS_GUSBCFG 寄存器中编程以下字段：
 - FS 超时校准字段
 - USB 周转时间字段
- 步骤 3. 软件必须取消对 USBFS_GINTMSK 寄存器中以下位的屏蔽：
 - 模式不匹配中断屏蔽
- 步骤 4. 通过读取 USBFS_GINTSTS 中的 CMOD 位，软件可确定 USBFS 控制器是在主机模式还是设备模式下工作。

35.4.2 主机初始化

背景信息

要将模块作为主机进行初始化，应用程序必须执行以下步骤：

操作步骤

- 步骤 1. 编程 USBFS_GINTMSK 寄存器中的 HPRTINT 以将对取消屏蔽。
- 步骤 2. 编程 USBFS_HCFG 寄存器以选择全速主机。
- 步骤 3. 将 USBFS_HPRT 中的 PWPR 位编程为 1，给 USB 总线提供 VBUS。
- 步骤 4. 等待 USBFS_HPRT 中的 PCDET 中断。这表示某设备已连接到主机端口。
- 步骤 5. 将 USBFS_HPRT 中的 PRST 位编程为 1，在 USB 总线上发出复位信号。
- 步骤 6. 至少等待 10ms，以便完成复位过程。
- 步骤 7. 将 USBFS_HPRT 中的 PRST 位编程为 0。
- 步骤 8. 等待 USBFS_HPRT 中的 PENCHNG 中断。
- 步骤 9. 读取 USBFS_HPRT 中的 PSPD 位以获取枚举速度。
- 步骤 10. 使用所选 PHY 时钟，相应地设置 HFIR 寄存器。
- 步骤 11. 根据步骤 9 中检测到的设备速度编程 USBFS_HCFG 寄存器中的 FSLSPCS 字段。如果 FSLSPCS 发生更改，则必须执行端口复位。

步骤 12. 编程 USBFS_GRXFSIZ 寄存器以选择接收 FIFO 的大小。

步骤 13. 编程 USBFS_HNPTXFSIZ 寄存器，以选择用于非周期性通信事务的非周期性发送 FIFO 的大小和起始地址。

步骤 14. 编程 USBFS_HPTXFSIZ 寄存器，以选择用于周期性事务的周期性通信发送 FIFO 的大小和起始地址。

后续任务

要与设备通信，系统软件必须初始化并使能至少一个通道。

35.4.3 设备初始化

背景信息

上电期间或者从主机模式切换为设备模式后，应用程序必须执行下列步骤来将模块作为设备进行初始化。

操作步骤

步骤 1. 在 USBFS_DCFG 寄存器中编程以下字段：

- 设备速度
- 非零长度状态 OUT 握手信号

步骤 2. 编程 USBFS_GINTMSK 寄存器以取消屏蔽以下中断：

- USB 复位
- 枚举完成
- 早期挂起
- USB 挂起
- SOF

步骤 3. 等待 USBFS_GINTSTS 中的 VBUSVINT 中断，表示进入供电状态。

步骤 4. 等待 USBFS_GINTSTS 中的 USBRST 中断。这表示已在 USB 上检测到复位信号，复位过程自接收到此中断后约持续 10ms。

步骤 5. 等待 USBFS_GINTSTS 中的 ENUMDNE 中断。此中断指示 USB 上复位过程结束。接收到此中断时，应用程序必须读取 USBFS_DSTS 寄存器以确定枚举速度并执行枚举完成时的端点初始化中所列的步骤。

操作结果

此时，设备已准备好接受 SOF 数据包并在控制端点 0 上执行控制传输。

35.4.4 DMA 模式

USB 使用 AHB 主接口来获取发送数据包数据（AHB 到 USB）和接收数据更新（USB 到 AHB）。AHB 主接口使用经过编程的 DMA 地址（主机模式下的 HCDMAx 寄存器和设备模式下的 DIEPDMAx/DOEPDMAx 寄存器）来访问数据缓冲区。

35.4.5 主机编程模型

35.4.5.1 通道初始化

背景信息

应用程序必须初始化一个或多个通道，之后才能与所连接的设备通信。

要初始化和使能通道，应用程序必须执行以下步骤：

操作步骤

步骤 1. 编程 USBFS_GINTMSK 寄存器以取消对以下位的中断屏蔽：

- 用于 OUT 事务的非周期性发送 FIFO 为空（在流水线事务级别工作且数据包计数字段编程值大 1 时适用）。

- 用于 OUT 事务的非周期性发送 FIFO 为半空（在流水线事务级别工作且数据包计数字段编程值大 1 时适用）。

步骤 2. 编程 USBFS_HAINTMSK 寄存器以使能所选通道中断。

步骤 3. 编程 USBFS_HCINTMSK 寄存器，以使能主机通道中断寄存器中反映的和通信事务有关的中断。

步骤 4. 编程所选通道的 USBFS_HCTSIZx 寄存器，指定以字节为单位的总传输大小和包括短数据包在内的预期数据包个数。应用程序必须使用初始数据 PID（用于第一个 OUT 事务或预期从第一个 IN 事务获取）编程 PID 字段。

步骤 5. 编程所选通道的 USBFS_HCCHARx 寄存器，指定设备的端点特性，例如类型、速度、方向等。（仅当应用程序准备好发送或接收数据包时，才能通过将通道使能位置 1 来使能通道）。

35.4.5.2 通道停止

应用程序可以通过编程 USBFS_HCCHARx 寄存器将 CHDIS 和 CHENA 位置 1 来禁止任何通道。这会使 USBFS 主机清空之前在该通道上发出的请求（如果有）并生成通道停止中断。应用程序在将通道重新分配给其它通信事务之前，必须等待 USBFS_HCINTx 中的 CHH 中断。USBFS 主机不会中断已在 USB 上启动的通信事务。禁止通道前，应用程序必须确保非周期性请求队列（禁止非周期性通道时）或周期性请求队列（禁止周期性通道时）中至少有一个空闲空间。应用程序可以在请求队列已满时（禁止通道之前），通过编程 USBFS_HCCHARx 寄存器将 CHDIS 位置 1 和将 CHENA 位清零，清空请求队列。出现以下任一情况时，应用程序将禁止通道：

- IN 或 OUT 通道的 USBFS_HCINTx 中接收到 STALL、TXERR、BBERR 或 DTERR 中断。应用程序在接收到通道停止信号之前，必须能够接收相同通道的其它中断（DTERR、Nak、Data、TXERR）。
- 接收到 USBFS_GINTSTS 中的 DISCINT（断开设备连接）中断。（应用程序将禁止所有已使能的通道）。
- 应用程序在传输正常完成之前将其中止。

在 DMA 模式时，应用程序不能同过写寄存器停止不可分割的周期性传输。

35.4.6 设备编程模型

35.4.6.1 USB 复位时端点初始化

操作步骤

步骤 1. 为所有 OUT 端点将 NAK 位置 1

- USBFS_DOEPTLx 中，SNAK=1（对于所有 OUT 端点）

步骤 2. 取消对以下中断位的屏蔽

- USBFS_DAINMSK 中，INEP0=1（控制 0 IN 端点）
- USBFS_DAINMSK 中，OUTEP0=1（控制 0 OUT 端点）
- DOEPMK 中，STUP=1
- DOEPMK 中，XFRC=1
- DIEPMK 中，XFRC=1
- DIEPMK 中，TOC=1

步骤 3. 为每个 FIFO 设置数据 FIFO RAM

- 对 USBFS_GRXFSIZ 寄存器进行编程，以能够接收控制传输的 OUT 数据和设置数据。该寄存器必须至少等于控制端点 0 的 1 个最大数据包大小+2 个字（用于控制 OUT 数据包的状态）+10 个字（用于 SETUP 数据包）。
- 对 USBFS_TX0FSIZ 寄存器进行编程（取决于所选的 FIFO 编号），以能够发送控制 IN 数据。该寄存器至少必须等于控制端点 0 的 1 个最大数据包大小。

步骤 4. 对端点相关寄存器中的以下字段进行编程，以使控制 OUT 端点 0 接收 SETUP 数据包

- USBFS_DOEPTSZ0 中的 STUPCNT=3（接收最多 3 个连续的 SETUP 数据包）

操作结果

此时，接收 SETUP 数据包所需的所有初始化工作便已完成。

35.4.6.2 USB 枚举完成时端点初始化

操作步骤

- 步骤 1. 在枚举完成中断 (USBFS_GINTSTS 中的 ENUMDNE) 中，读取 USBFS_DSTS 寄存器以确定设备的枚举速度。
- 步骤 2. 对 USBFS_DIEPCTL0 中的 MPSIZ 字段进行编程以设置最大数据包大小。该步骤配置控制端点 0。控制端点的最大数据包大小取决于枚举速度。

操作结果

此时，设备已准备好接收 SOF 数据包并配置为在控制端点 0 执行控制传输。

35.4.6.3 收到 SetAddress 命令时的端点初始化

背景信息

本节介绍了应用程序在 SETUP 数据包中接收到 SetAddress 命令时必须执行的操作。

操作步骤

- 步骤 1. 使用在 SetAddress 命令中接收到的设备地址来对 USBFS_DCFG 寄存器进行编程
- 步骤 2. 对模块进行编程以发出状态阶段的 IN 数据包

35.4.6.4 收到 SetConfiguration/SetInterface 命令时的端点初始化

背景信息

本节介绍了应用程序在 SETUP 包中接收 SetConfiguration 或 SetInterface 命令时必须执行的操作。

操作步骤

- 步骤 1. 接收到 SetConfiguration 命令时，应用程序必须对端点寄存器进行编程，以使用新配置中有效端点的特性来配置这些端点寄存器。
- 步骤 2. 接收到 SetInterface 命令时，应用程序必须对命令指定的端点的端点寄存器进行编程。
- 步骤 3. 在先前配置或其它设置中有效的端点在新的配置或其它设置中无效。必须停用这些无效端点。
- 步骤 4. 使用 USBFS_DAINMSK 寄存器使能有效端点的中断，屏蔽无效端点的中断。
- 步骤 5. 为每个 FIFO 设置数据 FIFO RAM。
- 步骤 6. 配置完所有必需的端点后，应用程序必须对模块进行编程以发送状态阶段的 IN 数据包。

操作结果

此时，设备模块已可以接收和发送任何类型的数据包。

35.4.6.5 端点激活

背景信息

本节介绍激活设备端点或者将现有设备端点配置为新类型所需的步骤。

操作步骤

- 步骤 1. 在 USBFS_DIEPCTLx 寄存器（对于 IN 或双向端点）或 USBFS_DOEPCTLx 寄存器（对于 OUT 或双向端点）的以下字段中，对所需端点的特性进行编程。
 - 最大数据包大小
 - USB 活动端点位置 1
 - 端点初始数据同步位（对于中断和批量端点）
 - 端点类型
 - TxFIFO 编号
- 步骤 2. 激活端点后，模块便开始解码发送到该端点的令牌，并在收到的令牌有效的情况下回复有效握手信号。

35.4.6.6 端点停用

背景信息

本节介绍停用现有端点所需的步骤。

操作步骤

- 步骤 1.** 在要停用的端点中，将 USBFS_DIEPCTLx 寄存器（对于 IN 或双向端点）或 USBFS_DOEPCTLx 寄存器（对于 OUT 或双向端点）中的 USB 活动端点位清零。
- 步骤 2.** 停用端点后，模块便会忽略发送到该端点的令牌，从而导致 USB 超时。

35.5 操作模型

35.5.1 SETUP 和 OUT 数据传输

本节介绍了数据 OUT 传输和 SETUP 事务期间的内部数据流和应用程序操作步骤。

数据包读取

介绍如何从接收 FIFO 读取数据包 (OUT 数据和 SETUP 数据包)。

1. 捕获到 RXFNE 中断 (USBFS_GINTSTS 寄存器) 时, 应用程序必须读取接收状态弹出寄存器 (USBFS_GRXSTSP)。
2. 应用程序可以通过写入 RXFNE=0 (在 USBFS_GINTMSK 中) 来屏蔽 RXFNE 中断 (在 USBFS_GINTSTS 中), 直到它把数据包从接收 FIFO 中读取出来。
3. 如果已接收数据包的字节计数不是 0, 则从接收数据 FIFO 中弹出这些数据并存储在存储器中。如果接收到的数据包字节计数为 0, 则不会从接收数据 FIFO 中弹出任何数据。
4. 从接收 FIFO 读出的数据包状态有以下几种状态:
 - 全局 OUT NAK:
PKTSTS=全局 OUT NAK, BCNT=0x000, EPNUM 和 DPID 的值无关紧要。
这些数据表示全局 OUT NAK 位已生效。
 - SETUP 数据包:
PKTSTS=SETUP, BCNT=0x008, EPNUM=控制 EP 编号, DPID=D0。这些数据表示指定端点上收到的 SETUP 数据包现在可从接收 FIFO 中读取。
 - 建立阶段完成:
PKTSTS=建立阶段完成, BCNT=0x0, EPNUM=控制 EP 编号, DPID 值无关紧要。
这些数据表示指定端点的建立阶段完成并且数据阶段已启动。在此状态条目从接收 FIFO 中弹出后, 模块将在该控制 OUT 端点上产生建立中断。
 - OUT 数据包:
PKTSTS=DataOUT, BCNT=接收的 OUT 数据包的大小(BCNT:0~1024), EPNUM=收到数据包的端点编号, DPID=实际数据 PID。
 - 数据传输完成:
PKTSTS=OUT 数据传输完成, BCNT=0x0, EPNUM=完成数据传输的 OUT EP 编号, DPID 值无关紧要。
这些数据表示指定 OUT 端点的 OUT 数据传输完成。在此状态条目从接收 FIFO 中弹出后, 模块将在指定的 OUT 端点上引发“传输完成”中断。
5. 从接收 FIFO 中弹出数据后, 必须取消对 RXFNE 中断的屏蔽(USBFS_GINTSTS)。
6. 每次应用程序检测到 USBFS_GINTSTS 中的 RXFNE 中断时, 都将重复步骤 1 到 5。读取空的接收 FIFO 可能导致未定义的模块行为。

SETUP 事务

介绍模块处理 SETUP 数据包的方式以及应用程序处理 SETUP 事务的顺序。

应用程序要求:

1. 要接收 SETUP 数据包, 必须将控制 OUT 端点 STUPCNT 字段(USBFS_DOEPTSIZE)编程为非零值。如果应用程序将 STUPCNT 字段编程为非零值, 模块会接收 SETUP 数据包并将其写入接收 FIFO, 而不考虑 NAK 状态和 USBFS_DOEPCTLx 中的 EPENA 位设置。控制端点每收到一个 SETUP 数据包后, STUPCNT 字段都会递减。如果在接收 SETUP 数据包之前, 未将 STUPCNT 字段编程为适当值, 模块仍能接收 SETUP 数据包并使 STUPCNT 字段递减, 但应用程序可能无法确定在控制传输的建立阶段中接收的 SETUP 数据包正确数量。
 - 在 USBFS_DOEPTSIZE 中, STUPCNT=3

- 应用程序必须始终在接收数据 FIFO 中分配一些额外空间，以便能够在控制端点上接收连续的最多三个 SETUP 数据包。
 - 预留空间 10 个字。第一个 SETUP 数据包需要 3 个字，“建立阶段完成”状态双字需要 1 个字，还需要 6 个字以存储两个额外的 SETUP 数据包。
 - 每个 SETUP 数据包需要 3 个字以存储 8 个字节的 SETUP 数据和 4 个字节的 SETUP 状态。模块将在接收 FIFO 中保留这些空间。
 - 这段 FIFO 仅用于存储 SETUP 包，绝对不会将该空间用于数据包。
- 应用程序必须从接收 FIFO 中读取 SETUP 数据包的 2 个字。
- 应用程序必须从接收 FIFO 中读取并丢弃“建立阶段完成”状态字。

内部数据流：

- 接收到 SETUP 数据包时，模块会将接收到的数据写入接收 FIFO，而不会检查接收 FIFO 中的可用空间，且不考虑端点的 NAK 和 STALL 位设置。
 - 模块会在内部将接收到 SETUP 数据包的控制 IN/OUT 端点的 IN NAK 和 OUTNAK 位置 1。
- USB 上接收到的每个 SETUP 数据包，模块会将 3 个字的数据写入接收 FIFO，并且将 STUPCNT 字段递减 1。
 - 第一个字包含由模块所使用的内部控制信息
 - 第二个字包含 SETUP 命令的前 4 个字节
 - 第三个字包含 SETUP 命令的最后 4 个字节
- 当建立阶段结束，数据 IN/OUT 阶段开始时，模块会将一个状态条目（“建立阶段完成”字）写入接收 FIFO，指示建立阶段完成。
- 在 AHB 端，SETUP 数据包被应用程序读取。
- 当应用程序从接收 FIFO 中弹出“建立阶段完成”字时，模块将使用 STUP 中断(USBFS_DOEPIINTx)来中断应用程序，指示其可以处理接收到的 SETUP 数据包。
 - 模块会将控制 OUT 端点的端点使能位清零。

应用程序编程顺序：

- 对 USBFS_DOEPTSIZx 寄存器进行编程。
 - STUPCNT=3
- 等待 RXFNE 中断(USBFS_GINTSTS)并且从接收 FIFO 中读取数据包。
- STUP 中断的触发(USBFS_DOEPIINTx)表示 SETUP 数据传输成功完成。
 - 发生该中断时，应用程序必须读取 USBFS_DOEPTSIZx 寄存器以确定接收的 SETUP 数据包数量并处理最后接收的 SETUP 数据包。

处理三个以上连续的 SETUP 数据包：

根据 USB2.0 规范，在 SETUP 数据包错误中，主机通常不会向同一个端点发送 3 个以上连续的 SETUP 数据包。但是，USB2.0 规范并未限制主机可以向同一个端点发送的连续 SETUP 数据包数量。发生这种情况时，USBFS 控制器将生成中断（USBFS_DOEPIINTx 中的 B2BSTUP）。

将全局 OUT NAK 置 1

内部数据流：

- 如果应用程序将全局 OUT NAK（USBFS_DCTL 中的 SGONAK 位）置 1，模块将停止向接收 FIFO 中写入 SETUP 数据包以外的数据。无论接收 FIFO 中可用空间大小如何，设备都会对主机发送的非同步 OUT 令牌回复 NAK，而对同步 OUT 数据包直接予以忽略。
- 模块将全局 OUT NAK 写入接收 FIFO。应用程序必须为此留出足够空间。
- 当应用程序从接收 FIFO 中弹出全局 OUT NAK 字时，模块会将 GONAKEFF 中断（USBFS_GINTSTS）置 1。

- 应用程序检测到该中断后，会认为模块处于全局 OUT NAK 模式。应用程序可以通过将 USBFS_DCTL 中的 SGONAK 位清零来清除该中断。

应用程序编程顺序：

- 要停止接收任何类型的数据到接收 FIFO 中，应用程序必须通过编程以下字段以将全局 OUT NAK 位置 1。
 - 在 USBFS_DCTL 中，SGONAK=1
- 等待 USBFS_GINTST 中的 GONAKEFF 中断。一旦被触发，该中断表示模块已停止接收 SETUP 数据包以外的任何类型数据。
 - 如果应用程序已将 USBFS_DCTL 中的 SGONAK 位置 1，则在模块引发 GONAKEFF 中断 (USBFS_GINTSTS) 之前，应用程序可以接收有效 OUT 数据包。
- 应用程序可通过对 USBFS_GINTMSK 寄存器中的 GINAKEFFM 位执行写操作来暂时屏蔽此中断。
 - 在 USBFS_GINTMSK 寄存器中，GINAKEFFM=0
- 当应用程序准备退出全局 OUT NAK 模式时，必须将 USBFS_DCTL 中的 SGONAK 位清零。此操作还会清除 GONAKEFF 中断 (USBFS_GINTSTS) 。
 - 在 CGONAK 中，USBFS_DCTL=1
- 如果应用程序在之前已屏蔽此中断，则必须按以下方式取消对该中断的屏蔽：
 - 在 GINTMSK 中，GINAKEFFM=1

将全局 OUT NAK 置 1

应用程序必须使用以下顺序禁止已使能的 OUT 端点。

应用程序编程顺序：

- 禁止任何 OUT 端点前，应用程序必须在模块中使能全局 OUT NAK 模式。
 - 在 USBFS_DCTL 中，SGONAK=1
- 等待 GONAKEFF 中断 (USBFS_GINTSTS)
- 通过编程以下字段来禁止 OUT 端点：
 - 在 USBFS_DOEPCTLx 中，EPDIS=1
 - 在 USBFS_DOEPCTLx 中，SNAK=1
- 等待 EPDISD 中断 (USBFS_DOEPINTx)，该中断表示已完全禁止 OUT 端点。引发 EPDISD 中断时，模块还会将以下位清零：
 - 在 USBFS_DOEPCTLx 中，EPDIS=0
 - 在 USBFS_DOEPCTLx 中，EPENA=0
- 应用程序必须将全局 OUT NAK 位清零，以开始从其它未禁止的 OUT 端点接收数据。
 - 在 USBFS_DCTL 中，SGONAK=0

通用非同步 OUT 数据传输

本节介绍一种常规非同步 OUT 数据传输（控制、批量或中断）。

应用程序要求：

- 建立 OUT 传输前，应用程序必须在存储器中分配一个缓冲区，以容纳要作为 OUT 传输的一部分而接收的所有数据。
- 对于 OUT 传输，端点的传输大小寄存器中的传输大小字段必须是端点的最大数据包大小的倍数（且以字对齐）。
 - 传输大小[EPNUM] = $n \times (\text{MPSIZ}[\text{EPNUM}] + 4 - (\text{MPSIZ}[\text{EPNUM}] \bmod 4))$
 - 数据包计数[EPNUM] = n
 - $n > 0$
- 发生 OUT 端点中断时，应用程序必须读取端点的传输大小寄存器以计算存储器中有效数据量。接收的有效数据量可能小于编程的传输大小。

- 存储器中的有效数据量 = 应用程序设置的初始传输量 - 模块更新后的剩余传输量
- 接收到 USB 数据包数 = 应用程序设置的初始数据包数 - 模块更新后的剩余数据包数

内部数据流：

1. 应用程序必须在端点相关寄存器中设置传输大小和数据包计数字段，将 NAK 位清零，并使能端点来接收数据。
2. NAK 位清零后，模块便开始接收数据并将数据写入接收 FIFO（只要接收 FIFO 中有空间）。对于 USB 上接收的每个数据包，数据包及其状态都会写入接收 FIFO。写入接收 FIFO 的每个数据包（数据量达到最大数据包大小的数据包或短数据包）都会使该端点的数据包计数字段递减 1。
 - 收到的数据包若 CRC 无效，则自动被从接收 FIFO 中清除。
 - 在 USB 上为数据包回复 ACK 后，模块将丢弃主机因无法检测到 ACK 而重新发送的非同步 OUT 数据包。应用程序不会在具有相同数据 PID 的相同端点上检测到多个连续的 OUT 数据包。在这种情况下，数据包计数不会递减。
 - 如果接收 FIFO 中没有空间，则会忽略同步或非同步数据包并且不会将它们写入接收 FIFO。此外，非同步 OUT 令牌将会收到 NAK 握手应答。
 - 在上述所有三种情况中，数据包计数都不会递减，因为没有任何数据写入接收 FIFO。
3. 当数据包计数变为 0 或者在端点上接收到短数据包时，该端点的 NAK 位将置 1。NAK 位置 1 后，将忽略同步或非同步数据包并且不会将它们写入接收 FIFO，同时非同步 OUT 令牌会收到 NAK 握手应答。
4. 在数据写入接收 FIFO 后，应用程序将从接收 FIFO 中读取数据并将数据写入外部存储器，一次一个数据包，逐个端点过来。
5. 在 AHB 上向外部存储器写入完每个数据包后，端点的传输大小都会自动减去该数据包的大小。
6. 在以下情况时，OUT 端点的 OUT 数据传输完成状态将写入接收 FIFO：
 - 传输大小为 0 并且数据包计数为 0
 - 写入接收 FIFO 的最后一个 OUT 数据包是短数据包
(数据包大小:0~最大数据包大小-1)
7. 当应用程序弹出此状态条目（OUT 数据传输完成），并生成该端点的传输完成中断，同时清零端点使能位。

应用程序编程顺序：

1. 使用传输大小和相应数据包个数对 USBFS_DOEPSIZx 寄存器进行编程。
2. 使用端点特性对 USBFS_DOEPCTLx 寄存器进行编程，并将 EPENA 和 CNAK 位置 1。
 - 在 USBFS_DOEPCTLx 中，EPENA=1
 - 在 USBFS_DOEPCTLx 中，CNAK=1
3. 等待 RXFNE 中断（在 USBFS_GINTSTS 中）并且从接收 FIFO 中读走数据包。
 - 此步骤可重复多次，具体取决于传输大小。
4. 触发 XFRC 中断（USBFS_DOEPINTx），以表示非同步 OUT 数据传输成功完成。
5. 读取 USBFS_DOEPSIZx 寄存器，以确定有效数据量。

通用同步 OUT 数据传输

本节介绍常规的同步 OUT 数据传输。

应用程序要求：

1. 非同步 OUT 数据传输的所有应用程序要求均适用于同步 OUT 数据传输。
2. 对于同步 OUT 数据传输中的传输大小和数据包计数字段，必须始终将其设置为单个帧中可接收的最大数据包大小的数据包数目。同步类型的 OUT 数据传输事务必须在一个帧内完成。
3. 在周期性帧结束（USBFS_GINTSTS 中的 EOPF 中断）之前，应用程序必须从接收 FIFO 中读取所有同步 OUT 数据包（数据条目和状态条目）。

4. 要接收下一帧中的数据，必须在 EOPF (USBFS_GINTSTS) 之后 SOF (USBFS_GINTSTS) 之前使能一个同步 OUT 端点。

内部数据流：

1. 同步 OUT 端点的内部数据流与非同步 OUT 端点的基本相同，但稍有差异。
2. 同步 OUT 端点通过将端点使能位置 1 并将 NAK 位清零来使能时，必须相应地将偶数/奇数帧位置 1。仅当符合以下条件时，模块才会在同步 OUT 端点上接收特定帧中的数据：
 - EONUM (在 USBFS_DOEPCTLx 中) = FNSOF[0] (在 USBFS_DSTS 中)
3. 当应用程序从接收 FIFO 中完整地读取一个同步 OUT 数据包 (数据和状态) 时，模块会根据从接收 FIFO 中读取的最后一个同步 OUT 数据包的数据 PID 更新 USBFS_DOEPSIZx 中的 RXDPID 字段。

应用程序编程顺序：

1. 使用传输大小和相应数据包计数对 USBFS_DOEPSIZx 寄存器进行编程。
2. 使用端点特性对 USBFS_DOEPCTLx 寄存器进行编程，并将端点使能位、清除 NAK 位和奇数/偶数帧位置 1。
 - EPENA1
 - CNAK=1
 - EONUM= (0: 偶数/1: 奇数)
3. 等待 RXFNE 中断 (在 USBFS_GINTSTS 中) 并且从接收 FIFO 中读走数据包。
 - 此步骤可重复多次，具体取决于传输大小。
4. XFRC 中断 (在 USBFS_DOEPINTx 中) 表示同步 OUT 数据传输完成。该中断不一定意味着存储器中的数据是有效的。
5. 对于同步 OUT 传输，应用程序可能并不总会检测到该中断。相反，应用程序可能检测到 USBFS_GINTSTS 中的 IISOXFRM 中断。
6. 读取 USBFS_DOEPSIZx 寄存器以确定接收的传输大小以及确定帧中接收的数据的有效性。仅当符合以下条件之一时，应用程序才必须将存储器中接收的数据视为有效数据：
 - RXDPID=D0 (在 USBFS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=1
 - RXDPID=D1 (在 USBFS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=2
 - RXDPID=D2 (在 USBFS_DOEPSIZx 中) 并且接收该有效数据的 USB 数据包数量=3

接收该有效数据的 USB 数据包数量=应用程序编程的初始数据包个数-模块更新后的剩余数据包个数。

应用程序可将无效数据包丢弃。

不完整的同步 OUT 数据传输

本节介绍了同步 OUT 数据包出现丢包时应用程序编程顺序。

内部数据流：

1. 对于同步 OUT 端点，可能不会始终引发 XFRC 中断 (在 USBFS_DOEPINTx 中)。如果模块丢弃同步 OUT 数据包，则在以下情况下，应用程序可能无法检测到 XFRC 中断 (USBFS_DOEPINTx)：
 - 在接收 FIFO 无法容纳完整的 ISO OUT 数据包时，模块将丢弃接收到的 ISO OUT 数据
 - 接收到的同步 OUT 数据包存在 CRC 错误
 - 模块接收到的同步 OUT 令牌损坏
 - 应用程序从接收 FIFO 中读取数据的速度非常缓慢
2. 如果模块在所有同步 OUT 端点的传输完成前检测到周期性帧结束，将触发未完成同步 OUT 数据中断 (USBFS_GINTSTS 中的 IISOXFRM)，指示至少有一个同步 OUT 端点上未触发 XFRC 中断 (在 USBFS_DOEPINTx 中)。此时，未完成传输的端点仍保持使能，但在 USB 的该端点上，没有进行中的有效传输。

应用程序编程顺序：

1. 硬件触发 IISOOXFRM 中断 (USBFS_GINTSTS) 表示当前帧中至少有一个同步 OUT 端点具有未完成的传输。
2. 如果因未从端点完全读取同步 OUT 数据而发生这种情况，应用程序必须确保首先从接收 FIFO 读取走所有同步 OUT 数据（包括数据条目和状态条目），然后再继续处理。
 - 从接收 FIFO 读取所有数据后，应用程序即可检测到 XFRC 中断 (USBFS_DOEPINTx)。在此情况下，应用程序必须重新使能端点以接收下一个帧中的同步 OUT 数据。
3. 当应用程序接收到 IISOOXFRM 中断（在 USBFS_GINTSTS 中）时，应用程序必须读取所有同步 OUT 端点的控制寄存器 (USBFS_DOEPCTLx)，以确定哪些端点在当前帧中具有不完整的传输。同时满足以下两个条件时，表示端点传输未完成：
 - EONUM 位（在 USBFS_DOEPCTLx 中）= FNSOF[0]（在 USBFS_DSTS 中）
 - EPENA=1（在 USBFS_DOEPCTLx 中）
4. 在检测到 SOF 中断（在 USBFS_GINTSTS 中）前，必须执行完成上一步操作，以确保当前帧编号未发生更改。
5. 对于具有不完整传输的同步 OUT 端点，应用程序必须丢弃存储器中的数据，并通过将 USBFS_DOEPCTLx 中的 EPDIS 位置 1 来禁止端点。
6. 等待 EPDIS 中断（在 USBFS_DOEPINTx 中），并且使能端点以在下一帧中接收新数据。
 - 由于模块可能需要一些时间才能禁止端点，因此应用程序在接收到无效同步数据后，可能无法接收下一个帧中的数据。

停止非同步 OUT 端点

本节介绍应用程序如何才能停止非同步端点。

1. 将模块置于全局 OUT NAK 模式。
2. 禁止所需的端点
 - 禁止端点时，请设置 STALL=1（在 USBFS_DOEPCTL 中），而不是将 USBFS_DOEPCTL 中的 SNAK 位置 1。STALL 位的优先级始终高于 NAK 位。
3. 当应用程序不再需要端点回复 STALL 握手信号时，必须将 STALL 位（在 USBFS_DOEPCTLx 中）清零。
4. 如果应用程序由于收到主机的 SetFeature.Endpoint Halt 或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点上的状态阶段传输前，将 STALL 位置 1 或清零。

35.5.2 IN 数据传输**数据包写入**

本节介绍在已使能专用发送 FIFO 的情况下应用程序如何将数据包写入端点 FIFO。

1. 应用程序可以选择轮询模式或中断模式。
 - 在轮询模式下，应用程序通过读取 USBFS_DTXFSTSx 寄存器来监视端点发送数据 FIFO 的状态，从而确定数据 FIFO 中是否有足够空间。
 - 在中断模式下，应用程序等待 TXFE 中断（在 USBFS_DIEPINTx 中），然后读取 USBFS_DTXFSTSx 寄存器以确定数据 FIFO 中是否有足够空间。
 - 要写入单个非零长度的数据包，数据 FIFO 中必须有足够的空间来容纳整个数据包。
 - 要写入零长度的数据包，应用程序不能查看 FIFO 空间。
2. 如果使用上述方法之一，当应用程序确定有足够空间来写入发送数据包时，应用程序必须首先对端点控制寄存器进行相应写操作，然后再将数据写入数据 FIFO。通常，应用程序必须对 USBFS_DIEPCTLx 寄存器执行读-修改-写操作，以避免在将端点使能位置 1 的同时，修改寄存器中的其它内容。

如果有足够空间，应用程序可将同一端点的多个数据包写入发送 FIFO。对于周期性 IN 端点，应用程序只能一次写入一个帧内的多个数据包。只有先前一个帧的通信事务传输完成之后，应用程序才会写入下一个帧内要发送的所有数据包。

将 IN 端点 NAK 置 1

内部数据流：

1. 当应用程序将特定端点的 IN NAK 置 1 时，模块将停止端点上的数据发送，而不考虑端点发送 FIFO 中的数据是否可用。
2. 非同步端点收到 IN 令牌，回复 NAK 握手应答。
 - 同步端点收到 IN 令牌，返回零长度数据包
3. 模块在 USBFS_DIEPINTx 中触发 INEPNE (IN 端点 NAK 有效) 中断以响应 USBFS_DIEPCTLx 中的 SNAK 位。
4. 应用程序检测到该中断后，便会认为端点处于 IN NAK 模式。应用程序可通过将 USBFS_DIEPCTLx 中的 CNAK 位置 1 来清除该中断。

应用程序编程顺序：

1. 要在特定 IN 端点上停止发送任何数据，应用程序必须将 IN NAK 位置 1。要将该位置 1，必须编程以下字段。
 - USBFS_DIEPCTLx 中的 SNAK=1
2. 等待 USBFS_DIEPINTx 中的 INEPNE 中断触发。该中断表示模块已在端点上停止发送数据。
3. 在应用程序将 NAK 位置 1 但“NAK 有效”中断尚未触发时，模块可以在端点上发送有效 IN 数据。
4. 应用程序可通过写入 DIEPMSK 中的 INEPNEM 位来临时屏蔽该中断。
 - 在 DIEPMSK 中，INEPNEM = 0
5. 要退出端点 NAK 模式，应用程序必须将 USBFS_DIEPCTLx 中的 NAK 状态位 (NAKSTS) 清零。此操作还会清除 INEPNE 中断 (在 USBFS_DIEPINTx 中)。
 - 在 USBFS_DIEPCTLx 中，CNAK=1
6. 如果应用程序已将该中断屏蔽，则必须按以下方式取消屏蔽：
 - 在 DIEPMSK 中，INEPNEM=1

禁止 IN 端点

使用以下顺序来禁止先前已使能的特定 IN 端点。

应用程序编程顺序：

1. 应用程序必须先停止在 AHB 上写入数据，之后才能禁止 IN 端点。
2. 应用程序必须将端点设置为 NAK 模式。
 - USBFS_DIEPCTLx 中的 SNAK=1
3. 等待 USBFS_DIEPINTx 中的 INEPNE 中断。
4. 将必须禁止的端点的 USBFS_DIEPCTLx 寄存器中的以下位置 1。
 - USBFS_DIEPCTLx 中的 EPDIS=1
 - USBFS_DIEPCTLx 中的 SNAK=1
5. USBFS_DIEPINTx 中的 EPDISD 中断的触发表示模块已完全禁止指定的端点。在触发中断的同时，模块还会将以下位清零：
 - 在 USBFS_DIEPCTLx 中，EPENA=0
 - 在 USBFS_DIEPCTLx 中，EPDIS=0
6. 应用程序必须为周期性 IN EP 读取 USBFS_DIEPTSIZx 寄存器，以计算端点上有多少数据是在 USB 上发送的。
7. 应用程序必须通过将 USBFS_GRSTCTL 寄存器中的以下字段置 1，来清空端点发送 FIFO 中的数据：

- TXFNUM (在 USBFS_GRSTCTL 中) = 端点发送 FIFO 编号
- TXFFLSH (在 USBFS_GRSTCTL 中) = 1

应用程序必须轮询 USBFS_GRSTCTL 寄存器, 直至模块将 TXFFLSH 位清零, 这表示 FIFO 清空操作结束。要在该端点上发送新数据, 应用程序可以在稍后重新使能该端点。

通用非周期性 IN 数据传输

应用程序要求:

1. 建立 IN 传输前, 应用程序必须确保组成一次 IN 传输的每个数据包都可以容纳在单个缓冲区中。
2. 对于 IN 传输, 端点传输大小寄存器中的传输大小字段表示本次传输的有效数据量, 它由多个最大数据包大小和单个短数据包组成。该短数据包在传输结束时发送。
 - 要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包:
 传输大小[EPNUM] = $x \times \text{MPSIZ}[\text{EPNUM}] + \text{sp}$
 如果 ($\text{sp} > 0$), 数据包计数[EPNUM] = $x + 1$ 。
 否则, 数据包计数[EPNUM] = x
 - 要发送单个零长度数据包:
 传输大小[EPNUM] = 0
 数据包计数[EPNUM] = 1
 - 要发送多个最大数据包大小的数据包并在传输结束时外加一个零长度数据包, 应用程序必须将传输拆分为两个部分。
 第一部分发送最大数据包大小的数据包, 第二部分仅发送零长度数据包。
 第一次传输: 传输大小[EPNUM] = $x \times \text{MPSIZ}[\text{epnum}]$; 数据包计数 = n ;
 第二次传输: 传输大小[EPNUM] = 0; 数据包计数 = 1;
3. 使能某个端点进行数据传输后, 模块会更新传输大小寄存器。在 IN 传输结束时, 应用程序必须读取传输大小寄存器, 以确定送入发送 FIFO 中的数据已有多少通过 USB 发送出去。
4. 送入发送 FIFO 中的数据量 = 应用程序编程的初始传输大小 - 模块更新后的最终传输大小
 - 通过 USB 已经发送的数据量 = (应用程序编程的初始数据包计数 - 模块更新后的最终数据包计数) $\times$ MPSIZ[EPNUM]
 - 要通过 USB 发送的剩余数据量 = (应用程序编程的初始传输大小 - 已通过 USB 发送的数据量)

内部数据流:

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段, 并使能该端点来发送数据。
2. 应用程序还必须向该端点的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包, 该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO, 直到该端点的传输大小变为 0。向 FIFO 写入数据后, “FIFO 中的数据包数”计数会递增 (这是一个 3 位计数, 由模块在内部进行维护, 每个 IN 端点发送 FIFO 对应一个。在 IN 端点 FIFO 中, 模块所维护的最大数据包数始终为八个)。对于零长度数据包, 每个 FIFO 均另有一个单独的标志, FIFO 中没有任何数据。
4. 当数据写入发送 FIFO 后, 模块会在接收到 IN 令牌时将这些数据送出。每个数据包发送出去并收到回复的 ACK 握手信号后, 该端点的数据包计数都会递减 1, 直到数据包计数变 0 为止。发生超时, 数据包计数不会递减。
5. 对于零长度数据包 (由内部零长度标志指示), 模块会针对 IN 令牌发出一个零长度数据包, 并递减数据包计数字段的值。
6. 如果接收到 IN 令牌的端点对应的 FIFO 中无数据, 且该端点的数据包计数字段为零, 则模块会针对该端点生成一个“TxFIFO 为空时接收到 IN 令牌” (ITTXFE) 中断 (前提是该端点的 NAK 位未置 1)。模块在该非同步端点上回复 NAK 握手信号。
7. 模块会在内部使 FIFO 指针重新返回到开头, 并且不会生成超时中断。

8. 当传输大小为 0 且数据包计数为 0 时，将生成该端点的传输完成 (XFRC) 中断，同时将端点使能清零。

应用程序编程顺序：

1. 使用传输大小和相应数据包计数对 USBFS_DIEPTSIZE 寄存器进行编程。
2. 使用端点特性对 USBFS_DIEPTCTL 寄存器进行编程，并将 CNAK 和 EPENA（端点使能）位置 1。
3. 发送非零长度数据包时，应用程序必须轮询 USBFS_DTXFSTS 寄存器（其中 x 为与该端点相关联的 FIFO 编号）以确定数据 FIFO 中是否有足够的空间。写入数据前，应用程序也可选用 TXFE 位（在 USBFS_DIEPTINT 中）。

通用周期性 IN 数据传输

本节介绍典型的周期性 IN 数据传输。

应用程序要求：

1. 通用非周期性 IN 数据传输的应用程序要求 1、2、3、4 对周期性 IN 数据传输同样适用（只是对要求 2 稍加修改）。
 - 应用程序只能发送若干个最大数据包大小的数据包或若干个最大数据包大小的包，外加传输结束时的一个短数据包。
要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包，必须满足以下条件：
传输大小[EPNUM] = $x \times \text{MPSIZE}[EPNUM] + sp$
(其中 x 是大于 0 的整数，且 sp 范围是 0~MPSIZE[EPNUM]-1)
如果 (sp>0)，数据包计数[EPNUM] = $x + 1$
否则，数据包计数[EPNUM] = x;
MCNT[EPNUM] = 数据包计数[EPNUM]
 - 应用程序无法在传输结束时发送零长度数据包。应用程序可以单独发送一个零长度数据包。
 - 要发送单个零长度数据包：
传输大小[EPNUM]=0
数据包计数[EPNUM]=1
MCNT[EPNUM]=数据包计数[EPNUM]
2. 应用程序一次只能安排一帧的数据传输。
 - $(MCNT - 1) \times \text{MPSIZE} \leq \text{XFERSIZE} \leq MCNT \times \text{MPSIZE}$
 - PKTCNT = MCNT（在 USBFS_DIEPTSIZE 中）
 - 如果 XFERSIZE < MCNT × MPSIZE，则传输的最后一个数据包为短数据包
 - 请注意：MCNT 位于 USBFS_DIEPTSIZE 中、MPSIZE 位于 USBFS_DIEPTCTL 中、PKTCNT 位于 USBFS_DIEPTSIZE 中、XFERSIZE 位于 USBFS_DIEPTSIZE 中
3. 接收到 IN 令牌前，应用程序必须将要在帧中发送的完整数据写入到发送 FIFO 中。在接收到 IN 令牌时，即使发送 FIFO 中该帧要发送的数据只差 1 个双字未写进来，模块也会执行 FIFO 为空时的操作。当发送 FIFO 为空时：
 - 同步端点上将回复零长度数据包
 - 中断端点上将回复 NAK 握手信号

内部数据流：

1. 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。
2. 应用程序还必须向与该端点相关联的发送 FIFO 写入必需的数据。
3. 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。
4. 当周期性端点接收到 IN 令牌时，模块将开始发送 FIFO 中的数据（如果 FIFO 中有数据）。如果 FIFO 中没有该帧要发送的数据的完整数据包，则模块将为该端点生成一个“TxFIFO 为空时接收到 IN 令牌”中断。

- 同步端点上将回复零长度数据包
 - 中断端点上将回复 NAK 握手信号
5. 端点的数据包计数会在下列情况下递减 1:
- 对于同步端点, 发送一个零长度或非零长度的数据包时
 - 对于中断端点, 在发送 ACK 握手信号时递减
 - 当传输大小和数据包计数均为 0 时, 将生成该端点的传输完成中断, 同时将端点使能位清零。
6. 在“周期性帧间隔”(由 USBFS_DCFG 中的 PFIVL 位控制)内, 当模块发现任何在当前帧内应为空的同步 IN 端点 FIFO 中的数据还未发送完成时, 都会在 USBFS_GINTSTS 中生成一个 IISOIXFR 中断。

应用程序编程顺序:

1. 使用端点特性对 USBFS_DIEPCTLx 寄存器进行编程, 并将 CNAK 和 EPENA 位置 1。
2. 将需要在下一帧中发送的数据写入发送 FIFO。
3. 硬件触发 ITTXFE 中断 (在 USBFS_DIEPINTx 中) 表示应用程序尚未将需要发送的全部数据写入发送 FIFO。
4. 如果在检测到中断前已使能中断端点, 则将忽略该中断。如果中断端点未使能, 则使能此端点, 以便数据能够在收到下一次 IN 令牌时发送出去。
5. 硬件触发 XFRC 中断 (在 USBFS_DIEPINTx 中) 时如果 USBFS_DIEPINTx 中未产生 ITTXFE 中断, 则表示成功完成同步 IN 传输。读取 USBFS_DIEPTSIZx 寄存器时始终得到传输大小=0 且数据包计数=0, 则表示所有数据都已通过 USB 发送完毕。
6. 置位 XFRC 中断 (在 USBFS_DIEPINTx 中) 时无论是否产生 ITTXFE 中断 (在 USBFS_DIEPINTx 中), 都表示成功完成中断 IN 传输。读取 USBFS_DIEPTSIZx 寄存器时始终得到传输大小=0 且数据包计数=0, 则表示所有数据都已通过 USB 发送完毕。
7. 在 USBFS_GINTSTS 中置位未完成的同步 IN 传输 (IISOIXFR) 中断时如果未产生任何前述中断, 则表示在当前帧中模块至少未收到 1 个周期性的 IN 令牌。

未完成同步 IN 数据传输

本节介绍应用程序针对未完成同步 IN 数据传输必须执行的操作。

内部数据流:

1. 符合下列条件之一时, 即认为同步 IN 传输未完成:
 - 模块在至少一个同步 IN 端点上接收到损坏的同步 IN 令牌。此时, 应用程序检测到未完成同步 IN 传输中断 (USBFS_GINTSTS 中的 IISOIXFR 位)。
 - 应用程序向发送 FIFO 写入数据的速度过慢, 在将完整数据写入 FIFO 之前便接收到 IN 令牌。此时, 应用程序在 USBFS_DIEPINTx 中检测到“TxFIFO 为空时接收到 IN 令牌”中断。应用程序可忽略此中断, 因为最终这将在周期性帧结束时产生一个未完成同步 IN 传输中断 (USBFS_GINTSTS 中的 IISOIXFR 位)。模块会通过 USB 发送一个零长度数据包来响应接收到的 IN 令牌。
2. 应用程序必须尽快停止向发送 FIFO 写入数据。
3. 应用程序必须将端点的 NAK 位和禁止位置 1。
4. 模块会禁止该端点, 将禁止位清零并触发端点的“端点禁止”中断。

应用程序编程顺序:

1. 应用程序可以在任何同步 IN 端点上忽略 USBFS_DIEPINTx 中的“TxFIFO 为空时接收到 IN 令牌”中断, 因为最终这将产生一个未完成同步 IN 传输中断 (在 USBFS_GINTSTS 中)。
2. 硬件触发未完成同步 IN 传输中断 (在 USBFS_GINTSTS 中) 表示在至少一个同步 IN 端点上存在未完成的同步 IN 传输。
3. 应用程序必须读取所有同步 IN 端点的“端点控制”寄存器来检测存在未完成 IN 数据传输的端点。
4. 应用程序必须停止向与这些端点相关联的“周期性发送 FIFO”写入数据。
5. 对 USBFS_DIEPCTLx 寄存器中的下列字段进行编程以禁止端点:

- USBFS_DIEPCTLx 中的 SNAK=1
 - USBFS_DIEPCTLx 中的 EPDIS=1
6. 硬件触发 USBFS_DIEPINTx 中的“端点禁止”中断表示模块已禁止该端点。
- 此时，应用程序必须清空相关联的发送 FIFO 中的数据，或者通过在下一帧中使能新传输的端点来覆盖 FIFO 中的现有数据。要刷新数据，应用程序必须使用 USBFS_GRSTCTL 寄存器。

停止非同步 IN 端点

本节介绍应用程序如何才能停止非同步端点。

应用程序编程顺序：

1. 禁止要停止的 IN 端点。同时将 STALL 位置 1。
2. USBFS_DIEPCTLx 中的 EPDIS=1（当端点已使能时）
 - USBFS_DIEPCTLx 中的 STALL=1
 - STALL 位的优先级始终高于 NAK 位
3. 硬件触发“端点禁止”中断（在 USBFS_DIEPINTx 中）可以让应用程序知道模块已禁止指定端点。
4. 应用程序必须根据端点类型清空非周期性或周期性发送 FIFO。对于非周期性端点，应用程序必须重新使能另一个无需停止的非周期性端点来发送数据。
5. 当应用程序准备好结束该端点的 STALL 握手信号时，必须将 USBFS_DIEPCTLx 的 STALL 位清零。
6. 如果应用程序因收到来自主机的 SetFeature.Endpoint Halt 命令或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点的状态阶段传输之前将 STALL 位置 1 或清零。

特例：停止控制 OUT 端点

如果在控制传输的数据阶段，主机发送的 IN/OUT 令牌数超过 SETUP 数据包指定的值，则模块必须对这些多余的 IN/OUT 令牌回复 STALL。在这种情况下，应用程序必须在控制传输的数据阶段使能 USBFS_DIEPINTx 的 ITTXFE 中断和 USBFS_DOEPINTx 的 OTEPDIS 中断（当模块已完成传输 SETUP 数据包指定的数据量后）。随后，当应用程序收到此中断时，必须将相应端点控制寄存器中的 STALL 位置 1 并清除此中断。

35.6 寄存器

应用程序通过 AHB 从接口对控制和状态寄存器进行读写操作，以此来控制 USBFS 模块。USBFS 模块所有寄存器为 32 位寄存器，其地址按 32 位对齐，因此只能以 32 位的方式访问。

控制和状态寄存器分为以下几类：

- USBFS 系统控制寄存器
- 模块全局寄存器
- 主机模式寄存器
- 设备模式寄存器
- 电源和时钟门控控制寄存器
- 数据 FIFO(DFIFO)访问寄存器

其中 USBFS 系统控制寄存器与其它寄存器基地址不同，该寄存器独立于 USBFS 模块以控制 USBFS 模块的相关设定。

只有模块全局寄存器、电源和时钟门控控制寄存器和数据 FIFO 访问寄存器可以在主机和设备模式下进行访问。当 USBFS 模块处以一种模式（主机或者设备）下，应用程序不得以另外一种角色模式访问寄存器，比如主机模式时，访问设备模式的寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器 USBFS.GINTSTS.NMIS 位反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。

控制状态寄存器存储器映射

主机和设备模式寄存器占用不同的地址。所有寄存器均在 AHB 时钟域内实现。

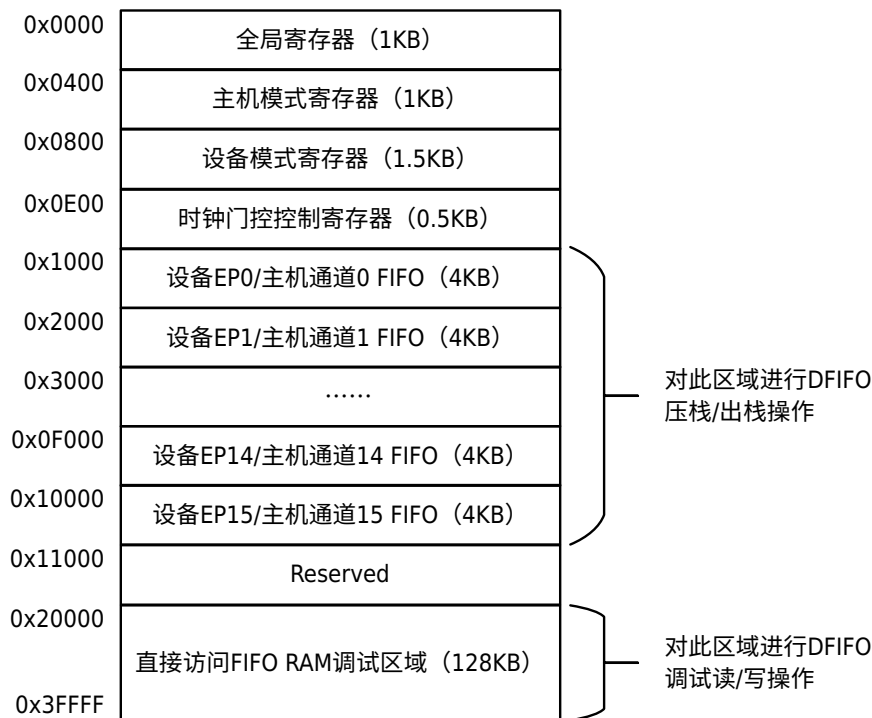


图 35-7 USBFS 控制状态寄存器存储器映射

35.6.1 寄存器总表

USB 系统控制寄存器基地址：0x40055400

表 35-3 USB 系统控制寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	USB_SYCTLREG	USB 系统控制寄存器	32

USBFS 模块寄存器基地址：0x40080000

表 35-4 USB2.0 全速模块 USBFS 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	USBFS_GVBUSCFG	VBUS 控制寄存器	32
0x08	USBFS_GAHBCFG	AHB 控制寄存器	32
0x0C	USBFS_GUSBCFG	USB 配置寄存器	32
0x10	USBFS_GRSTCTL	复位寄存器	32
0x14	USBFS_GINTSTS	模块中断寄存器	32
0x18	USBFS_GINTMSK	中断屏蔽寄存器	32
0x1C	USBFS_GRXSTSR	接收状态调试读取寄存器	32
0x20	USBFS_GRXSTSP	接收状态读取和出栈寄存器	32
0x24	USBFS_GRXFSIZ	接收 FIFO 大小寄存器	32
0x28	USBFS_HNPTXFSIZ	主机非周期性发送 FIFO 大小寄存器	32
0x28	USBFS_DIEPRXF0	设备端点 0 发送 FIFO 大小寄存器	32
0x2C	USBFS_HNPTXSTS	非周期性发送 FIFO/队列状态寄存器	32
0x3C	USBFS_CID	模块 ID 寄存器	32
0x54	USBFS_GLPMCFG	LPM 配置寄存器	32
0x100	USBFS_HPTXFSIZ	主机周期性发送 FIFO 大小寄存器	32
0x100+0x4*x	USBFS_DIEPTXFx	设备 IN 端点 n 发送 FIFO 大小寄存器 (x=1~15)	32
0x400	USBFS_HCFG	主机配置寄存器	32
0x404	USBFS_HFIR	主机帧时间间隔寄存器	32
0x408	USBFS_HFNUM	主机帧编号/帧剩余时间间隔寄存器	32
0x410	USBFS_HPTXSTS	主机周期性发送 FIFO/队列状态寄存器	32
0x414	USBFS_HAINT	主机全体通道中断寄存器	32
0x418	USBFS_HAINTMSK	主机全体通道中断屏蔽寄存器	32
0x440	USBFS_HPRT	USBFS 主机端口控制和状态寄存器	32
0x500+0x20*x	USBFS_HCCHARx	主机通道 x 特性寄存器	32

偏移地址	寄存器	描述	位宽
0x508+0x20*x	USBFS_HCINTx	主机通道 x 中断寄存器	32
0x50C+0x20*x	USBFS_HCINTMSKx	主机通道 x 中断屏蔽寄存器	32
0x510+0x20*x	USBFS_HCTSIZx	主机通道 x 传输大小寄存器	32
0x514+0x20*x	USBFS_HCDMAx	主机通道 x DMA 地址寄存器	32
0x800	USBFS_DCFG	设备配置寄存器	32
0x804	USBFS_DCTL	设备控制寄存器	32
0x808	USBFS_DSTS	设备状态寄存器	32
0x810	USBFS_DIEPMSK	设备 IN 端点通用中断屏蔽寄存器	32
0x814	USBFS_DOEPMSK	设备 OUT 端点通用中断屏蔽寄存器	32
0x818	USBFS_DAINTE	设备全体端点中断寄存器	32
0x81C	USBFS_DAINTEMSK	设备全体端点中断屏蔽寄存器	32
0x834	USBFS_DIEPEMPMSK	设备 IN 端点 FIFO 空中断屏蔽寄存器	32
0x900	USBFS_DIEPCTL0	设备 IN 端点 0 控制寄存器	32
0x900+0x20*x	USBFS_DIEPCTLx	设备 IN 端点 x 控制寄存器 (x=1~15)	32
0x908+0x20*x	USBFS_DIEPINTx	设备 IN 端点 x 中断寄存器	32
0x910	USBFS_DIEPTSIZ0	设备 IN 端点 0 传输大小寄存器	32
0x910+0x20*x	USBFS_DIEPTSIZx	设备 IN 端点 x 传输大小寄存器 (x=1~15)	32
0x914+0x20*x	USBFS_DIEPDMAx	设备 IN 端点 x DMA 地址寄存器	32
0x918+0x20*x	USBFS_DTXFSTSx	设备 IN 端点 x 发送 FIFO 状态寄存器	32
0xB00	USBFS_DOEPCTLx	设备 OUT 端点 0 控制寄存器	32
0xB00+0x20*x	USBFS_DOEPCTLx	设备 OUT 端点 x 控制寄存器 (x=1~15)	32
0xB08+0x20*x	USBFS_DOEPINTx	设备 OUT 端点 x 中断寄存器	32
0xB10	USBFS_DOEPSIZ0	设备 OUT 端点 0 传输大小寄存器	32
0xB10+0x20*x	USBFS_DOEPSIZx	设备 OUT 端点 x 传输大小寄存器 (x=1~15)	32
0xB14+0x20*x	USBFS_DOEPDMAx	设备 OUT 端点 x DMA 地址寄存器	32
0xE00	USBFS_GCCTL	时钟门控控制寄存器	32



说明

表中的 x 代表存在多个同类型寄存器，取值为 0~15。特殊标识的以寄存器行描述为准。

35.6.2 USB 系统控制寄存器

35.6.2.1 USB 系统控制寄存器 (USB_SYCTLREG)

Offset	Bit Position																																					
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved				USBHS_NFE		USBHS_NFS		Reserved				USBFS_NFE		USBFS_NFS		Reserved				USBHS_FSPHYE		USBHS_SOFEN		USBHS_DFB		Reserved				USBFS_SOFEN		USBFS_DFB					
Access					RW		RW						RW		RW						RW		RW		RW						RW		RW					

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26	USBHS_NFE	USBHS 滤波使能寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的开关 0b0：模拟滤波器关 0b1：模拟滤波器开，滤波范围请参考 USBHS_NFS 的设定	RW
25:24	USBHS_NFS	USBHS 滤波选择寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的滤波范围 0b00：滤波宽度档位 1 0b01：滤波宽度档位 2 0b10：滤波宽度档位 3 0b11：滤波宽度档位 4 各档位具体值请参考《数据手册》的“电气特性”章节中的“USB 片上全速 PHY STOP 模式下滤波特性”。	RW
23:19	Reserved	保留	-	-
18	USBFS_NFE	USBFS 滤波使能寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的开关 0b0：模拟滤波器关 0b1：模拟滤波器开，滤波范围请参考 USBFS_NFS 的设定	RW
17:16	USBFS_NFS	USBFS 滤波选择寄存器	用于控制 STOP 模式下，USBHS 片上全速 PHY DP/DM 模拟滤波器的滤波范围 0b00：滤波宽度档位 1 0b01：滤波宽度档位 2 0b10：滤波宽度档位 3 0b11：滤波宽度档位 4 各档位具体值请参考《数据手册》的“电气特性”章节中的“USB 片上全速 PHY STOP 模式下滤波特性”章节。	RW
15:11	Reserved	保留	-	-
10	USBHS_FSPHYE	USBHS 片上全速 PHY 使能位	0b0：USBHS 片上全速 PHY 挂起 0b1：USBHS 片上全速 PHY 使能	RW

位/位域	标记	位名	功能描述	读写
9	USBHS_SOFEN	USBHS SOF 脉冲输出使能位	USBHS 主机发出 SOF 或者设备成功接收到 SOF 时, 16 个 HCLK 时钟周期宽度的 SOF 脉冲从 PAD 输出使能 ● 0b0: SOF 脉冲不输出 ● 0b1: SOF 脉冲输出	RW
8	USBHS_DFB	USBHS VBUS/ID 管脚内部去抖滤波器旁路使能位	● 0b0: 模块内部去抖动滤波器有效 ● 0b1: 旁路模块内部去抖动滤波器  说明 设备模式和主机模式均有效。	RW
7:2	Reserved	保留	-	-
1	USBFS_SOFEN	USBFS SOF 脉冲输出使能位	USBFS 主机发出 SOF 或者设备成功接收到 SOF 时, 16 个 HCLK 时钟周期宽度的 SOF 脉冲从 PAD 输出使能 ● 0b0: SOF 脉冲不输出 ● 0b1: SOF 脉冲输出  说明 设备模式和主机模式均有效。	RW
0	USBFS_DFB	USBFS VBUS/ID 管脚内部去抖滤波器旁路使能位	● 0b0: 模块内部去抖动滤波器有效 ● 0b1: 旁路模块内部去抖动滤波器  说明 设备模式和主机模式均有效。	RW

35.6.3 USBFS 全局寄存器

这些寄存器在主机模式和设备模式下都可用，且在这两个模式间切换时无需对其进行重新编程。

35.6.3.1 VBUS 控制寄存器 (USBFS_GVBUSCFG)

该寄存器可用于设定 VBUS 值从而忽略 VBUS 管脚的状态。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00XX0000																															
Name	Reserved																								RW	VBUSVAL	RW	VBUSOVEN	Reserved			
Access																									RW	VBUSVAL	RW	VBUSOVEN				

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	VBUSVAL	VBUS 值 (VBUS Value)	用于设定 USBFS 的 VBUS 值，当设定为 1，且 VBUSOVEN 设定 1 后对 USBFS 完成上电。  说明 仅设备模式有效。	RW
6	VBUSOVEN	VBUS Override 使能	用于将 VBUSVAL 设定的值反映到 USBFS CORE 的状态。仅当该位设置为 1，VBUSVAL 的值才有效。  说明 仅设备模式有效。	RW
5:0	Reserved	保留	-	-

35.6.3.2 AHB 控制寄存器 (USBFS_GAHBCFG)

该寄存器可用于在上电后或更改角色模式时对模块进行配置，主要包含 AHB 系统相关的配置参数。
应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程，请勿在初始编程后更改该寄存器。

Offset	Bit Position																																			
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																								PTXFELVL		TXFELVL		Reserved		DMAEN		HBSTLEN		GINTMSK	
Access																									RW		RW				RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8	PTXFELVL	周期性 TxFIFO 空门限 (Periodic TxFIFO empty level)	指示何时触发模块中断寄存器中的周期性 TxFIFO 空中断位 (USBFS_GINTSTS 中的 PTXFE 位)。 0b0: PTXFE (位于 USBFS_GINTSTS) 中断指示周期性 TxFIFO 为半空状态 0b1: PTXFE (位于 USBFS_GINTSTS) 中断指示周期性 TxFIFO 为全空状态  说明 仅可主机模式下访问。	RW
7	TXFELVL	设备 TxFIFO 空门限 (TxFIFO empty level)	在设备模式下，该位指示何时触发 IN 端点发送 FIFO 空中断 (USBFS_DIEPINTx 中的 TXFE)。 0b0: TXFE (位于 USBFS_DIEPINTx) 中断指示 IN 端点 TxFIFO 为半空状态 0b1: TXFE (位于 USBFS_DIEPINTx) 中断指示 IN 端点 TxFIFO 为全空状态  说明 仅设备模式有效。	RW
6	Reserved	保留	-	-
5	DMAEN	DMA 使能 (DMA enable)	0b0: 模块以从模式运行 0b1: 模块以 DMA 模式运行  说明 设备模式和主机模式均有效。	RW
4:1	HBSTLEN	批量长度/类型 (Burst length/type)	0b0000: 单次 0b0001: INCR 0b0011: INCR4 0b0101: INCR8 0b0111: INCR16 - 其他: 禁止设置  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	GINTMSK	全局中断屏蔽 (Global interrupt mask)	该位用于屏蔽全局中断或对全局中断取消屏蔽。中断状态寄存器由模块进行更新，与此位的设置无关。 ● 0b0：屏蔽应用程序触发的中断 ● 0b1：取消对应用程序触发的中断的屏蔽  说明 设备模式和主机模式均有效。	RW

35.6.3.3 USB 配置寄存器 (USBFS_GUSBCFG)

该寄存器可用于在上电或更改角色模式后对模块进行配置。其中包含与 USB 和 USB-PHY 相关的配置参数。应用程序必须在开始任何 AHB 或 USB 事务前对该寄存器进行编程。请勿在初始编程后更改该寄存器。

Offset	Bit Position																																			
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00001440																																			
Name	Reserved	FDMOD	FHMOD	Reserved																TRDT				Reserved				PHYSEL	Reserved				TOTAL			
Access		RW	RW																	RW								RW					RW			

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30	FDMOD	强制设备模式 (Force device mode)	向该位写入 1 时, 可将模块强制为设备模式, 忽略 USBFS_ID 输入引脚的状态。 0b0: 正常模式, 取决 USBFS_ID 管脚的输入状态 0b1: 强制设备模式 将强制位置 1 后, 应用程序必须等待至少 25ms 后更改方可生效。  说明 设备模式和主机模式均有效。	RW
29	FHMOD	强制主机模式 (Force host mode)	向该位写入 1 时, 可将模块强制为主机模式, 忽略 USBFS_ID 输入引脚的状态。 0b0: 正常模式, 取决 USBFS_ID 管脚的输入状态 0b1: 强制主机模式 将强制位置 1 后, 应用程序必须等待至少 25ms 后更改方可生效。  说明 设备模式和主机模式均有效。	RW
28:14	Reserved	保留	-	-
13:10	TRDT	USB 周转时间 (USB turnaround time)	以 PHY 时钟数为单位设置周转时间。要计算 TRDT 的值, 请使用如下公式: $TRDT = 4 \times \text{AHB 时钟} + 1 \text{ 个 PHY 时钟}$ 例如: - 如果 AHB 时钟频率=84MHz (PHY 时钟频率=48MHz), 则 TRDT 设置为 9。 - 如果 AHB 时钟频率=48MHz (PHY 时钟频率=48MHz), 则 TRDT 设置为 5。  说明 仅设备模式有效。	RW
9:7	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
6	PHYSEL	全速串行收发器选择 (Full Speed serial transceiver select)	读出恒为 1，表示选择片上全速串行收发器。	RO
5:3	Reserved	保留	-	-
2:0	TOCAL	FS 超时校准 (FS timeout calibration)	PHY 引入的额外延迟包括应用程序在该字段中设置的 PHY 时钟数，以及模块的全速数据包间超时间隔。不同 PHY 引入的延迟对数据线状态的影响是不同的。全速操作的 USB 标准超时值为 16 到 18（含）个位时间。应用程序必须根据枚举速度编程该字段。每个 PHY 时钟增加的位时间数为 0.25 个位时间。  说明 设备模式和主机模式均有效。	RW

35.6.3.4 复位寄存器 (USBFS_GRSTCTL)

应用程序通过此寄存器复位模块中的各项硬件特性。

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80000000																															
Name	AHBIDL	DMAREQ	Reserved																			TXFNUM			TXFFLSH	RXFFLSH	Reserved	FCRST	HSRST	CSRST		
Access	RO	RO																				RW			RW	RW		RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31	AHBIDL	AHB 主器件空闲 (AHB master idle)	指示 AHB 主器件状态机处于空闲情况。  说明 设备模式和主机模式均有效。	RO
30	DMAREQ	DMA 请求信号 (DMA request signal)	该位指示 DMA 请求正在进行中。用于调试。  说明 设备模式和主机模式均有效。	RO
29:11	Reserved	保留	-	-
10:6	TXFNUM	TxFIFO 编号 (TxFIFO number)	使用 TxFIFO 刷新位进行 FIFO 刷新的 FIFO 编号。只有在模块将 TxFIFO 刷新位清零后，方可更改此字段。 0b00000：主机模式下刷新非周期性 TxFIFO、设备模式下刷新 TxFIFO0 0b00001：主机模式下刷新周期性 TxFIFO、设备模式下刷新 TxFIFO1 0b00010：设备模式下刷新 TxFIFO2 0b00011：设备模式下刷新 TxFIFO3 0b00101：设备模式下刷新 TxFIFO15 0b10000：在设备模式或主机模式下刷新所有的发送 FIFO - 其他：禁止设置  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
5	TXFFLSH	TxFIFO 刷新 (TxFIFO flush)	此位选择性地刷新一个或所有的发送 FIFO，但当模块处理通信事务时无法执行该操作。 只有在确认模块当前未对 TxFIFO 执行读写操作后，应用程序方可对此位执行写操作。使用以下寄存器进行确认： ● 读：NAK 有效中断可确保模块当前未对 FIFO 执行读操作 ● 写：USBFS_GRSTCTL 中的 AHBIDL 位可确保模块当前未对 FIFO 执行任何写操作  说明 设备模式和主机模式均有效。	RW
4	RXFFLSH	RxFIFO 刷新 (RxFIFO flush)	应用程序可使用此位刷新整个 RxFIFO，但必须首先确保模块当前未在处理通信事务。只有在确认模块当前未对 RxFIFO 执行读写操作后，应用程序方可对此位执行写操作。 应用程序必须等到此位清零后，方可执行其它操作。通常需要等待 8 个时钟周期（以 PHY 或 AHB 时钟中最慢的为准）。  说明 设备模式和主机模式均有效。	RW
3	Reserved	保留	-	-
2	FCRST	主机帧计数器复位 (Host frame counter reset)	应用程序对该位执行写操作时，模块中的帧数计数器复位。帧计数器复位后，由模块发送的下一个 SOF 的帧号为 0。  说明 设备模式和主机模式均有效。	RW
1	HSRST	HCLK 域逻辑软复位 (HCLK soft reset)	应用程序使用此位来刷新 AHB 时钟域中的控制逻辑。仅复位 AHB 时钟域流水线。 FIFO 不通过此位来刷新。 遵照协议终止 AHB 上的事务后，AHB 时钟域中的所有状态机均复位至空闲状态。 AHB 时钟域状态机所使用的 CSR 控制位清零。 要清零该中断，需要将由 AHB 时钟域状态机生成并用于控制中断状态的状态屏蔽位清零。 由于中断状态位并未清零，因此应用程序可以获取在该位置 1 后所发生的所有模块事件的状态。 此位为自清零位，模块将在其中所有必要逻辑复位后将该位清零。该过程需要若干个时钟的时间，具体取决于模块的当前状态。  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	CSRST	模块软复位 (Core soft reset)	按如下所述将 HCLK 和 PCLK 域复位： 除以下各位外，将各个中断和所有 CSR 寄存器位清零： ● USBFS_GCCTL 中的 GATEHCLK 位 ● USBFS_GCCTL 中的 STPPCLK 位 ● USBFS_HCFG 中的 FSLSPCS 位 ● USBFS_DCFG 中的 DSPD 位 将所有模块状态机（AHB 从器件除外）复位至空闲状态，并清空所有发送 FIFO 和接收 FIFO。 在 AHB 传输的最后数据阶段结束后，尽快终止 AHB 主器件上的所有事务。立即终止 USB 上的所有事务。 应用程序可在需要复位模块时随时对该位执行写操作。该位为自清零位，模块将在其中所有必要逻辑复位后将该位清零，该过程需要若干个时钟的时间，具体取决于模块的当前状态。该位一旦清零，软件必须等待至少 3 个 PHY 时钟后才可以访问 PHY 域（同步延迟）。此外，软件还必须在确定该寄存器中的位 31 置 1（AHB 主器件空闲）后方可开始运行。 软件复位通常在两种情况下使用，一是软件开发期间，二是用户动态更改以上所列 USB 配置寄存器中的 PHY 选择位后。用户更改 PHY 时，将为 PHY 选择相应的时钟并用于 PHY 域中。一旦选择了新的时钟，则必须复位 PHY 域，才能保证正常运行。  说明 设备模式和主机模式均有效。	RW

35.6.3.5 全局中断状态寄存器 (USBFS_GINTSTS)

该寄存器用于在当前模式（设备模式或主机模式）下借助系统级别的事件来中断应用程序。该寄存器可指示当前模式。

FIFO 状态中断为只读；如果软件在处理这些中断期间对 FIFO 执行读写操作，则 FIFO 中断标志将自动清零。使能中断位前，应用程序必须在初始化时将 USBFS_GINTSTS 寄存器清零，才可以避免在初始化前产生任何中断。

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x14000020																															
Name	WKUINT	VBUSVINT	DISCINT	CIDSCHG	LPMINT	PTXFE	HCINT	HPRTINT	Reserved	DATAFSUSP	IPXFR/INCOMPI	ISOIXFR	OEPI	IEPI	Reserved	EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Reserved	GONAKEFF	GINAKEFF	NPTXFE	RXFNE	SOF	Reserved	MMIS	CMOD		
Access	RW	RW	RW	RW	RW	RO	RO	RO		RW	RW	RW	RO	RO		RW	RW	RW	RW	RW	RW		RO	RO	RO	RO	RW		RW	RO	RO	

位/位域	标记	位名	功能描述	读写
31	WKUINT	检测到恢复/远程唤醒中断 (Resume/remote wakeup detected interrupt)	在设备模式下，当 USB 总线上检测到恢复信号时，将触发该中断。 在主机模式下，当 USB 上检测到远程唤醒时，将触发该中断。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
30	VBUSVINT	VBUS 有效中断 (VBUS valid interrupt)	设备模式下，当检测到 USBFS_VBUS 管脚由低变高时，将触发该中断。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
29	DISCINT	检测到断开连接中断 (Disconnect detected interrupt)	当检测到设备断开连接时触发该中断。 通过软件对该位写 1 清零。  说明 仅主机模式有效。	RW
28	CIDSCHG	连接器 ID 线状态更改 (Connector ID status change)	当连接器 ID 线状态发生更改时，模块将该位置 1。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
27	LPMINT	LPM 中断	● 设备模式：设备收到正确 LPM 传输且有效应答时该位置 1。 ● 主机模式：主机发送 LPM 后收到有效应答，或者超过设定的重试次数后该位置 1。	RW

位/位域	标记	位名	功能描述	读写
26	PTXFE	周期性 TxFIFO 空中断(Periodic TxFIFO empty interrupt)	当周期性发送 FIFO 为半空或全空状态，且周期性请求队列中存在可写入至少一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 USBFS_GAHBCFG 寄存器中的周期性 TxFIFO 空级别位 (USBFS_GAHBCFG 中的 PTXFELVL 位) 决定。  说明 仅主机模式有效。	RO
25	HCINT	主机通道中断 (Host channels interrupt)	模块将该位置 1 时，指示模块中一个通道上存在挂起的中断（在主机模式下）。应用程序必须读取主机 USBFS_HAINT 寄存器，以确定发生中断的通道的准确编号，然后读取相应的 USBFS_HCINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将 USBFS_HCINTx 寄存器的相应状态位清零，之后才能将该位清零。  说明 仅主机模式有效。	RO
24	HPRTINT	主机端口中断 (Host port interrupt)	模块将该位置 1 时，指示主机模式下 USBFS 控制器端口的状态发生变化。应用程序必须读取 USBFS_HPRT 寄存器，以确定引发此中断的确切事件。应用程序必须先将 USBFS_HPRT 寄存器的相应状态位清零，之后才能将该位清零。  说明 仅主机模式有效。	RO
23	Reserved	保留	-	-
22	DATAFSUSP	数据获取挂起 (Data fetch suspended)	该中断仅在 DMA 模式下有效。该中断指示，模块因 TxFIFO 空间或请求队列空间不可用而停止为 IN 端点获取数据。应用程序将该中断用于端点不匹配算法中。例如，在检测到端点不匹配后，应用程序将执行以下操作： ● 将全局非周期性 IN NAK 握手信号置 1 ● 禁止 IN 端点 ● 清空 FIFO ● 根据 IN 令牌序列学习队列确定令牌序列 ● 重新使能端点 ● 清零全局非周期性 IN NAK 握手信号 如果全局非周期性 IN NAK 已清零但模块尚未从 IN 端点获取数据，同时又接收到 IN 令牌，模块将产生“FIFO 为空时接收到 IN 令牌”中断。然后 USBFS 将 NAK 响应发送到主机。为避免上述情况，应用程序可以检查 USBFS_GINTSTS 中的 DATAFSUSP 中断，该中断可确保在 FIFO 存满后再将全局 NAK 握手信号清零。另外，应用程序可以在将全局 IN NAK 握手信号清零时屏蔽“当 FIFO 为空时接收到 IN 令牌中断”。通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
21	IPXFR/ INCOMPISOO UT	IPXFR: 未完成 周期性传输 (Incomplete periodic transfer) INCOMPISOO OUT: 未完成 OUT 同步传输 (Incomplete isochronous OUT transfer)	● IPXFR: 在主机模式下, 如果存在仍处于挂起状态的未完成周期性事务, 而这些事务计划在当前帧期间完成, 则模块会将该中断位置 1。 通过软件对该位写 1 清零。  说明 仅主机模式有效。 ● INCOMPISOOOUT: 在设备模式下, 模块将该中断置 1 时, 指示当前帧中至少有一个同步 OUT 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
20	IISOIXFR	未完成 IN 同步 传输 (Incomplete isochronous IN transfer)	模块将该中断置 1 时, 指示当前帧中至少有一个同步 IN 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断 (EOPF) 位一同触发。 通过软件对该位写 1 清零。	RW
19	OEPINT	OUT 端点中断 (OUT endpoint interrupt)	模块将该位置 1 时, 指示模块中一个 OUT 端点上存在挂起的中断 (在设备模式下)。应用程序必须读取主机 USBFS_DAIN_T 寄存器, 以确定发生中断的 OUT 端点的准确编号, 然后读取相应的 USBFS_DOEPINTx 寄存器, 以确定引发中断的确切原因。应用程序必须先将相应 USBFS_DOEPINTx 寄存器的相应状态位清零, 之后才能将该位清零。	RO
18	IEPINT	IN 端点中断(IN endpoint interrupt)	模块将该位置 1 时, 指示模块中一个 IN 端点上存在挂起的中断 (在设备模式下)。应用程序必须读取主机 USBFS_DAIN_T 寄存器, 以确定发生中断的 IN 端点的准确编号, 然后读取相应的 USBFS_DIEPINTx 寄存器, 以确定引发中断的确切原因。应用程序必须先将相应 USBFS_DIEPINTx 寄存器的相应状态位清零, 之后才能将该位清零。	RO
17:16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	EOPF	周期性帧结束中断(End of periodic frame interrupt)	指示当前帧已达到 USBFS_DCFG 寄存器中周期性帧间隔字段 (USBFS_DCFG 中的 PFIVL 位) 所指定的周期。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
14	ISOODRP	丢弃同步 OUT 数据包中断 (Isochronous OUT packet dropped interrupt)	如果由于 Rx FIFO 空间不足, 无法容纳同步 OUT 端点的最大数据包, 从而导致模块无法向 Rx FIFO 写入同步 OUT 数据包, 模块会将该位置 1。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
13	ENUMDNE	枚举完成中断 (Enumeration done interrupt)	模块将该位置 1 时, 指示速度枚举已完成。应用程序必须读取 USBFS_DSTS 寄存器来获取枚举速度。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
12	USBRST	USB 复位中断 (USB reset interrupt)	模块将该位置 1 时, 指示在 USB 上检测到复位信号。 通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
11	USBSUSP	USB 挂起中断 (USB suspend interrupt)	模块将该位置 1 时, 指示在 USB 上检测到挂起状态。当 USB 总线上的空闲状态保持 3ms, 模块便会进入挂起状态。通过软件对该位写 1 清零。  说明 仅设备模式有效。	RW
10	ESUSP	早期挂起中断 (Early suspend interrupt)	模块将该位置 1 时, 指示已检测到 USB 处于空闲状态的时间达到 3ms。  说明 仅设备模式有效。	RW
9:8	Reserved	保留	-	-
7	GONAKEFF	全局 OUT NAK 有效中断(Global OUT NAK effective interrupt)	指示 USBFS_DCTL 寄存器中由应用程序设置的“将全局 OUT NAK 置 1”位 (USBFS_DCTL 中的 SGONAK 位) 已在模块中生效。通过写入 USBFS_DCTL 寄存器中的“将全局 OUT NAK 清零”位 (USBFS_DCTL 中的 CGONAK 位), 可将该位清零。  说明 仅设备模式有效。	RO

位/位域	标记	位名	功能描述	读写
6	GINAKEFF	全局非周期性 IN NAK 有效中断 (Global IN nonperiodic NAK effective interrupt)	指示 USBFS_DCTL 寄存器中由应用程序设置的“将全局非周期性 IN NAK 置 1”位 (USBFS_DCTL 中的 SGINAK 位) 已在模块中生效。也就是说, 模块已对应用程序设置的全局 IN NAK 位进行采样, 结果已生效。通过清零 USBFS_DCTL 寄存器中的“将全局非周期性 IN NAK 清零”位 (USBFS_DCTL 中的 CGINAK 位), 可将该位清零。此中断不一定表示 USB 上已发送了一个 NAK 握手信号。STALL 位优先级高于 NAK 位。  说明 仅设备模式有效。	RO
5	NPTXFE	非周期性 Tx FIFO 空中断 (Non-periodic Tx FIFO empty interrupt)	当非周期性 Tx FIFO 为半空或全空状态, 且非周期性发送请求队列中至少存在可写入一个条目的空间时, 将触发该中断。该 FIFO 为半空状态还是全空状态由 USBFS_GAHBCFG 寄存器中的非周期性 Tx FIFO 空级别位 (USBFS_GAHBCFG 中的 TXFELVL 位) 决定。  说明 仅主机模式有效。	RO
4	RXFNE	Rx FIFO 非空中断 (Rx FIFO non-empty interrupt)	指示 Rx FIFO 中至少有一个数据包等待读取。  说明 设备模式和主机模式均有效。	RO
3	SOF	帧起始中断 (Start of frame interrupt)	在主机模式下, 模块将该位置 1 时, 指示 USB 上已发送一个 SOF (FS) 或 Keep-Alive (LS) 信号。应用程序必须将此位置 1 才可清除该中断。 在设备模式下, 模块将该位置 1 时, 指示 USB 上已接收到一个 SOF 令牌。应用程序可通过读取设备状态寄存器来获得当前的帧编号。只有在模块以 FS 模式运行时, 才会出现此中断。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW
2	Reserved	保留	-	-
1	MMIS	模式不匹配中断 (Mode mismatch interrupt)	当应用程序尝试访问以下寄存器时, 模块将该位置 1: ● 模块运行在设备模式下访问主机模式寄存器 ● 模块运行在主机模式下访问设备模式寄存器 寄存器访问在 AHB 上以 OKAY 响应结束, 但该访问在内部被模块忽略并且不会影响模块运行。 通过软件对该位写 1 清零。  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
0	CMOD	当前工作模式 (Current mode of operation)	指示当前模式。 ● 0b0：设备模式 ● 0b1：主机模式 说明 设备模式和主机模式均有效。	RO

35.6.3.6 全局中断屏蔽寄存器 (USBFS_GINTMSK)

该寄存器与全局中断状态寄存器 (USBFS_GINTSTS) 结合使用，以中断应用程序。如果将某个中断位屏蔽，则不会产生与该位相关的中断，但是与该中断相对应的寄存器位仍会置 1。

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	WKUIM	VBUSVIM	DISCIM	CIDSCHGM	LPMINTM	PTXFEM	HCIM	HPRTIM	Reserved	DATAFSUSPM	IPXFRM/INCOMPIISOOUTM	IISOXFRM	OEPIM	IEPIM	Reserved	EOPFM	ISOODRPM	ENUMDNEM	USBRSTM	USBSUSPM	ESUSPM	Reserved	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFNEM	SOFM	Reserved	MMISM	Reserved		
Access	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW			RW		

位/位域	标记	位名	功能描述	读写
31	WКУIM	检测到恢复/远程唤醒中断屏蔽 (Resume/remote wakeup detected interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
30	VBUSVIM	VBUS 有效中断屏蔽 (VBUS valid interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 仅设备模式有效。	RW
29	DISCIM	检测到断开连接中断屏蔽 (Disconnect detected interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 仅主机模式有效。	RW
28	CIDSCHGM	连接器 ID 线状态更改中断屏蔽 (Connector ID status change interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
27	LPMINTM	LPM 中断屏蔽 (LPM interrupt mask)	0b0: 屏蔽中断 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW

位/位域	标记	位名	功能描述	读写
26	PTXFEM	周期性 TxFIFO 空中断屏蔽 (Periodic TxFIFO empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
25	HCIM	主机通道中断屏蔽 (Host channels interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
24	HPRTIM	主机端口中断屏蔽 (Host port interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
23	Reserved	保留	-	-
22	DATAFSUSPM	数据获取挂起中断屏蔽 (Data fetch suspended interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
21	IPXFRM/ INCOMPISOO UTM	IPXFRM: 未完成周期性传输中断屏蔽 (Incomplete periodic transfer interrupt mask) INCOMPISOOOUTM: 未完成 OUT 同步传输中断屏蔽 (Incomplete isochronous OUT transfer interrupt mask)	● IPXFRM: ▶ 0b0: 屏蔽中断 ▶ 0b1: 使能中断  说明 仅主机模式有效。 ● INCOMPISOOOUTM: ▶ 0b0: 屏蔽中断 ▶ 0b1: 使能中断  说明 仅设备模式有效。	RW
20	IISOIXFRM	未完成 IN 同步传输中断屏蔽 (Incomplete isochronous IN transfer interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
19	OEPIIM	OUT 端点中断屏蔽 (OUT endpoint interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
18	IEPIM	IN 端点中断屏蔽 (IN endpoint interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
17:16	Reserved	保留	-	-
15	EOPFM	周期性帧结束中断屏蔽(End of periodic frame interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
14	ISOODRPM	丢弃同步 OUT 数据包中断屏蔽(Isochronous OUT packet dropped interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
13	ENUMDNEM	枚举完成中断屏蔽(Enumeration done interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
12	USBRSTM	USB 复位中断屏蔽(USB reset interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
11	USBSUSPM	USB 挂起中断屏蔽(USB suspend interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
10	ESUSPM	早期挂起中断屏蔽(Early suspend interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
9:8	Reserved	保留	-	-
7	GONAKEFFM	全局 OUT NAK 有效中断屏蔽(Global OUT NAK effective interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW

位/位域	标记	位名	功能描述	读写
6	GINAKEFFM	全局非周期性 IN NAK 有效中断屏蔽(Global IN nonperiodic NAK effective interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅设备模式有效。	RW
5	NPTXFEM	非周期性 TxFIFO 空中断屏蔽(Non-periodic TxFIFO empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 仅主机模式有效。	RW
4	RXFNEM	RxFIFO 非空中断屏蔽(RxFIFO non-empty interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
3	SOFM	帧起始中断屏蔽(Start of frame interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
2	Reserved	保留	-	-
1	MMISM	模式不匹配中断屏蔽(Mode mismatch interrupt mask)	● 0b0: 屏蔽中断 ● 0b1: 使能中断  说明 设备模式和主机模式均有效。	RW
0	Reserved	保留	-	-

35.6.3.7 接收状态调试读取/状态读取和出栈寄存器 (USBFS_GRXSTSR/USBFS_GRXSTSP)

读取接收状态调试读取寄存器将返回接收 FIFO 顶部的内容。读取接收状态读取和出栈寄存器将额外弹出 RxFIFO 顶部的数据条目。接收状态内容在主机模式和设备模式下的解释不同。

当接收 FIFO 为空时，模块会忽略对该寄存器的读取或出栈操作，并返回值 0x0000 0000。当模块中断寄存器的接收 FIFO 非空位 (USBFS_GINTSTS 中的 RXFNE 位) 置位时，应用程序必须仅弹出接收状态 FIFO。

读取的偏移地址 (Offset) : 0x1C

出栈的偏移地址 (Offset) : 0x20

主机模式:

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												RO PKTSTS		RO DPID		RO BCNT										RO CHNUM_EPNUM					
Access													RO		RO		RO										RO					

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:17	PKTSTS	数据包状态 (Packet status)	指示接收的数据包的状态 0b0010: 接收到 IN 数据包 0b0011: IN 传输完成 (触发中断) 0b0101: 数据同步错误 (触发中断) 0b0111: 暂停通道 (触发中断) - 其他: Reserved	RO
16:15	DPID	数据 PID (Data PID)	指示接收的数据包的数据 PID 0b00: DATA0 0b01: DATA2 0b10: DATA1 0b11: MDATA	RO
14:4	BCNT	字节计数(Byte count)	指示接收的 IN 数据包的字节数	RO
3:0	CHNUM_EPNUM	通道编号 (Channel number)	指示当前接收的数据包所属的通道编号	RO

设备模式：

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												RO PKTSTS				RO DPID		RO BCNT								RO CHNUM_EPNUM					
Access													RO				RO		RO								RO					

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:17	PKTSTS	数据包状态 (Packet status)	指示接收的数据包的状态 0b0001: 全局 OUT NAK (触发中断) 0b0010: 接收到 OUT 数据包 0b0011: OUT 传输完成 (触发中断) 0b0100: SETUP 事务完成 (触发中断) 0b0110: 接收到 SETUP 数据包 - 其他: Reserved	RO
16:15	DPID	数据 PID (Data PID)	指示接收的 OUT 数据包的数据 PID 0b00: DATA0 0b01: DATA2 0b10: DATA1 0b11: MDATA	RO
14:4	BCNT	字节计数(Byte count)	指示接收的数据包的字节数	RO
3:0	CHNUM_EPNUM	通道编号 (Channel number)	指示当前接收的数据包所属的端点编号	RO

35.6.3.8 接收 FIFO 大小寄存器 (USBFS_GRXFSIZ)

此应用程序可以对必须分配给 RxFIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000280																															
Name	Reserved																				RXFD											
Access																					RW											

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10:0	RXFD	RxFIFO 深度 (RxFIFO depth)	以 32 位字为单位。取值范围是 16~256 上电复位值为最大 Rx 数据 FIFO 深度	RW

35.6.3.9 主机非周期性发送 FIFO 大小寄存器 (USBFS_HNPTXFSIZ) / 端点 0 发送 FIFO 大小 (USBFS_DIEPTXF0)

此应用程序可以对必须分配给 TxFIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x02800280																															
Name	NPTXFD/TX0FD																NPTXFS/TX0FSA															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	NPTXFD/ TX0FD	NPTXFD: 非周期性 TxFIFO 深度 TX0FD: 端点 0 TxFIFO 深度	- 主机模式: NPTXFD, 以 32 位字为单位。取值范围是 16~256 - 设备模式: TX0FD, 以 32 位字为单位。取值范围是 16~256	RW
15:0	NPTXFS/TX0FSA	NPTXFS: 非周期性发送 RAM 起始地址 TX0FSA: 端点 0 发送 RAM 起始地址	- 主机模式: NPTXFS, 此字段包含非周期性发送 FIFO RAM 的存储器起始地址 - 设备模式: TX0FSA, 此字段包含端点 0 发送 FIFO RAM 的存储器起始地址	RW

35.6.3.10 非周期性发送 FIFO/队列状态寄存器 (USBFS_HNPTXSTS)

此只读寄存器包含非周期性 TxFIFO 和非周期性发送请求队列的自由空间信息。
此寄存器仅在主机模式有效，设备模式无效。

Offset	Bit Position																																
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00080280																																
Name	Reserved	NPTXQTOP								NPTQXSAV								NPTXFSAV															
Access		RO								RO								RO															

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:24	NPTXQTOP	非周期性发送请求队列顶部(Top of the non-periodic transmit request queue)	非周期性发送请求队列中 MAC 目前正在处理的条目。 [30:27]: 通道/端点编号 [26:25]: 0b00: IN/OUT 令牌 0b01: 长度为零的发送数据包 0b10: Reserved 0b11: 通道停止命令 [24]: 结束 (所选通道/端点的最后一个条目)	RO
23:16	NPTQXSAV	非周期性发送请求队列可用空间 (Non-periodic transmit request queue space available)	指示非周期性发送请求队列中的可用空闲空间大小，取值范围是 0~8。 在主机模式下，此队列保存 IN 和 OUT 请求。 0: 非周期性发送请求队列已满 1: 1 个位置可用 2: 2 个位置可用 8: 8 个位置可用	RO
15:0	NPTXFSAV	非周期性 TxFIFO 可用空间(Non-periodic TxFIFO space available)	指示非周期性 TxFIFO 中的可用空闲空间大小，以 32 位字为单位，取值范围是 0~256。 0: 非周期性 TxFIFO 已满 1: 1 个字可用 2: 2 个字可用 256: 256 个字可用	RO

35.6.3.11 模块 ID 寄存器 (USBFS_CID)

该寄存器为可编程用户配置 ID 寄存器。

Offset	Bit Position																															
0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x12345678																															
Name	PRODUCT_ID																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	PRODUCT_ID	产品 ID 字段 (Product ID field)	可通过应用程序编程的 ID 字段。	RW

35.6.3.12 LPM 配置寄存器 (USBFS_GLPMCFG)

该寄存器为可编程 LPM 配置寄存器。

Offset	Bit Position																																								
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
Reset	0x00000000																																								
Name	Reserved	ENBSEL		LPMRCNTSTS				SENDLPM		LPMRCNT				LPMCHIDX				L1RSMOK		SLPSTS		LPMRSP		L1DSEN		BSELTHRS				L1SSEN		REMWAKE		BSEL				LPMACK		LPMEN	
Access		RW		RO				RW		RW				RW				RO		RO		RO		RW		RW				RW		RW		RW				RW		RW	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28	ENBSEL	BSEL 使能位	0b0: 遵循 USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007 0b1: 遵循 Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007	RW
27:25	LPMRCNTSTS	LPM 重试状态	记录 LPM 剩余的重试次数  说明 仅主机模式有效。	RO
24	SENDLPM	发送 LPM 传输	0b0: 不发送 LPM 传输 0b1: 发送 LPM 传输 (EXT 和 LPM 令牌) 当收到设备返回的应答 (ACK, STALL 或 NYET) 后该位自动清零。  说明 仅主机模式有效。	RW
23:21	LPMRCNT	LPM 重试次数	当收到 ERROR 应答时, 主机发送 LPM 传输的重试次数, 直到收到有效的应答 (ACK, STALL 或 NYET)  说明 仅主机模式有效。	RW
20:17	LPMCHIDX	LPM 发送通道索引	LPM 传输所用主机通道索引, 硬件自动插入该通道设定的设备地址以及端点信息。  说明 仅主机模式有效。	RW
16	L1RSMOK	L1 状态位	0b0: 当前状态不能从 L1 恢复 0b1: 当前状态可以从 L1 恢复	RO
15	SLPSTS	sleep 状态位	设备模式: 当设备发送 ACK 响应 LPM 传输并经过协议规定的时间后, 该位置 1, 表示进入 SLEEP 模式。 当总线状态变化, 或者断连, 或者发送远程唤醒信号时, 该位自动清 0, 退出 SLEEP 模式。 主机模式:	RO

位/位域	标记	位名	功能描述	读写
15	SLPSTS	sleep 状态位	主机发送 LPM 传输，接收到设备响应的 ACK 应答后，该位置 1，表示进入 SLEEP 模式。 当主机接收到远程唤醒信号，或者主机发起唤醒，或者主机发起复位时，该位自动清 0，退出 SLEEP 模式。 ● 0b0: Not in L1 ● 0b1: In L1	RO
14:13	LPMRSP	LPM 应答	主机接收或者从机发送的 LPM 应答 ● 0b00: ERROR(无应答) ● 0b01: STALL ● 0b10: NYET ● 0b11: ACK	RO
12	L1DSEN	L1 deep sleep 使能位	最大限度的降低功耗，应用程序需要将此位设定为 1	RW
11:8	BSELTHRS	BSEL 阈值设定	设备模式：当设备接收到 BSEL 值大于等于该寄存器设定时，进入 L1。 主机模式：主机发送 resume 信号的时间。	RW
7	L1SSEN	L1 shallow sleep 使能位	最大限度的降低功耗，应用程序需要将此位设定为 1	RW
6	REMWAKE	bRemoteWake 值	主机发送或者设备接收到的 bRemoteWake 值	RW
5:2	BSEL	BSEL 设置	设备模式：BSEL bmAttribute 值，有效应答 LPM 传输后自动更新。 主机模式：主机发送 LPM 传输的 BSEL 值，或者 resume 信号长度值。 ● 0b0000: 125us ● 0b0001: 150us ● 0b0010: 200us ● 0b0100: 400us ● 0b0101: 500us ● 0b0110: 1000us ● 0b0111: 2000us ● 0b1000: 3000us ● 0b1001: 4000us ● 0b1010: 5000us ● 0b1011: 6000us ● 0b1100: 7000us ● 0b1101: 8000us ● 0b1110: 9000us ● 0b1111: 10000us ● 其他: 禁止设置	RW
1	LPMACK	LPM 应答	● 0b0: NYET ● 0b1: ACK 即使该位写 1，但在以下场景（LPM 传输有错误）中不会发送 ACK 应答： ● 存在 PID/CRC 错误，该场景发送 ERROR 应答 ● bLinkState 不等于 0001b，该场景发送 STALL 应答	RW

位/位域	标记	位名	功能描述	读写
1	LPMACK	LPM 应答	● 存在挂起的数据传输，该场景发送 NYET 应答  说明 仅设备模式有效。	RW
0	LPMEN	LPM 使能位	● 0b0：禁止 ● 0b1：使能	RW

35.6.3.13 主机周期性发送 FIFO 大小寄存器 (USBFS_HPTXFSIZ)

此应用程序可以对必须分配给周期 Tx FIFO 的 RAM 大小进行编程。

Offset	Bit Position																															
0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x02800500																															
Name	PTXFD													PTXSA																		
Access	RW													RW																		

位/位域	标记	位名	功能描述	读写
31:16	PTXFD	主机周期性 Tx FIFO 深度	以 32 位字为单位，最小值为 16	RW
15:0	PTXSA	主机周期性 Tx FIFO 起始地址	上电复位值是最大 Rx FIFO 深度与最大非周期性 Tx FIFO 深度之和	RW

35.6.3.14 设备 IN 端点发送 FIFO 大小寄存器 (USBFS_DIEPTXFx) (x=1~15)

此应用程序可以对必须分配给设备 Tx FIFO 的大小进行编程。

偏移地址 (Offset) : 0x100+0x4*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x02800500+0x280*(x-1)																															
Name	INEPTXFD																INEPTXSA															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	INEPTXFD	设备 IN 端点 Tx FIFO 深度	以 32 位字为单位，最小值为 16	RW
15:0	INEPTXSA	IN 端点 Tx FIFOx RAM 起始地址	此字段包含 IN 端点发送 FIFOx 的存储器起始地址。该地址必须与 32 位存储器位置对齐。	RW

35.6.4 USBFS 主机模式寄存器

主机模式寄存器会影响主机模式下的模块操作。在设备模式下不得访问主机模式寄存器，因为产生的结果不明确。

35.6.4.1 主机配置寄存器 (USBFS_HCFG)

此寄存器将在上电后对模块进行配置，请勿在初始化主机后更改此寄存器。

Offset	Bit Position																															
0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000200																															
Name	Reserved																												FSLSS			
Access																													RW	RW		

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	FSLSS	仅支持 FS 和 LS 功能控制位	应用程序使用此位控制模块的枚举速度。使用此位，应用程序可使模块工作为 FS 主机，即使所连接的设备支持 HS 通信也是如此。请勿在初始编程后更改此字段。 0b0：禁止设置 0b1：仅限 FS/LS，即使所连接设备可支持 HS	RW
1:0	FSLSPCS	FS/LS PHY 时钟选择(FS/LS PHY clock select)	当模块处于 FS 主机模式时： 0b01：PHY 时钟以 48MHz 运行 - 其他：禁止设置 当模块处于 LS 主机模式时： 0b01：PHY 时钟以 48MHz 运行 0b10：PHY 时钟以 6MHz 运行 - 其他：禁止设置  注意 当设备连上主机时，必须依照所连接设备的速度设置 FSLSPCS（更改此位后，必须进行软件复位）。	RW

35.6.4.2 主机帧时间间隔寄存器 (USBFS_HFIR)

此寄存器用于存储 USBFS 控制器对已连接设备当前速度所设定的帧间隔信息。

Offset	Bit Position																															
0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000EA60																															
Name	Reserved																FRIVL															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	FRIVL	帧间隔 (Frame interval)	应用程序在此字段编程的值用于指定两个连续 SOF(FS)或 Keep-Alive 令牌(LS)之间的时间间隔。此字段包含构成所需帧间隔的 PHY 时钟数。只有将主机端口控制和状态寄存器的端口使能位 (USBFS_HPRT 的 PENA 位) 置 1 后, 应用程序才能向此寄存器中写入值。如果未对值进行编程, 模块将根据在主机配置寄存器的 FS/LS PHY 时钟选择字段 (USBFS_HCFG 中的 FSLSPCS) 中指定的 PHY 时钟来计算。请勿在初始配置后更改此字段的值。 设定值=帧间隔 (ms) × (PHY 时钟频率) -1  注意 当应用程序需要更改帧间隔时间时, 可对 FRIVL 位进行修改。	RW

35.6.4.3 主机帧编号/帧剩余时间寄存器 (USBFS_HFNUM)

Offset	Bit Position																															
0x408	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00003FFF																															
Name	FTREM																FRNUM															
Access	RO																RO															

位/位域	标记	位名	功能描述	读写
31:16	FTREM	帧剩余时间 (Frame time remaining)	指示当前帧的剩余时间 (以 PHY 时钟数为单位)。每过去 1 个 PHY 时钟, 此字段递减 1。当值达到零时, 此字段将重新装载帧间隔寄存器中的值, 并由模块在 USB 上发送一个新 SOF。	RO
15:0	FRNUM	帧编号 (Frame number)	当在 USB 上发送 1 个新 SOF 时此字段的值将递增 1, 当达到 0x3FFF 时会清零。	RO

35.6.4.4 主机周期性发送 FIFO/队列状态寄存器 (USBFS_HPTXSTS)

此只读寄存器包含周期性 TxFIFO 和周期性发送请求队列的空闲空间信息。

Offset	Bit Position																															
0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00080280																															
Name	PTXQTOP								PTXQSAV								PTXFSAVL															
Access	RO								RO								RO															

位/位域	标记	位名	功能描述	读写
31:24	PTXQTOP	周期性发送请求队列顶部(Top of the periodic transmit request queue)	指示周期性 Tx 请求队列中 MAC 当前正在处理的项。该寄存器用于调试。 [31]: 奇数/偶数帧 ● 0b0: 以偶数帧发送 ● 0b1: 以奇数帧发送 [30:27]: 通道/端点编号 [26:25]: 类型 ● 0b00: 输入/输出 ● 0b01: 零长度数据包 ● 0b10: Reserved ● 0b11: 禁止通道命令 [24]: 结束 (所选通道的最后一个条目)	RO
23:16	PTXQSAV	周期性发送请求队列可用空间 (Periodic transmit request queue space available)	指示可供写入的周期性发送请求队列的空闲位置的数量, 取值范围是 0~8。该队列既包含 IN 请求, 又包含 OUT 请求。 ● 0: 周期性发送请求队列已满 ● 1: 1 个位置可用 ● ● 8: 8 个位置可用	RO
15:0	PTXFSAVL	周期性发送数据 FIFO 可用空间 (Periodic transmit data FIFO space available)	指示可供写入的周期性 TxFIFO 的空闲位置的数量。以 32 位字为单位。 ● 0: 周期性 TxFIFO 已满 ● 1: 1 个字可用 ● ● n: n 个字可用 (n 取值为寄存器 USBFS_HPTXFSIZ.PTXFD 的值)	RO

35.6.4.5 主机全体通道中断寄存器 (USBFS_HAINT)

当通道上有事件发生时，主机全体通道中断寄存器会使用模块中断寄存器中的主机通道中断位（USBFS_GINTSTS 中的 HCINT 位）中断应用程序。每个通道对应 1 个中断位，最多有 16 个位。
当应用程序通过相应主机通道 x 中断寄存器清零中断时，该寄存器中的位也会清零。

Offset	Bit Position																															
0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																HAINT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	HAINT	通道中断 (Channel interrupt)	对应通道 15~通道 0： ● 0b0：该通道没有产生中断 ● 0b1：该通道产生了中断  说明 每个 Bit 对应一个通道。	RO

35.6.4.6 主机全体通道中断屏蔽寄存器 (USBFS_HAINTMSK)

主机全体通道中断屏蔽寄存器与主机全体通道中断寄存器结合使用，进而在通道上发生事件时中断应用程序。
每个通道对应 1 个中断屏蔽位，最多有 16 个位。

Offset	Bit Position																															
0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																HAINTM															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	HAINTM	通道中断屏蔽 (Channel interrupt mask)	对应通道 15~通道 0： ● 0b0：屏蔽中断 ● 0b1：使能中断  说明 每个 Bit 对应一个通道。	RW

35.6.4.7 主机端口控制和状态寄存器 (USBFS_HPRT)

该寄存器仅在主机模式下可用。当前，USBFS 主机仅支持一个端口。

该寄存器包含 USB 端口相关的信息，如 USB 复位、使能、挂起、恢复、连接状态。该寄存器中的 PENCHNG/PCDET 位可通过模块中断寄存器中的主机端口中断位 (USBFS_GINTST 中 HPRTINT 位) 触发应用程序中断。发生端口中断时，应用程序必须读取该寄存器，并将引起中断的位清零。应用程序必须向该位写入 1 以清除该中断。

Offset	Bit Position																															
0x440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												PSPD	Reserved				PWPR	PLSTS		Reserved	PRST	PSUSP	PRES	Reserved		PENCHNG	PENA	PCDET	PCSTS		
Access													RO					RW	RO			RW	RW	RW				RW	RW	RW	RO	

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18:17	PSPD	端口速度(Port speed)	指示连接到该端口的设备的速度。 ● 0b01: 全速 ● 0b10: 低速 ● 0b00/0b11: Reserved	RO
16:13	Reserved	保留	-	-
12	PWPR	端口电源(Port power)	应用程序使用该字段控制该端口的电源。由于本 USBFS 内置 PHY 不具备供电能力，所以此位设置为 1 时，通过 USBFS_DRVVBUS 使能外部 USB 电源芯片供电。 ● 0b0: 掉电 ● 0b1: 通电	RW
11:10	PLSTS	端口线状态	指示 USB 数据线的当前逻辑电平。 [11]: USBFS_DM 的逻辑电平 [10]: USBFS_DP 的逻辑电平	RO
9	Reserved	保留	-	-
8	PRST	端口复位(Port reset)	应用程序将该位置 1 时，会在该端口上启动复位序列。应用程序必须为复位周期定时，并在复位序列完成后将该位清零。 ● 0b0: 端口未处于复位状态 ● 0b1: 端口处于复位状态 应用程序必须将该位置 1 并最少保持 10ms，以在端口上启动复位。	RW
7	PSUSP	端口挂起(Port suspend)	应用程序将此位置 1 以将此端口置于挂起模式。只有此位置 1 时，模块才会停止发送 SOF。要停止 PHY 时钟，应用程序必须将端口时钟停止位置 1，这会使命 PHY 的挂起输入引脚。 此位的读取值反映该端口的当前挂起状态。检测到远程唤醒信号，或者应用程序将此寄存器中的端口复位位或端口恢复位置 1 后，模块可将此位清零；或应用程序将模块中断寄存器中的恢复/远程唤醒检测中断位或断开连接检测中断位（分别为 USBFS_GINTSTS	RW

位/位域	标记	位名	功能描述	读写
7	PSUSP	端口挂起 (Port suspend)	中的 WKUINT 或 DISCINT) 置 1, 模块也可将此位清零。 ● 0b0: 端口未处于挂起模式 ● 0b1: 端口处于挂起模式	RW
6	PRES	端口恢复 (Port resume)	应用程序将此位置 1 以在该端口上驱动恢复信号。模块会持续驱动恢复信号直到应用程序将此位清零。 如模块中断寄存器中的端口恢复/远程唤醒检测中断位 (USBFS_GINTSTS 中 WKUINT 位) 指示, 如果模块检测到 USB 远程唤醒序列, 则开始驱动恢复信号, 而无需应用程序进行干预; 如果模块检测到断开连接的情况, 则将此位清零。 此位的读取值指示当前模块是否正在驱动恢复信号。 ● 0b0: 不驱动恢复信号 ● 0b1: 驱动恢复信号	RW
5:4	Reserved	保留	-	-
3	PENCHNG	端口使能/禁止变化 (Port enable/disable change)	该寄存器中的端口使能位 2 的状态发生变化时, 模块将此位置 1。 通过软件对该位写 1 清零。	RW
2	PENA	端口使能 (Port enable)	端口执行复位序列后, 只能由模块使能, 并且可以由过流状况、断开连接状况或应用程序将此位清零来禁止。应用程序无法通过对寄存器执行写操作将此位置 1。只能将此位清零来禁止端口。对此位的操作不会触发应用程序的任何中断。 ● 0b0: 禁止端口 ● 0b1: 使能端口	RW
1	PCDET	检测到端口连接 (Port connect detected)	当检测到设备连接时, 模块将此位置 1, 以使用模块中断寄存器中的主机端口中断位 (USBFS_GINTSTS 中的 HPRTINT 位) 触发应用程序的中断。 应用程序必须将此位置 1 才可清除该中断。	RW
0	PCSTS	端口连接状态 (Port connect status)	● 0b0: 端口未连接设备 ● 0b1: 端口已连接设备	RO

35.6.4.8 主机通道 x 特性寄存器 (USBFS_HCCHARx) (x=0~15)

该寄存用于设定主机通道特性。

偏移地址 (Offset) : 0x500+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHENA	CHDIS	ODDFRM	DAD								Reserved	EPTYP	LSDEV	Reserved	EPDIR	EPNUM								MPSIZ							
Access	RW	RW	RW	RW									RW	RW		RW	RW								RW							

位/位域	标记	位名	功能描述	读写
31	CHENA	通道使能(Channel enable)	此字段由应用程序软件置 1，并由 USBFS 主机硬件清零。 ● 0b0：禁止通道 ● 0b1：使能通道	RW
30	CHDIS	通道禁止(Channel disable)	应用程序将此位置 1 以停止通过通道发送/接收数据，即使通过该通道的传输还未完成，停止操作仍然生效。 应用程序必须等待禁止通道的中断以确认通道已经被禁止。	RW
29	ODDFRM	奇数帧(Odd frame)	此字段由应用程序置位或复位，以分别指示 USBFS 主机必须传输奇数帧或偶数帧。此字段只适用于周期性（同步和中断）事务。 ● 0b0：偶数帧 ● 0b1：奇数帧	RW
28:22	DAD	设备地址(Device address)	此字段用于指定要与该主机通信的特定设备。	RW
21:20	Reserved	保留	-	-
19:18	EPTYP	端点类型(Endpoint type)	指示选择的传输类型。 ● 0b00：控制 ● 0b01：同步 ● 0b10：批量 ● 0b11：中断	RW
17	LSDEV	低速设备 (Low-speed device)	此字段由应用程序置 1，表示此通道正在与一个低速设备进行通信。	RW
16	Reserved	保留	-	-
15	EPDIR	端点方向(Endpoint direction)	指示通信事务的方向是输入还是输出。 ● 0b0：输出 ● 0b1：输入	RW
14:11	EPNUM	端点编号 (Endpoint number)	指示要与该主机通道通信的 USB 设备的端点号。	RW
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	指示与该主机通道通信的设备端点的最大数据包大小。	RW

35.6.4.9 主机通道 x 中断寄存器 (USBFS_HCINTx) (x=0~15)

该寄存器指示在出现 USB 和 AHB 相关事件时通道的状态。当模块中断寄存器中的主机通道中断位 (USBFS_GINTSTS 中的 HCINT 位) 置 1 时, 应用程序必须读取该寄存器。在对寄存器执行读操作之前, 应用程序必须先读取主机全体通道中断(USBFS_HAINT)寄存器, 以获取主机通道 x 中断寄存器的准确通道编号。应用程序必须将该寄存器中的相应位清零, 才能将 USBFS_HAIN 和 USBFS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0x508+0x20*x

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																							DTERR	FRMOR	BBERR	TXERR	Reserved	ACK	NAK	STALL	AHBERR	CHH	XFRC
Access																								RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10	DTERR	数据同步错误	应用程序需通过写 1 清除该位。	RW
9	FRMOR	帧溢出错误	应用程序需通过写 1 清除该位。	RW
8	BBERR	串扰错误	产生串扰事件的典型原因是端点发送了一个数据包, 但是数据包长度超过了端点的最大包长。 应用程序需通过写 1 清除该位。	RW
7	TXERR	通信事务错误	指示 USB 上发生下列错误之一: ● CRC 校验失败 ● 超时 ● 位填充错误 ● 错误的 EOP 应用程序需通过写 1 清除该位。	RW
6	Reserved	保留	-	-
5	ACK	收到/发出 ACK 响应	应用程序需通过写 1 清除该位。	RW
4	NAK	收到 NAK 响应	应用程序需通过写 1 清除该位。	RW
3	STALL	收到 STALL 响应	应用程序需通过写 1 清除该位。	RW
2	AHBERR	AHB 错误	仅当处于内部 DMA 模式下且 AHB 读/写操作期间发生 AHB 错误时才生成此错误。应用程序可通过读取相应的 DMA 通道地址寄存器来获取错误地址。 应用程序需通过写 1 清除该位。	RW
1	CHH	通道停止	因任意 USB 事务错误或为响应应用程序的禁止请求而导致传输非正常结束。 应用程序需通过写 1 清除该位。	RW
0	XFRC	传输完成	未出现任何错误, 正常完成传输。 应用程序需通过写 1 清除该位。	RW

35.6.4.10 主机通道 x 中断屏蔽寄存器 (USBFS_HCINTMSKx) (x=0~15)

该寄存器用于选择屏蔽主机通道中断。

偏移地址 (Offset) : 0x50C+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																					DTERRM	FRMORM	BBERRM	TXERRM	Reserved		NAKM	STALLM	AHBERRM	CHHM	XFRM
Access																						RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10	DTERRM	数据同步错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
9	FRMORM	帧溢出屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
8	BBERRM	串扰错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
7	TXERRM	通信事务错误屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
6	Reserved	保留	-	-
5	ACKM	ACK 响应接收/发送中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
4	NAKM	NAK 响应接收中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
3	STALLM	收到 STALL 响应中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
2	AHBERRM	AHBERR 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
1	CHHM	通道停止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

35.6.4.11 主机通道 x 传输大小寄存器 (USBFS_HCTSIZx) (x=0~15)

该寄存器用于设定主机通道传输大小以及数据 PID。

偏移地址 (Offset) : 0x510+0x20*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved	DPID		PKTCNT										XFRSIZ																			
Access		RW		RW										RW																			

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	DPID	数据 PID(Data PID)	应用程序在此字段设置数据通信的初始同步 PID。 主机在此次传输事务过程中保留该字段的设置。 ● 0b00: DATA0 ● 0b01: DATA2 ● 0b10: DATA1 ● 0b11: MDATA/SETUP	RW
28:19	PKTCNT	数据包计数 (Packet count)	应用程序在此字段中设置将要发送或接收的数据包数。 主机每成功发送或接收一个数据包便递减一次计数值。此值达到 0 后，将中断应用程序来指示操作正常完成。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	对于 OUT 操作，此字段为传输期间主机发送的数据字节数。 对于 IN 操作，此字段为应用程序保留给传输的缓冲区大小。对于 IN 事务（周期性和非周期性），应用程序会将此字段编程为最大数据包大小的整数倍。	RW

35.6.4.12 主机通道 x DMA 地址寄存器 (USBFS_HCDMAX) (x=0~15)

该寄存器用于设定主机 DMA 模式时 DMA 地址。

偏移地址 (Offset) : 0x514+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	此字段存储主机从设备端点获取数据或往设备端点发送数据所用 DMA 传输的存储器的地址。每次 AHB 传输结束，该寄存器都会递增。	RW

35.6.5 USBFS 设备模式寄存器

设备模式寄存器会影响设备模式下的模块操作。在主机模式下不得访问设备模式寄存器，因为产生的结果不明确。

35.6.5.1 设备配置寄存器 (USBFS_DCFG)

此寄存器在上电、执行某些控制命令或枚举后，会将模块配置为设备模式。请勿在初始编程后更改该寄存器。

Offset	Bit Position																															
0x800	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x08200000																															
Name	Reserved																PFIVL		DAD				Reserved		NZLSOHSK				DSPD			
Access																	RW		RW						RW				RW			

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:11	PFIVL	周期性帧间隔 (Periodic frame interval)	指示一帧内必须使用周期性帧中断通知应用程序的时间点。此功能可用于确定该帧的所有同步通信是否完成。 0b00: 80%帧间隔 0b01: 85%帧间隔 0b10: 90%帧间隔 0b11: 95%帧间隔	RW
10:4	DAD	设备地址 (Device address)	应用程序必须在执行每个 SetAddress 控制命令后根据命令参数对该字段进行设置。	RW
3	Reserved	保留	-	-
2	NZLSOHSK	非零长度状态 OUT 握手信号 (Non-zero-length status OUT handshake)	在控制传输状态阶段的 OUT 事务期间，当模块收到非零长度数据包后，应用程序可以使用此字段选择要发送的握手信号。 0b0: 将收到的 OUT 数据包（零长度或非零长度）发送给应用程序，并基于设备端点控制寄存器中端点的 NAK 和 STALL 位回复握手信号。 0b1: 收到非零长度状态 OUT 事务时，回复 STALL 握手信号，收到的 OUT 数据包不发送给应用程序。	RW
1:0	DSPD	设备速度 (Device speed)	指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成 chirp 序列后才能确定，同时此速度基于与模块连接的 USB 主机的速度。 0b00: 禁止设置 0b01: 禁止设置 0b10: 禁止设置 0b11: 全速 (USB1.1 收发器时钟为 48MHz)	RW

35.6.5.2 设备控制寄存器 (USBFS_DCTL)

Offset	Bit Position																															
0x804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																				POPGRDNE	CGONAK	SGONAK	CGINAK	SGINAK	Reserved	GONSTS	GINSTS	SDIS	RWUSIG		
Access																					RW	RW	RW	RW	RW		RO	RO	RW	RW		

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	POPGRDNE	上电编程完成 (Power-on programming done)	应用程序使用此位指示寄存器从掉电模式唤醒后已完成编程。	RW
10	CGONAK	清零全局 OUT NAK (Clear global OUT NAK)	对此位执行写操作会将全局 OUT NAK 清零。	RW
9	SGONAK	置位全局 OUT NAK (Set global OUT NAK)	对此位执行写操作会将全局 OUT NAK 置 1。 应用程序使用此位在所有 OUT 端点发送 NAK 握手信号。 应用程序只有确定模块中断寄存器中全局 OUT NAK 有效位 (USBFS_GINTSTS 中 GONAKEFF 位) 已清零时, 才可以将此位置 1。	RW
8	CGINAK	清零全局 IN NAK (Clear global IN NAK)	对此位执行写操作会将全局 IN NAK 清零。	RW
7	SGINAK	置位全局 IN NAK (Set global IN NAK)	对此字段执行写操作会将全局非周期性 IN NAK 置 1。应用程序使用此位使所有非周期性 IN 端点发送 NAK 握手信号。 应用程序只有确定模块中断寄存器中全局 IN NAK 有效位 (USBFS_GINTSTS 中 GINAKEFF 位) 已清零时, 才可以将此位置 1。	RW
6:4	Reserved	保留	-	-
3	GONSTS	全局 OUT NAK 状态 (Global OUT NAK status)	0b0: 将根据 FIFO 状态和 NAK 和 STALL 位设置发送握手信号。 0b1: 无论 RxFIFO 中是否还有空闲空间都不接收数据。除 SETUP 事务之外, 对所有收到的数据包回复 NAK 握手信号。所有同步类型的 OUT 数据包都将被丢弃。	RO
2	GINSTS	全局 IN NAK 状态 (Global IN NAK status)	0b0: 将根据发送 FIFO 中的数据可用性回复握手信号。 0b1: 使所有非周期性 IN 端点回复 NAK 握手信号, 无需考虑发送 FIFO 中的数据可用性。	RO

位/位域	标记	位名	功能描述	读写
1	SDIS	软断连(Soft disconnect)	应用程序使用该位向 USBFS 模块发出执行软断开的信号。该位置 1 时，主机不会看到设备已连接，且该设备也不会接收 USB 上的信号。在应用程序将此位清零之前，模块会保持断开状态。 ● 0b0：正常工作。此位在软断连之后清零，会使主机收到设备已连接的事件。重新连接设备之后，USB 主机会重新启动设备枚举。 ● 0b1：使主机收到设备断开连接的事件。 全速时，软断连的最小时间规定如下： ● 挂起状态：最小时间为 1ms+2.5μs ● 空闲状态：2.5μs ● 非空闲或挂起状态：2.5μs	RW
0	RWUSIG	发送远程唤醒信号(Remote wakeup signaling)	应用程序将此位置 1 时，模块会启动远程唤醒信号，以唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出挂起状态。根据 USB2.0 规范，应用程序必须在将此位置 1 之后的 1ms 到 15ms 内将其清零。	RW

35.6.5.3 设备状态寄存器 (USBFS_DSTS)

此寄存器指示模块在出现 USB 相关事件时的状态。发生中断时，必须从设备全体段端点中断寄存器 (USBFS_DAINTE) 读取发生中断的端点信息。

Offset	Bit Position																															
0x808	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved								LNSTS		FNSOF												Reserved				EERR		ENUMSPD		SUSPSTS	
Access									RO		RO																RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:22	LNSTS	USB 总线状态	[23]: D+的逻辑电平 [22]: D-的逻辑电平	RO
21:8	FNSOF	接收 SOF 帧编号	-	RO
7:4	Reserved	保留	-	-
3	EERR	不定错误(Erratic error)	模块将该位置 1 以报告任何不定错误。 由于不定错误，USBFS 控制器会进入挂起状态，并且会 USBFS_GINTSTS 寄存器的早期挂起位 (USBFS_GINTSTS 中的 ESUSP 位) 生成一个中断。 如果早期挂起中断是由不定错误触发，则应用程序只能执行软断开来恢复通信。	RO
2:1	ENUMSPD	枚举速度 (Enumerated speed)	指示 USBFS 控制器通过 chirp 序列检测速度后被枚举成的速度。 ● 0b00: Reserved ● 0b01: Reserved ● 0b10: Reserved ● 0b11: 全速 (PHY 时钟运行频率为 48MHz)	RO
0	SUSPSTS	挂起状态 (Suspend status)	在设备模式下，只要在 USB 上检测到挂起状态，该位就会置 1。当 USB 总线上的空闲状态保持 3ms，模块便会进入挂起状态。出现以下情况时，模块会退出挂起状态： ● USB 数据线上有活动 ● 应用程序对 USBFS_DCTL 寄存器的远程唤醒信号位 (USBFS_DCTL 中的 RWUSIG 位) 执行写操作。	RO

35.6.5.4 设备 IN 端点通用中断屏蔽寄存器 (USBFS_DIEPMSK)

此寄存器与全体端点的各个 USBFS_DIEPINTx 寄存器配合使用，以便在每个 IN 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBFS_DIEPINTx 寄存器中的 IN 端点中断。默认情况下，状态中断都被屏蔽。

Offset	Bit Position																																	
0x810	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																			NAKM	Reserved							INENEM	INENMM	TTXFEMSK	TOM	Reserved	EPDM	XFCM
Access																				RW								RW	RW	RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	NAKM	NAK 中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
12:7	Reserved	保留	-	-
6	INEPNEM	IN 端点 NAK 有效中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
5	INEPNMM	EP 不匹配时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
4	TTXFEMSK	TxFIFO 为空时接收到 IN 令牌中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
3	TOM	超时中断屏蔽 (非同步端点)	0b0: 屏蔽中断 0b1: 使能中断	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW
0	XFRM	传输完成中断屏蔽	0b0: 屏蔽中断 0b1: 使能中断	RW

35.6.5.5 设备 OUT 端点通用中断屏蔽寄存器 (USBFS DOEPMASK)

此寄存器与全体端点的各个 USBFS_DOEPINTx 寄存器配合使用，以便在每个 OUT 端点上生成中断。通过对此寄存器的相应位执行写操作，可屏蔽 USBFS_DOEPINTx 寄存器中的 OUT 端点中断。默认情况下，状态中断都被屏蔽。

Offset	Bit Position																																
0x814	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																												OTEPDM	STUPM	Reserved	EPDM	XFCM
Access																													RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4	OTEPDM	端点禁止时接收到 OUT 令牌中断屏蔽	● 0b0：屏蔽中断 ● 0b1：使能中断  说明 仅适用于控制 OUT 端点。	RW
3	STUPM	SETUP 阶段完成中断屏蔽	● 0b0：屏蔽中断 ● 0b1：使能中断  说明 仅适用于控制端点。	RW
2	Reserved	保留	-	-
1	EPDM	端点禁止中断屏蔽	● 0b0：屏蔽中断 ● 0b1：使能中断	RW
0	XFCRM	传输完成中断屏蔽	● 0b0：屏蔽中断 ● 0b1：使能中断	RW

35.6.5.6 设备全体端点中断寄存器 (USBFS_DAIN_T)

当端点上发生有效事件时，USBFS_DAIN_T 寄存器将通过 USBFS_GINTSTS 寄存器中的设备 OUT 端点中断位或设备 IN 端点中断位（分别为 USBFS_GINTSTS 中的 OEPINT 或 IEPINT 位）来中断应用程序。每个端点对应一个中断位，OUT 端点和 IN 端点均最多有 16 个中断位。双向端点将使用相应的 IN 和 OUT 中断位。当应用程序将相应设备端点 x 中断寄存器(USBFS_DIEPINT_x/USBFS_DOEPINT_x)中的位置 1 和清零时，此寄存器中的相应位也将置 1 和清零。

Offset	Bit Position																															
0x818	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OEPINT																IEPINT															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	OEPINT	OUT 端点中断位 (OUT endpoint interrupt bits)	对应 OUT 端点 15~OUT 端点 0： ● 0b0：该端点未发生中断 ● 0b1：该端点发生中断  说明 每个 Bit 对应一个 OUT 端点。	RW
15:0	IEPINT	IN 端点中断位 (IN endpoint interrupt bits)	对应 IN 端点 15~IN 端点 0： ● 0b0：该端点未发生中断 ● 0b1：该端点发生中断  说明 每个 Bit 对应一个 IN 端点。	RW

35.6.5.7 设备全体端点中断屏蔽寄存器 (USBFS_DAINMSK)

USBFS_DAINMSK 寄存器与设备端点中断寄存器结合使用，在设备端点上发生事件时中断应用程序。但是，与该中断相对应的 USBFS_DAIN 寄存器位仍会置 1。

Offset	Bit Position																															
0x81C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OEPINTM																IEPINTM															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	OEPINTM	OUT 端点中断屏蔽位(OUT endpoint interrupt mask bits)	对应 OUT 端点 15~OUT 端点 0: 0b0: 屏蔽中断 0b1: 使能中断  说明 每个 Bit 对应一个 OUT 端点。	RW
15:0	IEPINTM	IN 端点中断屏蔽位(IN endpoint interrupt mask bits)	对应 IN 端点 15~IN 端点 0: 0b0: 屏蔽中断 0b1: 使能中断  说明 每个 Bit 对应一个 IN 端点。	RW

35.6.5.8 设备 IN 端点 FIFO 空中断屏蔽寄存器 (USBFS_DIEPEMPMSK)

此寄存器用于控制 IN 端点 FIFO 空中断的生成。

Offset	Bit Position																															
0x834	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																INEPTXFEM															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	INEPTXFEM	IN EP Tx FIFO 空中断屏蔽位(IN EP Tx FIFO empty interrupt mask bits)	这些位用作 USBFS_DIEPINTx 的屏蔽位。 对应 IN 端点 15~IN 端点 0: 0b0: 屏蔽中断 0b1: 使能中断  说明 每个 Bit 对应一个 IN 端点的 TXFE 中断。	RW

35.6.5.9 设备控制 IN 端点 0 控制寄存器 (USBFS_DIEPCTL0)

此寄存器用于控制控制传输端点 0。

Offset	Bit Position																															
0x900	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008000																															
Name	EPENA	EPDIS	Reserved		SNAK	CNAK		TXFNUM			STALL	Reserved		EPTYP	NAKSTS	Reserved	USBAEP	Reserved											MPSIZ			
Access	RW	RW			RW	RW		RW			RW			RO	RO		RO												RW			

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点 0 上启动数据发送。在此端点上触发以下任一中断之前，模块会将此位清零： ● SETUP 阶段完成 ● 端点禁止 ● 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传输完成之前，应用程序也可将此位置 1，以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后，才能将端点视为禁止端点。在端点禁止中断位置 1 前，模块会将此位清零。只有在该端点的端点使能位置 1 后，应用程序才可将该位置 1。	RW
29:28	Reserved	保留	-	-
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位，应用程序可以控制端点上 NAK 握手信号的发送。模块也可在端点接收到 SETUP 数据包后将该端点的此位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	TXFNUM	TxFIFO 编号 (TxFIFO number)	该值设置为分配给 IN 端点 0 的 FIFO 编号。只能使用 TX-FIFO0。	RW
21	STALL	STALL 握手 (STALL handshake)	应用程序只能将此位置 1，端点接收到 SETUP 令牌时，模块会将此位清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位均置 1，则 STALL 位优先。	RW
20	Reserved	保留	-	-
19:18	EPTYP	端点类型 (Endpoint type)	硬件设置为'0b00'，表示控制类型的端点。	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0：模块根据 FIFO 状态回复非 NAK 握手 ● 0b1：模块在此端点上回复 NAK 握手 当此位置 1 时（无论是被应用程序还是被模块），即使 TxFIFO 中仍有数据可用，模块也会停止发送数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	USBAEP	USB 活动端点 (USB active endpoint)	此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。	RO
14:2	Reserved	保留	-	-
1:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。 ● 0b00: 64 字节 ● 0b01: 32 字节 ● 0b10: 16 字节 ● 0b11: 8 字节	RW

35.6.5.10 设备 IN 端点 x 控制寄存器 (USBFS_DIEPCTLx) (x=1~15)

应用程序使用此寄存器控制各个逻辑端点 (端点 0 除外) 的行为。

偏移地址 (Offset) : 0x900+0x20*x

Offset	Bit Position																																				
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	EPENA	EPDIS	SODDFRM	SD0PID/SEVNF	RM	SNAK	CNAK	TXFNUM					STALL	Reserved	EPTYP	NAKSTS	EONUM/DPID	USBAEP	Reserved					MPSIZ													
Access	RW	RW	RW	RW	RW	RW		RW	TXFNUM					RW		RO	RO	RO	RO						RW	MPSIZ											

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点上启动数据发送。在此端点上触发以下任一中断之前, 模块会将此位清零: - 端点禁止 - 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传输完成之前, 应用程序也可将此位置 1, 以停止端点上的数据发送。应用程序必须等到发生端点禁止中断后, 才能将端点视为禁止端点。在端点禁止中断位置 1 前, 模块会将此位清零。只有在该端点的端点使能位置 1 后, 应用程序才可将该位置 1。	RW
29	SODDFRM	设置奇数帧 (Set odd frame)	仅适用于同步 IN 和 OUT 端点。对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为奇数帧。	RW
28	SD0PID/SEVNFRM	设置 DATA0 PID/设置偶数帧	SD0PID: 仅适用于中断/批量 IN 端点。对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。 SEVNFRM: 仅适用于同步 IN 端点。对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。	RW
27	SNAK	置位 NAK 位 (Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位, 应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后, 模块也可以将 IN 端点的这个位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	TXFNUM	TxFIFO 编号 (TxFIFO number)	这些位用于指定与此端点相关联的 FIFO 编号。必须为每个有效的 IN 端点设置单独的 FIFO 编号。此字段仅针对 IN 端点有效。	RW
21	STALL	STALL 握手 (STALL handshake)	应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1, 则 STALL 位优先。只有应用程序能够将此位清零, 而模块则不能。	RW
20	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
19:18	EPTYP	端点类型 (Endpoint type)	● 0b00: 控制 ● 0b01: 同步 ● 0b10: 批量 ● 0b11: 中断	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0: 模块根据 FIFO 状态回复非 NAK 握手 ● 0b1: 模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时: ● 对于非同步 IN 端点: 即使 TxFIFO 中存在可用数据, 模块也会停止通过 IN 端点发送任何数据。 ● 对于同步 IN 端点: 即使 TxFIFO 中存在可用数据, 模块也会发送长度为零的数据包。 无论此位如何设置, 模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	EONUM/DPID	偶数/奇数帧/端点数据 PID	EONUM: 仅适用于同步 IN 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程, 以便此端点发送/接收同步数据。 ● 0b0: 偶数帧 ● 0b1: 奇数帧 DPID: 仅适用于中断/批量 IN 端点。 包含此端点上将要接收或发送的数据包的 PID。端点激活后, 应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SDOPID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。 ● 0b0: DATA0 ● 0b1: DATA1	RO
15	USBAEP	USB 活动端点 (USB active endpoint)	指示此端点在当前配置和接口中是否激活。检测到 USB 复位后, 模块会为所有端点 (端点 0 除外) 将此位清零。接收到 SetConfiguration 和 SetInterface 命令后, 应用程序必须相应地对端点寄存器进行编程并将此位置 1。	RO
14:11	Reserved	保留	-	-
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。	RW

35.6.5.11 设备 IN 端点 x 中断寄存器 (USBFS_DIEPINTx) (x=0~15)

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 IN 端点中断位 (USBFS_GINTSTS 中的 IEPINT 位) 置 1 时, 应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前, 必须先读取设备全体端点中断(USBFS_DAINTEP)寄存器, 以获取设备端点 x 中断寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零, 才能将 USBFS_DAINTEP 和 USBFS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0x908+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000080																															
Name	Reserved																								TXFE	INEPNE	Reserved	TTXFE	TOC	Reserved	EPDISD	XFRC
Access																									RO	RW		RW	RW		RW	RW

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	TXFE	发送 FIFO 为空 (Transmit FIFO empty)	当此端点的 TxFIFO 为半空或全空时, 此中断被置位。TxFIFO 为半空还是全空状态由 USBFS_GAHBCFG 寄存器中的 TxFIFO 空白别位 (USBFS_GAHBCFG 中的 TXFELVL 位) 决定。	RO
6	INEPNE	IN 端点 NAK 有效 (IN endpoint NAK effective)	当应用程序通过向 USBFS_DIEPCTLx 中的 CNAK 位写入数据来将 IN 端点 NAK 清零时, 此位可被清零。该中断指示模块已对 (由应用程序或模块) 置 1 的 NAK 采样, 结果已生效。该中断指示由应用程序置 1 的 IN 端点 NAK 位已在模块中起作用。此中断不保证在 USB 上发送了 NAK 握手信号。STALL 位的优先级高于 NAK 位。软件写 1 也可将此位清零。	RW
5	Reserved	保留	-	-
4	TTXFE	TxFIFO 为空时接收到 IN 令牌 (IN token received when TxFIFO is empty)	仅适用于非周期性 IN 端点。当和该端点相对应的 TxFIFO (周期性/非周期性) 为空时, 接收到 IN 令牌, 从而产生中断。通过软件写 1 清零。	RW
3	TOC	超时条件 (Timeout condition)	仅适用于控制 IN 端点。指示该端点对最近收到的 IN 令牌响应超时。通过软件写 1 清零。	RW
2	Reserved	保留	-	-
1	EPDISD	端点禁止中断 (Endpoint disabled interrupt)	此位指示该端点已经由应用程序禁止掉。通过软件写 1 清零。	RW
0	XFRC	传输完成中断 (Transfer completed interrupt)	此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。通过软件写 1 清零。	RW

35.6.5.12 设备 IN 端点 0 传输大小寄存器 (USBFS_DIEPTSIZ0)

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位 (USBFS_DIEPCTL0 中的 EPENA) 使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。
非零端点使用端点 1~15 的寄存器。

Offset	Bit Position																																
0x910	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved												PKTCNT	Reserved										XFRSIZ									
Access													RW											RW									

位/位域	标记	位名	功能描述	读写
31:21	Reserved	保留	-	-
20:19	PKTCNT	数据包计数 (Packet count)	指示端点 0 的一次数据传输包含的数据包个数。每次从 TxFIFO 读取数据包（最大大小或短数据包）时，此字段将递减。	RW
18:7	Reserved	保留	-	-
6:0	XFRSIZ	传输大小 (Transfer size)	指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。每次向 TxFIFO 写入来自外部存储器的数据包时，模块会使此字段递减。	RW

35.6.5.13 设备 IN 端点 x 传输大小寄存器 (USBFS_DIEPTSIZE) (x=1~15)

在使能该端点之前，应用程序必须修改此寄存器。通过 USBFS_DIEPCTLx 寄存器中的端点使能位 (USBFS_DIEPCTLx 中的 EPENA 位) 使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

偏移地址 (Offset) : 0x910+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	MCNT		PKTCNT										XFRSIZ																		
Access		RW		RW										RW																		

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	MCNT	多重计数(Multi count)	对于周期性 IN 端点，此字段指示在 USB 上每帧必须发送的数据包数。模块使用此字段计算同步 IN 端点的数据 PID。 ● 0b00：禁止设置 ● 0b01：1 个数据包 ● 0b10：2 个数据包 ● 0b11：3 个数据包	RW
28:19	PKTCNT	数据包计数 (Packet count)	指示该端点上的一次数据传输包含的数据包个数。每次从 TxFIFO 读取数据包（最大大小或短数据包）时，此字段将递减。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。每次向 TxFIFO 写入来自外部存储器的数据包时，模块会使此字段递减。	RW

35.6.5.14 设备 IN 端点 x DMA 地址寄存器 (USBFS_DIEPDMAx) (x=0~15)

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。

偏移地址 (Offset) : 0x914+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	该位包含使用 DMA 进行端点上数据存储时的外部存储区起始地址。  注意 对于控制端点，该字段所指向的存储区也用于存储控制 OUT 数据包以及 SETUP 事务数据包。连续接收到三个以上的 SETUP 数据包时，存储器中的 SETUP 数据包将被覆盖。每次进行 AHB 传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	RW

35.6.5.15 设备 IN 端点发送 FIFO 状态寄存器 (USBFS_DTXFSTx) (x=0~15)

偏移地址 (Offset) : 0x918+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000280																															
Name	Reserved																INEPTFSAV															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	INEPTFSAV	IN 端点 Tx FIFO 可用空间(IN endpoint Tx FIFO space available)	指示端点 Tx FIFO 中的可用空闲空间大小。以 32 位字为单位： ● 0：端点 Tx FIFO 已满 ● 1：1 个字可用 ● 2：2 个字可用 ● ● n：n 个字可用（n 取值为寄存器 USBFS_DIEPTFx.INEPTXFD 的值）	RO

35.6.5.16 设备控制 OUT 端点 0 控制寄存器 (USBFS_DOEPCTL0)

此寄存器用于控制控制传输端点 0。

Offset	Bit Position																															
0xB00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008000																															
Name	EPENA	EPDIS	Reserved		SNAK	CNAK	Reserved				STALL	SNPM	EPTYP	NAKSTS	Reserved	USBAEP	Reserved										MPSIZ					
Access	RW	RW			RW	RW					RW	RW	RO	RO		RO											RW					

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点 0 上启动数据发送。在此端点上触发以下任一中断之前，模块会将此位清零： ● SETUP 阶段完成 ● 端点禁止 ● 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	应用程序无法禁止控制 OUT 端点 0。	RW
29:28	Reserved	保留	-	-
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位，应用程序可以控制端点上 NAK 握手信号的发送。模块也可在端点接收到 SETUP 数据包后将该端点的此位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	Reserved	保留	-	-
21	STALL	STALL 握手 (STALL handshake)	此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RW
20	SNPM	监听模式 (Snoop mode)	此位用于将端点配置为监听模式。在监听模式下，模块不会在将 OUT 数据包传输到应用存储区前检查其是否正确。	RW
19:18	EPTYP	端点类型 (Endpoint type)	硬件设置为‘0b00’，表示控制类型的端点。	RO
17	NAKSTS	NAK 状态(NAK status)	● 0b0：模块根据 FIFO 状态回复非 NAK 握手 ● 0b1：模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时，即使 Rx FIFO 中存在空间可继续容纳收到的数据包，模块也会停止接收数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
15	USBAEP	USB 活动端点 (USB active endpoint)	此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。	RO
14:2	Reserved	保留	-	-
1:0	MPSIZ	最大数据包大小 (Maximum packet size)	控制 OUT 端点 0 的最大数据包大小与在控制 IN 端点 0 中进行编程的值相同。 ● 0b00: 64 字节 ● 0b01: 32 字节 ● 0b10: 16 字节 ● 0b11: 8 字节	RW

35.6.5.17 设备 OUT 端点 x 控制寄存器 (USBFS_DOEPCTLx) (x=1~15)

应用程序使用此寄存器控制各个逻辑端点 (端点 0 除外) 的行为。

偏移地址 (Offset) : 0xB00+0x20*x

Offset	Bit Position																																	
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	EPENA	EPDIS	SD1PID/SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	Reserved					STALL	SNPM	EPTYP	NAKSTS	EONUM/DPID	USBAEP	Reserved																
Access	RW	RW	RW	RW	RW	RW						RW	RW	RO	RO	RO	RO						RW											

位/位域	标记	位名	功能描述	读写
31	EPENA	端点使能 (Endpoint enable)	应用程序将此位置 1 以在端点上启动数据发送。在此端点上触发以下任一中断之前, 模块会将此位清零: - 端点禁止 - 传输完成	RW
30	EPDIS	端点禁止 (Endpoint disable)	即使在该端点上的传送完成之前, 应用程序也可将此位置 1, 以停止端点上的数据发送/接收。应用程序必须等到发生端点禁止中断后, 才能将端点视为禁止端点。在端点禁止中断位置 1 前, 模块会将此位清零。只有在该端点的端点使能位置 1 后, 应用程序才可将该位置 1。	RW
29	SD1PID/SODDFRM	设置 DATA1 PID/设置奇数帧	SD1PID : 仅适用于中断/批量 OUT 端点。对此字段进行写操作会将此寄存器中的端点数据 PID(DPID)字段设置为 DATA1。 SODDFRM : 仅适用于同步 OUT 端点。对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为奇数帧。	RW
28	SD0PID/SEVNFRM	设置 DATA0 PID/设置偶数帧	SD0PID : 仅适用于中断/批量 OUT 端点。对此字段进行写操作会将此寄存器中的端点数据 PID(DPID)字段设置为 DATA0。 SEVNFRM : 仅适用于同步 OUT 端点。对此字段进行写操作会将偶数/奇数帧(EONUM)字段设置为偶数帧。	RW
27	SNAK	置位 NAK 位(Set NAK)	对此位进行写操作会将端点的 NAK 位置 1。通过此位, 应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后, 模块也可以将 OUT 端点的这个位置 1。	RW
26	CNAK	清零 NAK 位 (Clear NAK)	对此位进行写操作会将端点的 NAK 位清零。	RW
25:22	Reserved	保留	-	-
21	STALL	STALL 握手 (STALL handshake)	此端点接收到 SETUP 令牌时, 应用程序只能将此位置 1, 而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1, 则 STALL 位优先。只有应用程序能够将此位清零, 而模块则不能。	RW

位/位域	标记	位名	功能描述	读写
20	SNPM	监听模式(Snoop mode)	此位用于将端点配置为监听模式。在监听模式下，模块不会再检查接收数据的正确性。	RW
19:18	EPTYP	端点类型 (Endpoint type)	0b00: 控制 0b01: 同步 0b10: 批量 0b11: 中断	RO
17	NAKSTS	NAK 状态(NAK status)	0b0: 模块根据 FIFO 状态回复非 NAK 握手 0b1: 模块在此端点上回复 NAK 握手 当应用程序或模块将此位置 1 时，即使 RxFIFO 存在空间可容纳传入数据包，模块也会停止在 OUT 端点上接收任何数据。 无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。	RO
16	EONUM/DPID	偶数/奇数帧/端点数据 PID	EONUM: 仅适用于同步 OUT 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步数据。 0b0: 偶数帧 0b1: 奇数帧 DPID: 仅适用于中断/批量 OUT 端点。 包含此端点上将要接收或发送的数据包的 PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SD0PID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。 0b0: DATA0 0b1: DATA1	RO
15	USBAEP	USB 活动端点 (USB active endpoint)	指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。	RO
14:11	Reserved	保留	-	-
10:0	MPSIZ	最大数据包大小 (Maximum packet size)	应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。	RW

35.6.5.18 设备 OUT 端点 x 中断寄存器 (USBFS_DOEPINTx) (x=0~15)

此寄存器指示端点在出现 USB 和 AHB 相关事件时的状态。当 USBFS_GINTSTS 寄存器中的 OUT 端点中断位 (USBFS_GINTSTS 中的 OEPINT 位) 置 1 时，应用程序必须读取此寄存器。在应用程序能够读取此寄存器之前，必须先读取 USBFS_DAINTE 寄存器，以获取 USBFS_DOEPINTx 寄存器的准确端点编号。应用程序必须将此寄存器中的相应位清零，才能将 USBFS_DAINTE 和 USBFS_GINTSTS 寄存器中的对应位清零。

偏移地址 (Offset) : 0xB08+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000080																															
Name	Reserved																								B2BSTUP	Reserved	OTEPDIS	STUP	Reserved	EPDISD	XFRC	
Access																									RW		RW	RW		RW	RW	

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6	B2BSTUP	接收到连续的 SETUP 数据包	仅适用于控制 OUT 端点。此位指示该端点已接收到三个以上的连续 SETUP 数据包。软件写 1 也可将此位清零。	RW
5	Reserved	保留	-	-
4	OTEPDIS	端点禁止时接收到 OUT 令牌	仅适用于控制 OUT 端点。指示在尚未使能端点时接收到 OUT 令牌，从而产生中断。通过软件写 1 清零。	RW
3	STUP	SETUP 阶段完成	仅适用于控制 OUT 端点。指示控制端点的 SETUP 阶段已完成，当前控制传输中不再接收到连续的 SETUP 数据包。在此中断上，应用程序可以对接收到的 SETUP 数据包进行解码。通过软件写 1 清零。	RW
2	Reserved	保留	-	-
1	EPDISD	端点禁止中断	此位指示该端点已经由应用程序禁止掉。通过软件写 1 清零。	RW
0	XFRC	传输完成中断	此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。通过软件写 1 清零。	RW

35.6.5.19 设备 OUT 端点 0 传输大小寄存器 (USBFS_DOEPTSIZ0)

在使能端点 0 之前，应用程序必须修改此寄存器。通过设备控制端点 0 控制寄存器中的端点使能位 (USBFS_DIEPCTL0 中的 EPENA) 使能端点 0 后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

非零端点使用端点 1~15 的寄存器。

Offset	Bit Position																															
0xB10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	STUPCNT		Reserved								PKTCNT	Reserved												XFRSIZ							
Access		RW										RW													RW							

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	STUPCNT	SETUP 数据包计数 (SETUP packet count)	此字段指定端点能连续接收的 SETUP 数据包数量。 ● 0b00: 禁止设置 ● 0b01: 1 个数据包 ● 0b10: 2 个数据包 ● 0b11: 3 个数据包	RW
28:20	Reserved	保留	-	-
19	PKTCNT	数据包计数 (Packet count)	一次传输中应该接收到数据包数量。 在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。	RW
18:7	Reserved	保留	-	-
6:0	XFRSIZ	传输大小 (Transfer size)	指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从 RxFIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。	RW

35.6.5.20 设备 OUT 端点 x 传输大小寄存器 (USBFS_DOEPTSIZE_x) (x=1~15)

在使能该端点之前，应用程序必须修改此寄存器。通过 USBFS_DOEPTL_x 寄存器中的端点使能位 (USBFS_DOEPTL_x 中的 EPENA 位) 使能该端点后，模块对此寄存器进行修改。仅当模块将端点使能位清零后，应用程序才能读取此寄存器。

偏移地址 (Offset) : 0xB10+0x20*x

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved		RXDPID/STUPCNT				PKTCNT								XFRSIZ																							
Access			RW				RW								RW																							

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:29	RXDPID/ STUPCNT	接收到的数据 PID/SETUP 数据 包计数	RXDPID : 仅适用于同步 OUT 端点。这是此端点收到的上一个数据包的 PID。 ● 0b00: DATA0 ● 0b01: DATA2 ● 0b10: DATA1 ● 0b11: MDATA STUPCNT : 仅适用于控制 OUT 端点。此字段指定端点能连续接收的 SETUP 数据包数量。 ● 0b00: 禁止设置 ● 0b01: 1 个数据包 ● 0b10: 2 个数据包 ● 0b11: 3 个数据包	RW
28:19	PKTCNT	数据包计数 (Packet count)	指示该端点上的一次数据传输包含的数据包个数。每次向 RxFIFO 写入数据包 (最大大小或短数据包) 后, 此字段将递减。	RW
18:0	XFRSIZ	传输大小 (Transfer size)	此字段包含当前端点的一次数据传输包含的数据量, 以字节为单位。仅当应用程序传输完这些数据后, 模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小, 以在每个数据包结束时中断。每次从 RxFIFO 读取数据包并将其写入外部存储器时, 模块会使此字段递减。	RW

35.6.5.21 设备 OUT 端点 x DMA 地址寄存器 (USBFS_DOEPMxAx) (x=0~15)

该寄存器用于设定设备端点 DMA 模式时 DMA 地址。。

偏移地址 (Offset) : 0xB14+0x20*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DMAADDR																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DMAADDR	DMA 地址(DMA address)	该位包含使用 DMA 进行端点上数据发送时的外部存储区起始地址。  注意 对于控制端点，该字段所指向的存储区也用于存储控制 OUT 数据包以及 SETUP 事务数据包。连续接收到三个以上的 SETUP 数据包时，存储器中的 SETUP 数据包将被覆盖。每次进行 AHB 传输，该寄存器都会递增。应用程序必须设定一个双字对齐地址。	RW

35.6.6 USBFS 时钟门控控制寄存器

通过门控时钟控制寄存器控制 HCLK 和 PHY 时钟从而降低功耗。

35.6.6.1 时钟门控控制寄存器 (USBFS_GCCTL)

此寄存器在主机模式和设备模式下均可用。

Offset	Bit Position																																			
0xE00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																								SUSP		PHYSLEEP		ENL1GTG		Reserved		GATEHCLK		STPPCLK	
Access																									RO		RO		RW				RW		RW	

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7	SUSP	Deep Sleep (L1 suspended) 状态位	0b0: 未处于 Deep Sleep 状态 0b1: 处于 Deep Sleep 状态	RO
6	PHYSLEEP	PHY SLEEP 模式状态位 (PHY in sleep)	0b0: PHY 未处于 Sleep 状态 0b1: PHY 处于 Sleep 状态	RO
5	ENL1GTG	L1 时钟门控 (Enable Sleep Clock Gating)	该位用于 L1 模式时时钟门控控制。	RW
4:2	Reserved	保留	-	-
1	GATEHCLK	门控 HCLK (Gate HCLK)	当 USB 通信挂起或会话无效时，应用程序会将此位置 1，以停止对除 AHB 总线从接口、主接口和唤醒逻辑之外的模块提供时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。	RW
0	STPPCLK	停止 PHY 时钟 (Stop PHY clock)	当 USB 通信挂起、会话无效或设备断开连接时，应用程序将此位置 1 以停止 PH 时钟。当 USB 恢复通信时，应用程序将此位清零。	RW

36 CAN FD 控制器 (CAN FD)

36.1 简介

CAN (Controller Area Network) 总线是一种可以在无主机情况下实现微处理器或者设备之间相互通信的总线标准。

CAN FD 控制器 (CAN1/CAN2) 遵循 CAN 总线协议 2.0A 和 2.0B, 并向上兼容 CAN-FD。

CAN 总线控制器可以处理总线上的数据收发, 在本产品中, CAN FD 控制器具有 16 组筛选器。筛选器用于为应用程序选择要接收的消息。

CAN FD 控制器中应用程序可通过 1 个高优先级的主发送缓冲器 (Primary Transmit Buffer, 以下简称 PTB) 和 3 个辅发送缓冲器 (Secondary Transmit Buffer, 以下简称 STB) 将发送数据送至总线, 由发送调度器决定邮箱发送顺序。通过 8 个接收缓冲器 (Receive Buffer, 以下简称 RB) 获取总线数据。3 个 STB 以及 8 个 RB 可以理解为一个 3 级 FIFO 和一个 8 级 FIFO, FIFO 完全由硬件控制。

CAN FD 总线控制器同时也可以支持时间触发 CAN 通信 (Time-trigger communication)。

36.2 主要特性

- 完全支持 CAN2.0A/CAN2.0B/CAN FD 协议
- CAN2.0 支持最高通信波特率 1Mbit/s
- 支持 1~1/256 的波特率预分频, 灵活配置波特率
- 8 个接收缓冲器
 - ▶ FIFO 方式
 - ▶ 错误或者不被接收的数据不会覆盖存储的消息
- 1 个高优先主发送缓冲器 PTB
- 3 个副发送缓冲器 STB
 - ▶ FIFO 方式
 - ▶ 优先级仲裁方式
- 16 组独立的筛选器
 - ▶ 支持 11 位标准 ID 和 29 位扩展 ID
 - ▶ 可编程 ID CODE 位以及 MASK 位
- PTB/STB 均支持单次发送模式
- 支持静默模式
- 支持回环模式
- 支持捕捉传输的错误种类以及定位仲裁失败位置
- 可编程的错误警告值
- 支持 ISO11898-4 规定时间触发 CAN 以及接收时间戳

36.3 功能说明

36.3.1 系统框图

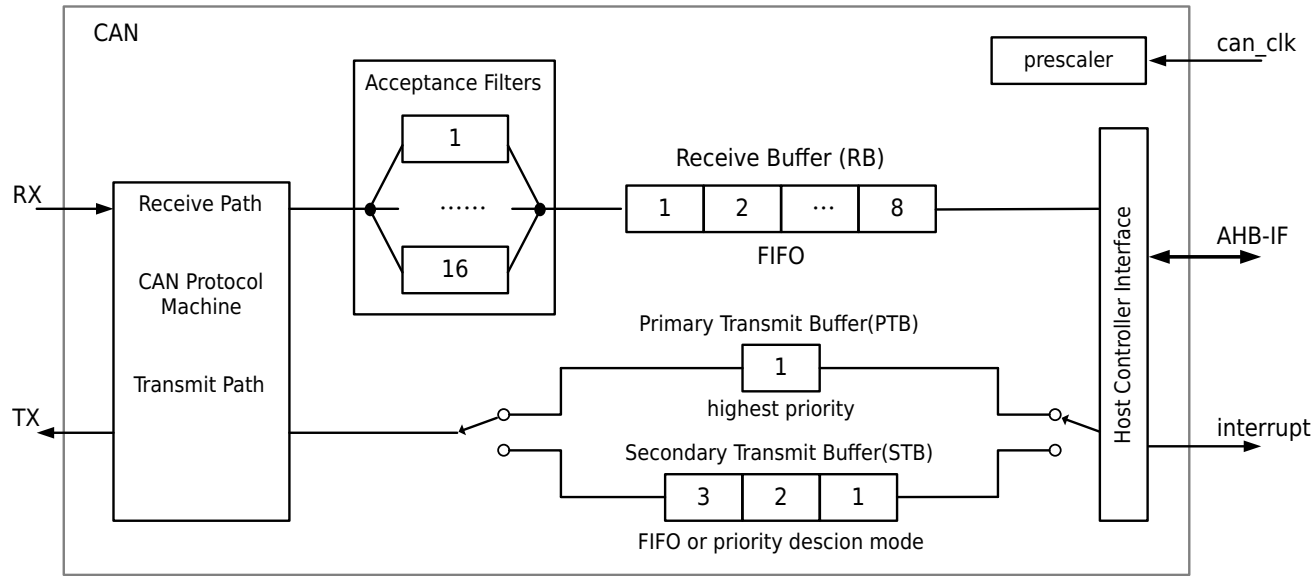


图 36-1 CAN FD 系统框图

36.3.2 管脚说明

表 36-1 CAN 管脚说明

管脚名	方向	功能描述
CANx_RX (x=1~2)	输入	CAN 接收数据信号
CANx_TX (x=1~2)	输出	CAN 发送数据信号
CANx_TST_SAMPLE (x=1~2)	输出	仅用于观测，采样位置（采样点后一周期的通信时钟宽度）
CANx_TST_CLOCK (x=1~2)	输出	仅用于观测，波特率（一位开始前一周期的通信时钟宽度）

36.3.3 工作模式

36.3.3.1 动作模式

使用 CAN FD 功能之前，需先配置 CAN FD 功能使能，详见寄存器章节关于 CAN 全局控制寄存器的说明。CAN FD 控制器存在两个操作模式，复位模式（CAN_CFG_STAT.RESET=1）和动作模式（CAN_CFG_STAT.RESET=0）。模块初始化时，首先应该在复位模式中设定只能在复位模式下操作的寄存器（详见寄存器章节），然后退出复位模式，在动作模式中操作其余寄存器。

36.3.3.2 回环模式

CAN_CTRL 支持以下两种回环模式：

- 内部回环
- 外部回环

两种回环模式都可以接收自己发出的数据帧，主要用于测试用途。

内部回环模式，模块内部将接收数据线连接到发送数据线，并且发送数据不输出。内部回环模式下，节点会生成自应答信号以避免 ACK 错误。

外部回环模式保持和收发器的连接因此发送的数据仍能出现在 CAN 总线上，在收发器的帮助下，CAN 能收到自己发送的数据。外部回环模式可以通过 RCTRL 寄存器的 SACK 位来决定是否生成自应答信号，SACK=0 时，不生成自应答信号；SACK=1 时，生成自应答信号。

外部回环模式，SACK=0 时，会出现以下两种情况：

- 其它节点也收到本节点发送的数据帧并发送应答信号，该情况下本节点能够成功收发数据。
- 如果没有其它节点返回应答信号，则会产生应答错误，会重新发送数据并增加错误计数器。此时推荐采用单次发送模式。

从回环模式返回到正常模式时，除了清除模式位以外，还需要软件复位 CAN_CTRL。

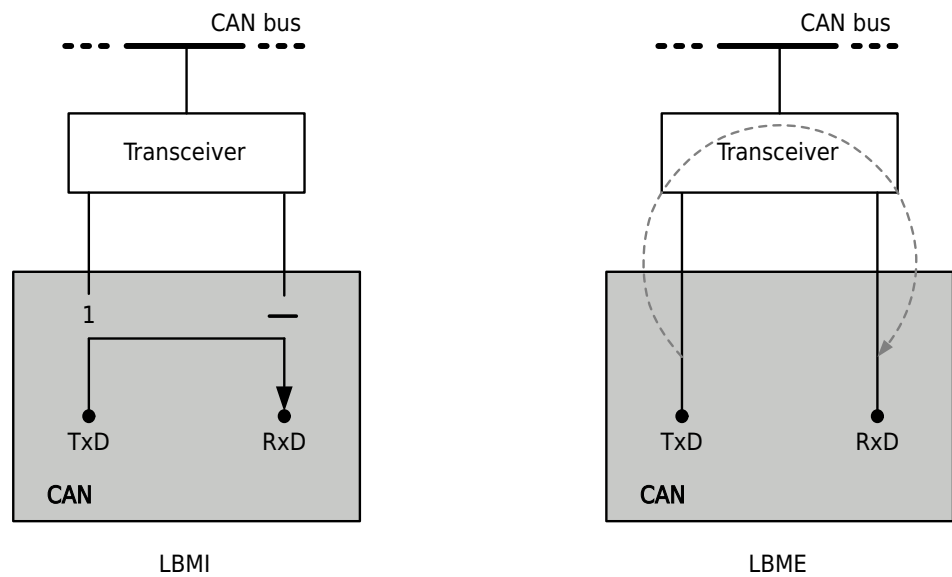


图 36-2 CANFD LBMI 和 LBME 示意图

36.3.3.3 静默模式

静默模式可以用来监控 CAN 网络数据。在静默模式下，可以从 CAN 总线接收数据，不向总线发送任何数据。将 TCMD 寄存器中的 LOM 置 1，使 CAN 总线控制器进入静默模式，将其清 0 可以离开静默模式。

外部回环模式可以和静默模式组合成外部回环静默模式，此时 CAN 可以认为一个安静的接收者，但在有必要的时候可以发送数据。外部回环静默模式下，帧包含自应答信号允许被发送，但是该节点不会产生错误标志和过载帧。

36.3.4 波特率设定

CAN 的通信时钟 (can_clk) 用于生成波特率，可选择系统时钟分频或独立时钟（请参见《时钟控制器 (CMU)》章节“工作时钟规格”），由寄存器 CMU_CANCKCFGR 指定。时钟选择时必须满足 PCLK0 (CAN 控制逻辑时钟) 是 CAN 通信时钟的 1.5 倍及以上的设置条件。

下图给出 CAN 位时间定义图，虚线上部分为 CAN 协议规定的位时间，虚线下部分为本 CAN 控制器 CAN_CTRL 定义的位时间。其中 segment1 和 segment2 可以通过寄存器 SBT 和 FBT 设定。SBT 和 FBT 寄存器只能在 CAN_CFG_STAT.RESET=1 即 CAN 软件复位时设定。SBT 寄存器用于 CAN2.0 和 CAN FD 的仲裁段，FBT 寄存器用于 CAN FD 数据段。

FD 通信时，建议通信时钟选取 20MHz/40MHz/80MHz。

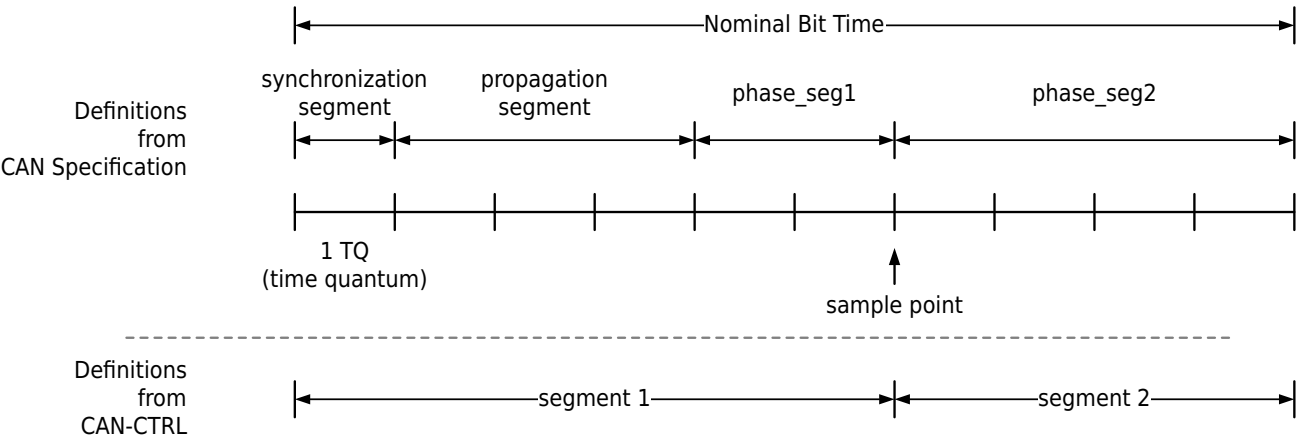


图 36-3 CAN 位时间定义图

TQ 计算方法请参考以下公式，其中 PRESC 通过 SBT 寄存器的 S_PRESC 位或 FBT 寄存器的 F_PRESC 位设定。 f_{can_clk} 为 CAN 通信时钟频率。

$$S\_TQ = \frac{S\_PRESC + 1}{f_{can\_clk}}$$

$$F\_TQ = \frac{F\_PRESC + 1}{f_{can\_clk}}$$

位时间计算方法请参考以下公式，其中 S_segment1 和 S_segment2 通过 SBT 寄存器的 S_SEG_1 位和 S_SEG_2 位设定, F_segment1 和 F_segment2 通过 FBT 寄存器的 F_SEG_1 位和 F_SEG_2 位设定。

$$Slow\_BT = t_{S\_segment1} + t_{S\_segment2} = ((S\_SEG\_1 + 2) + (S\_SEG\_2 + 1)) \times S\_TQ$$

$$Fast\_BT = t_{F\_segment1} + t_{F\_segment2} = ((F\_SEG\_1 + 2) + (F\_SEG\_2 + 1)) \times F\_TQ$$

表 36-2 CAN 位时间设定规则

位	设定范围	规则
SBT 寄存器的 S_SEG_1 位	[0..63] CAN2.0 bits (slow) [0..63] CAN FD nominal bits (slow)	SEG_1≥SEG_2+1 SEG_2≥SJW
SBT 寄存器的 S_SEG_2 位	[0..7] CAN2.0 bits (slow) [0..31] CAN FD nominal bits (slow)	
SBT 寄存器的 S_SJW 位	[0..15] CAN2.0 bits (slow) [0..15] CAN FD nominal bits (slow)	
FBT 寄存器的 F_SEG_1 位	[0..15] CAN FD data bits (fast)	SEG_1≥SEG_2 SEG_2≥SJW
FBT 寄存器的 F_SEG_2 位	[0..7] CAN FD data bits (fast)	
FBT 寄存器的 F_SJW 位	[0..7] CAN FD data bits (fast)	

以下给出 CANFD 的波特率设定推荐，仅供参考。

- PSP: Primary Sample Point
- SSP: Secondary Sample Point
- Seg 1: Segment 1
- Seg 2: Segment 2
- TDC: Transmitter Delay Compensation

表 36-3 20MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 (仲裁)	80	-	1	80	64	16	16	-
0.5 (仲裁)	80	-	1	40	32	8	8	-
0.5	80	禁止	1	40	32	8	8	-
1	80	80	1	20	16	4	4	16
2	80	80	1	10	8	2	2	8
4	80	80	1	5	4	1	1	4
5	75	75	1	4	3	1	1	3

表 36-4 40MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 (仲裁)	80	-	2	80	64	16	16	-
0.5 (仲裁)	80	-	1	80	64	16	16	-
0.5	80	禁止	2	40	32	8	8	-
1	80	80	1	40	32	8	8	32
2	80	80	1	20	16	4	4	16
4	80	80	1	10	8	2	2	8
5	75	75	1	8	6	2	2	6
8	80	80	1	5	4	1	1	4

表 36-5 80MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 (仲裁)	80	-	4	80	64	16	16	-
0.5 (仲裁)	80	-	2	80	64	16	16	-
0.5	80	禁止	4	40	32	8	8	-
1	80	80	2	40	32	8	8	64
2	80	80	2	20	16	4	4	32
4	80	80	1	20	16	4	4	16
5	75	75	1	16	12	4	4	12
8	80	80	1	10	8	2	2	8

36.3.5 发送处理

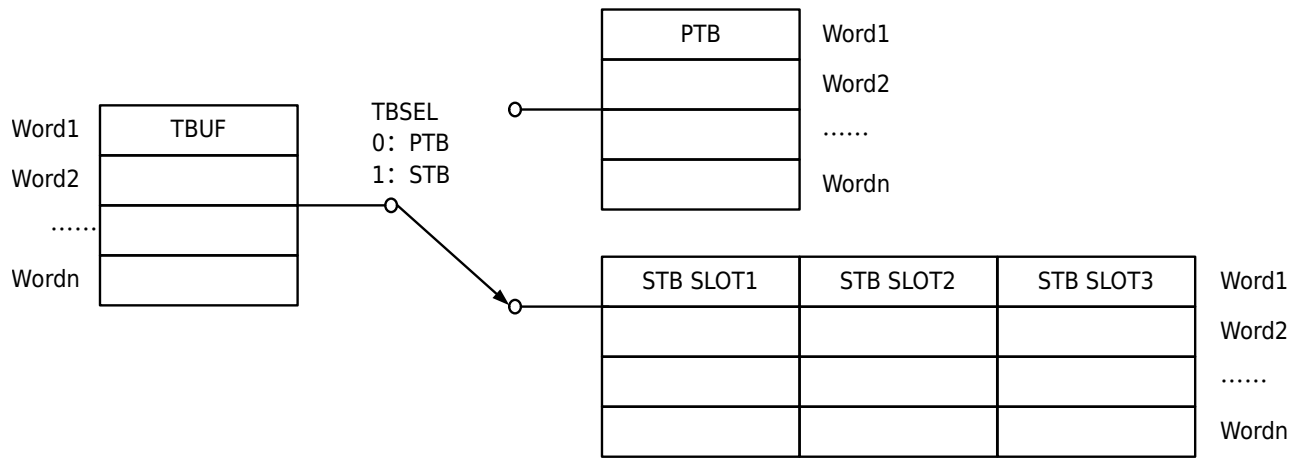
36.3.5.1 发送缓冲器

CAN_CTRL 提供两种发送缓冲器用于发送数据，主发送数据缓冲器 PTB 和副发送缓冲器 STB。PTB 具有最高的优先级，但只能缓冲一帧数据。STB 优先级比 PTB 低，但可以缓冲 3 帧数据，且 STB 内 3 帧数据可以工作在 FIFO 模式或者优先级仲裁模式。

STB 中的 3 帧数据可以通过 TCMD 寄存器的 TSALL 位设定为 1 全部发送，在 FIFO 模式下，最先写入的数据先发送，在优先级模式下，ID 小的数据先发送。

PTB 中的数据具有最高优先级，所以 PTB 发送能推迟 STB 发送，但是已经赢得仲裁并开始发送的 STB 不能够被 PTB 发送推迟。

PTB 和 STB 可以通过 TBUF 寄存器进行访问。通过 TCMD 寄存器的 TBSEL 位选择 PTB 或者 STB，TBSEL=0，选择 PTB；TBSEL=1，选择 STB。通过 TCTRL 寄存器的 TSNEXT 位选择 STB 中的下一个 SLOT。对应关系如下图所示：



Set TSNEXT bit to mark a slot filled and to jump to the next message slot

图 36-4 CANFD TBUF 寄存器写发送缓冲器和示意图

36.3.5.2 数据发送

在开始发送前必须保证 PTB 或者 STB 待发送的数据已被填充完毕，然后再启动 PTB 或者 STB 发送。发送过程中不允许再次填充数据。

发送数据设定步骤如下：

- 步骤 1. 设定 TBSEL 从 PTB 和 STB 中选择发送 BUF；
- 步骤 2. 通过 TBUF 寄存器写需要发送的数据；
- 步骤 3. 如果选择的时 STB，设置 TSNEXT=1 以完成全部 STB SLOT 的装载；
- 步骤 4. 发送使能：
 - PTB 发送使用 TPE；
 - STB 发送使用 TSALL 或者 TSONE。
- 步骤 5. 发送完成状态确认：
 - PTB 发送完成使用 TPIF，TPIE 用于使能 TPIF；
 - STB 采用 TSONE 发送完成时使用 TSIF，TSIE 用于使能 TSIF；
 - STB 采用 TSALL 发送完成时使用 TSIF，此时需要设定的全部 STB SLOT 数据发送完成后，TSIF 才置位，TSIE 用于使能 TSIF。

36.3.5.3 单次数据发送

不需要使用自动重新发送功能时，可以通过寄存器设定为单次发送模式，CFG_STAT 寄存器的 TPSS 位用于 PTB 的单次发送模式设定，TSSS 位用于 STB 的单次发送模式。数据成功发送时单次发送和正常发送模式时动作相同。但是数据没有成功发送时会出现以下结果：

- TPIF 置位 (TPIE=1)，对应的 BUF SLOT 数据会被清除；
- 有错误发送时，KOER 更新，BEIF 置位 (BEIE=1)；
- 仲裁失败，ALIF 置位 (ALIE=1)。

单次发送模式，不能单独依靠 TPIF 来判断发送完成，需要同 BEIF 和 ALIF 一起判断发送是否完成。

36.3.5.4 取消数据发送

可以通过 TPA 或者 TSA 取消已请求但还没有被执行的数据发送。取消数据发送会出现以下几种情况：

- 仲裁中：
 - ▶ 节点仲裁失败，则取消数据发送；
 - ▶ 节点仲裁成功，则继续发送。
- 数据发送中：
 - ▶ 成功发送数据且收到 ACK，对应的标志和状态正常置位。数据发送不取消；
 - ▶ 成功发送数据但没有收到 ACK，数据发送取消，错误计数器增加；
 - ▶ TSALL=1 设定的发送数据，正在发送的 STB SLOT 数据正常发送，没有开始发送的 STB SLOT 被取消。

取消数据发送的结果有以下两种情况：

- TPA 释放 PTB，且使 TPE=0；
- TSA 释放一个 STB SLOT 或者全部 STB SLOT 取决是 TSONE 还是 TSALL 使能的发送。

36.3.6 接收处理

36.3.6.1 接收缓冲器

CAN_CTRL 提供 8 个 SLOT 的接收缓冲器用于存储接收到的数据，该 8 SLOT 的接收缓冲器工作在 FIFO 模式。RB SLOT 通过 RBUF 寄存器来读取接收到的数据，总是最先读取最早接收到的数据，并通过 RCTRL 寄存器的 RREL 设置为 1 释放已经读取的 RB SLOT，并指向下一个 RB SLOT。

通过 RBUF 读取 RB SLOT 示意图如下。



Set RREL bit to jump to the next RB slot

图 36-5 CAN RBUF 寄存器读接收缓冲器示意图

36.3.6.2 接收筛选寄存器组

CAN_CTRL 提供 16 组 32 位筛选器用于过滤接收到的数据从而降低 CPU 负荷，筛选器可以支持标准格式 11 位 ID 或者扩展格式 29 位 ID。每组筛选器有一个 32 位 ID CODE 寄存器和一个 32 位 ID MASK 寄存器，ID CODE 寄存器用于比较接收到 CAN ID，而 ID MASK 寄存器用于选择比较的 CAN ID 位。对应的 ID MASK 位为 1 时，不比较该位的 ID CODE。

接收到的数据只要通过 16 组筛选器的任意一组，则被接收，接收到的数据存储在 RB 中，否则数据不被接收，也不被存储。

每组筛选器通过 ACFEN 寄存器使能或者禁止。ID CODE 和 ID MASK 通过 ACFCTRL 寄存器的 SELMASK 位设定，SELMASK=0 时，指向 ID CODE；SELMASK=1 时，指向 ID MASK。筛选器通过 ACFCTRL 寄存器的 ACFADR 位选择。ID CODE 和 ID MASK 通过 ACF 寄存器访问且只能在 CFG_STAT.RESET=1 即 CAN 软件复位时设定。ACF 寄存访问筛选寄存器组的方式请参考下图。

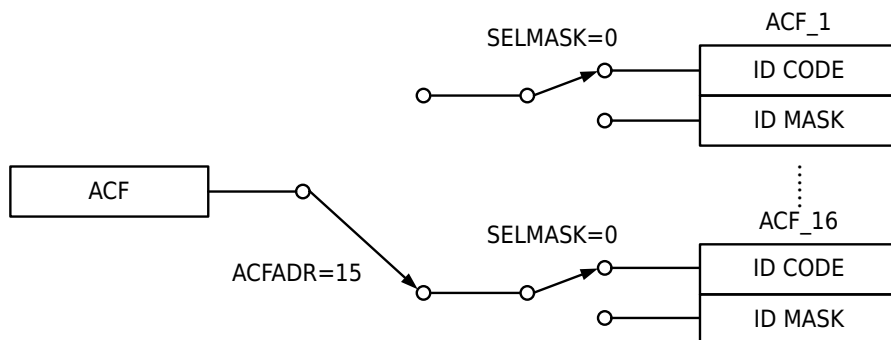


图 36-6 CAN ACF 寄存器访问筛选器组示意图

36.3.6.3 数据接收

接收筛选器组可以过滤掉不需要的接收数据，减少中断的发生和 RB 的读取，从而降低 CPU 负荷。接收数据设定步骤如下：

- 步骤 1. 设定筛选器组；
- 步骤 2. 设定 RFIE, RAFIE 和 AFWL；
- 步骤 3. 等待 RFIF 或者 RAFIF；
- 步骤 4. 通过 RBUF 从 RB FIFO 中读取最早接收到的数据；
- 步骤 5. 设置 RREL=1，选择下一个 RB SLOT；
- 步骤 6. 重复步骤 4 和 5 直到通过 RSTAT 确认 RB 为空。

36.3.7 错误处理

CAN_CTRL 不仅可以自动处理部分错误，比如自动重发数据或者丢弃接收到含有错误的帧；还可以通过中断将错误向 CPU 报告。

CAN 节点有以下三种错误状态：

- 错误主动：节点检测到错误时自动发送主动错误标志。
- 错误被动：节点检测到错误时自动发送被动错误标志。
- 节点关闭：关闭状态下此节点不再影响整个 CAN 网络。

CAN_CTRL 提供 TECNT 和 RECNT 两个计数器用于计数错误。TECNT 和 RECNT 计数器按照 CAN 协议规定的规则进行增减。另外提供可编程的 CAN 错误警告 LIMIT 寄存器用于产生错误中断通知 CPU。

CAN 通信过程中有以下 5 种错误类型，错误类型可以通过 EALCAP 寄存器的 KOER 位识别。

- 位错误
- 形式错误
- 填充错误
- 应答错误
- CRC 错误

36.3.8 节点关闭

当发送错误数大于 255 时，CAN 节点自动进入节点关闭状态从而不参与 CAN 通信，直到返回到错误主动状态。可以通过 CFG_STAT 寄存器的 BUSOFF 位确认 CAN 节点关闭状态。BUSOFF 被置位的同时 EIF 中断产生。

CAN 从节点关闭状态恢复到错误主动状态有以下两种方法：

- 上电复位
- 接收到连续 128 个 11 位的隐形位序列（恢复序列）

节点关闭状态下, TECNT 值保持不变, RECNT 用于计数恢复序列。从节点关闭状态恢复后, TECNT 和 RECNT 被复位为 0。

36.3.9 仲裁失败位置捕捉

CAN_CTRL 能够精确捕捉到仲裁失败位的位置并反映到 ALC 寄存器中。ALC 寄存器中保存着最近一次仲裁失败位的位置, 如果节点赢得仲裁, 则 ALC 位不更新。

ALC 值定义如下:

SOF 位后, 第一个 ID 数据位 ALC 为 0, 第二个 ID 数据位 ALC 为 1, 依次类推。因为仲裁只发生在仲裁场内, 所以 ALC 的最大值为 31。比如一个标准格式远程帧和一个扩展帧仲裁, 扩展帧在 IDE 位失败, 则 ALC=12。

36.3.10 时间触发 TTCAN

CAN-CTRL 为 ISO11898-4 规定的时间触发通信方式提供部分 (level 1) 硬件支持。本章节从以下 5 个部分介绍 TTCAN 功能。

36.3.10.1 TTCAN 模式下的 TBUF 行为

TTTBM=1

TTTBM=1 时, PTB 和 STB SLOT 一样组成 TB SLOT, 通过 TBPTR 寄存器指定发送 BUF, 其中 TBPTR=0 时指向 PTB, TBPTR=1 时指向 STB SLOT1, 依次类推。主机可以通过 TBE 和 TBF 寄存器来标记发送 BUF SLOT。此时 TBSEL 和 TSNEXT 寄存器无任何意义从而可以被忽略。

TTTBM=1 时, PTB 不具有任何特殊的属性, 和 STB SLOT 一样, 传送完成标志也采用 TSIF。

TTCAN 模式时, 发送 BUF 没有 FIFO 模式和优先级仲裁模式, 同时也只有一个选定的 SLOT 可以发送数据。

TTCAN 模式下, 传输开始需要采用时间触发方式, TPE、TSONE、TSALL、TPSS 和 TPA 被固定为 0 且被忽略。

TTTBM=0

TTTBM=0 时, 组合使用事件驱动通信和接收时间戳功能。在该模式下, PTB 和 STB 的功能和 TTEN=0 时一致, 因此 PTB 始终具有最高的优先级, 而 STB 可以工作在 FIFO 模式或者优先级仲裁模式。

36.3.10.2 TTCAN 功能描述

上电后, Time Master 需要根据 ISO 11898-4 协议进行初始化。一个 CAN 网络中, 最多可以有 8 个潜在的 Time Master。每一个 Time Master 都具有自己的参考消息 ID (ID 最后 3 位)。这些潜在的 Time Master 根据自己的优先级发送各自的参考消息。

TTEN=1 后, 16 位的计数器开始工作, 当参考消息被成功接收或者 Time Master 成功发送参考消息时, CAN 控制器将 Sync_Mark 拷贝给 Ref_Mark, Ref_Mark 将 cycle time 设置为 0。成功接收参考消息置位 RIF 标志而成功发送参考消息置位 TPIF 标志或者 TSIF 标志。此时主机需要准备下一个动作的触发条件。

触发条件可以是接收触发。该触发仅触发中断可用于检测期待的消息没有被收到。

触发条件也可以是发送触发。该触发开始发送通过 TTPTR 寄存器指定的 TBUF SLOT 里的数据。如果选定的 TBUF SLOT 被标记为空, 则不开始发送, 但置位中断标志。

36.3.10.3 TTCAN 时序

CAN_CTRL 支持 ISO11898-4 level 1。包含的一个 16 位计数器工作在 S_PRESC、S_SEG_1、S_SEG_2 定义的位时间下。如果 TTEN=1, 则有一个额外的预分频器 T_PRESC。

一帧数据的 SOF 时, 计数器的值为 Sync_Mark。如果该帧数据为参考消息, 则将 Sync_Mark 拷贝给 Ref_Mark。cycle time 等于计数器的值减去 Ref_Mark。该时间用作为接收消息的时间戳或者发送消息的触发时间基准。

36.3.10.4 TTCAN 触发方式

通过 TTYPE 寄存器定义 TTCAN 的触发方式, TTPTR 寄存器指定发送 SLOT, 而 TT_TRIG 指定触发器的 cycle time。

包含以下五种触发方式:

- 立即触发
- 时间触发
- 单次发送触发
- 发送开始触发
- 发送停止触发

除了立即触发方式外, 所有的触发器都使用 TTIF 标志。TTTBM=1 时, 只支持时间触发方式。

立即触发

通过写 TT_TRIG 的高位 (不在意写入的值), 启动触发器。此模式下, TTPTR 选定的 TBUF SLOT 内的数据会立即发送。TTIF 不置位。

时间触发

时间触发方式仅通过置位 TTIF 标志产生中断, 并无其他功能。如果一个节点期待在特定的时间窗口内收到期待的数据, 则可以使用时间触发方式。如果 TT_TRIG 值小于实际的 cycle time, 则 TEIF 置位且无其它动作。

单次发送触发

单次发送触发方式用于在执行时间窗口内发送数据。此时, 忽略 TSSS 位。

通过 TEW 位设定 ISO11898-4 规定的最多 16 个 cycle time 的 Tick, 设定范围为 1~16。如果在规定的发送使能时间窗口内数据没有开始发送, 则帧被丢弃。对应的发送 BUF SLOT 被标记为空, 并且置位 AIF, 对应的发送 BUF 内的数据不会被改写, 因为可以通过置位 TPF 再次发送。

如果 TT_TRIG 值小于实际的 cycle time, 则 TEIF 置位且无其它动作。

发送开始触发

发送开始触发方式用于仲裁时间窗口内, 参与仲裁。TSSS 用于决定是否自动重发或者单次发送模式。如果 TTPTR 寄存器指定的消息没有被成功发送, 可以使用发送停止触发来停止该发送。

如果 TT_TRIG 值小于实际的 cycle time, 则 TEIF 置位且无其它动作。

发送停止触发

发送停止触发方式用于停止通过发送开始触发方式已经开始的发送。如果发送被停止, 则发送帧被舍弃, 置位 AIF 并将选定的 TBUF SLOT 标记为空, 但 TBUF SLOT 内的数据不会被改写, 可以通过置位 TPF 就可以再次发送。

如果 TT_TRIG 值小于实际的 cycle time 则 TEIF 置位且执行停止。

36.3.10.5 TTCAN 触发看门时间

TTCAN 触发看门时间功能类似于看门狗功能, 在 TTTBM=1 时使用。用来看门从上次成功接收到参考消息开始的时间。参考消息可以在周期 cycle time 中或者一个事件后被接收, 应用程序应该根据具体情况设定合适的看门时间。

如果 cycle count 等于 TT_WTRIG, 则置位 WTIF。通过 WTIE 写 0, 关闭看门触发。

如果 TT_WTRIG 比实际的 cycle time 小, 则 TEIF 置位。

36.3.11 TDC 和 RDC

CANFD 通信时, 数据的延迟可能超过一位时间, 因此需要补偿。TDC (Transmitter Delay Compensation) 和 RDC (Receiver Delay Compensation) 用于数据延迟的补偿。其中 TDC 需要软件控制开关, 而 RDC 不需要, 自动有效。TDC 中采用辅助采样点 SSP (Secondary Sample Point) 的方式去补偿数据延迟。如下图:

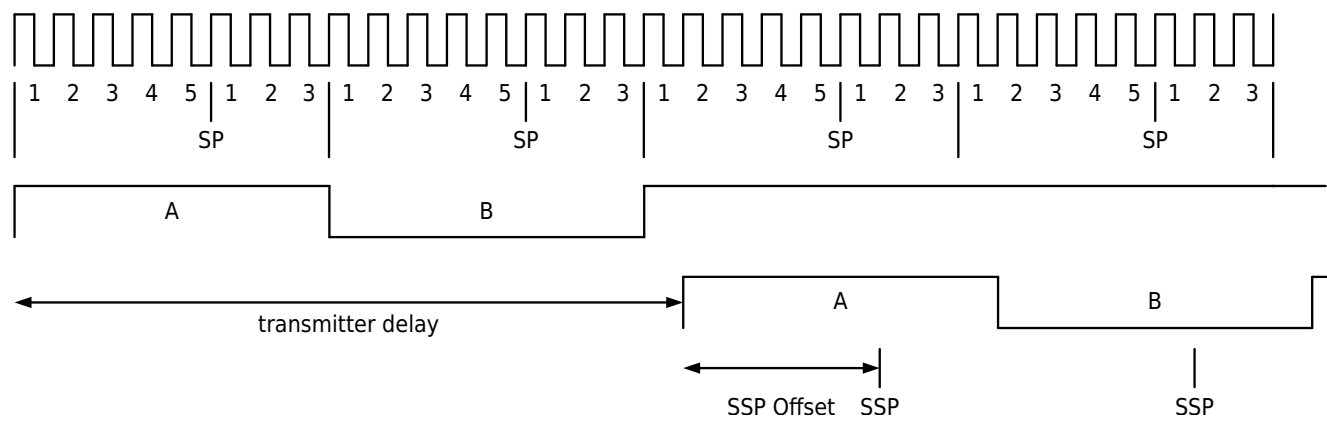


图 36-7 TDC 功能示意图

软件使能 TDC 功能时，本控制器可以自动确定 transmitter delay，通过设定寄存器 TDC.SSPOFF 来设定 SSP Offset。建议 SSP Offset 设定值和 $t_{F\_segment1}$ 设定值相同。

36.3.12 向上兼容 CAN-FD 功能

CAN-CTRL 在 CAN FD 功能禁止时即使在包含 CAN FD 网络中收到 CAN FD 的帧，接收器会自动忽略这些帧，不返回 ACK，等到总线空闲时，再发送或者接收下一个 CAN2.0B 的帧。

36.3.13 RAM ECC 功能

每个 CAN 模块内部包含 1 个 RAM 存储器，用作 CAN 模块的发送缓冲器 (TBUF)、接收缓冲器 (RBUF) 和接收筛选器 (ACF)。

CAN RAM 中的发送缓冲器 (TBUF)，起始地址 0x00000000，每个 TBUF 的大小是 72 字节，按地址从低到高依次是 PTB、STB1~STB3；接收缓冲器 (RBUF)，起始地址 0x00000120，每个 RBUF 的大小是 72 字节，按地址从低到高依次是 RBUF1~RBUF8；接收筛选器 (ACF)，起始地址 0x000003A8，每个 ACF 的大小是 8 字节，按地址从低到高依次是 ACF1~ACF16。

表 36-6 CAN RAM 地址映射

CAN RAM 元素	CAN RAM 地址区间
PTB	0x00000000~0x00000047
STB	0x00000048~0x0000011F
RBUF	0x00000120~0x000003A7
ACF	0x000003A8~0x00000427

RAM 带有 ECC 校验，ECC 校验为纠一检二码，即可以纠正 1-bit 错误，检查 2-bit 错误。ECC 校验带有错误注入功能。根据错误注入使能寄存器 CAN_RAM_EIEN 和错误注入位使能寄存器 CAN_RAM_EIBIT0/CAN_RAM_EIBIT1 的设定，硬件电路将从 RAM 读取的数据按位取反，以达到错误注入的目的。通过设定错误注入位使能寄存器 CAN_RAM_EIBIT0/CAN_RAM_EIBIT1，可以制造 1-bit 错误或多 bit 错误，由于 ECC 校验为纠一检二码，因此超过 2-bit 错误有可能不会被检出。CAN RAM 校验错误地址寄存器

(CAN_RAM_ECCERRADDR) 记录 CAN RAM 发生 1-bit 或 2-bit ECC 错误时的访问地址（如上表所示）。

36.3.14 软件复位功能

通过设定寄存器 CFG_STAT 寄存器的 RESET 位为 1，实现软件复位功能，软件复位功能的复位范围如下表所示。

表 36-7 软件复位范围表

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
ACFADR	否	-	F_SEG_1	是	只能在软件复位时可写
ACODE	否	只能在软件复位时可写	F_SEG_2	是	只能在软件复位时可写
AE_x	否	-	F_SJW	是	只能在软件复位时可写
AFWL	否	-	KOER	是	-
AIF	是	-	LBME	是	-
ALC	是	-	LBMI	是	-
ALIE	否	-	RACTIVE	是	接收立即停止，并不生成 ACK
ALIF	是	-	RAFIE	否	-
AMASK	否	只能在软件复位时可写	RAFIF	是	-
BEIE	否	-	RBALL	是	-
BEIF	是	-	RBUF	是	RB 被标记为空，数值不定
BUSOFF	否	通过写 1 清除	RECNT	否	通过 BUSOFF 写 1 清零
EIEF	否	-	REF_ID	否	-
EIF	否	-	REF_IDE	否	-
EPASS	否	-	RFIE	否	-
EPIE	否	-	RFIF	是	-
EPIF	是	-	RIE	否	-
EWARN	否	-	RIF	是	-
EWL	是	-	ROIE	否	-
FD_ISO	否	只能在软件复位时可写	ROIF	是	-
F_FRESC	否	只能在软件复位时可写	ROM	否	-
ROV	是	-	TSMODE	否	-
RREL	是	-	TSNEXT	是	-
RSTAT	是	-	TSONE	是	-
SACK	是	-	TPIE	否	-
SELMASK	否	-	TPIF	是	-
S_PRESC	否	只能在软件复位时可以写	TPSS	是	-
S_SEG_1	否	只能在软件复位时可以写	TSFF	是	所有 STB SLOT 被标记为空

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
S_SEG_2	否	只能在软件复位时可以写	TSIE	否	-
S_SJW	否	只能在软件复位时可以写	TSIF	是	-
SSPOFF	是	-	TSSS	是	-
TACTIVE	是	发送立即停止	TSSTAT	是	所有 STB SLOT 被标记为空
TBE	是	-	TTEN	是	-
TBF	否	-	TTIF	是	-
TBPTR	否	-	TTIE	否	-
TBSEL	是	-	TTPTR	否	-
TBUF	是	STB 被标记为空，指向 PTB	TTTBM	否	-
TDCEN	是	-	TTYPE	否	-
TECNT	否	可通过 BUSOFF=1 清除	TT_TRIG	否	-
TEIF	是	-	TT_WTRIG	否	-
TPA	是	-	T_PRESC	否	-
TPE	是	-	WTIE	否	-
TSA	是	-	WTIF	是	-
TSALL	是	-			

36.3.15 中断

表 36-8 CAN 中断表

符号	中断标志	描述
CAN_x_HOST(x=1~2)	RIF	接收中断
	ROIF	接收上溢中断
	ROIF	接收 BUF 满中断
	RAFIF	接收 BUF 将满中断
	TPIF	PTB 发送中断
	TSIF	STB 发送中断
	EIF	错误中断
	AIF	取消发送中断
	EPIE	错误被动中断
	ALIF	仲裁失败中断
	BEIF	总线错误中断
	WTIF	触发看门中断
	TEIF	触发错误中断
	TTIF	时间触发中断

36.4 寄存器

36.4.1 寄存器总表

CAN 全局控制寄存器基地址：0x40055400

表 36-9 CAN 全局控制寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x18	CAN_SYCTLREG	CAN 全局控制寄存器	32

- CAN1 基地址：0x40009000
- CAN2 基地址：0x40078000

表 36-10 CAN FD 控制器 (CAN FD) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00~0x4F	CAN_RBUF	CAN 接收 BUF 寄存器	-
0x50~0x97	CAN_TBUF	CAN 发送 BUF 寄存器	-
0xA0	CAN_CFG_STAT	CAN 配置和状态寄存器	8
0xA1	CAN_TCMD	CAN 命令寄存器	8
0xA2	CAN_TCTRL	CAN 发送控制寄存器	8
0xA3	CAN_RCTRL	CAN 接收控制寄存器	8
0xA4	CAN_RTIE	CAN 接收和发送中断使能寄存器	8
0xA5	CAN_RTIF	CAN 接收和发送中断标志寄存器	8
0xA6	CAN_ERRINT	CAN 错误中断使能和标志寄存器	8
0xA7	CAN_LIMIT	CAN 警告限定寄存器	8
0xA8	CAN_SBT	CAN Slow 位时序寄存器	32
0xAC	CAN_FBT	CAN Fast 位时序寄存器	32
0xB0	CAN_EALCAP	CAN 错误和仲裁失败捕捉寄存器	8
0xB1	CAN_TDC	CAN 发送器延迟补偿寄存器	8
0xB2	CAN_RECNT	CAN 接收错误计数器寄存器	8
0xB3	CAN_TECNT	CAN 发送错误计数器寄存器	8
0xB4	CAN_ACFCTRL	CAN 筛选器组控制寄存器	8
0xB6	CAN_ACFEN	CAN 筛选器组使能寄存器	16
0xB8	CAN_ACF	CAN 筛选器组 code 和 mask 寄存器	32
0xBE	CAN_TB SLOT	TTCAN TB slot 指针寄存器	8
0xBF	CAN_TTCFG	TTCAN 时间触发配置寄存器	8

偏移地址	寄存器	描述	位宽
0xC0	CAN_REF_MSG	TTCAN 参考消息寄存器	32
0xC4	CAN_TRG_CFG	TTCAN 触发配置寄存器	16
0xC6	CAN_TT_TRIG	TTCAN 触发时间寄存器	16
0xC8	CAN_TT_WTRIG	TTCAN 触发看门时间寄存器	16
0x100	CAN_RAM_CKCR	CAN RAM 校验控制寄存器	32
0x104	CAN_RAM_CKSR	CAN RAM 校验状态寄存器	32
0x108	CAN_RAM_EIEN	CAN RAM 错误注入使能寄存器	32
0x10C	CAN_RAM_EIBIT0	CAN RAM 错误注入位使能寄存器 0	32
0x110	CAN_RAM_EIBIT1	CAN RAM 错误注入位使能寄存器 1	32
0x114	CAN_RAM_ECCERRADDR	CAN RAM 校验错误地址寄存器	32

36.4.2 CAN 接收 BUF 寄存器 (CAN_RBUF)

偏移地址 (Offset) : 0x00~0x4F

复位值 (Reset) : 0xFFFFFFFF

RBUF 寄存器指向最早接收到的 CAN 邮箱的 RB SLOT 地址, RBUF 寄存器可以按照任意顺序读取。

KOER 位即为寄存器 EALCAP.KOER, 仅在 RBALL=1 时有意义。

TX 位表示在回环模式下接受到自己发送的消息。

CYCLE_TIME 位仅在 TTCAN 模式时有效, 表示 SOF 开始时的 cycle time。

CAN_RBUF 仅支持 WORD 访问。

CAN 接收邮箱的数据格式如下:

表 36-11 标准格式 CAN 接收邮箱格式

地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	功能
RBUF	ID[7:0]								ID
RBUF+1	Reserved					ID[10:8]			ID
RBUF+2	Reserved								ID
RBUF+3	Reserved								ID
RBUF+4	IDE=0	RTR	FDF	BRS	DLC[3:0]				Control
RBUF+5	KOER[2:0]			TX	Reserved				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
...	...								Data
RBUF+70	DATA63								Data
RBUF+71	DATA64								Data

表 36-12 扩展格式 CAN 接收邮箱格式

地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	功能
RBUF	ID[7:0]								ID
RBUF+1	ID[15:8]								ID
RBUF+2	ID[23:16]								ID
RBUF+3	Reserved			ID[28:24]					ID
RBUF+4	IDE=1	RTR	FDF	BRS	DLC[3:0]				Control
RBUF+5	KOER[2:0]			TX	Reserved				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
...	...								Data
RBUF+70	DATA63								Data

地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	功能
RBUF+71	DATA64								Data

控制位和状态位含义如下：

分类	位名	功能描述																								
控制位	IDE(IDentifier Extension)	● 0b0：标准格式 ● 0b1：扩展格式																								
	RTR(Remote Transmission Request)	● 0b0：数据帧 ● 0b1：远程帧（仅适用于 CAN2.0,CAN FD 时固定为 0）																								
	FDF(CAN FD Frame)	● 0b0：CAN2.0 帧 ● 0b1：CAN FD 帧																								
	BRS(Bit Rate Switch)	● 0b0：整帧为低速波特率 ● 0b1：数据和 CRC 为快速波特率（仅适用于 CAN FD，FDF=0 时固定为 0）																								
	DLC(Data Length Code)	数据长度码，CAN2.0 设定范围为 0~8，对应数据长度为 0~8Byte；CAN FD 设定范围为 0~15，对应数据长度为 0~64Byte。 表 36-14 DLC 控制位 <table><tr><th>Frame Type</th><th>DLC（二进制）</th><th>Payload in Bytes</th></tr><tr><td rowspan="2">CAN2.0</td><td>0b0000~0b1000</td><td>0~8</td></tr><tr><td>0b1001~0b1111</td><td>8</td></tr><tr><td rowspan="8">CAN FD</td><td>0b0000~0b1000</td><td>0~8</td></tr><tr><td>0b1001</td><td>12</td></tr><tr><td>0b1010</td><td>16</td></tr><tr><td>0b1011</td><td>20</td></tr><tr><td>0b1100</td><td>24</td></tr><tr><td>0b1101</td><td>32</td></tr><tr><td>0b1110</td><td>48</td></tr><tr><td>0b1111</td><td>64</td></tr></table>	Frame Type	DLC（二进制）	Payload in Bytes	CAN2.0	0b0000~0b1000	0~8	0b1001~0b1111	8	CAN FD	0b0000~0b1000	0~8	0b1001	12	0b1010	16	0b1011	20	0b1100	24	0b1101	32	0b1110	48	0b1111
Frame Type	DLC（二进制）	Payload in Bytes																								
CAN2.0	0b0000~0b1000	0~8																								
	0b1001~0b1111	8																								
CAN FD	0b0000~0b1000	0~8																								
	0b1001	12																								
	0b1010	16																								
	0b1011	20																								
	0b1100	24																								
	0b1101	32																								
	0b1110	48																								
	0b1111	64																								
状态位	KOER	同 EALCAP.KOER																								
	TX	回环模式中接收自己发送的数据时此位置 1																								

36.4.3 CAN 发送 BUF 寄存器 (CAN_TBUF)

偏移地址 (Offset) : 0x50~0x97

复位值 (Reset) : 0xFFFFFFFF

TBUF 寄存器指向下一个空的 CAN 发送 BUF SLOT，TBUF 寄存器可以按照任意顺序读取。通过软件将 TSNEXT 写 1 来标记对应的 TBUF SLOT 已经写入数据，从而指向下一个 TBUF SLOT。

TBUF 只能 WORD 访问。

CAN 发送邮箱的数据格式如下：

表 36-15 标准格式 CAN 发送邮箱格式

地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	功能
TBUF	ID[7:0]								ID
TBUF+1	Reserved					ID[10:8]			ID
TBUF+2	Reserved								ID
TBUF+3	Reserved								ID
TBUF+4	IDE=0	RTR	FDF	BRS	DLC[3:0]				Control
TBUF+5	Reserved								-
TBUF+6	Reserved								-
TBUF+7	Reserved								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
TBUF+70	DATA63								Data
TBUF+71	DATA64								Data

表 36-16 扩展格式 CAN 发送邮箱格式

地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	功能
TBUF	ID[7:0]								ID
TBUF+1	ID[15:8]								ID
TBUF+2	ID[23:16]								ID
TBUF+3	Reserved			ID[28:24]					ID
TBUF+4	IDE=1	RTR	FDF	BRS	DLC[3:0]				Control
TBUF+5	Reserved								-
TBUF+6	Reserved								-
TBUF+7	Reserved								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
TBUF+70	DATA63								Data
TBUF+71	DATA64								Data

控制位含义请参见“CAN 接收 BUF 寄存器”章节相关描述。

36.4.4 CAN 配置和状态寄存器 (CAN_CFG_STAT)

Offset	Bit Position							
0xA0	7	6	5	4	3	2	1	0
Reset	0x80							
Name	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF
Access	RW	RW	RW	RW	RW	RO	RO	RW

位/位域	标记	位名	功能描述	读写
7	RESET	复位请求	0b0: 不请求局部复位 0b1: 请求局部复位 部分寄存器只能在 RESET=1 时进行写操作, 具体请参见 软件复位功能 。  说明 当 RESET=0 后需要 11 个 CAN bit times 该节点才能参与通信。	RW
6	LBME	外部回环模式使能	0b0: 禁止 0b1: 使能  说明 通信中禁止设定该位。	RW
5	LBMI	内部回环模式使能	0b0: 禁止 0b1: 使能  说明 通信中禁止设定该位。	RW
4	TPSS	PTB 单次传输模式	0b0: 禁止 PTB 单次传输模式 0b1: 使能 PTB 单次传输模式	RW
3	TSSS	STB 单次传输模式	0b0: 禁止 STB 单次传输模式 0b1: 使能 STB 单次传输模式	RW
2	RACTIVE	接收中状态信号	0b0: 非接收中 0b1: 接收中	RO
1	TACTIVE	发送中状态信号	0b0: 非发送中 0b1: 发送中	RO
0	BUSOFF	总线关闭状态	0b0: 总线有效状态 0b1: 总线关闭状态  说明 写 1 可以清零 TECNT 和 RECNT 寄存器, 但仅限用于调试用途。	RW

36.4.5 CAN 命令寄存器 (CAN_TCMD)

Offset	Bit Position							
0xA1	7	6	5	4	3	2	1	0
Reset	0x00							
Name	TBSEL	LOM	Reserved	TPE	TPA	TSONE	TSALL	TSA
Access	RW	RW		RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	TBSEL	发送 BUF 选择位	0b0: PTB 0b1: STB 当 TTEN=1&TTTBM=1 时, TBSEL 被复位成复位值。  说明 写 TBUF 寄存器或者 TSNEXT 位时, 此位需要保持定值。	RW
6	LOM	静默模式使能位	0b0: 禁止静默模式 0b1: 使能静默模式 LOM=1&LBME=0 时禁止发送。 LOM=1&LBME=1 时禁止应答相应接收到的帧以及错误帧, 但可以发送数据。  说明 通信中禁止设定该位。	RW
5	Reserved	保留	-	-
4	TPE	PTB 发送使能位	0b0: 禁止 PTB 发送 0b1: 使能 PTB 发送 此位使能后, PTB 中的 Mailbox 将在下一个可以发送的位置被发送。已经开始的 STB 发送将继续, 但是下一个等待的 STB 发送会被延迟到 PTB 发送完成后再进行。 该位写 1 后将保持为 1 直到 PTB 发送完成或者通过 TPA 取消发送。软件不能通过写 0 清除该位。 以下情况 TPE 被硬件复位成复位值: - RESET=1 - BUSOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTBM=1	RW

位/位域	标记	位名	功能描述	读写
3	TPA	PTB 发送取消位	● 0b0: 不取消 ● 0b1: 取消已经通过 TPE 置 1 请求但还未开始的 PTB 发送 该位软件写 1 但是通过硬件清零。通过写 1 可以清零 TPE 位, 因此和 TPE 同时写 1。 以下情况 TPE 被硬件复位成复位值: ● RESET=1 ● BUSOFF=1 ● TTEN=1&TTTBM=1	RW
2	TSONE	发送一帧 STB 数据设定位	● 0b0: 不发送 ● 0b1: 发送一帧 STB 数据 FIFO 模式中, 发送最早写入的数据, 优先级模式里发送最高优先级的数据。 该位写 1 后将保持为 1 直到 STB 发送完成或者通过 TSA 取消发送。软件不能通过写 0 清除该位。 以下情况 TSONE 被硬件复位成复位值: ● RESET=1 ● BUFOFF=1 ● LOM=1&LBME=0 ● TTEN=1&TTTBM=1	RW
1	TSALL	发送所有的 STB 数据设定位	● 0b0: 不发送 ● 0b1: 发送 STB 中所有的数据 该位写 1 后将保持为 1 直到 STB 发送完成或者通过 TSA 取消发送。软件不能通过写 0 清除该位。 以下情况 TSALL 被硬件复位成复位值: ● RESET=1 ● BUSOFF=1 ● LOM=1&LBME=0 ● TTEN=1&TTTBM=1	RW
0	TSA	STB 发送取消位	● 0b0: 不取消 ● 0b1: 取消已经通过 TSONE 或者 TSALL 置 1 请求但还未开始的 STB 发送 该位通过软件写 1 但是通过硬件清零。写 1 可以清零 TSONE 或者 TSALL 位。 以下情况 TSA 被硬件复位成复位值: ● RESET=1 ● BUSOFF=1	RW

36.4.6 CAN 发送控制寄存器 (CAN_TCTRL)

Offset	Bit Position							
0xA2	7	6	5	4	3	2	1	0
Reset	0x90							
Name	FD_ISO	TSNEXT	TSMODE	TTTBM	Reserved		TSSTAT	
Access	RW	RW	RW	RW			RO	

位/位域	标记	位名	功能描述	读写
7	FD_ISO	CAN FD ISO 模式	0b0: Bosch CAN FD 模式 0b1: 11898-1:2015 CAN FD 模式  说明 仅能在 CAN_CFG_STAT.RESET=1 时写入该位。	RW
6	TSNEXT	下一个 STB SLOT	0b0: 无动作 0b1: 当前 STB SLOT 已填充, 指向下一个 SLOT 应用程序将 TBUF 中的数据写完后, 通过置位 TSNEXT 位标识当前 STB SLOT 已经被填充, 从而硬件将 TBUF 指向下一个 STB SLOT。 被 TSNEXT 位标识的 STB SLOT 中的数据可以通过 TSONE 或者 TSALL 位发送。 该位通过应用程序写 1, 硬件清零。 所有的 STB SLOT 被填满后, TSNEXT 保持为 1 直到有 STB SLOT 被释放。  说明 TTCAN 模式时该位固定为 0。	RW
5	TSMODE	STB 发送模式	0b0: FIFO 模式 0b1: 优先级模式 FIFO 模式根据数据帧写入的先后顺序发送。 优先级模式根据 ID 自动判断, ID 越小, 优先级越高。 无论何种模式, PTB 具有最高的优先级。  说明 TSMODE 位只能在 STB 空时设定。	RW
4	TTTBM	TTCAN BUF 模式	0b0: TSMODE 决定, PTB 和 STB 0b1: 通过 TBPTR 和 TTPTR 设定 TTCAN 模式时, 只需要接收时间戳功能时, 该位可以设置为 0, 通过 TSMODE 决定使用 PTB 还是 STB。  说明 TSMODE 位只能在 STB 空时设定。 TTEN=0 时, TTTBM 被忽略。	RW
3:2	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
1:0	TSSTAT	STB 状态	TTEN=0 或 TTEN=1&TTTBM=0 情况下： ● 0b00: STB 空 ● 0b01: STB 小于等于半满 ● 0b10: STB 大于半满 ● 0b11: STB 满 TTEN=1 且 TTTBM=1 情况下： ● 0b00: PTB 和 STB 空 ● 0b01: PTB 和 STB 非满 ● 0b10: Reserved ● 0b11: PTB 和 STB 满	RO

36.4.7 CAN 接收控制寄存器 (CAN_RCTRL)

Offset	Bit Position							
0xA3	7	6	5	4	3	2	1	0
Reset	0x00							
Name	SACK	ROM	ROV	RREL	RBALL	Reserved	RSTAT	
Access	RW	RW	RO	RW	RW		RO	

位/位域	标记	位名	功能描述	读写
7	SACK	自应答	● 0b0: 无自应答 ● 0b1: LBME=1 时, 使能自应答功能	RW
6	ROM	接收 BUF 上溢模式设定位	● 0b0: 最早接收到的数据被覆盖 ● 0b1: 新接收到的数据不被存储	RW
5	ROV	接收 BUF 上溢标志位	● 0b0: 无上溢 ● 0b1: 上溢, 最少有一个数据丢失通过写 RREL 为 1 清零。	RO
4	RREL	释放接收 BUF	● 0b0: 不释放 ● 0b1: 表示该接收 BUF 已经被读取过, RBUF 寄存器指向下一个 RB SLOT	RW
3	RBALL	接收 BUF 数据存储所有的数据帧	● 0b0: 正常模式 ● 0b1: 存储所有的数据包括有错误的数据	RW
2	Reserved	保留	-	-
1:0	RSTAT	接收 BUF 状态	● 0b00: RBUF 空 ● 0b01: RBUF 非空但小于 AFWL 编程值 ● 0b10: RBUF 大于等于 AFWL 编程值但未满 ● 0b11: 满 (上溢时保持此值)	RO

36.4.8 CAN 接收和发送中断使能寄存器 (CAN_RTIE)

Offset	Bit Position							
0xA4	7	6	5	4	3	2	1	0
Reset	0xFE							
Name	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF
Access	RW	RW	RW	RW	RW	RW	RW	RO

位/位域	标记	位名	功能描述	读写
7	RIE	接收中断使能	0b0: 禁止 0b1: 使能	RW
6	ROIE	接收上溢中断使能	0b0: 禁止 0b1: 使能	RW
5	RFIE	接收 BUF 满中断使能	0b0: 禁止 0b1: 使能	RW
4	RAFIE	接收 BUF 将满中断使能	0b0: 禁止 0b1: 使能	RW
3	TPIE	PTB 发送中断使能	0b0: 禁止 0b1: 使能	RW
2	TSIE	STB 发送中断使能	0b0: 禁止 0b1: 使能	RW
1	EIE	错误中断使能	0b0: 禁止 0b1: 使能	RW
0	TSFF	发送 BUF 满标志	TTEN=0 or TTTBM=0: STB 满标志 (Transmit Secondary buffer Full Flag) 0b0: STB SLOT 没有被全部填充 0b1: STB SLOT 被全部填充 TTEN=1 and TTTBM=1: TB 满标志 (Transmit buffer Full Flag) 0b0: TBPTR 选择的发送 BUF 没有被全部填充 0b1: TBPTR 选择的发送 BUF 被全部填充	RO

36.4.9 CAN 接收和发送中断标志寄存器 (CAN_RTIF)

Offset	Bit Position							
0xA5	7	6	5	4	3	2	1	0
Reset	0x00							
Name	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF
Access	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	RIF	接收中断标志	写 1 清零，写 0 无效 读取： ● 0b0：未收到数据帧 ● 0b1：接收到有效的数据帧或者远程帧	RW
6	ROIF	接收上溢中断标志	写 1 清零，写 0 无效 读取： ● 0b0：无 RB 被覆盖 (overwrite) ● 0b1：RB 至少有一个被覆盖  说明 上溢时 ROIF 和 RFIF 同时置 1。	RW
5	RFIF	接收 BUF 满中断标志	写 1 清零，写 0 无效 读取： ● 0b0：RB FIFO 未满 ● 0b1：RB FIFO 满	RW
4	RAFIF	接收 BUF 将满中断标志	写 1 清零，写 0 无效 读取： ● 0b0：被填充的 RB SLOT 数目小于 AFWL 设定值 ● 0b1：被填充的 RB SLOT 数目大于等于 AFWL 设定值	RW
3	TPIF	PTB 发送中断标志	写 1 清零，写 0 无效 读取： ● 0b0：没有 PTB 发送完成 ● 0b1：请求的 PTB 发送成功完成  说明 TTCAN 模式时，TPIF 无效，仅使用 TSIF 标志。	RW
2	TSIF	STB 发送中断标志	写 1 清零，写 0 无效 读取： ● 0b0：没有 STB 发送完成 ● 0b1：请求的 STB 发送成功完成  说明 TTCAN 模式时，TPIF 无效，仅使用 TSIF 标志。	RW

位/位域	标记	位名	功能描述	读写
1	EIF	错误中断标志	写 1 清零，写 0 无效 读取： ● 0b0：BUSOFF 位未发生变化，或者错误计数器的值与 ERROR warning limit 设定值的相对关系未发生变化 ● 0b1：BUSOFF 位发生变化，或者错误计数器的值与 ERROR warning limit 设定值的相对关系发生变化。比如错误计数器的值从小于设定值变为大于设定值，或者从大于设定值变为小于设定值	RW
0	AIF	取消发送中断标志	写 1 清零，写 0 无效 读取： ● 0b0：未取消发送数据 ● 0b1：通过 TPA 和 TSA 请求的发送消息被成功取消	RW

36.4.10 CAN 错误中断使能和标志寄存器 (CAN_ERRINT)

Offset	Bit Position							
0xA6	7	6	5	4	3	2	1	0
Reset	0x00							
Name	EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF
Access	RO	RO	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7	EWARN	到达设定的 ERROR WARNING LIMIT	0b0: RECNT 或者 TECNT 小于 EWL 设定值 0b1: RECNT 或者 TECNT 大于等于 EWL 设定值	RO
6	EPASS	错误被动	0b0: 节点是主动错误节点 0b1: 节点是被动错误节点	RO
5	EPIE	错误被动中断使能	0b0: 禁止 0b1: 使能	RW
4	EPIF	错误被动中断标志	写 1 清零, 写 0 无效 读取: 0b0: 未发生错误主动到错误被动或者错误被动到错误主动的变化 0b1: 发生错误主动到错误被动或者错误被动到错误主动的变化	RW
3	ALIE	仲裁失败中断使能	0b0: 禁止 0b1: 使能	RW
2	ALIF	仲裁失败中断标志	写 1 清零, 写 0 无效 读取: 0b0: 仲裁成功 0b1: 仲裁失败	RW
1	BEIE	总线错误中断使能	0b0: 禁止 0b1: 使能	RW
0	BEIF	总线错误中断标志	写 1 清零, 写 0 无效 读取: 0b0: 无总线错误 0b1: 总线错误	RW

36.4.11 CAN Slow 位时序寄存器 (CAN_SBT)

Offset	Bit Position																																	
0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x01020203																																	
Name	S_PRESC								Reserved	S_SJW								Reserved	S_SEG_2								S_SEG_1							
Access	RW									RW									RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	S_PRESC	预分频设定	此寄存器设定值时将模块通信时钟设定为 (S_PRESC+1) 分频作为 TQ	RW
23	Reserved	保留	-	-
22:16	S_SJW	再同步补偿宽度时间设定	再同步补偿宽度时间 = (S_SJW + 1) * TQ	RW
15	Reserved	保留	-	-
14:8	S_SEG_2	位段 2 时间设定	位段 2 时间 = (S_SEG_2 + 1) * TQ	RW
7:0	S_SEG_1	位段 1 时间设定	位段 1 时间 = (S_SEG_1 + 2) * TQ	RW

36.4.12 CAN Fast 位时序寄存器 (CAN_FBT)

Offset	Bit Position																															
0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x01020203																															
Name	F_PRESC								Reserved				F_SJW				Reserved				F_SEG_2				Reserved				F_SEG_1			
Access	RW												RW								RW								RW			

位/位域	标记	位名	功能描述	读写
31:24	F_PRESC	预分频设定	此寄存器设定值时将模块通信时钟设定为 (F_PRESC+1) 分频作为 TQ	RW
23:20	Reserved	保留	-	-
19:16	F_SJW	再同步补偿宽度时间设定	再同步补偿宽度时间 = (F_SJW + 1) * TQ	RW
15:12	Reserved	保留	-	-
11:8	F_SEG_2	位段 2 时间设定	位段 2 时间 = (F_SEG_2 + 1) * TQ	RW
7:5	Reserved	保留	-	-
4:0	F_SEG_1	位段 1 时间设定	位段 1 时间 = (F_SEG_1 + 2) * TQ	RW

36.4.13 CAN 发送器延迟补偿寄存器 (CAN_TDC)

Offset	Bit Position							
0xB1	7	6	5	4	3	2	1	0
Reset	0x00							
Name	TDCEN		SSPOFF					
Access	RW		RW					

位/位域	标记	位名	功能描述	读写
7	TDCEN	发送器延迟补偿功能使能位	用于 CAN FD BRS=1 的情况： ● 0b0：禁止 ● 0b1：使能	RW
6:0	SSPOFF		设定单位为 TQ	RW

36.4.14 CAN 错误和仲裁失败捕捉寄存器 (CAN_EALCAP)

Offset	Bit Position							
0xB0	7	6	5	4	3	2	1	0
Reset	0x00							
Name	KOER				ALC			
Access	RO				RO			

位/位域	标记	位名	功能描述	读写
7:5	KOER	错误类别	● 0b000：无错误 ● 0b001：位错误 ● 0b010：形式错误 ● 0b011：填充错误 ● 0b100：应答错误 ● 0b101：CRC 错误 ● 0b110：其他错误 ● 0b111：Reserved  说明 有错误时 KOER 位更新，正常发送接收时 KOER 位保持不变。	RO
4:0	ALC	仲裁失败位置捕捉	仲裁失败时 ALC 记录一帧数据中仲裁失败时的位置	RO

36.4.15 CAN 警告限定寄存器 (CAN_LIMIT)

Offset	Bit Position							
0xA7	7	6	5	4	3	2	1	0
Reset	0x1B							
Name	AFWL				EWL			
Access	RW				RW			

位/位域	标记	位名	功能描述	读写
7:4	AFWL	接收 BUF 将满 Warning Limit	设定值范围为 1~8。 AFWL=0 无意义, 当做 AFWL=1 处理。	RW
3:0	EWL	Error Waring Limit 编程值	Error Waring Limit = (EWL + 1) * 8 该寄存器设定值影响 EIF 标志。	RW

36.4.16 CAN 接收错误计数器寄存器 (CAN_RECNT)

Offset	Bit Position							
0xB2	7	6	5	4	3	2	1	0
Reset	0x00							
Name	RECNT							
Access	RO							

位/位域	标记	位名	功能描述	读写
7:0	RECNT	接收错误计数器	接收错误计数器根据 CAN 协议规定的错误计数增加或者减少。该计数器不存在上溢, 255 为最大值	RO

36.4.17 CAN 发送错误计数器寄存器 (CAN_TECNT)

Offset	Bit Position							
0xB3	7	6	5	4	3	2	1	0
Reset	0x00							
Name	TECNT							
Access	RO							

位/位域	标记	位名	功能描述	读写
7:0	TECNT	接收错误计数器	发送错误计数器根据 CAN 协议规定的错误计数增加或者减少。该计数器不存在上溢, 255 为最大值  说明 进入 Bus_Off 时, TECNT 可能溢出, 因此可能会保持一个较小的值。	RO

36.4.18 CAN 筛选器组控制寄存器 (CAN_ACFCTRL)

Offset	Bit Position							
0xB4	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved		SELMASK	Reserved	ACFADR			
Access			RW		RW			

位/位域	标记	位名	功能描述	读写
7:6	Reserved	保留	-	-
5	SELMASK	选择筛选器的屏蔽寄存器	0b0: ACF 指向筛选器 ID 寄存器 0b1: ACF 指向筛选器 MASK 寄存器 通过 ACFADR 选择具体的筛选寄存器组。	RW
4	Reserved	保留	-	-
3:0	ACFADR	筛选器地址	ACFADR 指向具体的筛选器，通过 SELMASK 去区分 ID 和 MASK: 0b0000: 指向 ACF_1 0b0001: 指向 ACF_2 0b0010: 指向 ACF_3 0b0011: 指向 ACF_4 0b0100: 指向 ACF_5 0b0101: 指向 ACF_6 0b0110: 指向 ACF_7 0b0111: 指向 ACF_8 0b1000: 指向 ACF_9 0b1001: 指向 ACF_10 0b1010: 指向 ACF_11 0b1011: 指向 ACF_12 0b1100: 指向 ACF_13 0b1101: 指向 ACF_14 0b1110: 指向 ACF_15 0b1111: 指向 ACF_16	RW

36.4.19 CAN 筛选器组使能寄存器 (CAN_ACFEN)

Offset	Bit Position															
0xB6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0001															
Name	AE_16	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	AE_16	ACF_16 使能	0b0: 禁止 0b1: 使能	RW
14	AE_15	ACF_15 使能	0b0: 禁止 0b1: 使能	RW
13	AE_14	ACF_14 使能	0b0: 禁止 0b1: 使能	RW
12	AE_13	ACF_13 使能	0b0: 禁止 0b1: 使能	RW
11	AE_12	ACF_12 使能	0b0: 禁止 0b1: 使能	RW
10	AE_11	ACF_11 使能	0b0: 禁止 0b1: 使能	RW
9	AE_10	ACF_10 使能	0b0: 禁止 0b1: 使能	RW
8	AE_9	ACF_9 使能	0b0: 禁止 0b1: 使能	RW
7	AE_8	ACF_8 使能	0b0: 禁止 0b1: 使能	RW
6	AE_7	ACF_7 使能	0b0: 禁止 0b1: 使能	RW
5	AE_6	ACF_6 使能	0b0: 禁止 0b1: 使能	RW
4	AE_5	ACF_5 使能	0b0: 禁止 0b1: 使能	RW
3	AE_4	ACF_4 使能	0b0: 禁止 0b1: 使能	RW
2	AE_3	ACF_3 使能	0b0: 禁止 0b1: 使能	RW
1	AE_2	ACF_2 使能	0b0: 禁止 0b1: 使能	RW
0	AE_1	ACF_1 使能	0b0: 禁止 0b1: 使能	RW

36.4.20 CAN 筛选器组 code 和 mask 寄存器 (CAN_ACF)

Offset	Bit Position																															
0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name				ACODE/AMASK																												
	Reserved	AIDEE	AIDE																													
Access		RW	RW	RW																												

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30	AIDEE	IDE 位比较使能	0b0: 筛选器接收标准格式和扩展格式帧 0b1: 筛选器接收 AIDE 位定义的标准格式或者扩展格式帧  说明 仅在 SELMASK=1 时有效。	RW
29	AIDE	IDE 位 MASK	0b0: 筛选器仅接收标准格式 0b1: 筛选器仅接收扩展格式	RW
28:0	ACODE/ AMASK	筛选器 CODE/筛选器 MASK	ACODE: - 通过 ACFADR 指向具体的筛选器。 - SELMASK=0 时表示筛选器的 CODE。 - 标准格式时使用 Bit10~Bit0, 扩展格式时使用 Bit28~Bit0。 AMASK: - 通过 ACFADR 指向具体的筛选器。 - SELMASK=1 时表示筛选器的 MASK。 - 标准格式时使用 Bit10~Bit0, 扩展格式时使用 Bit28~Bit0。	RW

36.4.21 TTCAN TB slot 指针寄存器 (CAN_TBSLOT)

Offset	Bit Position							
0xBE	7	6	5	4	3	2	1	0
Reset	0x00							
Name	TBE	TBF	Reserved			TBPTR		
Access	RW	RW				RW		

位/位域	标记	位名	功能描述	读写
7	TBE	设置 TB 为空	0b0: 无操作 0b1: 被 TBPTR 选择的 SLOT 被标记为空 当 SLOT 被标记为空并且 TSFF=0 时, TBE 自动复位为 0。 如果设定此位为 1 时, 被选定的 SLOT 中存在数据正在发送状态则 TBE=1, 则等到发送完成、发送错误或者发送取消后 TBE 复位为 0。 TBE 优先级高于 TBF。	RW
6	TBF	设置 TB 为已填充	0b0: 无操作 0b1: 被 TBPTR 选择的 SLOT 被标记为已填充 当 SLOT 被标记为已填充并且 TSFF=1 时, TBE 自动复位为 0。	RW
5:3	Reserved	保留	-	-
2:0	TBPTR	TB SLOT 指针	0b000: 指向 PTB 0b001: 指向 STB SLOT1 0b010: 指向 STB SLOT2 0b011: 指向 STB SLOT3 - 其他: 禁止设置 被指向的 TB SLOT 可以通过 TBUF 进行读写访问, 并且可以通过 TBE 和 TBF 来标记是否已经被填充。 TTCAN 模式时, TBSEL 和 TSNEXT 寄存器无效。	RW
			 说明 仅可以在 TSFF=0 时对该位进行写操作。	

36.4.22 TTCAN 时间触发配置寄存器 (CAN_TTCFG)

Offset	Bit Position							
0xBF	7	6	5	4	3	2	1	0
Reset	0x90							
Name	WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC		TTEN
Access	RW	RW	RW	RW	RW	RW		RW

位/位域	标记	位名	功能描述	读写
7	WTIE	触发看门中断使能	0b0: 禁止 0b1: 使能	RW
6	WTIF	触发看门中断标志	写 1 清零, 写 0 无效 读取: 0b0: 未产生标志 0b1: 产生标志 当 CYCLE COUNT 值=TT_WTRIG 设定值时且 WTIE=1 时, WTIF 置位。	RW
5	TEIF	触发错误中断标志	写 1 清零, 写 0 无效 读取: 0b0: 未产生标志 0b1: 产生标志 TT_TTIG 设定值小于实际的 CYCLE_TIME 时, TEIF 置位。	RW
4	TTIE	时间触发中断使能	0b0: 禁止 0b1: 使能	RW
3	TTIF	时间触发中断标志	写 1 清零, 写 0 无效 读取: 0b0: 未产生标志 0b1: 产生标志 当 CYCLE COUNT 值=TT_TRIG 设定值时且 TTIE=1 时, TTIF 置位。 如果 TT_TRIG 没有更新, 则 TTIF 只置位 1 次, 下一个基本 CYCLE 不置位。	RW
2:1	T_PRESC	TTCAN 计数器预分频	0b00: SBT 寄存器设定的位时间的 1 分频 0b01: SBT 寄存器设定的位时间的 2 分频 0b10: SBT 寄存器设定的位时间的 4 分频 0b11: SBT 寄存器设定的位时间的 8 分频  说明 T_PRESC 可在 TTEN=0 时进行写操作或者写 TTEN=1 时同时操作。	RW
0	TTEN	TTCAN 使能	0b0: 禁止 0b1: 使能 TTCAN, 计数器开始计数	RW

36.4.23 TTCAN 参考消息寄存器 (CAN_REF_MSG)

Offset	Bit Position																															
0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	REF_IDE	Reserved	REF_ID																													
Access	RW																															

位/位域	标记	位名	功能描述	读写
31	REF_IDE	参考消息的 IDE 位	0b0: 标准格式 0b1: 扩展格式	RW
30:29	Reserved	保留	-	-
28:0	REF_ID	参考消息的 ID 位	- REF_IDE=0: REF_ID[10:0]有效 - REF_IDE=1: REF_ID[28:0]有效 REF_ID 用于检测参考消息，适用于发送和接收。检测到参考消息后，当前帧的 Sync_Mark 则变成 Ref_Mark。 REF_ID[2:0]固定为 0，并不检查其值，这样最多可以支持 8 个潜在的 time master。 当 REF_MSG 的最高字节写操作后，则需要等待 6 个 CAN 时钟周期以完成 REF_MSG 向 CAN 时钟域的传递。	RW

36.4.24 TTCAN 触发配置寄存器 (CAN_TRG_CFG)

Offset	Bit Position															
0xC4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TEW				Reserved	TTYPE				Reserved				TTPTR		
Access	RW					RW								RW		

位/位域	标记	位名	功能描述	读写
15:12	TEW	发送使能窗口	用于 TTCAN 的单次发送触发模式 (Single Shot Transmit Trigger)，可以设定 TEW+1 个 cycle time 的窗口，发送仅在此窗口内被允许	RW
11	Reserved	保留	-	-
10:8	TTYPE	触发类型	0b000: 立即触发 (Immediate Trigger for immediate transmission) 0b001: 时间触发 (Time Trigger for receive triggers) 0b010: 单次发送触发 (Single Shot Transmit Trigger for exclusive time windows) 0b011: 发送开始触发 (Transmit Start Trigger for merged arbitrating time windows) 0b100: 发送停止触发 (Transmit Stop Trigger for merged arbitrating time windows) - 其他: 禁止设置 触发时间通过 TT_TRIG 寄存器设定，TB Slot 通过 TTPTR 选择。	RW
7:3	Reserved	保留	-	-
2:0	TTPTR	发送触发器 TB slot 指针	0b000: 指向 PTB 0b001: 指向 STB SLOT1 0b010: 指向 STB SLOT2 0b011: 指向 STB SLOT3 - 其他: 禁止设置 如果指向的 TB SLOT 被标记为空，当到达触发时间后，TEIF 置位。	RW

36.4.25 TTCAN 触发时间寄存器 (CAN_TT_TRIG)

Offset	Bit Position															
0xC6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	TT_TRIG															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	TT_TRIG	触发时间	用于指定触发器的 cycle time，对于发送触发器来说发送 SOF 时间大约是 TT_TRIG 设定值+1。 当 TT_TRIG 的最高字节写操作后，TT_TRIG 值开始向 CAN 时钟域的传递。因此如果 BYTE 操作，需先写低字节再写高字节。	RW

36.4.26 TTCAN 触发看门时间寄存器 (CAN_TT_WTRIG)

Offset	Bit Position															
0xC8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFF															
Name	TT_WTRIG															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	TT_WTRIG	触发时间	用于指定看门触发器的 cycle time。 当 TT_WTRIG 的最高字节写操作后，TT_WTRIG 值开始向 CAN 时钟域的传递。因此如果 BYTE 操作，需先写低字节再写高字节。	RW

36.4.27 CAN RAM 校验控制寄存器 (CAN_RAM_CKCR)

Offset	Bit Position																															
0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															ECCMOD
Access																																RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1:0	ECCMOD	CAN RAM 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断	RW

36.4.28 CAN RAM 校验状态寄存器 (CAN_RAM_CKSR)

Offset	Bit Position																															
0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														RAM_2ERR	RAM_1ERR
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RAM_2ERR	CAN RAM 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: 0b0: 无 2-bit ECC 错误发生 0b1: 有 2-bit ECC 错误发生	RW
0	RAM_1ERR	CAN RAM 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: 0b0: 无 1-bit ECC 错误发生 0b1: 有 1-bit ECC 错误发生	RW

36.4.29 CAN RAM 错误注入使能寄存器 (CAN_RAM_EIEN)

Offset	Bit Position																															
0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 禁止 0b1: 使能	RW

36.4.30 CAN RAM 错误注入位使能寄存器 0 (CAN_RAM_EIBIT0)

Offset	Bit Position																															
0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 CAN RAM 输出数据的 31~0 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

36.4.31 CAN RAM 错误注入位使能寄存器 1 (CAN_RAM_EIBIT1)

Offset	Bit Position																															
0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 CAN RAM 输出数据的 38~32 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

36.4.32 CAN RAM 校验错误地址寄存器 (CAN_RAM_ECCERRADDR)

Offset	Bit Position																															
0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ECCERRADDR											
Access																																

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10:0	ECCERRADDR	校验错误地址	记录 CAN RAM 发生 1-bit 或 2-bit ECC 错误时的访问地址	RO

36.4.33 CAN 全局控制寄存器 (CAN_SYCTLREG)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																														CAN2FDE	CAN1FDE
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	CAN2FDE	CAN2 FD 功能使能	0b0: 禁止 0b1: 使能	RW
0	CAN1FDE	CAN1 FD 功能使能	0b0: 禁止 0b1: 使能	RW

36.5 注意事项

36.5.1 CAN 总线抗干扰措施

CAN 总线广泛应用于汽车、工业控制等行业，如果 CAN 应用现场电磁环境比较恶劣，存在电路不平衡性、空间电磁场、电网进线等因素，会导致 CAN 总线因辐射、传导干扰而产生大量通信噪声，致使总线错误帧增加、重发频繁，正确数据不能及时到达等情况，严重影响数据通信质量。所以实际应用中应该致力于消除噪声干扰，保障 CAN 总线网络稳定工作。

以下是几类常用的 CAN 总线抗干扰措施（包括但不限于）：

- 增加 CAN 总线接口电气隔离
- 共收发器的信号地
- 使用屏蔽双绞线缆并正确接地
- 提高 CAN 传输线双绞程度
- 增加信号保护器
- 改进网络拓扑
- 应用层软件抗干扰机制

36.5.2 CAN 控制器噪声制约

在 CAN 总线网络中应确保通信的位时间满足标准协议的要求，若引入不满足位时间宽度的噪声干扰，可能引起 CAN 控制器异常动作。

37 控制器局域网 (MCAN)

37.1 简介

控制器局域网络 (CAN) 子系统由两个 CAN 模块、一个共享消息 RAM 存储器组成。关于这三个组成部分的基地址, 请参见《存储器映射 (Memory Mapping)》章节。

两个 CAN 模块 (MCAN1 和 MCAN2) 均符合 ISO 11898-1:2015 (CAN 协议规范第 2.0 版 A、B 部分) 和 CAN FD 协议规范第 1.0 版 (CAN with Flexible Data-Rate Specification Version 1.0)。

4KB 的消息 RAM 存储器可实现过滤器 (Rx Filter)、接收 FIFO (Rx FIFO)、接收缓冲区 (Rx Buffer)、发送事件 FIFO (Tx Event FIFO)、发送缓冲区 (Tx Buffer) 功能。该消息 RAM 在 MCAN1 和 MCAN2 模块之间共用。

37.2 主要特性

- 适用 ISO 11898-1:2015 标准
- CAN FD 最多支持收发 64 个数据字节
- 支持 CAN 错误日志
- 支持 AUTOSAR 和 SAE J1939
- 支持接收过滤功能
- 两个可配置的接收 FIFO
- 接收到高优先级消息时单独发出信号指示
- 最多 64 个专用接收缓冲区
- 最多 32 个专用发送缓冲区
- 可配置的发送 FIFO 或队列
- 可配置的发送事件 FIFO
- 主机 CPU 可直接访问消息 RAM
- MCAN1 和 MCAN2 两个模块共享同一个消息 RAM
- 支持可编程回环测试模式
- 支持可屏蔽模块中断
- 两个时钟域: CAN 内核时钟和 APB 总线接口时钟

37.3 功能说明

37.3.1 MCAN 系统框图

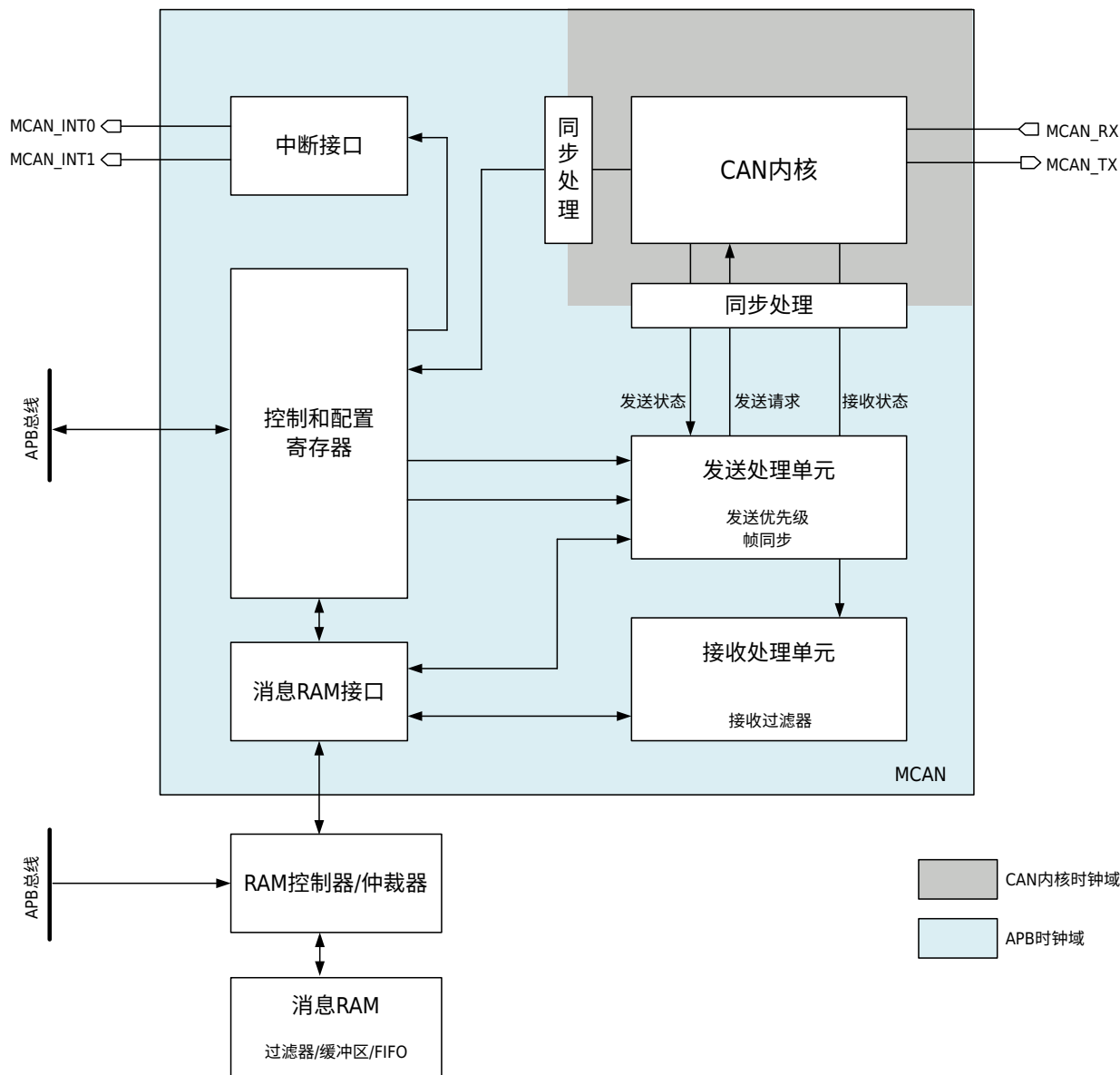


图 37-1 MCAN 系统框图

CAN 内核

CAN 内核包含协议控制器、接收与发送移位寄存器。它可处理 ISO 11898-1:2015 的所有协议功能，并支持 11 位标准 ID 和 29 位扩展 ID。

同步

同步模块将来自 APB 时钟域的信号同步到 CAN 内核时钟域，以及将来自 CAN 内核时钟域的信号同步到 APB 时钟域。

APB 时钟域的时钟为 MCAN 控制逻辑时钟，CAN 内核时钟域的时钟为 MCAN 通信时钟，请参见《时钟控制器（CMU）》章节“工作时钟规格”。为了保证 MCAN 通信的稳定性，控制逻辑时钟频率必须大于等于通信时钟频率。

发送处理单元

该处理单元控制消息从外部消息 RAM 到 CAN 内核的传输。最多可配置 32 个发送缓冲区进行发送。发送缓冲区可用作专用发送缓冲区、发送 FIFO（或队列），或二者的组合。发送事件 FIFO 将发送时间戳与对应的消息 ID 存储在一起。另外还支持取消发送。

接收处理单元

该处理单元控制接收到的消息从 CAN 内核到外部消息 RAM 的传输。接收处理单元支持两个接收 FIFO（每个 FIFO 的大小均可配置）和最多 64 个专用接收缓冲区，用于存储所有通过接收过滤的消息。与接收 FIFO 有所不同，专用接收缓冲区仅用于存储特定 ID 的消息。接收时间戳与其消息存储在一起。对于 11 位标准 ID，最多可定义 128 个过滤器；对于 29 位扩展 ID，最多可定义 64 个过滤器。

APB 接口

将 MCAN 连接至 APB 总线。

消息 RAM

接口通过 RAM 控制器和仲裁器将 MCAN 访问连接到外部 4KB 消息 RAM。

双中断线

MCAN 提供 MCAN_INT0 和 MCAN_INT1 两条中断线。对 MCAN_ILE.EINT0 和 MCAN_ILE.EINT1 位进行编程，可分别使能或禁止这两条中断线。

37.3.2 引脚说明

表 37-1 MCAN 引脚说明

引脚名	方向	功能描述
MCAN_RX	输入	MCAN 接收引脚
MCAN_TX	输出	MCAN 发送引脚

37.3.3 工作模式

37.3.3.1 软件初始化

软件初始化是通过置位 MCAN_CCCR.INIT 开始的。软件复位、硬件复位或进入 Bus_Off 会置位 MCAN_CCCR.INIT。当 MCAN_CCCR.INIT 被置位时，CAN 总线上传入和传出的消息都将停止，MCAN_TX 引脚的状态变为隐性（高电平）。错误逻辑管理单元（EML）的计数器保持不变。将 MCAN_CCCR.INIT 置位不会改变任何配置寄存器。将 MCAN_CCCR.INIT 清零可以完成软件初始化，随后，位流处理单元（BSP）等待总线上出现 11 个连续隐性位的序列（Bus_Idle），以此将其自身与 CAN 总线上的数据传输进行同步，然后才能参与总线活动并开始消息传输。

仅当 MCAN_CCCR.INIT 和 MCAN_CCCR.CCE 均置位时，才能访问 MCAN 配置寄存器。MCAN 配置寄存器如下表所示。

表 37-2 MCAN 配置寄存器列表

寄存器	说明
MCAN_DBTP	MCAN 数据位时间和预分频寄存器
MCAN_TEST	MCAN 测试寄存器
MCAN_RWD	MCAN RAM 看门狗寄存器
MCAN_CCCR	MCAN CC 控制寄存器

寄存器	说明
MCAN_NBTP	MCAN 标称位时间和预分频寄存器
MCAN_TSCC	MCAN 时间戳计数器配置寄存器
MCAN_TOCC	MCAN 超时计数器配置寄存器
MCAN_TDCR	MCAN 发送延迟补偿寄存器
MCAN_GFC	MCAN 全局过滤器配置寄存器
MCAN_SIDFC	MCAN 标准 ID 过滤器配置寄存器
MCAN_XIDFC	MCAN 扩展 ID 过滤器配置寄存器
MCAN_XIDAM	MCAN 扩展 ID 与掩码寄存器
MCAN_RXF0C	MCAN 接收 FIFO0 配置寄存器
MCAN_RXBC	MCAN 接收缓冲区配置寄存器
MCAN_RXF1C	MCAN 接收 FIFO1 配置寄存器
MCAN_RXESC	MCAN 接收缓冲区和 FIFO 元素大小配置寄存器
MCAN_TXBC	MCAN 发送缓冲区配置寄存器
MCAN_TXESC	MCAN 发送缓冲区元素大小配置寄存器
MCAN_TXEFC	MCAN 发送事件 FIFO 配置寄存器

MCAN_CCCR.CCE 仅在 MCAN_CCCR.INIT 置位时才能被置位或清零，在 MCAN_CCCR.INIT 清零时自动清零。

MCAN_CCCR.CCE 置位后，以下寄存器会被复位：

- MCAN_HPMS：高优先级消息状态
- MCAN_RXF0S：接收 FIFO 0 状态
- MCAN_RXF1S：接收 FIFO 1 状态
- MCAN_TXFQS：发送 FIFO/队列状态
- MCAN_TXBRP：发送缓冲区请求挂起
- MCAN_TXBTO：发送缓冲区发送发生
- MCAN_TXBCF：发送缓冲区取消完成
- MCAN_TXEFS：发送事件 FIFO 状态
- MCAN_TXBAR：发送缓冲区添加请求
- MCAN_TXBCR：发送缓冲区取消请求

仅当 MCAN_CCCR.INIT 和 MCAN_CCCR.CCE 均已置位时，主机才能将 MCAN_CCCR.TEST 和 MCAN_CCCR.MON 置位，主机可随时复位 MCAN_CCCR.TEST 和 MCAN_CCCR.MON。仅当 MCAN_CCCR.INIT 和 MCAN_CCCR.CCE 均已置位时，才能将 MCAN_CCCR.DAR 置位或清零。

37.3.3.2 正常工作

MCAN 完成初始化且 MCAN_CCCR.INIT 清零后，MCAN 会将其自身与 CAN 总线同步并准备好进行通信。接收消息时，通过接收过滤器接收到的消息（包含消息 ID 和 DLC）会存储到专用的接收缓冲区，或接收 FIFO0，或接收 FIFO1。

发送消息时，先初始化或更新专用发送缓冲区和（或）发送 FIFO（队列）。不支持在接收到远程帧后自动发送。

37.3.3.3 CAN FD 操作

CAN FD 帧传输有两种类型，第一种是不带比特率切换的 CAN FD 帧；第二种是带比特率切换的 CAN FD 帧，其控制场（Control field）、数据场（Data field）和 CRC 场（CRC field）的使用比帧开始和结束时更高的比特率传输。

经典 CAN（Classical CAN）标准帧中的 r0 位（IDE 位之后，标准帧格式中的保留位）、扩展帧中的 r1 位（RTR 位之后，扩展帧格式中的第一个保留位），在 CAN FD 帧格式中解码为 FDF 位。FDF 位为隐性（逻辑 1）表示 CAN FD 帧，为显性（逻辑 0）表示经典 CAN 帧。在 CAN FD 帧中，FDF 位之后的 res 和 BRS（Bit Rate Switch，比特率切换）两个位，决定是否切换 CAN FD 帧中的比特率，并通过 res 显性和 BRS 隐性指示已切换比特率。res 位隐性的编码保留，供将来协议扩展使用。如果 MCAN 接收到的帧中 FDF 位和 res 位都是隐性，则会通过将 MCAN_PSR.PXE 置位的方式发出协议异常事件。如果使能协议异常处理

（MCAN_CCCR.PXHD=0），MCAN 将在下一采样点将工作状态从接收（MCAN_PSR.ACT=0b10）变为同步（MCAN_PSR.ACT=0b00）。如果禁止协议异常处理（MCAN_CCCR.PXHD=1），MCAN 会将隐性 res 位当作格式错误处理，并将以错误帧进行响应。

CAN FD 模式通过编程 MCAN_CCCR.FDOE 来使能。如果 MCAN_CCCR.FDOE=1，则会使能 CAN FD 帧的发送和接收。经典 CAN 帧始终可进行发送和接收。是发送 CAN FD 帧还是经典 CAN 帧，可通过相应发送缓冲区元素中的 FDF 位配置。如果 MCAN_CCCR.FDOE=0，接收到的帧会被当作经典 CAN 帧，从而会在接收到 CAN FD 帧时发送错误帧。如果 CAN FD 模式已禁止，即使发送缓冲区元素的 FDF 位置 1，也不会发送 CAN FD 帧。仅当 MCAN_CCCR.INIT 和 MCAN_CCCR.CCE 均置位时，才能更改 MCAN_CCCR.FDOE 和 MCAN_CCCR.BRSE。

如果 MCAN_CCCR.FDOE=0，则会忽略 FDF 和 BRS 位的设置，帧以经典 CAN 格式发送。如果 MCAN_CCCR.FDOE=1 且 MCAN_CCCR.BRSE=0，则只会评估发送缓冲区元素的 FDF 位。如果 MCAN_CCCR.FDOE=1 且 MCAN_CCCR.BRSE=1，则会使能比特率切换发送 CAN FD 帧。所有 FDF 和 BRS 位置位的发送缓冲区元素均会以比特率切换的 CAN FD 格式发送。

建议仅当以下条件满足时，才在 CAN 操作期间切换模式：

- CAN FD 数据阶段（Data phase）的故障率显著高于 CAN FD 仲裁阶段（Arbitration phase）的故障率。在这种情况下，应禁止发送的 CAN FD 比特率切换。
- 系统启动期间，所有节点在被验证能够以 CAN FD 格式进行通信之前，都将发送经典 CAN 消息。如果通过验证，所有节点会切换为 CAN FD 操作。
- CAN 局部网络（CAN Partial Networking）中的唤醒消息必须以经典 CAN 格式发送。
- EOL 编程（End-of-line programming）时，并非所有节点都支持 CANFD，非 CANFD 的节点会保持在静默模式直到编程结束。随后，所有节点切换回经典 CAN 通信。

在 CANFD 格式中，DLC 代码 0 到 8 的编码（对应的数据字节长度）与经典 CAN 相同，代码 9 到 15 的编码与经典 CAN 不同，如下表所示。

表 37-3 MCAN 中的 DLC 编码

DLC 编码		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
数据字节数	经典 CAN	0	1	2	3	4	5	6	7	8	8	8	8	8	8	8	8
	CAN FD	0	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64

在 CAN FD 帧中，如果 BRS 位为隐性，则位时长将在 BRS（比特率切换）位后，在帧内切换。在 BRS 位之前，在 CAN FD 仲裁阶段（Arbitration phase），位时长按照寄存器 MCAN_NBTP（Nominal Bit Timing & Prescaler Register，标称位时长和预分频寄存器）的定义执行。在接下来的 CAN FD 数据阶段（Data

phase)，位时长按照寄存器 MCAN_DBTP (Data Bit Timing & Prescaler Register，数据位时长和预分频寄存器) 的定义执行。位时长在 CRC 分隔符处，或检测到错误时从数据阶段 (Data phase) 位时长切换回来，以先发生的事件为准。

CAN FD 数据阶段 (Data phase) 的最大可配置比特率取决于 CAN 通信时钟频率。举例来说，如果 CAN 通信时钟频率为 20MHz，最短可配置的位时间为 4 个时间片 (tq)，则数据阶段的比特率为 5Mbps。

在带比特率切换和不带比特率切换的 CAN FD 两种数据帧格式中，位 ESI (Error Status Indicator，错误状态指示符) 的值，由发送开始时发送节点的错误状态决定。如果发送节点为错误被动状态，则 ESI 以隐性位发送，否则以显性位发送。

37.3.3.4 发送延迟补偿

在 CAN FD 传输的数据阶段 (Data phase)，只有一个节点进行发送，其他所有节点都是接收节点。总线的长度没有影响。当通过引脚 MCAN_TX 发送时，MCAN 通过引脚 MCAN_RX 从其本地 CAN 收发器接收发送的数据。接收数据因发送延迟而延迟，如果该延迟大于 TSEG1 (采样点之前的时间段)，则会检测到位错误。为了使数据阶段 (Data phase) 位时间可以比发送延迟更短，引入了延迟补偿。如果没有发送延迟补偿，CAN FD 帧数据阶段 (Data phase) 的比特率将受到发送延迟的限制。

MCAN 的协议单元实现了延迟补偿机制来补偿发送延迟，从而实现在 CAN FD 数据阶段 (Data phase) 以更高的比特率进行传输，而不受特定 CAN 收发器延迟的影响。

为了检查发送节点的数据阶段 (Data phase) 是否存在位错误，会将延迟的发送数据与第二采样点 SSP 处的接收数据进行比较。如果检测到位错误，发送节点将在下一个常规采样点对该位错误作出反应。在仲裁阶段，延迟补偿始终被禁用。

发送延迟补偿支持数据位时间短于发送延迟的配置，ISO 11898-1:2015 对此进行了详细描述，它通过置位 MCAN_DBTP.TDC 使能。

在第二采样点 SSP 将接收到的位与发送的位进行比较。SSP 的位置定义为，从 MCAN 发送输出引脚 MCAN_TX 通过收发器到接收输入引脚 MCAN_RX 测得的延迟，与由 MCAN_TDCR.TDCO 配置的发送延迟补偿偏移之和。发送延迟补偿偏移，用于调整 SSP 在已接收到的位中的位置 (例如数据阶段 Data phase 的位时间的一半)。第二采样点的位置向下取整到下一个整数 mtq (Minimum time quantum，即最小时间片，为一个 MCAN 通信时钟周期)。

MCAN_PSR.TDCV 显示实际的发送延迟补偿值。当 MCAN_CCCR.INIT 置位时，MCAN_PSR.TDCV 被清零；当 MCAN_DBTP.TDC 置位时，MCAN_PSR.TDCV 在每次发送 FD 帧时都会更新。

对于在 MCAN 中实现的发送延迟补偿，必须考虑以下边界条件：

- 测得的 MCAN_TX 到 MCAN_RX 的延迟与配置的发送延迟补偿偏移 MCAN_TDCR.TDCO 之和，必须小于数据阶段 (Data phase) 的 6 个位时间。
- 测得的 MCAN_TX 到 MCAN_RX 的延迟与配置的发送延迟补偿偏移 MCAN_TDCR.TDCO 之和，必须小于或等于 127mtq。如果此和大于 127mtq，则发送延迟补偿使用最大值 127mtq。
- 数据阶段 (Data phase) 在 CRC 分隔符的采样点结束，之后将不在 SSP 检查接收位。

如果通过编程 MCAN_DBTP.TDC=1 使能发送延迟补偿，则会在每个 CAN FD 帧的 FDF 位到 res 位的下降沿处开始测量。当在发送节点的 MCAN_RX 引脚上检测到该边沿时，停止测量。该测量的分辨率为一个 mtq。

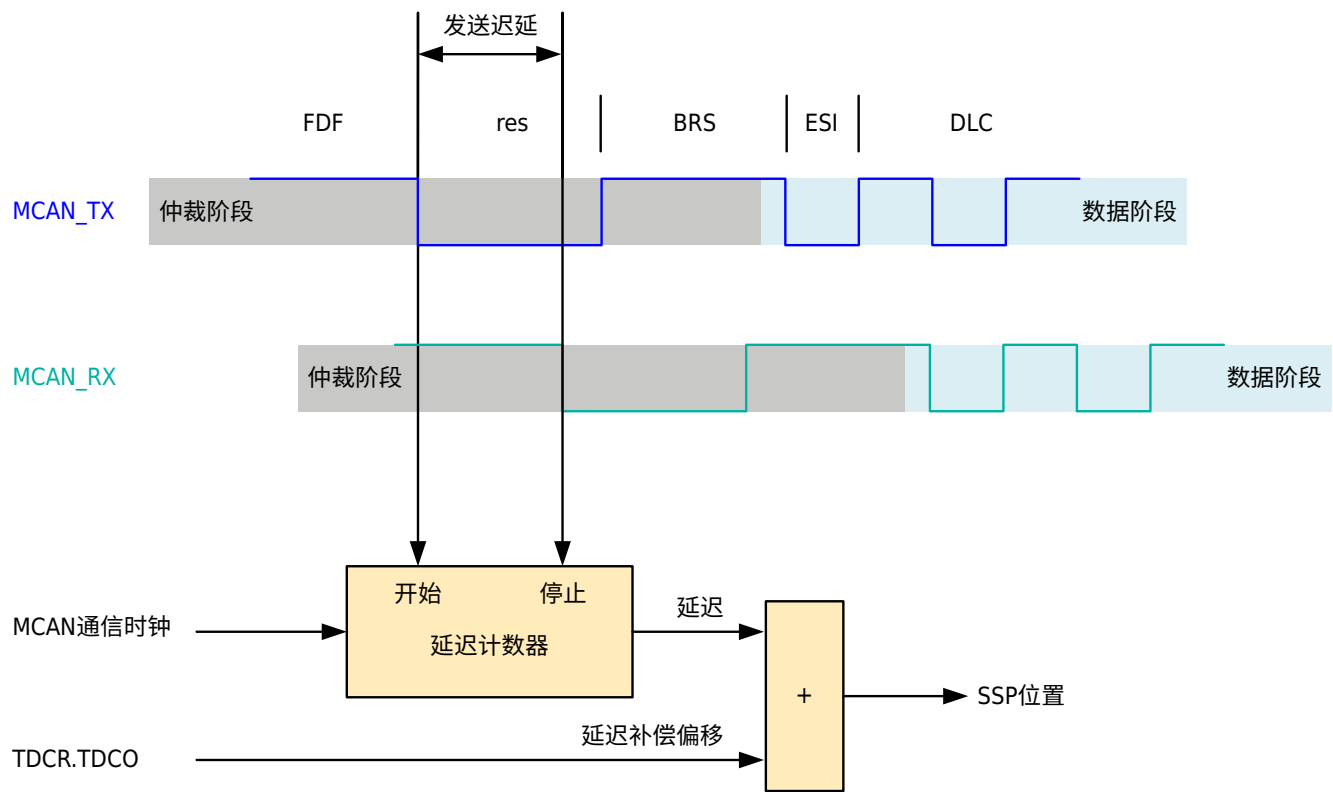


图 37-2 收发器延迟测量

为了避免接收到的 FDF 位中的显性毛刺，在接收到的 res 位的下降沿之前结束延迟补偿测量，从而导致 SSP 位置提前，可通过编程 MCAN_TDCR.TDCF 来使能发送延迟补偿滤波窗口，从而定义 SSP 位置的最小值。MCAN_RX 引脚上的显性边缘会导致 SSP 位置更加提前，但对于发送延迟测量而言这将被忽略。当 SSP 位置至少为 MCAN_TDCR.TDCF 且 MCAN_RX 引脚为低电平时，测量停止。

说明

MCAN 使用的发送延迟为实时测量值，不支持配置固定的发送延迟。

37.3.3.5 受限工作模式

在受限工作模式下，节点能够接收数据帧和远程帧，并能对有效帧进行确认，但不会发送数据帧、远程帧、主动错误帧或过载帧。在出现错误或过载情况时，不会发送显性位，而是等待出现总线空闲条件，以重新同步到 CAN 通信。错误记录 (MCAN_TDCR.CEL) 动作时，错误计数器 (MCAN_TDCR.REC、MCAN_TDCR.TEC) 被冻结。主机可通过置位 MCAN_CCCR.ASM 将 MCAN 设置为受限工作模式，仅当 MCAN_CCCR.CCE 和 MCAN_CCCR.INIT 均设为 1 时，主机才能将 MCAN_CCCR.ASM 置位。MCAN_CCCR.ASM 可随时通过主机清零。

当发送处理单元无法及时从消息 RAM 中读取数据时，会自动进入受限工作模式。要退出受限工作模式，主机 CPU 必须清零 MCAN_CCCR.ASM。

受限工作模式可用于适应不同 CAN 比特率的应用。在这种情况下，应用程序会测试不同的比特率，并在收到有效帧后退出受限操作模式。

说明

受限工作模式不得与回环模式（内部或外部）结合使用。

37.3.3.6 总线监听模式

配置 MCAN_CCCR.MON 为 1，将 MCAN 设置为总线监听模式。在总线监听模式下（见 ISO 11898-1:2015，10.14 Bus monitoring），MCAN 能够接收有效数据帧和有效远程帧，但不能启动发送。在该模式下，MCAN 仅在 CAN 总线上发送隐性位，如果 MCAN 需要发送一个显性位（ACK 位、过载标志、主动错误标志），该

位将在内部被改道发送，以便 MCAN 可以监听该显性位，但 CAN 总线可以保持隐性状态。在总线监听模式下，寄存器 MCAN_TXBRP 保持复位状态。

总线监听模式可用于分析 CAN 总线上的流量，同时又不会因发送显性位而对总线造成影响。下图显示了总线监听模式下 MCAN_TX 和 MCAN_RX 信号与 MCAN 的连接。

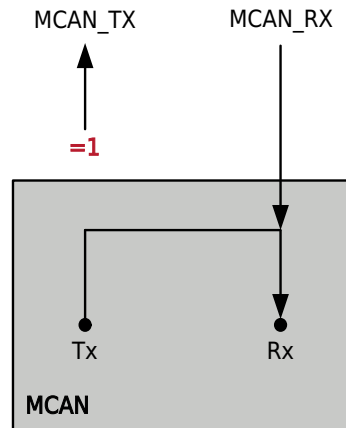


图 37-3 总线监听模式下的引脚控制

37.3.3.7 禁止自动重发 (DAR) 模式

根据 CAN 规范（见 ISO 11898-1:2015, 8.3.4 Recovery Management），MCAN 可自动重发仲裁失败或在发送期间被错误干扰的帧。默认使能自动重发。

在 DAR 模式下，所有发送在 CAN 总线上启动后都会自动取消。发送缓冲区发送请求挂起位 MCAN_TXBRP.TRPx 在成功发送后复位，如果取消时发送尚未开始、发送已因仲裁失败而中止、或帧发送期间发生错误，该位也会复位。

- 成功发送：
 - ▶ 对应的发送缓冲区发送已发生位 MCAN_TXBTO.TOx 置位
 - ▶ 对应的发送缓冲区取消已完成位 MCAN_TXBCF.CFx 不置位
- 即使已取消但也成功发送：
 - ▶ 对应的发送缓冲区发送已发生位 MCAN_TXBTO.TOx 置位
 - ▶ 对应的发送缓冲区取消已完成位 MCAN_TXBCF.CFx 置位
- 仲裁失败或帧发送受到干扰：
 - ▶ 对应的发送缓冲区发送已发生位 MCAN_TXBTO.TOx 不置位
 - ▶ 对应的发送缓冲区取消已完成位 MCAN_TXBCF.CFx 置位

如果帧发送成功，并且使能了发送事件的存储，则会写入发送事件 FIFO 元素，其中事件类型 ET=0b10（即使已取消也会进行发送）。

37.3.3.8 掉电（休眠模式）

MCAN 可通过 CC 控制寄存器 MCAN_CCCR.CSR 设置为掉电模式。只要时钟停止请求有效，MCAN_CCCR.CSR 位就读为 1。

当所有挂起的发送请求都完成后，MCAN 会一直等待直到检测到总线空闲状态。检测到总线空闲状态后，MCAN 将 MCAN_CCCR.INIT 设置为 1 以防止任何进一步的 CAN 传输。然后，MCAN 通过将 MCAN_CCCR.CSA 置 1 确认其已准备好进入掉电状态。在模块时钟（控制逻辑时钟和通信时钟）关闭之前，可继续访问寄存器，但 MCAN_CCCR.INIT 保持为 1。

**说明**

在 CAN 总线受到严重干扰的情况下, 可能永远不会达到空闲状态, 因此 MCAN 不会置位 MCAN_CCCR.INIT。这种情况可通过轮询 MCAN_PSR.ACT 来检测。如果 MCAN 没有进入空闲状态, 软件可以写 CCCR.INIT=1, 这将立即停止 MCAN 的 CAN 通信, 无论是否正在进行发送或接收。

要退出掉电模式, 应用程序必须在复位 CC 控制寄存器标志 MCAN_CCCR.CSR 之前开启模块时钟。MCAN 将通过复位 MCAN_CCCR.CSA 确认时钟已开启。之后, 应用程序可通过复位 MCAN_CCCR.INIT 重新开始 CAN 通信。

37.3.3.9 测试模式

要使能对 MCAN 测试寄存器 MCAN_TEST (请参见“[MCAN 测试寄存器 \(MCAN_TEST\)](#)”)的写访问, 必须将 MCAN_CCCR.TEST 置 1, 从而能够配置测试模式和测试功能。

通过编程 MCAN_TEST.TX, 发送引脚 MCAN_TX 可实现四种输出功能。除了串行数据输出这一默认功能之外, 还可驱动 CAN 采样点信号来监测 MCAN 的位时长, 并可输出恒定的显性或隐性电平。从 MCAN_TEST.RX 可读取引脚 MCAN_RX 的实际值。这两种功能 (MCAN_TEST.TX 和 MCAN_TEST.RX) 均可用于检查 CAN 总线的物理层。

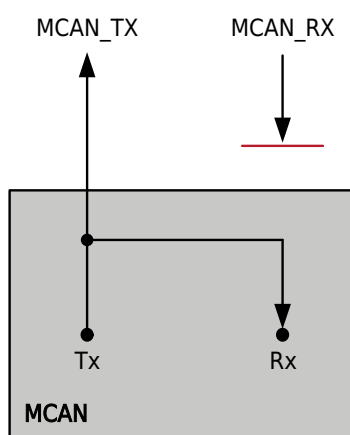
由于 CAN 通信时钟与控制逻辑时钟之间的同步机制, 从配置 MCAN_TEST.TX 到引脚 MCAN_TX 实现指定功能之间, 可能有几个控制逻辑时钟周期的延迟。这一点还适用于通过 MCAN_TEST.RX 读取引脚 MCAN_RX 的情况。

**说明**

测试模式仅限用于生产测试和自检。对 MCAN_TX 引脚进行软件控制会干扰所有 CAN 协议功能。不建议实际应用时使用测试模式。

37.3.3.10 回环模式**外部回环模式**

配置 MCAN_TEST.LBCK 为 1, 将 MCAN 设置为外部回环模式。在回环模式下, MCAN 将其自身发送的消息作为接收的消息来处理, 并将消息存储 (如果这些消息通过了接收过滤) 到接收缓冲区或接收 FIFO 中。下图显示了外部回环模式下 MCAN_TX 和 MCAN_RX 引脚信号与 MCAN 的连接。



外部回环模式

图 37-4 外部回环模式的引脚控制

该模式用于硬件自检。为了不受外部影响, MCAN 在回环模式下忽略 ACK 错误 (在数据帧或远程帧的 ACK 域采样到隐性位)。在此模式下, MCAN_TX 输出的信号在内部直接反馈到接收处理单元, MCAN_RX 输入引脚的实际值被忽略。从引脚 MCAN_TX 可以监听到发送的消息。

内部回环模式

配置 MCAN_TEST.LBCK 和 MCAN_CCCR.MON 为 1，将 MCAN 设置为内部回环模式。该模式可用于“热自检 (Hot Selftest)”，也就是说，MCAN 可以进行检测，同时又不会影响接入的正在运行的 CAN 系统。在此模式下，引脚 MCAN_RX 与 MCAN 断开连接，引脚 MCAN_TX 保持隐性。下图显示了内部回环模式下 MCAN_TX 和 MCAN_RX 引脚信号与 MCAN 的连接。

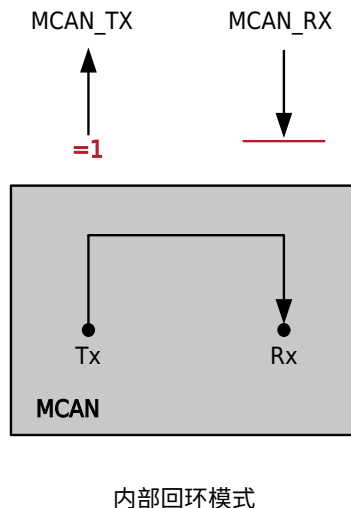


图 37-5 内部回环模式的引脚控制

37.3.4 时间戳生成

MCAN 提供了一个 16 位循环计数器来生成内部时间戳。计数时间单位可通过预分频器 MCAN_TSCC.TCP 配置为 CAN 位时间 1~16 倍。计数器值可通过 MCAN_TSCV.TCV 读取。对寄存器 MCAN_TSCV 进行写访问会将计数器复位为 0。时间戳计数器循环时，中断标志 MCAN_IR.TSW 会置位。

在接收到或发送 SOF (Start Of Frame, 帧起始) 时，计数值被捕获并存储到接收缓冲区或接收 FIFO (RXTS[15:0]) 或发送事件 FIFO (TXTS[15:0]) 元素的时间戳部分。

37.3.5 超时计数器

MCAN 提供了一个 16 位超时计数器来指示接收 FIFO0、接收 FIFO1 和发送事件 FIFO 的超时条件。它作为递减计数器工作，并且与时间戳计数器一样，使用 MCAN_TSCC.TCP 控制的预分频器。超时计数器通过寄存器 MCAN_TOCC 配置。实际计数器值可从 MCAN_TOCV.TOC 读取。仅当 MCAN_CCCR.INIT=0 时，才会启动超时计数器。当 MCAN_CCCR.INIT=1 时，超时计数器停止计数，例如当 CAN 进入 Bus_Off 状态时。

工作模式通过 MCAN_TOCC.TOS 选择。在连续模式下工作时，计数器在 MCAN_CCCR.INIT 复位时启动。对 MCAN_TOCV 进行写操作，会将计数器预设为 MCAN_TOCC.TOP 配置的值，并继续递减计数。

当超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为 MCAN_TOCC.TOP 配置的值。存储了第一个 FIFO 元素后，开始递减计数。对 MCAN_TOCV 执行写入操作不起作用。

计数器达到 0 时，中断标志 MCAN_IR.TOO 置位。在连续模式下，计数器在达到 MCAN_TOCC.TOP 后立即重启。



说明

超时计数器的时钟信号来自 CAN 内核采样点信号。因此，由于 CAN 内核同步（重同步）机制的原因，超时计数器递减的时间点可能有所不同。如果在 CAN FD 中使用了比特率切换功能，则超时计数器在仲裁阶段 (Arbitration phase) 和数据阶段 (Data phase) 中使用不同的时钟计数。

37.3.6 接收处理

接收处理单元控制着接收过滤、已接收消息到接收缓冲区或其中一个接收 FIFO（共两个）的传输以及接收 FIFO 放入和获取索引。

37.3.6.1 接收过滤

MCAN 能够配置两组接收过滤器，一组供标准 ID 使用，另一组供扩展 ID 使用。这两组过滤器可分配给接收缓冲区、接收 FIFO0 或接收 FIFO1。对于接收过滤，每个过滤器列表从第 0 个元素开始执行比较，直到第一个匹配的元素。接收过滤在第一个匹配的元素处停止。不会为此消息评估剩下的过滤器元素。

- 主要特性有：
 - ▶ 每个过滤器元素可配置为
 - ◆ ID 范围过滤器（起止范围）
 - ◆ 一个或两个专用 ID 的过滤器
 - ◆ ID 位屏蔽的经典过滤器
 - ▶ 每个过滤器元素均可配置为实现接收过滤或拒绝过滤
 - ▶ 每个过滤器元素均可单独使能/禁止
 - ▶ 按顺序比较过滤器元素，到第一个匹配的过滤器元素后停止执行
- 相关配置寄存器位：
 - ▶ 全局过滤器配置 (MCAN_GFC)
 - ▶ 标准 ID 过滤器配置 (MCAN_SIDFC)
 - ▶ 扩展 ID 过滤器配置 (MCAN_XIDFC)
 - ▶ 扩展 ID 与运算掩码 (MCAN_XIDAM)
- 根据过滤器元素的配置 (SFEC/EFEC)，发生匹配时会触发下列操作之一：
 - ▶ 将接收到的帧存储到接收 FIFO0 或 FIFO1 中
 - ▶ 将接收到的帧存储到接收缓冲区中
 - ▶ 拒绝接收到的帧
 - ▶ 将高优先级消息中断标志 MCAN_IR.HPM 置 1
 - ▶ 将高优先级消息中断标志 MCAN_IR.HPM 置 1，并将接收到的帧存储到接收 FIFO0 或 FIFO1 中

接收到完整的 ID 后，开始接收过滤。接收过滤完成后，如果找到匹配的接收缓冲区或接收 FIFO，消息处理单元便会开始将接收到的消息数据以 32 位的形式写入匹配的接收缓冲区或接收 FIFO 中。如果 CAN 协议控制器检测到错误条件（例如 CRC 错误），则会丢弃此消息，丢弃后的影响如下：

- 接收缓冲区

匹配接收缓冲区的新数据标志不会置 1，但接收缓冲区（部分）会被接收到的数据覆盖。有关错误类型，请参见 MCAN_PSR.LEC 和 MCAN_PSR.DLEC。
- 接收 FIFO

匹配接收 FIFO 的放入索引不会更新，但相关接收 FIFO 元素（部分）会被接收到的数据覆盖。有关错误类型，请参见 MCAN_PSR.LEC 和 MCAN_PSR.DLEC。如果匹配的接收 FIFO 在覆盖模式下工作，必须考虑接收 FIFO 覆盖模式中介绍的边界条件。



说明

当接收的消息写入两个接收 FIFO 之一或写入接收缓冲区中时，未匹配的已接收 ID 会独立于所用过滤器之外进行存储。接收过滤的过程很大程度上取决于所配置的过滤器元素的顺序。

37.3.6.2 ID 范围过滤器

该过滤器会将所有接收到的帧与 SF1ID/SF2ID 和 EF1ID/EF2ID 定义的范围内的消息 ID 进行匹配检查。范围过滤与扩展帧一起使用时，有两种可能的情况：

- EFT=0b00：应用范围过滤器之前，会将已接收帧的消息 ID 与扩展 ID 与运算掩码 (MCAN_XIDAM) 进行与运算
- EFT=0b11：不会为范围过滤使用扩展 ID 与运算掩码 (MCAN_XIDAM)

37.3.6.3 专用 ID 过滤器

可将过滤器元素配置为过滤一个或两个特定的消息 ID。要过滤一个特定的消息 ID，过滤器元素必须配置为 SF1ID=SF2ID 和 EF1ID=EF2ID。

37.3.6.4 ID 位屏蔽过滤器

ID 位屏蔽过滤用于通过屏蔽已接收消息 ID 的某些位来过滤消息 ID 组。进行 ID 位屏蔽过滤时，SF1ID/EF1ID 用作消息 ID 过滤器，SF2ID/EF2ID 用作过滤器 ID 掩码 (ID Mask)。

ID 掩码中为 0 的位将屏蔽掉已配置 ID 过滤器的对应位，即，已接收消息 ID 在该位位置的值不与接收过滤器对应位的值进行比较，只有对应掩码位为 1 的已接收消息 ID 的位才进行比较。

如果所有掩码位均为 1，则仅当接收到的消息 ID 和消息 ID 过滤器完全相同时，才会出现匹配。如果所有掩码位均为 0，则所有消息 ID 都匹配。

37.3.6.5 标准消息 ID 过滤

下图介绍了标准消息 ID (11 位 ID) 的过滤流程。“标准消息 ID 过滤器元素”介绍了标准消息 ID 过滤器元素。

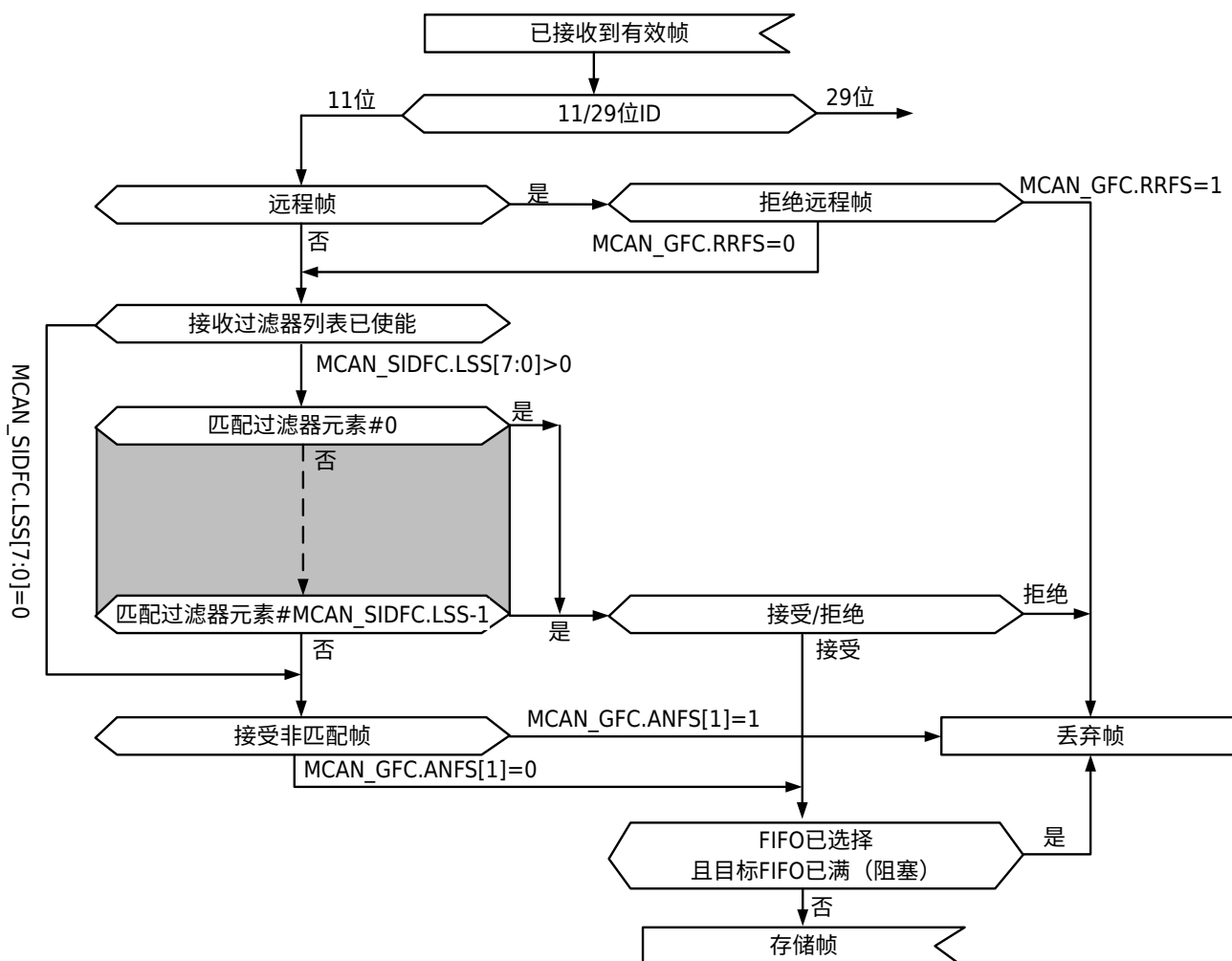


图 37-6 标准消息 ID 过滤器路径

在全局过滤器配置寄存器 (MCAN_GFC) 和标准 ID 过滤器配置寄存器 (MCAN_SIDFC) 控制下, 接收消息的 ID、远程传输请求位 (RTR) 和 ID 扩展位 (IDE) 将与配置的过滤器元素列表进行比较。

37.3.6.6 扩展消息 ID 过滤

下图介绍了扩展消息 ID（29 位 ID）的过滤流程。“[扩展消息 ID 过滤器元素](#)”介绍了扩展消息 ID 过滤器元素。

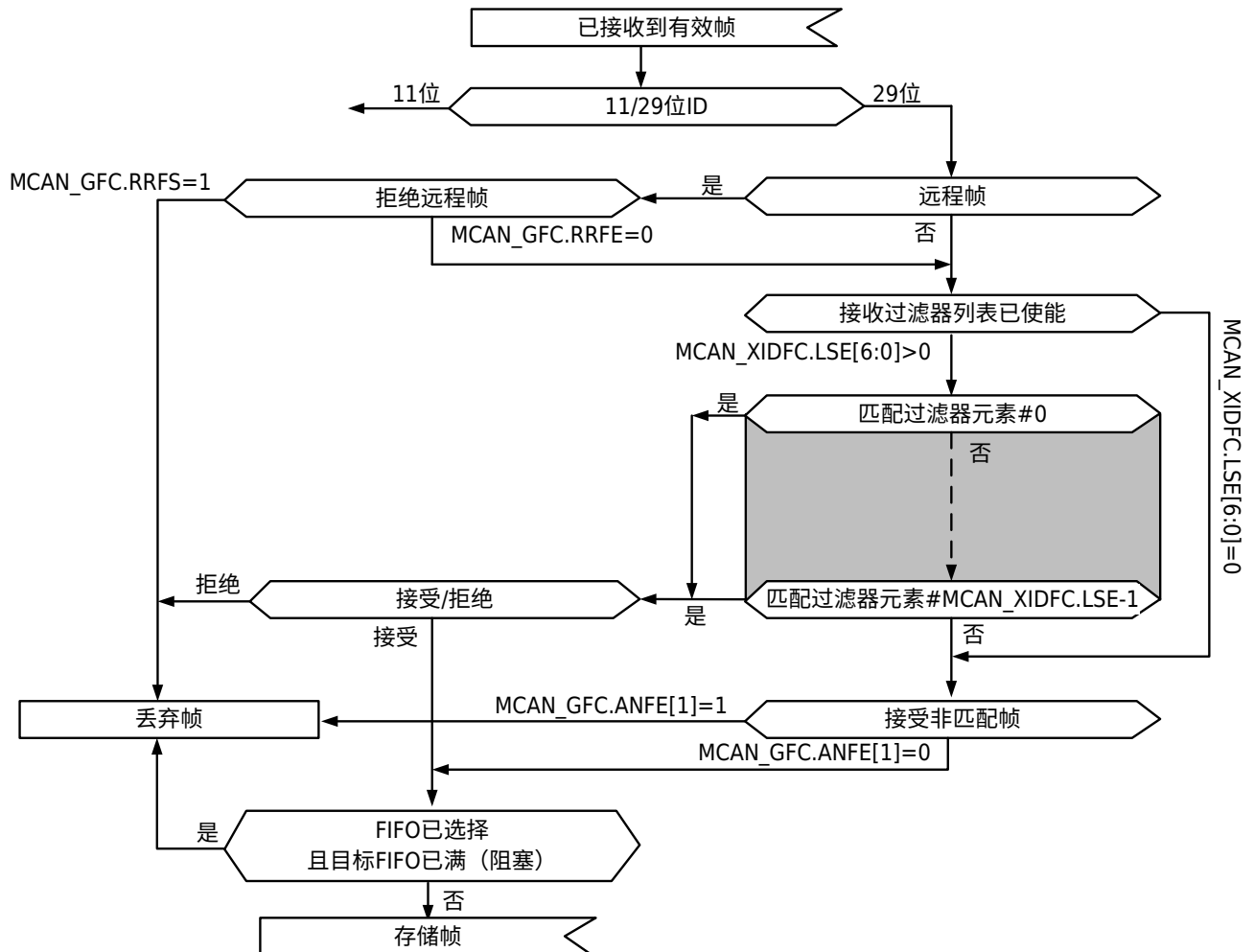


图 37-7 扩展消息 ID 过滤器路径

在全局过滤器配置寄存器 (MCAN_GFC) 和扩展 ID 过滤器配置寄存器 (MCAN_XIDFC) 控制下, 接收消息的 ID、远程传输请求位 (RTR) 和 ID 扩展位 (IDE) 将与配置的过滤器元素列表进行比较。扩展 ID 与掩码 (MCAN_XIDAM) 会在执行过滤器列表之前与接收到的 ID 进行与运算。

37.3.6.7 接收 FIFO

接收 FIFO0 和接收 FIFO1 分别最多可保存 64 个元素（实际元素个数与消息 RAM 容量有关）。两个接收 FIFO 的配置通过寄存器 MCAN_RXF0C 和 MCAN_RXF1C 完成。

通过接收过滤的消息会传输到匹配过滤器元素配置的接收 FIFO 中。有关可用于接收 FIFO0 和接收 FIFO1 的过滤器机制的说明，请参见“[接收过滤](#)”。“[接收缓冲区和 FIFO 元素](#)”介绍了接收缓冲区和 FIFO 元素。

为了避免接收 FIFO 溢出，可使用接收 FIFO 水位线（Watermark）。当接收 FIFO 填充级别达到 MCAN_RXFnC.FnWM 配置的接收 FIFO 水位线时，中断标志 MCAN_IR.RFnW 置位。当接收 FIFO 的放入索引和获取索引相遇时，接收 FIFO 满的条件产生，标志位 MCAN_RXFnS.FnF 置位，中断标志位 MCAN_IR.RFnF 置位。

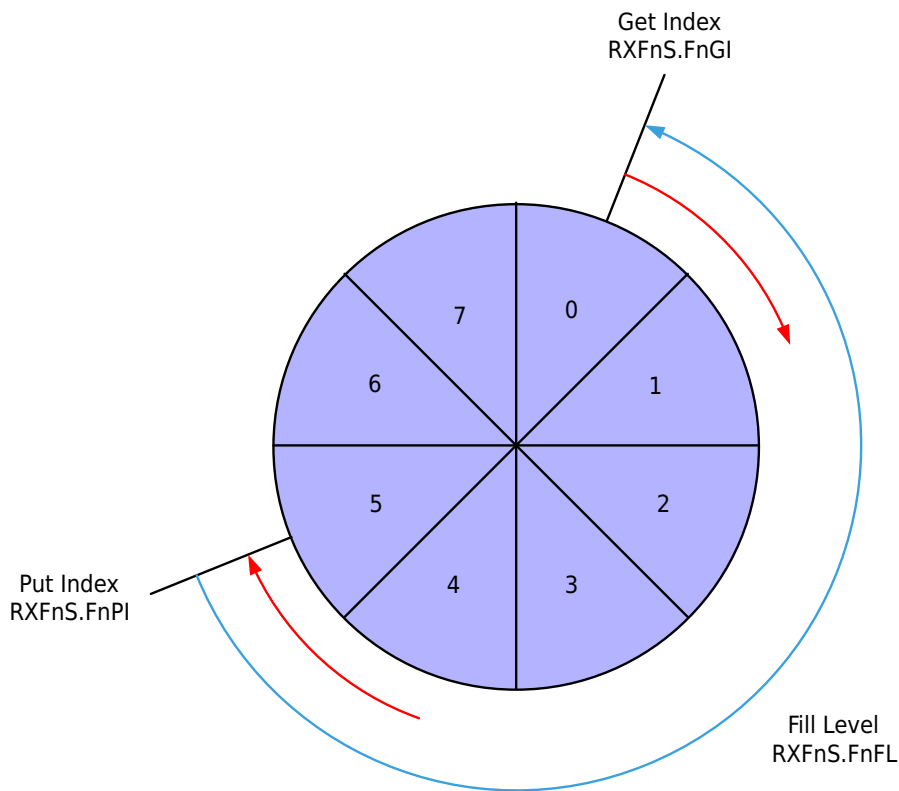


图 37-8 接收 FIFO 状态

当从接收 FIFO 中读取消息时，接收 FIFO 获取索引 MCAN_RXFnS.FnGI 乘以 FIFO 元素大小（字），再乘以 4 后，必须再加上起始地址 MCAN_RXFnC.FnSA。

表 37-4 接收缓冲器和 FIFO 元素大小

RXESC.RBDS[2:0] RXESC.FnDS[2:0]	数据场（字节）	元素大小（字）
0b000	8	4
0b001	12	5
0b010	16	6
0b011	20	7
0b100	24	8
0b101	32	10
0b110	48	14
0b111	64	18

37.3.6.8 接收 FIFO 阻塞模式

接收 FIFO 阻塞模式是通过 MCAN_RXFnC.FnOM=0 配置的，该模式是接收 FIFO 的默认工作模式。达到接收 FIFO 已满条件时（MCAN_RXFnS.FnPI=MCAN_RXFnS.FnGI），标志位 RXFnS.FnF=1，中断标志 MCAN_IR.RFnF=1。此时，MCAN 不会继续向相应的接收 FIFO 写入消息，除非从相应的接收 FIFO 读取消息且其获取索引已递增。

如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，并会通过 MCAN_RXFnS.RFnL=1 指示消息丢失。此外，中断标志 MCAN_IR.RFnL 也会置位。

37.3.6.9 接收 FIFO 覆盖模式

接收 FIFO 覆盖模式是通过 MCAN_RXFnC.FnOM=1 配置的。

达到接收 FIFO 已满条件时 (MCAN_RXFnS.FnPI=MCAN_RXFnS.FnGI)，标志位 RXFnS.FnF=1。此时，FIFO 接收的下一条消息将覆盖最早收到的消息，放入索引和获取索引都会加 1。

如果接收 FIFO 在覆盖模式下工作，并且接收 FIFO 已满，则应至少从获取索引加 1 处开始读取接收 FIFO 元素。这是因为，当 CPU 正在从消息 RAM 读取消息时，恰好有接收到的消息正在被写入消息 RAM，这可能导致 CPU 读到的数据与实际不一致，从接收 FIFO 读取数据时向获取索引中添加偏移量可避免此问题。偏移量取决于 CPU 访问接收 FIFO 的速度。下图显示了读取接收 FIFO 时获取索引的偏移量为 2。在这种情况下，存储在元素 1 和 2 中的两条消息会丢失。

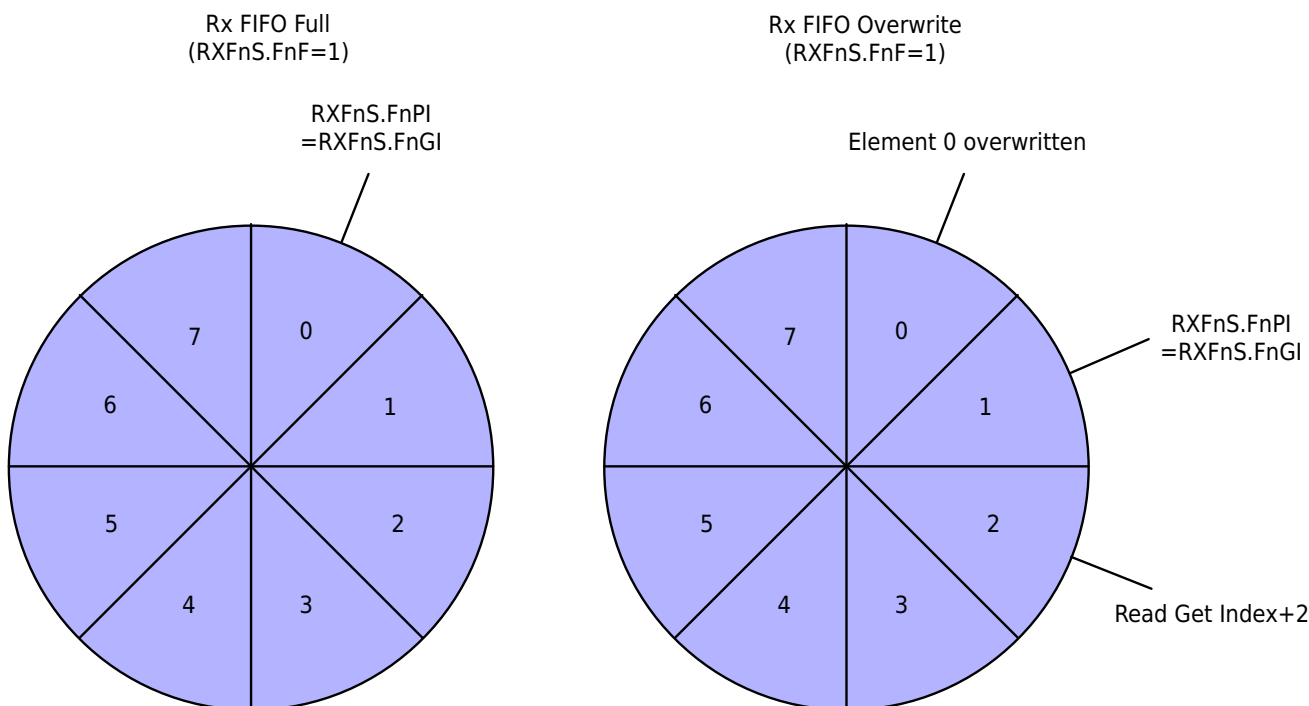


图 37-9 接收 FIFO 溢出处理

从接收 FIFO 读取数据后，必须将读取的最后一个元素的编号写入接收 FIFO 确认索引寄存器 MCAN_RXFnA.FnA 中，使增加后的获取索引指向下一个元素。如果放入索引未增加到该接收 FIFO 元素编号，接收 FIFO 已满标志会复位 (MCAN_RXFnS.FnF=0)。

37.3.6.10 专用接收缓冲区

MCAN 支持多达 64 个专用接收缓冲区。专用接收缓冲区的起始地址是通过 RXBC.RBSA 配置的。

要使用专用接收缓冲区，对于标准 ID 消息，必须配置标准消息 ID 过滤器元素的 SFEC 位为 0b111、SFID2 位为 0b00 (请参见“标准消息 ID 过滤器元素”)；对于扩展 ID 消息，必须配置扩展消息 ID 过滤器元素的 EFEC 位为 0b111、EFID2 位为 0b00 (请参见“扩展消息 ID 过滤器元素”)。下表为专用接收缓冲区的过滤器配置示例。

接收到的消息被过滤器元素接受后，会存储在过滤器元素引用的消息 RAM 中的接收缓冲区中。存储格式与接收 FIFO 元素的格式相同。此外，中断寄存器中的标志 MCAN_IR.DRX (专用接收缓冲区中存储的消息) 也会置 1。

表 37-5 接收缓冲区过滤器配置示例

过滤器元素	SFID1[10:0]	SFID2[10:9]	SFID2[5:0]
	EFID1[28:0]	EFID2[10:9]	EFID2[5:0]
0	ID 消息 1	0b00	0b000000
1	ID 消息 2	0b00	0b000001
2	ID 消息 3	0b00	0b000010

匹配的接收到的消息最后一个字写入消息 RAM 后，寄存器 MCAN_NDAT1/MCAN_NDAT2 中对应的新数据标志会置位。只要新数据标志保持置位的状态，相应的接收缓冲区就会被锁定，以免更新为新的匹配消息。新数据标志必须由主机向相应的位写 1 进行复位。

接收缓冲区新数据标志置位时，其对应的消息 ID 过滤器元素将不进行接收过滤，其他的过滤器元素会继续进行接收过滤。其他消息 ID 过滤器元素可能导致接收到的消息存储到另一接收缓冲区或接收 FIFO 中，或者被拒绝，具体取决于过滤器配置。

37.3.6.11 接收缓冲区处理

- 复位中断标志 MCAN_IR.DRX
- 读取新数据寄存器
- 从消息 RAM 中读取消息
- 复位已处理消息的新数据标志

37.3.6.12 过滤调试消息

要过滤调试消息，需为三个调试消息分别配置一个标准或扩展消息 ID 过滤器元素。要使能过滤器元素过滤调试消息，必须将 SFEC/EFEC 编程为 0b111。在这种情况下，SFID1/SFID2 和 EFID1/EFID2 的含义不同（请参见“标准消息 ID 过滤器元素”和“扩展消息 ID 过滤器元素”）。SFID2/EFID2[10:9]控制着调试消息处理状态机，而 SFID2/EFID2[5:0]控制着收到的调试消息的存储位置。下表为调试消息过滤器配置示例。

存储调试消息时，相应的新数据标志和 MCAN_IR.DRX 均不会置位，可通过 MCAN_RXF1S.DMS 监视调试消息的接收情况。

表 37-6 调试消息过滤器配置示例

过滤器元素	SFID1[10:0]	SFID2[10:9]	SFID2[5:0]
	EFID1[28:0]	EFID2[10:9]	EFID2[5:0]
0	ID 调试消息 A	0b01	0b111101
1	ID 调试消息 B	0b10	0b111110
2	ID 调试消息 C	0b11	0b111111

37.3.7 发送处理

发送处理单元处理专用发送缓冲区、发送 FIFO 和发送队列的发送请求。它控制着发送消息到 CAN 内核的传输、放入和获取索引以及发送事件 FIFO。最多可为消息发送设置 32 个发送缓冲区，每个发送缓冲区可以独立配置发送模式（经典 CAN 模式或 CAN FD 模式）。发送缓冲区的描述请参见“发送缓冲区元素”。下表描述了发送帧的可能配置。

表 37-7 发送帧的可能配置

MCAN_CCCR		发送缓冲区元素		发送帧
BRSE	FDOE	FDF	BRS	
忽略	0	忽略	忽略	经典 CAN
0	1	0	忽略	经典 CAN
0	1	1	忽略	不切换比特率的 FD
1	1	0	忽略	经典 CAN
1	1	1	0	不切换比特率的 FD
1	1	1	1	切换比特率的 FD

 **说明**

AUTOSAR 至少需要三个发送队列缓冲区，并支持发送取消。

当发送缓冲区请求挂起寄存器 MCAN_TXBRP 更新时，发送处理单元会启动发送扫描来检查优先级最高的挂起的发送请求（消息 ID 最小的发送缓冲区）。MCAN_TXBRP 更新的条件是，一个发送缓冲区发送完成，或主机通过写 MCAN_TXBAR 添加发送请求，或主机通过写 MCAN_TXBCR 取消已挂起的发送。

37.3.7.1 发送暂停

发送暂停功能用于 CAN 消息 ID（永久性地）被指定为特定值、并且无法轻易更改的 CAN 系统中。这些消息 ID 的 CAN 仲裁优先级可能高于其他已定义的消息，而在特定的应用中，其相对仲裁优先级应反转。这样可能会导致如下情况：一个 ECU 突发多条较高仲裁优先级的 CAN 消息造成另一个仲裁优先级较低的 ECU CAN 消息被延迟。

举例来说，如果 CAN ECU-1 启用了此功能，并且其应用软件请求发送四条消息，成功发送第一条消息后，将等待两个总线空闲的 CAN 位时间，然后才允许开始发送下一条请求的消息。如果其他 ECU 有挂起的消息，则会在空闲时间开始发送这些消息，不需要与 ECU-1 的下一条消息进行仲裁。接收到消息后，ECU-1 可在接收到的消息释放 CAN 总线后立即开始进行下一次发送。

此功能是通过 MCAN_CCCR.TXP 控制的。如果此位置位，则每次成功发送消息后，MCAN 都将暂停两个 CAN 位时间，然后再开始进行下一次发送。这样，即使网络中其他节点的 ID 优先级较低，也能发送消息。默认禁止此功能（MCAN_CCCR.TXP=0）。此功能可使来自单个节点突发的多条消息变得稀疏一些，避免发送较低优先级消息的 ECU 出现应用程序错误地认为自己请求过多传输。

37.3.7.2 专用发送缓冲区

专用发送缓冲区可完全在主机 CPU 的控制下发送消息。每个专用发送缓冲区都配置了特定的消息 ID。如果多个发送缓冲区配置为相同消息 ID，则会先发送编号最小的发送缓冲区中的消息，这些发送缓冲区应按升序请求发送，即先请求发送缓冲区编号最小的，或者，通过单次写入 MCAN_TXBAR 同时请求发送。

如果缓冲区中的消息有更新，通过置位 MCAN_TXBAR.ARn 请求发送。已请求发送的消息，会根据消息 ID 先在内部与发送 FIFO 或发送队列中的消息进行仲裁，然后在外部与 CAN 总线上的消息进行仲裁，最后根据仲裁结果发送出去。

消息 RAM 中分配的专用发送缓冲区元素大小为若干个 32 位字（4 字节）。因此，消息 RAM 中专用发送缓冲区的起始地址，是通过将发送缓冲区索引（0~31）乘以元素大小（字），再乘以 4（转换为字节），然后与 MCAN_TXBC.TBSA 相加计算得到。

表 37-8 发送缓冲区、FIFO（队列）元素大小

TXESC.TBDS2[2:0]	数据场（字节）	元素大小（字）
0b000	8	4
0b001	12	5
0b010	16	6
0b011	20	7
0b100	24	8
0b101	32	10
0b110	48	14
0b111	64	18

37.3.7.3 发送 FIFO

发送 FIFO 模式是通过将 MCAN_TXBC.TFQM 配置为 0 实现的。存储在发送 FIFO 中的消息，是从获取索引 MCAN_TXFQS.TFGI 指向的消息开始发送的。每次发送后，获取索引会循环递增，直至发送 FIFO 为空。发送 FIFO 会按消息写入的顺序，发送来自不同发送缓冲区但 ID 相同的消息。MCAN 通过计算获取索引与放入索引之差，得到可用（空闲）的发送 FIFO 元素个数（TXFQS.TFFL）。

新消息必须写入以放入索引 MCAN_TXFQS.TFQPI 指向的发送缓冲区开始的发送 FIFO 中。添加发送请求会使放入索引递增，指向下一空闲的发送 FIFO 元素。当放入索引与获取索引相遇时，会指示发送 FIFO 已满（MCAN_TXFQS.TFQF=1）。这种情况下，在下一条消息已发送且获取索引递增之前，不应继续向发送 FIFO 写入消息。

如果有单条消息添加到发送 FIFO，通过对发送 FIFO 放入索引对应的 MCAN_TXBAR 位写 1 来请求发送。如果有多条（假设有 n 条）消息添加到发送 FIFO，则会写入以放入索引开始的 n 个连续发送缓冲区中。随后通过 MCAN_TXBAR 请求发送，放入索引循环递增 n。请求发送的发送缓冲区数不应超过发送 FIFO 空闲级别（MCAN_TXFQS.TFFL）指示的空闲发送缓冲区数。

如果获取索引指向的发送缓冲区的发送请求已取消，那么获取索引会递增到下一个有发送请求挂起的发送缓冲区，并重新计算发送 FIFO 空闲级别（MCAN_TXFQS.TFFL）。如果取消其他任何发送缓冲区的发送，获取索引和 FIFO 空闲级别保持不变。

消息 RAM 中分配的发送 FIFO 元素大小为若干个 32 位字（4 字节）。因此，下一个可用（空闲）的发送 FIFO 的起始地址，是通过将放入索引 MCAN_TXFQS.TFQPI（0~31）乘以元素大小（字），再乘以 4，然后与 MCAN_TXBC.TBSA 相加计算得到。

37.3.7.4 发送队列

发送队列模式是通过将 MCAN_TXBC.TFQM 配置为 1 实现的。存储在发送队列中的消息，是先从消息 ID 最小（优先级最高）的开始发送的。如果多个发送队列缓冲区配置了相同的消息 ID，则发送顺序取决于存储消息的发送队列缓冲区的编号，而这些编号依赖于当前放入索引的状态，因此无法预测发送顺序。

新消息必须写入放入索引 MCAN_TXFQS.TFQPI 指向的发送缓冲区中。放入索引一直指向发送队列中编号最小的空闲缓冲区。如果发送队列已满（MCAN_TXFQS.TFQF=1），则放入索引无效，并且在至少有一个请求的消息已发出或挂起的发送请求已取消之前，不应继续向发送队列写入消息。

应用程序可使用寄存器 MCAN_TXBRP 来代替放入索引，并可将消息放入任何没有挂起发送请求的发送缓冲区中。

消息 RAM 中分配的发送队列缓冲区元素大小为若干个 32 位字（4 字节）。因此，下一个可用（空闲）的发送队列缓冲区的起始地址，是通过将放入索引 MCAN_TXFQS.TFQPI（0~31）乘以元素大小（字），再乘以 4，然后与 MCAN_TXBC.TBSA 相加计算得到。

37.3.7.5 混合使用专用发送缓冲区和发送 FIFO

在这种情况下，消息 RAM 中的发送缓冲区会被划分为一组专用发送缓冲区和一个发送 FIFO。专用发送缓冲区的数量是通过 MCAN_TXBC.NDTB 配置的。分配给发送 FIFO 的发送缓冲区数量是通过 MCAN_TXBC.TFQS 配置的。如果 MCAN_TXBC.TFQS 配置为 0，则仅会使用专用发送缓冲区。

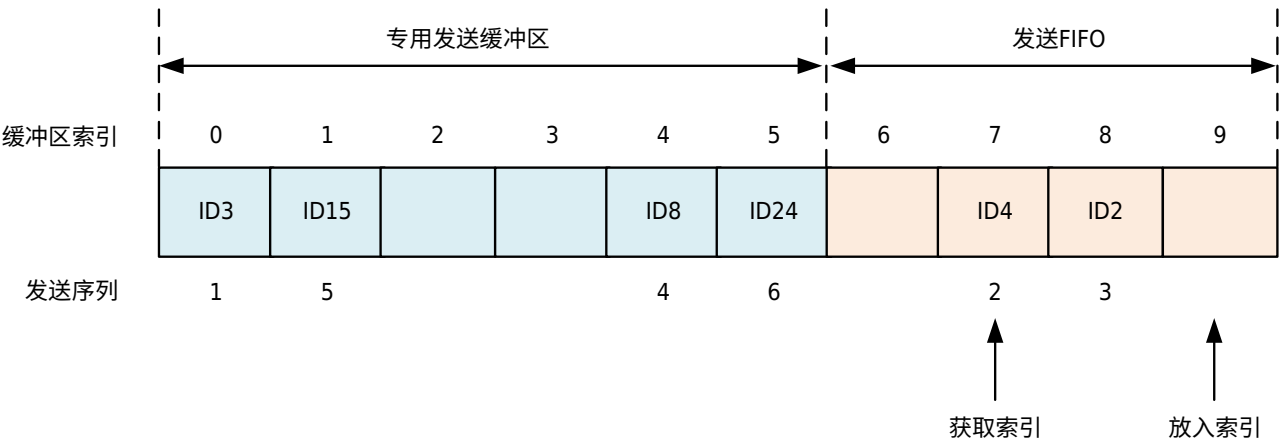


图 37-10 混合配置专用发送缓冲区和发送 FIFO 示例

发送优先级：

- 扫描所有激活了发送请求的专用发送缓冲区和最早挂起发送请求的发送 FIFO 缓冲区（MCAN_TXEFS.EFGI 指向的缓冲区）
- 消息 ID 最小的缓冲区优先级最高，下次将发送该缓冲区的消息

37.3.7.6 混合使用专用发送缓冲区和发送队列

在这种情况下，消息 RAM 中的发送缓冲区会被划分为一组专用发送缓冲区和一个发送队列。专用发送缓冲区的数量是通过 MCAN_TXBC.NDTB 配置的。发送队列缓冲区的数量是通过 MCAN_TXBC.TFQS 配置的。如果 MCAN_TXBC.TFQS 配置为 0，则仅会使用专用发送缓冲区。

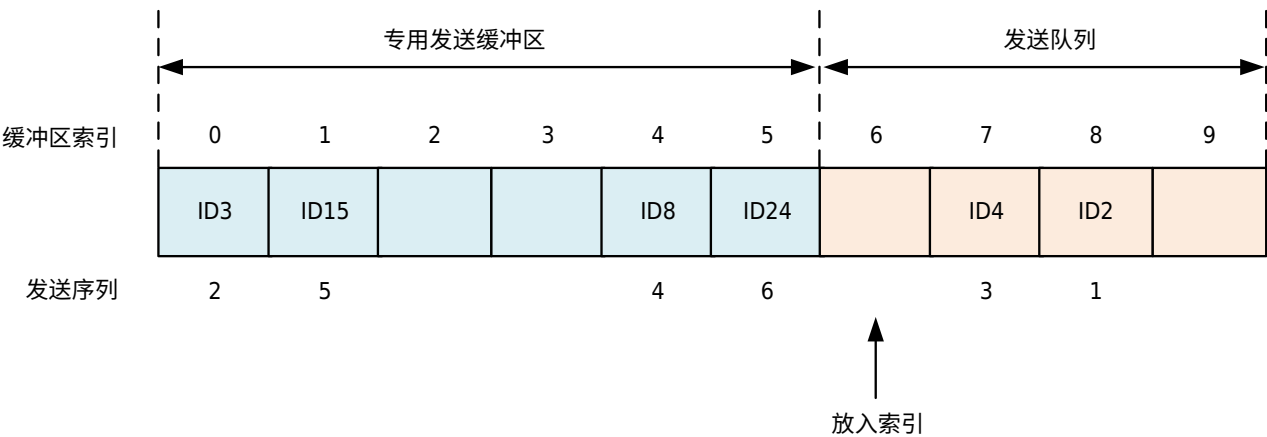


图 37-11 混合配置专用发送缓冲区和发送队列示例

发送优先级：

- 扫描所有激活了发送请求的专用发送缓冲区和发送队列缓冲区
- 消息 ID 最小的发送缓冲区优先级最高，下次将发送该缓冲区的消息

37.3.7.7 发送取消

MCAN 支持发送取消功能。这个功能特别适用于网关应用和基于 AUTOSAR 的应用。要取消专用发送缓冲区或发送队列缓冲区的发送请求，主机必须向寄存器 MCAN_TXBCR 中相应的位（发送缓冲区索引对应的位）写 1。发送取消不适用于发送 FIFO 模式。

成功取消后，寄存器 MCAN_TXBCF 的相应位会置 1。

如果在缓冲区发送过程中请求取消发送，则只要发送正在进行中，MCAN_TXBRP 相应的位就会保持置 1 状态。如果发送成功，MCAN_TXBTO 和 MCAN_TXBCF 相应的位会置 1。如果发送不成功，则不会重复发送，只会将 MCAN_TXBCF 相应的位置位。



说明

如果一个挂起的发送在开始发送之前被立即取消，即使该节点中有另一消息挂起，取消后也会有一个短暂的窗口期，在此期间不会开始任何发送。这样，另一节点便可以发送优先级可能比该节点中第二条消息低的消息。

37.3.7.8 发送事件处理

MCAN 用发送事件 FIFO 来支持发送事件的处理。MCAN 在 CAN 总线上发送消息后，消息 ID 和时间戳会存储在发送事件 FIFO 元素中。为了将发送事件关联到发送事件 FIFO 元素，已发送的发送缓冲区中的消息标记会被复制到发送事件 FIFO 元素中。

发送事件 FIFO 最多可配置 32 个元素，“[发送事件 FIFO 元素](#)”描述了发送事件 FIFO 元素。

发送事件 FIFO 的用途是将处理发送状态信息与处理发送消息分开，也就是让发送缓冲区仅保存要发送的消息，而将发送状态单独存储在发送事件 FIFO 中。这样做是有优势的，尤其是在处理动态管理的发送队列时，发送缓冲区可在发送成功后立即用于新消息。重写发送缓冲区之前，不需要保存该发送缓冲区的发送状态信息。

当 MCAN_IR.TEFF 指示发送事件 FIFO 已满时，在至少已读取一个元素且发送事件 FIFO 获取索引递增之前，不会继续向发送事件 FIFO 写入元素。如果在发送事件 FIFO 已满时发生发送事件，此事件会被丢弃，中断标志 MCAN_IR.TEFL 置位。

为了避免发送事件 FIFO 溢出，可使用发送事件 FIFO 水位线。当发送事件 FIFO 填充数量达到由 TXEFC.EFWM 配置的发送事件 FIFO 水位线时，中断标志 MCAN_IR.TEFW 置位。

从发送事件 FIFO 读取数据时，必须将发送事件 FIFO 获取索引 TXEFS.EFGI 乘以 2（发送事件 FIFO 元素大小为两个字），再乘以 4，然后与发送事件 FIFO 起始地址 TXEFC.EFSA 相加，以得到目标数据所在的地址。

37.3.8 FIFO 确认处理

接收 FIFO0、接收 FIFO1 和发送事件 FIFO 的获取索引，是通过对相应 FIFO 确认索引寄存器进行写操作来控制的，请参见“[MCAN 接收 FIFO0 确认寄存器 \(MCAN_RXF0A\)](#)”、“[MCAN 接收 FIFO1 确认寄存器 \(MCAN_RXF1A\)](#)”和“[MCAN 发送事件 FIFO 确认寄存器 \(MCAN_TXEFA\)](#)”。对 FIFO 确认索引进行写操作会将 FIFO 获取索引设置为 FIFO 确认索引加 1，进而会更新 FIFO 填充级别。有两种用例：

1. 如果只从 FIFO 读取了一个元素（获取索引指向的元素），则应将获取索引值写入 FIFO 确认索引寄存器中。
2. 如果已从 FIFO 中读取了一个序列的元素，则仅需在该序列读取结束时，将最后一个元素的索引值写入 FIFO 确认索引寄存器，即可更新 FIFO 获取索引。

由于 CPU 可自由访问 MCAN 的消息 RAM，在以任意顺序（不考虑获取索引）读取 FIFO 元素时需要特别注意，特别是在从两个接收 FIFO 之一读取高优先级消息时。在这种情况下，不对 FIFO 确认索引寄存器进行写操作，否则会导致获取索引指向错误的位置，还会改变 FIFO 填充级别，一些较早的 FIFO 元素可能会丢失。



说明

应用程序必须确保写入到 FIFO 确认索引寄存器中的值是正确的，MCAN 不会检查数值是否正确。

37.3.9 消息 RAM

37.3.9.1 消息 RAM 配置

消息 RAM 的宽度为 32 位。MCAN 模块最多可在消息 RAM 中分配 1024 个字，如下图所示。实际应用中不要
求对所有部分都进行配置，各部分之间的顺序也没有限制。

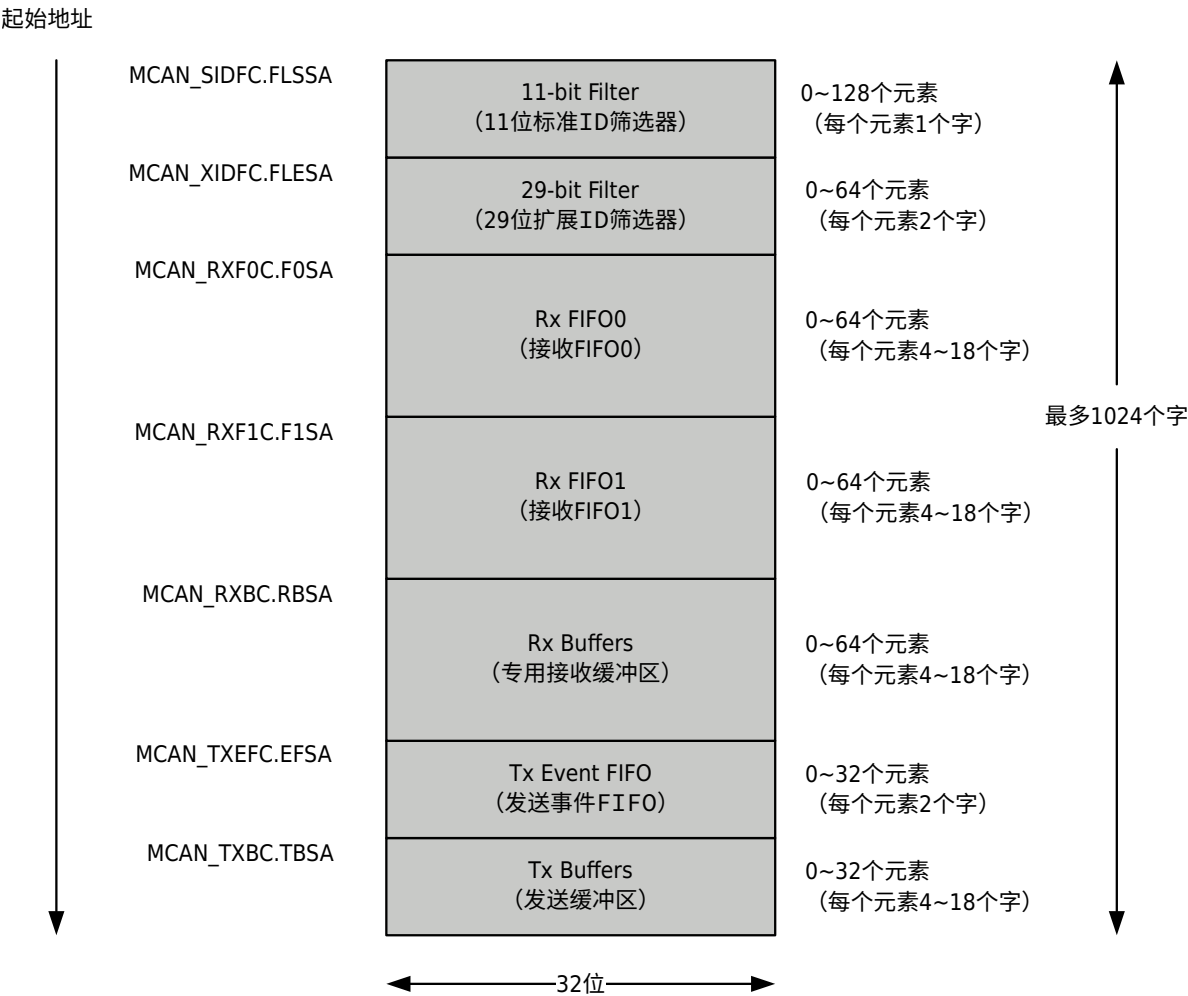


图 37-12 消息 RAM 配置

MCAN 访问消息 RAM 时需要按 32 位对齐的地址进行寻址，而不是字节对齐的地址。可配置的起始地址是 32 位字地址，即仅评估地址的 15 到 2 位，忽略两个最低有效位。

消息 RAM 带有 ECC 校验（Error Checking and Correcting）。ECC 校验为纠一检二码，即可以纠正 1-bit 错误，检查 2-bit 错误。ECC 校验带有错误注入功能。根据错误注入使能寄存器 MCANRAM_EIEN 和错误注入位使能寄存器 MCANRAM_EIBIT0/ MCANRAM_EIBIT1 的设定，硬件电路将从消息 RAM 读取的数据按位取反，以达到错误注入的目的。通过设定错误注入位使能寄存器 MCANRAM_EIBIT0/MCANRAM_EIBIT1，可以制造-1bit 错误或多 bit 错误，由于 ECC 校验为纠一检二码，因此超过 2-bit 错误有可能不会被检出。消息 RAM 校验错误地址寄存器（MCANRAM_ECCERRADDR）记录消息 RAM 发生 1-bit 或 2-bit ECC 错误时的访问地址（相对于消息 RAM 基地址的偏移地址）。

 说明

MCAN 不会检查消息 RAM 的配置是否存在错误，因此配置不同部分的起始地址以及各部分元素的数量时必须多加留意，以免造成数据入侵或丢失。

37.3.9.2 接收缓冲区和 FIFO 元素

最多可在消息 RAM 中配置 64 个接收缓冲区和 2 个接收 FIFO。每个接收 FIFO 均可配置为最多存储 64 个消息。接收缓冲区和接收 FIFO 元素的结构以及相关说明如下表所示。
可通过寄存器 MCAN_RXESC 配置元素大小，CAN FD 消息的数据场（Data field）最多可达 64 字节。

表 37-9 接收缓冲区和 FIFO 元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	ESI	XTD	RTR	ID																												
R1	ANMF	FIDX							Res.	FDF	BRS	DLC					RXTS															
R2	DB3								DB2							DB1							DB0									
R3	DB7								DB6							DB5							DB4									
.....																																
Rn	DBm								DBm-1							DBm-2							DBm-3									

表 37-10 接收缓冲区和 FIFO 元素说明

位/位域	标记	功能描述
R0[31]	ESI	错误状态指示符（Error State Indicator） ● 0b0：发送节点为错误主动状态 ● 0b1：发送节点为错误被动状态
R0[30]	XTD	扩展 ID 标志位（Extended Identifier） 向主机指示接收到的帧的 ID 是标准 ID 还是扩展 ID。 ● 0b0：11 位标准 ID ● 0b1：29 位扩展 ID
R0[29]	RTR	远程发送请求（Remote Transmission Request） 向主机指示接收到的是于数据帧还是远程帧。 ● 0b0：接收到的帧是数据帧 ● 0b1：接收到的帧是远程帧  说明 CAN FD 没有远程帧。在 CAN FD 帧中（FDF=1），显性位 RRS（Remote Request Substitution）替代了 RTR 位。
R0[28:0]	ID[28:0]	ID（Identifier） 标准 ID 或扩展 ID 取决于 XTD 位，标准 ID 存储在 ID[28:18]中。

位/位域	标记	功能描述
R1[31]	ANMF	接收到非匹配帧 (Accepted Non-matching Frame) 可通过 MCAN_GFC.ANFS 和 MCAN_GFC.ANFE 使能接受非匹配帧。 ● 0b0: 接收到的帧与过滤器索引 FIDX 指向的过滤器元素匹配 ● 0b1: 接收到的帧与任何接收过滤器元素都不匹配
R1[30:24]	FIDX[6:0]	过滤器索引 (Filter Index) ● 0~127: 匹配的接收过滤器元素的索引 (ANMF=1 时无效) 范围为 0 到 MCAN_SIDFC.LSS]减 1 或 MCAN_XIDFC.LSE 减 1。
R1[21]	FDF	FD 格式 (FD Format) ● 0b0: 经典 CAN 帧格式 ● 0b1: CAN FD 帧格式 (新的 DLC 编码和 CRC)
R1[20]	BRS	比特率切换 (Bit Rate Switch) ● 0b0: 接收帧没有比特率切换 ● 0b1: 接收帧有比特率切换
R1[19:16]	DLC[3:0]	数据长度代码 (Data Length Code) ● 0~8: 经典 CAN 和 CAN FD: 接收到的帧包含 0~8 个数据字节 ● 9~15: CAN: 接收到的帧包含 8 个数据字节 ● 9~15: CAN FD: 接收到的帧包含 12/16/20/24/32/48/64 个数据字节
R1[15:0]	RXTS[15:0]	接收时间戳 (Rx Timestamp) 接收到 SOF (Start Of Frame) 时捕获的时间戳计数器值。 分辨率取决于时间戳计数器预分频器 MCAN_TSCC.TCP 的配置。
R2[31:24]	DB3[7:0]	数据字节 3 (Data Byte 3)
R2[23:16]	DB2[7:0]	数据字节 2 (Data Byte 2)
R2[15:8]	DB1[7:0]	数据字节 1 (Data Byte 1)
R2[7:0]	DB0[7:0]	数据字节 0 (Data Byte 0)
R3[31:24]	DB7[7:0]	数据字节 7 (Data Byte 7)
R3[23:16]	DB6[7:0]	数据字节 6 (Data Byte 6)
R3[15:8]	DB5[7:0]	数据字节 5 (Data Byte 5)
R3[7:0]	DB4[7:0]	数据字节 4 (Data Byte 4)
...	...	...

位/位域	标记	功能描述
Rn[31:24]	DBm[7:0]	数据字节 m（Data Byte m）
Rn[23:16]	DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Rn[15:8]	DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Rn[7:0]	DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

37.3.9.3 发送缓冲区元素

发送缓冲区可配置为专用发送缓冲区和发送 FIFO（或队列）。如果发送缓冲区由专用发送缓冲区和发送 FIFO（或队列）共享，则专用发送缓冲区在发送缓冲区的起始处，其后是发送 FIFO（或队列）。发送处理单元会评估发送缓冲区配置 MCAN_TXBC.TFQS 和 MCAN_TXBC.NDTB，以区分专用发送缓冲区与发送 FIFO（队列）。发送缓冲区元素的结构以及相关说明如下表所示。可通过寄存器 MCAN_TXESC 配置元素大小，CAN FD 消息的数据场（Data field）最多可达 64 字节。

表 37-11 发送缓冲区元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T0	ESI	XTD	RTR	ID																												
T1	MM[7:0]								EFC	Res.	FDF	BRS	DLC				MM[15:8]							Res.								
T2	DB3								DB2					DB1							DB0											
T3	DB7								DB6					DB5							DB4											
.....																																
Tn	DBm								DBm-1					DBm-2							DBm-3											

表 37-12 发送缓冲区元素说明

位/位域	标记	功能描述
T0[31]	ESI	错误状态指示符（Error State Indicator） 0b0：CAN FD 格式的 ESI 位仅取决于错误被动标志 0b1：CAN FD 格式的 ESI 位以隐性位发送  说明 发送缓冲区的 ESI 位会与错误被动标志进行或运算，以确定将发送的 FD 帧 ESI 位的值。按照 CAN FD 协议规范的要求，错误主动节点可选择发送隐性 ESI 位，但错误被动节点将始终发送隐性 ESI 位。
T0[30]	XTD	扩展 ID 标志位（Extended Identifier） 0b0：11 位标准 ID 0b1：29 位扩展 ID

位/位域	标记	功能描述
T0[29]	RTR	远程发送请求 (Remote Transmission Request) ● 0b0: 发送数据帧 ● 0b1: 发送远程帧  说明 当 RTR=1 时, 即使 MCAN_CCCR.FDOE 使能以 CAN FD 格式进行发送, MCAN 也会按照 ISO 11898-1:2015 的规定发送远程帧。
T0[28:0]	ID[28:0]	ID (Identifier) 标准 ID 或扩展 ID, 取决于 XTD 位, 标准 ID 必须写在 ID[28:18]中。
T1[31:24]	MM[7:0]	消息标记 (Message Marker) 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中, 用于标识发送消息状态。
T1[23]	EFC	事件 FIFO 控制 (Event FIFO Control) ● 0b0: 不存储发送事件 ● 0b1: 存储发送事件
T1[21]	FDF	FD 格式 (FD Format) ● 0b0: 以经典 CAN 格式发送帧 ● 0b1: 以 CAN FD 格式发送帧
T1[20]	BRS	比特率切换 (Bit Rate Switch) ● 0b1: CAN FD 帧发送时切换比特率 ● 0b0: CAN FD 帧发送时不切换比特率  说明 仅当 CAN FD 使能 (MCAN_CCCR.FDOE=1) 时, 才会评估 ESI、FDF 和 BRS 位。仅当额外设置 MCAN_CCCR.BRSE=1 时, 才会评估 BRS 位。
T1[19:16]	DLC[3:0]	数据长度代码 (Data Length Code) ● 0~8: 经典 CAN 和 CAN FD: 发送帧包含 0~8 个数据字节 ● 9~15: CAN: 发送帧包含 8 个数据字节 ● 9~15: CAN FD: 发送帧包含 12/16/20/24/32/48/64 个数据字节
T1[15:8]	MM[15:8]	宽消息标记的高字节 (High byte of Wide Message Marker) 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中, 用于标识发送消息状态。仅当 MCAN_CCCR.WMM=1 时有效。

位/位域	标记	功能描述
T2[31:24]	DB3[7:0]	数据字节 3（Data Byte 3）  说明 根据 MCAN_TXESC 寄存器配置的元素大小，2~16（Tn=2~17）个 32 位字用于存储 CAN 消息的数据场（Data field）。
T2[23:16]	DB2[7:0]	数据字节 2（Data Byte 2）
T2[15:8]	DB1[7:0]	数据字节 1（Data Byte 1）
T2[7:0]	DB0[7:0]	数据字节 0（Data Byte 0）
T3[31:24]	DB7[7:0]	数据字节 7（Data Byte 7）
T3[23:16]	DB6[7:0]	数据字节 6（Data Byte 6）
T3[15:8]	DB5[7:0]	数据字节 5（Data Byte 5）
T3[7:0]	DB4[7:0]	数据字节 4（Data Byte 4）
...	...	...
Tn[31:24]	DBm[7:0]	数据字节 m（Data Byte m）
Tn[23:16]	DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Tn[15:8]	DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Tn[7:0]	DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

37.3.9.4 发送事件 FIFO 元素

每个元素存储已发送消息相关的信息。通过读取发送事件 FIFO，主机 CPU 按消息的发送顺序获取这些信息。关于发送事件 FIFO 的状态信息，可从寄存器 MCAN_TXEFS 中获取。发送事件 FIFO 元素的结构以及相关说明如下表所示。

- E1A：当 MCAN_CCCR.WMM=0 时，E1A.TXTS[15:0]保存由 MCAN 的内部时间戳逻辑生成的 16 位时间戳。
- E1B：当使能 16 位宽消息标记（MCAN_CCCR.WMM=1）时，E1B.MM[15:8]为宽消息标记的高 8 位。

表 37-13 发送事件 FIFO 元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
E0	ESI	XTD	RTR	ID																																	
E1A	MM[7:0]								ET		FDF	BRS	DLC			TXTS																					
E1B	MM[7:0]								ET		FDF	BRS	DLC			MM[15:8]							Res.														

表 37-14 发送事件 FIFO 元素说明

位/位域	标记	功能描述
E0[31]	ESI	错误状态指示符 (Error State Indicator) ● 0b0: 发送节点为错误主动状态 ● 0b1: 发送节点为错误被动状态
E0[30]	XTD	扩展 ID 标志位 (Extended Identifier) ● 0b0: 11 位标准 ID ● 0b1: 29 位扩展 ID
E0[29]	RTR	远程发送请求 (Remote Transmission Request) ● 0b0: 发送帧为数据帧 ● 0b1: 发送帧为远程帧
E0[28:0]	ID[28:0]	标识符 (Identifier) 标准 ID 或扩展 ID, 取决于 XTD 位。标准 ID 存储在 ID[28:18]中。
E1A/B[31:24]	MM[7:0]	消息标记 (Message Marker) 从发送缓冲区复制到发送事件元素, 用于标识发送消息状态。
E1A/B[23:22]	ET[1:0]	事件类型 (Event Type) ● 0b00: Reserved ● 0b01: 发送事件 ● 0b10: 取消后仍发送 (在 DAR 模式下发送时始终为该值) ● 0b11: Reserved
E1A/B[21]	FDF	FD 格式 (FD Format) ● 0b0: 经典 CAN 帧格式 ● 0b1: CAN FD 帧格式 (新的 DLC 编码和 CRC)
E1A/B[20]	BRS	比特率切换 (Bit Rate Switch) ● 0b0: CAN FD 帧发送时没有切换比特率 ● 0b1: CAN FD 帧发送时有切换比特率
E1A/B[19:16]	DLC[3:0]	数据长度代码 (Data Length Code) ● 0~8: 经典 CAN 和 CAN FD: 发送帧包含 0~8 个数据字节 ● 9~15: 经典 CAN: 发送帧包含 8 个数据字节 ● 9~15: CAN FD: 发送帧包含 12/16/20/24/32/48/64 个数据字节

位/位域	标记	功能描述
E1A[15:0]	TXTS[15:0]	发送时间戳 (Tx Timestamp) 发送 SOF (Start Of Frame) 时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 MCAN_TSCC.TCP 的配置。
E1B[15:8]	MM[15:8]	宽消息标记的高字节 (High byte of Wide Message Marker) 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中, 用于标识发送消息状态。

37.3.9.5 标准消息 ID 过滤器元素

最多可为 11 位标准 ID 配置 128 个过滤器元素 (每个元素大小为 1 个字)。其地址为过滤器元素索引 (0~127) 乘以 4 (转换成字节), 加上起始地址 MCAN_SIDFC.FLSSA。标准消息 ID 元素的结构以及相关说明如下表所示。

表 37-15 标准消息 ID 过滤器元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S0	SFT		SFEC			SFID1											Res.				SFID2											

表 37-16 标准消息 ID 过滤器元素说明

位/位域	标记	功能描述
S0[31:30]	SFT[1:0]	标准过滤器类型 (Standard Filter Type) ● 0b00: 从 SFID1 到 SFID2 的范围过滤器 (SFID2≥SFID1) ● 0b01: SFID1 或 SFID2 定义的双 ID 过滤器 ● 0b10: 经典过滤器: SFID1=ID, SFID2=掩码 ● 0b11: 禁止该过滤器元素  说明 当 SFT=0b11 时, 该过滤器元素被禁止, 接收过滤继续进行 (与 SFEC=0b000 时的行为相同)。
S0[29:27]	SFEC[2:0]	标准过滤器元素配置 (Standard Filter Element Configuration) 所有使能的过滤器元素都用于标准帧的接收过滤。当匹配到一个已使能的过滤器元素, 或到达过滤器列表结尾处时, 停止接收过滤。如果 SFEC=0b100、0b101 或 0b110, 出现匹配时中断标志 MCAN_IR.HPM 置位, 并产生中断 (如果使能)。在这种情况下, 寄存器 MCAN_HPMS 会更新为优先级匹配的状态。 ● 0b000: 禁止该过滤器元素 ● 0b001: 如果匹配, 则存储到接收 FIFO0 中 ● 0b010: 如果匹配, 则存储到接收 FIFO1 中 ● 0b011: 如果匹配, 则拒绝 ID ● 0b100: 如果匹配, 则设置优先级, 不存储

位/位域	标记	功能描述
S0[29:27]	SFEC[2:0]	● 0b101：如果匹配，则设置优先级并存储到 FIFO0 中 ● 0b110：如果匹配，则设置优先级并存储到 FIFO1 中 ● 0b111：存储到接收缓冲区中或作为调试消息，忽略 SFT[1:0]的配置
S0[26:16]	SFID1[10:0]	标准过滤器 ID1（Standard Filter ID 1） 标准 ID 过滤器元素的第一个 ID。当为接收缓冲区或调试消息过滤时，此位域定义要存储的标准消息的 ID。接收的 ID 必须完全匹配，不使用掩码机制。
S0[10:0]	SFID2[10:0]	标准过滤器 ID2（Standard Filter ID 2） 该位域具有不同含义，具体视 SFEC 的配置而定： ● SFEC=0b001~0b110 时：标准 ID 过滤器元素的第二个 ID ● SFEC=0b111 时：用于接收缓冲区或调试消息的过滤器
	SFID2[10:9]	确定接收的消息存储到接收缓冲区中，还是作为调试消息序列中的消息 A、B 或 C 进行处理。 ● 0b00：将消息存储到接收缓冲区中 ● 0b01：调试消息 A ● 0b10：调试消息 B ● 0b11：调试消息 C
	SFID2[5:0]	定义存储匹配消息的接收缓冲区到起始地址 RXBC.RBSA 的偏移（缓冲区索引）。

37.3.9.6 扩展消息 ID 过滤器元素

最多可为 29 位扩展 ID 配置 64 个过滤器元素（每个元素大小为 2 个字）。访问扩展消息 ID 过滤器元素时，其地址为过滤器元素索引（0~63）乘以 2，再乘以 4（转换成字节），然后加上起始地址 MCAN_XIDFC.FLESA。扩展消息 ID 元素的结构以及相关说明如下表所示。

表 37-17 扩展消息 ID 过滤器元素

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F0	EFEC			EFID1																												
F1	EFT		Res.	EFID2																												

表 37-18 扩展消息 ID 过滤器元素说明

位/位域	标记	功能描述
F0[31:29]	EFEC[2:0]	扩展过滤器元素配置（Extended Filter Element Configuration） 所有使能的过滤器元素都用于扩展帧的接收过滤。当匹配到一个已使能的过滤器元素，或到达过滤器列表结尾处时，停止接收过滤。 ● 0b000：禁止该过滤器元素

位/位域	标记	功能描述
F0[31:29]	EFEC[2:0]	● 0b001: 如果匹配, 则存储到接收 FIFO0 中 ● 0b010: 如果匹配, 则存储到接收 FIFO1 中 ● 0b011: 如果匹配, 则拒绝 ID ● 0b100: 如果匹配, 则设置优先级 ● 0b101: 如果匹配, 则设置优先级并存储到 FIFO0 中 ● 0b110: 如果匹配, 则设置优先级并存储到 FIFO1 中 ● 0b111: 存储到接收缓冲区中或作为调试消息, 忽略 EFT[1:0]的配置 如果 EFEC=0b100、0b101 或 0b110, 出现匹配时中断标志 MCAN_IR.HPM 置位, 并产生中断 (若使能)。在这种情况下, 寄存器 MCAN_HPMS 会更新为优先级匹配的状态。
F0[28:0]	EFID1[28:0]	扩展过滤器 ID1 (Extended Filter ID 1) 扩展 ID 过滤器元素的第一个 ID。当为接收缓冲区或调试消息过滤时, 此位域定义要存储的扩展消息的 ID。接收的 ID 必须完全匹配, 仅使用 MCAN_XIDAM 掩码机制。
F1[31:30]	EFT[1:0]	扩展过滤器类型 (Extended Filter Type) ● 0b00: 从 EFID1 到 EFID2 的范围过滤器(EFID2≥EFID1) ● 0b01: EFID1 或 EFID2 定义的双 ID 过滤器 ● 0b10: 经典过滤器: EFID1=ID, EFID2=掩码 ● 0b11: 从 EFID1 到 EFID2 的范围过滤器 (EFID2≥EFID1) , 不使用 MCAN_XIDAM 掩码
F1[28:0]	EFID2[28:0]	扩展过滤器 ID2 (Extended Filter ID 2) 该位域具有不同含义, 具体视 EFEC 的配置而定: ● EFEC=0b001~0b110 时: 扩展 ID 过滤器元素的第二个 ID ● EFEC=0b111 时: 用于接收缓冲区或调试消息的过滤器
	EFID2[10:9]	确定接收的消息存储到接收缓冲区中, 还是作为调试消息序列中的消息 A、B 或 C 进行处理。 ● 0b00: 将消息存储到接收缓冲区中 ● 0b01: 调试消息 A ● 0b10: 调试消息 B ● 0b11: 调试消息 C
	EFID2[5:0]	定义存储匹配消息的接收缓冲区到起始地址 MCAN_RXBC.RBSA 的偏移 (缓冲区索引) 。

37.4 寄存器

37.4.1 寄存器总表

- MCAN1 基地址: 0x4002B000
- MCAN2 基地址: 0x4002B400

表 37-19 控制器局域网 (MCAN) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x04	MCAN_ENDN	MCAN 字节序寄存器	32
0x0C	MCAN_DBTP	MCAN 数据位时间和预分频寄存器	32
0x10	MCAN_TEST	MCAN 测试寄存器	32
0x14	MCAN_RWD	MCAN RAM 看门狗寄存器	32
0x18	MCAN_CCCR	MCAN CC 控制寄存器	32
0x1C	MCAN_NBTP	MCAN 标称位时间和预分频寄存器	32
0x20	MCAN_TSCC	MCAN 时间戳计数器配置寄存器	32
0x24	MCAN_TSCV	MCAN 时间戳计数器值寄存器	32
0x28	MCAN_TOCC	MCAN 超时计数器配置寄存器	32
0x2C	MCAN_TOCV	MCAN 超时计数器值寄存器	32
0x40	MCAN_ECR	MCAN 错误计数器寄存器	32
0x44	MCAN_PSR	MCAN 协议状态寄存器	32
0x48	MCAN_TDCR	MCAN 发送延迟补偿寄存器	32
0x50	MCAN_IR	MCAN 中断寄存器	32
0x54	MCAN_IE	MCAN 中断许可寄存器	32
0x58	MCAN_ILS	MCAN 中断线选择寄存器	32
0x5C	MCAN_ILE	MCAN 中断线使能寄存器	32
0x80	MCAN_GFC	MCAN 全局过滤器配置寄存器	32
0x84	MCAN_SIDFC	MCAN 标准 ID 过滤器配置寄存器	32
0x88	MCAN_XIDFC	MCAN 扩展 ID 过滤器配置寄存器	32
0x90	MCAN_XIDAM	MCAN 扩展 ID 与掩码寄存器	32
0x94	MCAN_HPMS	MCAN 高优先级消息状态寄存器	32
0x98	MCAN_NDAT1	MCAN 新数据 1 寄存器	32
0x9C	MCAN_NDAT2	MCAN 新数据 2 寄存器	32
0xA0	MCAN_RXF0C	MCAN 接收 FIFO0 配置寄存器	32

偏移地址	寄存器	描述	位宽
0xA4	MCAN_RXF0S	MCAN 接收 FIFO0 状态寄存器	32
0xA8	MCAN_RXF0A	MCAN 接收 FIFO0 确认寄存器	32
0xAC	MCAN_RXBC	MCAN 接收缓冲区配置寄存器	32
0xB0	MCAN_RXF1C	MCAN 接收 FIFO1 配置寄存器	32
0xB4	MCAN_RXF1S	MCAN 接收 FIFO1 状态寄存器	32
0xB8	MCAN_RXF1A	MCAN 接收 FIFO1 确认寄存器	32
0xBC	MCAN_RXESC	MCAN 接收缓冲区和 FIFO 元素大小配置寄存器	32
0xC0	MCAN_TXBC	MCAN 发送缓冲区配置寄存器	32
0xC4	MCAN_TXFQS	MCAN 发送 FIFO/队列状态寄存器	32
0xC8	MCAN_TXESC	MCAN 发送缓冲区元素大小配置寄存器	32
0xCC	MCAN_TXBRP	MCAN 发送缓冲区请求挂起寄存器	32
0xD0	MCAN_TXBAR	MCAN 发送缓冲区添加请求寄存器	32
0xD4	MCAN_TXBCR	MCAN 发送缓冲区取消请求寄存器	32
0xD8	MCAN_TXBTO	MCAN 发送缓冲区发送已发生寄存器	32
0xDC	MCAN_TXBCF	MCAN 发送缓冲区取消已完成寄存器	32
0xE0	MCAN_TXBTIE	MCAN 发送缓冲区发送中断使能寄存器	32
0xE4	MCAN_TXBCIE	MCAN 发送缓冲区取消已完成中断使能寄存器	32
0xF0	MCAN_TXEFC	MCAN 发送事件 FIFO 配置寄存器	32
0xF4	MCAN_TXEFS	MCAN 发送事件 FIFO 状态寄存器	32
0xF8	MCAN_TXEFA	MCAN 发送事件 FIFO 确认寄存器	32

MCANRAM ECC 基地址: 0x4002D000

表 37-20 MCANRAM ECC 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	MCANRAM_CKCR	MCAN 消息 RAM 校验控制寄存器	32
0x04	MCANRAM_CKSR	MCAN 消息 RAM 校验状态寄存器	32
0x08	MCANRAM_EIEN	MCAN 消息 RAM 错误注入使能寄存器	32
0x0C	MCANRAM_EIBIT0	MCAN 消息 RAM 错误注入位使能寄存器 0	32
0x10	MCANRAM_EIBIT1	MCAN 消息 RAM 错误注入位使能寄存器 1	32
0x14	MCANRAM_ECCERRADDR	MCAN 消息 RAM 校验错误地址寄存器	32

37.4.2 MCAN 字节序寄存器（MCAN_ENDN）

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x87654321																															
Name	ETV																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	ETV	字节序测试值	测试值为 0x87654321	RO

37.4.3 MCAN 数据位时间和预分频寄存器（MCAN_DBTP）

仅当 MCAN_CCCR.CCE 和 MCAN_CCCR.INIT 位均被置位时，才能对该寄存器进行写访问。CAN 位时间的可编程范围为 4 到 49 个时间片（Time quanta, tq）。CAN 时间片的可编程范围为 1 到 32 个 MCAN 通信时钟周期（Minimum time quantum, mtq）。 $tq = (DBRP + 1)mtq$ 。

DTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。DTSEG2 为 Phase_Seg2。

因此，位时间的长度为（编程值） $[DTSEG1 + DTSEG2 + 3] tq$ 或（功能值） $[Sync\_Seg + Prop\_Seg + Phase\_Seg1 + Phase\_Seg2] tq$ 。

信息处理时间（IPT）为零，这意味着下一位数据在采样点之后的第一个时钟边沿可用。

Offset	Bit Position																																
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000A33																																
Name	Reserved								TDC	Reserved				DBRP				Reserved				DTSEG1				DTSEG2				DSJW			
Access									RWA					RWA								RWA				RWA				RWA			

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23	TDC	发送延迟补偿	0b0：禁止 0b1：使能	RWA
22:21	Reserved	保留	-	-
20:16	DBRP	数据比特率预分频	生成位时间片的 MCAN 通信时钟预分频值，位时间为该时间片的倍数。预分频有效设定值为 0~31，当 TDC=1 时，范围被限制在 0 到 1。硬件实际使用值为编程值加 1。	RWA
15:13	Reserved	保留	-	-
12:8	DTSEG1	采样点之前的数据时间	有效设定值为 0~31，硬件实际使用值为编程值加 1。	RWA
7:4	DTSEG2	采样点之后的数据时间	有效设定值为 1~15，硬件实际使用值为编程值加 1。	RWA
3:0	DSJW		有效设定值为 0~15，硬件实际使用值为编程值加 1。	RWA

 **说明**

- 当 MCAN 通信时钟为 8MHz 时，复位值 0x00000A33 会将 MCAN 数据阶段（Data phase）的比特率配置为 500kbps。
- 通过 DBTP 配置的 CAN FD 数据阶段（Data phase）比特率不能小于通过 NBTP 配置的仲裁阶段（Arbitration phase）比特率。

37.4.4 MCAN 测试寄存器 (MCAN_TEST)

仅当 MCAN_CCCR.TEST 位被置 1 后，才能对该测试寄存器进行写访问。当 MCAN_CCCR.TEST 位被清 0 时，所有测试寄存器功能都会设为其复位值。

回环模式和软件控制 MCAN_TX 引脚属于硬件测试模式。TX 配置为非 0b00 时可能会干扰 CAN 总线上的消息传输。

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000000X0（Bit7 的复位值由 MCAN_RX 引脚的实际电平决定）																															
Name	Reserved										SVAL	TXBNS				Reserved	PVAL	TXBNP				RX	TX	LBCK	Reserved							
Access											RO	RO					RO	RO				RO	RWA	RWA								

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21	SVAL	已开始有效	0b0: TXBNS 的值无效 0b1: TXBNS 的值有效	RO
20:16	TXBNS	已开始发送的缓冲区编号	最后开始发送的发送缓冲区编号。 当 SVAL 位被置 1 时有效，有效设定值为 0~31。	RO
15:14	Reserved	保留	-	-
13	PVAL	已准备有效	0b0: TXBNP 的值无效 0b1: TXBNP 的值有效	RO
12:8	TXBNP	已准备好发送的缓冲区编号	准备好发送的发送缓冲区编号。 当 PVAL 位被置 1 时有效，有效设定值为 0~31。	RO
7	RX	接收引脚	监测到的 MCAN_RX 引脚的实际值 0b0: CAN 总线为显性 (MCAN_RX=0) 0b1: CAN 总线为隐性 (MCAN_RX=1)	RO
6:5	TX	发送引脚控制	0b00: 复位值，MCAN_TX 引脚由 CAN 内核控制，在 CAN 位时间结束时更新 0b01: 可以在 MCAN_TX 引脚监测采样点 0b10: MCAN_TX 引脚输出显性电平 (0) 0b11: MCAN_TX 引脚输出隐性电平 (1)	RWA
4	LBCK	回环模式	0b0: 复位值，禁止回环模式 0b1: 使能回环模式 (详见"回环模式")	RWA
3:0	Reserved	保留	-	-

37.4.5 MCAN RAM 看门狗寄存器（MCAN_RWD）

RAM 看门狗会监视消息 RAM 的 READY 信号输出。对消息 RAM 进行访问时，会启动消息 RAM 看门狗计数器，该计数器的起始值由 MCAN_RWD.WDC 位进行配置。

当消息 RAM 通过激活其 READY 信号输出，指示访问成功完成时，该计数器会重新载入 MCAN_RWD.WDC 的值。如果计数器递减为 0 之前消息 RAM 没有响应，计数器将停止计数，并将中断标志 MCAN_IR.WDI 位置 1。RAM 看门狗计数器由 MCAN 控制逻辑时钟计数。

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																WDV								WDC							
Access																	RO								RWA							

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	WDV	看门狗计数值	消息 RAM 看门狗计数器的实际值	RO
7:0	WDC	看门狗配置	消息 RAM 看门狗计数器的起始值。 如果使用初始值 0x00，则计数器被禁止。	RWA

37.4.6 MCAN CC 控制寄存器 (MCAN_CCCR)

寄存器各个位的设置条件请参见“[软件初始化](#)”。

Offset	Bit Position																																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
Reset	0x00000001																																															
Name	Reserved																	NISO		TXP		EFBI		PXHD		WMM		Reserved		BRSE		FDOE		TEST		DAR		MON		CSR		CSA		ASM		CCE		INIT
Access																		RWA	RWA	RWA	RWA	RWA			RWA	RWA	RWA	RW	RO	RWA	RWA	RW																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	NISO	非 ISO 模式选择	0b0: 使用 ISO 11898-1:2015 规定的 CAN FD 帧格式 0b1: 使用 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式	RWA
14	TXP	传输暂停	0b0: 禁止传输暂停 0b1: 使能传输暂停  说明 若该位被置为 1, 则 MCAN 在成功传输完一帧数据后会暂停两个位时间, 再进行下一帧数据的传输。	RWA
13	EFBI	总线同步期间的边沿滤波	0b0: 禁止边沿滤波 0b1: 需要两个连续的显性 tq 才能检测硬同步边沿	RWA
12	PXHD	协议异常处理禁止	0b0: 使能协议异常处理 0b1: 禁止协议异常处理  说明 当协议异常处理被禁止时, MCAN 将在检测到协议异常条件时传输错误帧。	RWA
11	WMM	宽消息标记	0b0: 使用 8 位消息标记 0b1: 使用 16 位消息标记, 替换发送事件 FIFO 中的 16 位时间戳  说明 当使用 16 位宽消息标记 (WMM=1) 时, 发送事件 FIFO 中的 16 位内部时间戳是无效的。	RWA
10	Reserved	保留	-	-
9	BRSE	比特率切换	0b0: 禁止发送时进行比特率切换 0b1: 使能发送时进行比特率切换  说明 当 CAN FD 模式无效 (FDOE=0) 时, BRSE 的设置不起作用。	RWA

位/位域	标记	位名	功能描述	读写
8	FDOE	FD 模式使能	0b0: 禁止 FD 模式 0b1: 使能 FD 模式	RWA
7	TEST	测试模式使能	写 1 使能测试模式, 写 0 无效 读取: 0b0: 正常模式, 测试寄存器 MCAN_TEST 保持复位值 0b1: 测试模式, 使能写访问测试寄存器 MCAN_TEST	RWA
6	DAR	禁止自动重发	0b0: 使能允许自动重发未成功发送的消息 0b1: 禁止自动重发	RWA
5	MON	总线监听模式	写 1 使能总线监听模式, 写 0 无效 读取: 0b0: 禁止总线监听模式 0b1: 使能总线监听模式  说明 仅当 CCE 和 INIT 位均置 1 时, 主机才能将 MON 位置 1。此位可随时通过主机复位。	RWA
4	CSR	时钟停止请求	0b0: 未请求时钟停止 0b1: 已请求时钟停止  说明 当已请求时钟停止时, 在所有挂起的传输请求均已完成且 CAN 总线达到空闲状态之后, INIT 位和 CSA 位将会依次被置 1。	RW
3	CSA	时钟停止确认	0b0: 未确认时钟停止请求 0b1: 已确认时钟停止请求, 可以通过停止控制逻辑时钟和通信时钟将 MCAN 设为掉电状态	RO
2	ASM	受限操作模式	写 1 使能受限工作模式, 写 0 无效 读取: 0b0: 正常模式 0b1: 受限工作模式  说明 仅当 CCE 和 INIT 均置 1 时, 主机才能将 ASM 位置 1。此位可随时通过主机复位。 有关受限工作模式的说明, 请参见“受限工作模式”。	RWA
1	CCE	配置更改使能	0b0: CPU 对受保护的配置寄存器无写访问权限 0b1: CPU 对受保护的配置寄存器有写访问权限 (当 INIT=1 时)	RWA
0	INIT	初始化	0b0: 正常工作 0b1: 启动初始化  说明 由于两个时钟域之间存在同步机制, INIT 的值在写入后可能会有一定延迟才能被正确读取。因此, 在重新设定 INIT 值之前, 请确认之前的设定值确实已被写入。	RW

37.4.7 MCAN 标称位时间和预分频寄存器（MCAN_NBTP）

仅当 MCAN_CCCR.CCE 和 MCAN_CCR.INIT 位均被置 1 时，才能对该寄存器进行写访问。CAN 标称位时间的可编程范围为 4 到 385 个时间片（Time quanta, tq）。CAN 时间片的可编程范围为 1 到 512 个 MCAN 通信时钟周期（Minimum time quantum, mtq）。 $tq = (NBRP + 1)mtq$ 。

NTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。NTSEG2 为 Phase_Seg2。

因此，位时间的长度为（编程值） $[NTSEG1 + NTSEG2 + 3] tq$ 或（功能值） $[Sync\_Seg + Prop\_Seg + Phase\_Seg1 + Phase\_Seg2] tq$ 。

信息处理时间（IPT）为零，这意味着下一位数据在采样点之后的第一个时钟边沿可用。

Offset	Bit Position																																
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x06000A03																																
Name	NSJW								NBRP								NTSEG1								Reserved	NTSEG2							
Access	RWA								RWA								RWA									RWA							

位/位域	标记	位名	功能描述	读写
31:25	NSJW	标称（重）同步 跳转宽度	有效设定值为 0~127，硬件实际使用值为编程值加 1。	RWA
24:16	NBRP	标称比特率预分 频	生成位时间片的 MCAN 通信时钟预分频值，位时间为该时间片的倍数。预分频有效设定值为 0~511，硬件实际使用值为编程值加 1。	RWA
15:8	NTSEG1	采样点之前的标 称时间段	有效设定值为 1~255，硬件实际使用值为编程值加 1。	RWA
7	Reserved	保留	-	-
6:0	NTSEG2	采样点之后的标 称时间段	有效设定值为 1~127，硬件实际使用值为编程值加 1。	RWA

 说明

当 MCAN 时钟为 8MHz 时，复位值 0x06000A03 会将 MCAN 的比特率配置为 500kbps。

37.4.8 MCAN 时间戳计数器配置寄存器（MCAN_TSCC）

该寄存器用于配置内部 16 位时间戳计数器。内部时间戳处理的介绍请参见["时间戳生成"](#)。

Offset	Bit Position																															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												TCP		Reserved																TSS	
Access													RWA																		RWA	

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19:16	TCP	时间戳计数器预分频	将时间戳和超时计数器时间单位配置为 CAN 位时间的 1~16 倍。有效设定值为 0~15，硬件实际使用值为编程值加 1。	RWA
15:2	Reserved	保留	-	-
1:0	TSS	时间戳选择	0b00：时间戳计数器的值一直为 0x0000 0b01：时间戳计数器的值根据 TCP 递增 0b10：禁止设置 0b11：时间戳计数器的值一直为 0x0000（等同于设定值“0b00”）	RWA

37.4.9 MCAN 时间戳计数器值寄存器（MCAN_TSCV）

Offset	Bit Position															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x00000000															
Name	Reserved								TSC							
Access									RWC							

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TSC	时间戳计数器	在接收到或发送 SOF（Start Of Frame）时捕获的内部时间戳计数器值。 当 MCAN_TSCC.TSS=0b01 时，时间戳计数器以 CAN 位时间的 1~16 倍进行递增，具体倍数由 MCAN_TSCC.TCP 的设定值决定。计数回卷时，中断标志 MCAN_IR.TSW 会被置 1。写操作会将计数器清零。 说明 “回卷”是指时间戳计数器的值从非零变为零，并不是由于 MCAN_TSCV 的写操作引起的清零。 - 字节访问时，对该寄存器的任何一个字节进行写操作都会导致时间戳寄存器被清零。	RWC

37.4.10 MCAN 超时计数器配置寄存器（MCAN_TOCC）

超时计数器的介绍请参见“[超时计数器](#)”。

Offset	Bit Position																																		
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0xFFFF0000																																		
Name	TOP																Reserved																TOS		ETOC
Access																																	RW		
																	RWA																		

位/位域	标记	位名	功能描述	读写
31:16	TOP	超时周期	超时计数器（递减计数器）的起始值，配置超时周期	RW
15:3	Reserved	保留	-	-
2:1	TOS	超时选择	在连续模式下操作时，对 MCAN_TOCV 进行写访问会将计数器预设为 MCAN_TOCC.TOP 配置的值，并继续向下递减。当超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为 MCAN_TOCC.TOP 配置的值。当第一个 FIFO 元素被存储后，计数器开始递减计数： ● 0b00：连续模式 ● 0b01：由发送事件 FIFO 控制超时 ● 0b10：由接收 FIFO0 控制超时 ● 0b11：由接收 FIFO1 控制超时	RWA
0	ETOC	超时计数器使能	● 0b0：禁止 ● 0b1：使能	RWA

37.4.11 MCAN 超时计数器值寄存器（MCAN_TOCV）

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																TOC															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TOC	超时计数器	超时计数器以 CAN 位时间的 1~16 倍进行递减，具体倍数由 MCAN_TSCC.TCP 的设定值决定。当计数器递减到零时，中断标志位 MCAN_IR.TOO 会被置 1，计数器停止计数。可以通过 MCAN_TOCC.TOS 来设置启动和复位（或重启）条件。  说明 字节访问时，当 MCAN_TOCC.TOS=0b00 时，对该寄存器的任何一个字节进行写操作都会将超时计数器预设为 MCAN_TOCC.TOP 配置的值。	RW

37.4.12 MCAN 错误计数器寄存器 (MCAN_ECR)

Offset	Bit Position																																
0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved								RC CEL								RP	REC								TEC							
Access									RC								RO	RO								RO							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	CEL	CAN 错误记录	当 CAN 协议错误导致发送错误计数器 TEC 或接收错误计数器 REC 递增时，该错误记录计数器 CEL 都会递增。计数器值达到 0xFF 时，会停止计数。只有 RP 位置位但 REC 没有变化时，CEL 不会递增。CEL 在 REC 和 TEC 增加后递增。 对 CEL 进行读访问会复位计数器。计数器值达到 0xFF 时，停止计数，TEC 或 REC 下一次增加时中断标志 MCAN_IR.ELO 置位。  说明 字节访问时，读取寄存器的字节 2 会将 CEL 清零，读取字节 3、1 或 0 没有影响。	RC
15	RP	接收错误被动	0b0：接收错误计数器低于错误被动级别 128 0b1：接收错误计数器已达到错误被动级别 128	RO
14:8	REC	接收错误计数器	接收错误计数器的实际状态，计数值范围为 0~127	RO
7:0	TEC	发送错误计数器	发送错误计数器的实际状态，计数值范围为 0~255	RO



说明

如果 MCAN_CCCR.ASM 被置为 1，则检测到 CAN 协议错误时，CAN 协议控制器不会使 TEC 和 REC 递增，但 CEL 仍会递增。

37.4.13 MCAN 协议状态寄存器 (MCAN_PSR)

Offset	Bit Position																															
0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000707																															
Name	Reserved								TDCV								Reserved	PXE	RFDF	RBRS	RESI	DLEC				BO	EW	EP	ACT		LEC	
Access									RO									RO	RO	RO	RO	RO				RO	RO	RO	RO		RS	

位/位域	标记	位名	功能描述	读写
31:23	Reserved	保留	-	-
22:16	TDCV	发送延迟补偿值	实际的发送延迟补偿值，有效设定值为 0~127 个 mtq。 第二采样点 (Secondary Sample Point, SSP) 的位置，由 MCAN_TX 引脚到 MCAN_RX 引脚之间测得的延迟值与 MCAN_TDCR.TDCO 之和决定。在数据阶段 (Data phase)，SSP 位置是发送位的起始点与第二采样点间的 mtq 数。	RO
15	Reserved	保留	-	-
14	PXE	协议异常事件	0b0: 自上次读访问后未发生协议异常事件 0b1: 已发生协议异常事件  说明 字节访问时，读取寄存器的字节 0 会将该位清零，读取字节 3/2/1 没有影响。	RO
13	RFDF	收到 CAN FD 消息	该位的置位与接收过滤无关。 0b0: 自从上次被 CPU 清零后未收到 CAN FD 格式的消息 0b1: 收到 FDF 标志为 1 的 CAN FD 格式的消息  说明 字节访问时，读取寄存器的字节 0 会将该位清零，读取字节 3/2/1 没有影响。	RO
12	RBRS	上次收到的 CAN FD 消息的 BRS 标志	该位与 RFDF 一起置位，与接收过滤无关。 0b0: 上次收到的 CAN FD 消息的 BRS 标志未置位 0b1: 上次收到的 CAN FD 消息的 BRS 标志已置位  说明 字节访问时，读取寄存器的字节 0 会将该位清零，读取字节 3/2/1 没有影响。	RO

位/位域	标记	位名	功能描述	读写
11	RESI	上次收到的 CAN FD 消息的 ESI 标志	该位与 RFDF 一起被设置，与验收过滤器无关。 0b0: 上次收到的 CAN FD 消息的 ESI 标志未置位 0b1: 上次收到的 CAN FD 消息的 ESI 标志已置位  说明 字节访问时，读取寄存器的字节 0 会将该位清零，读取字节 3/2/1 没有影响。	RO
10:8	DLEC	数据阶段最后一次错误码	在设置了 BRS 标志的 CAN FD 帧的数据阶段 (Data phase) 中发生的最后一个错误的类型。错误码与 LEC 相同。当设置了 BRS 标志的 CAN FD 帧传输 (接收或发送) 没有错误时，DLEC 将被清零。  说明 字节访问时，读取寄存器的字节 0 会将该位设置为 0b111，读取字节 3/2/1 没有影响。	RO
7	BO	Bus_Off 状态	0b0: MCAN 未处于 Bus_Off 状态 0b1: MCAN 处于 Bus_Off 状态	RO
6	EW	警告状态	0b0: 两个错误计数器均小于错误警告 (Error_Warning) 上限 96 0b1: 至少有一个错误计数器已达到错误警告 (Error_Warning) 上限 96	RO
5	EP	错误被动	0b0: MCAN 处于错误主动 (Error_Active) 状态。MCAN 正常参与总线通信，并在检测到错误时发送主动错误标志 0b1: MCAN 处于错误被动 (Error_Passive) 状态	RO
4:3	ACT	通信状态	监测 MCAN 的通信状态： 0b00: 同步中，节点正在进行 CAN 通信同步 0b01: 空闲，节点既不是接收节点也不是发送节点 0b10: 接收，节点作为接收节点工作 0b11: 发送，节点作为发送节点工作  说明 协议异常事件会将该位设置为 0b00。	RO
2:0	LEC	最后一次错误码	0b000: 没有错误 (No Error)。自成功接收或发送使 LEC 复位之后，没有错误发生 0b001: 填充错误 (Stuff Error)。在已接收到的消息中，某一部分出现了 5 个以上的相同位，这种情况是不允许的 0b010: 格式错误 (Form Error)。已接收帧的固定格式部分有格式错误 0b011: ACK 错误 (ACK Error)。MCAN 发送的消息没有被其他节点确认 0b100: Bit1 错误 (Bit1 Error)。在消息的发送过程中 (仲裁场 Arbitration field 除外)，设备发送了一个隐性电平 (逻辑值 1)，但是监测到的总线电平为显性 (逻辑值 0) 0b101: Bit0 错误 (Bit0 Error)。在消息的发送过程中 (包括 ACK 位、主动错误标志、过载标志)，设备发送了一个显性电平 (数据或 ID 的逻	RS

位/位域	标记	位名	功能描述	读写
2:0	LEC	最后一次错误码	辑值 0)，但是监测到的总线电平为隐性（逻辑值 1）。在总线 Bus_Off 恢复期间，每次监测到 11 位隐性位序列时，都会设置为该状态。这样，CPU 就能够监测总线恢复序列的进程（表明总线未卡在显性状态或持续受到干扰） ● 0b110: CRC 错误 (CRC Error)。已接收消息的 CRC 有错误。收到的 CRC 和用接收到的数据计算出的 CRC 不匹配 ● 0b111: 无变化 (No Change)。对协议状态寄存器的任何读访问会将 LEC 重新初始化为 7。当 LEC 显示值为 7 时，表示自上次 CPU 对协议状态寄存器进行读访问之后，没有检测到 CAN 总线事件  说明 1. 当设置了 BRS 标志的 CAN FD 帧已经到达数据阶段 (Data phase) 时，下一个 CAN 事件（错误或有效帧）将显示在 DLEC 中而不是 LEC 中。CAN FD CRC 序列中的固定填充位的错误将显示为格式错误 (Form Error) 而不是填充错误 (Stuff Error)。 2. Bus_Off 恢复序列（请参考 ISO11898-1:2015）不能通过置位或清零 MCAN_CCCR.INIT 来缩短。如果设备进入 Bus_Off 状态，它将自动置位 MCAN_CCCR.INIT，停止所有总线活动。一旦 MCAN_CCCR.INIT 被 CPU 清零，设备在恢复正常操作之前，将等待 129 个总线空闲（129*11 个连续的隐性位）。在 Bus_Off 恢复序列结束时，错误管理计数器将被复位。在清零 MCAN_CCCR.INIT 后的等待期间，每次监测到一个 11 位隐性序列时，就向 MCAN_PSR.LEC 写入一个 Bit0 错误码，使 CPU 能够随时检查 CAN 总线是否处于显性状态或持续被干扰，并监测 Bus_Off 恢复序列。MCAN_ECR.REC 用于对这些序列进行计数。 3. 字节操作时，读取寄存器的字节 0 会将该位设置为 0b111，读取字节 3/2/1 没有影响。	RS

37.4.14 MCAN 发送延迟补偿寄存器（MCAN_TDCR）

Offset	Bit Position																															
0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TDCO						Reserved	TDCF								
Access																	RWA							RWA								

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14:8	TDCO	发送延迟补偿 SSP 偏移	该偏移值指从 MCAN_TX 引脚到 MCAN_RX 引脚测得的延迟与第二采样点（SSP）之间的距离。有效设定值为 0~127mtq	RWA
7	Reserved	保留	-	-
6:0	TDCF	发送延迟补偿滤波窗口宽度	定义 SSP 位置的最小值，对于发送延迟的测量，将忽略 MCAN_RX 引脚上会导致 SSP 位置提前的显性边沿。当 TDCF 设定值大于 TDCO 时，该功能被启用。有效设定值为 0~127mtq	RWA

37.4.15 MCAN 中断寄存器 (MCAN_IR)

当如下列表中的一个条件被检测到时，对应标志位会置位。在主机将标志位清零之前，保持置位状态。**所有标志位写 1 清零，写 0 无效。**硬复位会将寄存器清零。是否产生中断，由寄存器 MCAN_IE 控制；哪条中断线产生中断，由寄存器 MCAN_ILS 控制。

Offset	Bit Position																																																												
0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																													
Reset	0x00000000																																																												
Name	Reserved																																																												
Access		RW	ARA	RW	PED	RW	PEA	RW	WDI	RW	BO	RW	EW	RW	EP	RW	ELO	RW	BEU	RW	BEC	RW	DRX	RW	TOO	RW	MRAF	RW	TSW	RW	TEFL	RW	TEFF	RW	TEFW	RW	TEFN	RW	TFE	RW	TCF	RW	TC	RW	HPM	RW	RF1L	RW	RF1F	RW	RF1W	RW	RF1N	RW	RF0L	RW	RF0F	RW	RF0W	RW	RF0N

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	ARA	访问保留地址	0b0: 未对保留地址进行访问 0b1: 已对保留地址进行访问	RW
28	PED	数据阶段协议错误 (使用数据位时间)	0b0: 未检测到数据阶段协议错误 0b1: 检测到数据阶段协议错误 (MCAN_PSR.DLEC≠0b000/0b111)	RW
27	PEA	仲裁阶段协议错误 (使用标称位时间)	0b0: 未检测到仲裁阶段协议错误 0b1: 检测到仲裁阶段协议错误 (MCAN_PSR.LEC≠0b000/0b111)	RW
26	WDI	看门狗中断	0b0: 未发生消息 RAM 看门狗事件 0b1: 因 READY 信号缺失而发生消息 RAM 看门狗事件	RW
25	BO	Bus_Off 状态	0b0: Bus_Off 状态未改变 0b1: Bus_Off 状态已改变	RW
24	EW	警告状态	0b0: 错误警告 (Error_Warning) 状态未改变 0b1: 错误警告 (Error_Warning) 状态已改变	RW
23	EP	错误被动	0b0: 错误被动 (Error_Passive) 状态未改变 0b1: 错误被动 (Error_Passive) 状态已改变	RW
22	ELO	错误记录溢出	0b0: CAN 错误记录计数器未溢出 0b1: CAN 错误记录计数器已溢出	RW
21	BEU	位错误未纠正	0b0: 从消息 RAM 读取时未检测到位错误 0b1: 当 MCANRAM_CKCR.ECCMOD 不为 0 时，从消息 RAM 读取时检测到位错误且未纠正  说明 BEU 置位的同时 MCAN_CCCR.INIT 置位。	RW
20	BEC	位错误已纠正	0b0: 从消息 RAM 读取时未检测到位错误 0b1: 当 MCANRAM_CKCR.ECCMOD 为 0b10 或 0b11 时，从消息 RAM 读取时检测到位错误且已纠正	RW

位/位域	标记	位名	功能描述	读写
19	DRX	消息存储到专用接收缓冲区	● 0b0: 接收缓存区未更新 ● 0b1: 至少有一条消息已存储到接收缓冲区  说明 每当收到的消息已存储到专用接收缓冲区时，都会置位该标志。	RW
18	TOO	发生超时	● 0b0: 未发生超时 ● 0b1: 已发生超时	RW
17	MRAF	消息 RAM 访问失败	● 0b0: 未发生消息 RAM 访问失败 ● 0b1: 发生了消息 RAM 访问失败 接收处理中置位条件: ● 在接收到后续消息的仲裁场 (Arbitration field) 之前，尚未完成对已接收消息的接收过滤或存储。在这种情况下，接收过滤或消息存储被终止，接收处理单元将开始处理后续消息。 ● 无法向消息 RAM 写入消息，消息存储将终止。  说明 在这两种情况下，FIFO 放入索引都不会更新，专用接收缓冲区的新数据标志不会置位，当下一条消息存储到该位置时，已部分存储的消息会被覆盖。 发送处理中置位条件: ● 发送处理单元无法及时从消息 RAM 读取消息。在这种情况下，消息发送将终止。在发送处理单元访问失败的情况下，MCAN 被切换到受限工作模式（详见受限工作模式）。要退出受限工作模式，主机 CPU 必须复位 MCAN_CCCR.ASM。	RW
16	TSW	时间戳回卷	● 0b0: 时间戳计数器未回卷 ● 0b1: 时间戳计数器已回卷	RW
15	TEFL	发送事件 FIFO 元素丢失	● 0b0: 发送事件 FIFO 元素未丢失 ● 0b1: 发送事件 FIFO 元素有丢失；对空间大小为 0 的发送事件 FIFO 进行写操作时，该位也会置位	RW
14	TEFF	发送事件 FIFO 已满	● 0b0: 发送事件 FIFO 未滿 ● 0b1: 发送事件 FIFO 已滿	RW
13	TEFW	发送事件 FIFO 到达水位线	● 0b0: 发送事件 FIFO 的填充级别低于水位线 ● 0b1: 发送事件 FIFO 的填充级别达到水位线	RW
12	TEFN	发送事件 FIFO 新条目	● 0b0: 发送事件 FIFO 无变化 ● 0b1: 发送处理单元写入了发送事件 FIFO 元素	RW
11	TFE	发送 FIFO 为空	● 0b0: 发送 FIFO 非空 ● 0b1: 发送 FIFO 为空	RW
10	TCF	发送取消已完成	● 0b0: 发送取消未完成 ● 0b1: 发送取消已完成	RW
9	TC	发送完成	● 0b0: 发送未完成 ● 0b1: 发送已完成	RW
8	HPM	高优先级消息	● 0b0: 未收到高优先级消息 ● 0b1: 已收到高优先级消息	RW

位/位域	标记	位名	功能描述	读写
7	RF1L	接收 FIFO1 消息丢失	0b0: 接收 FIFO1 消息未丢失 0b1: 接收 FIFO1 消息有丢失; 对空间大小为 0 的接收 FIFO1 进行写操作时, 该位也会置位	RW
6	RF1F	接收 FIFO1 已满	0b0: 接收 FIFO1 未满 0b1: 接收 FIFO1 已满	RW
5	RF1W	接收 FIFO1 到达水位线	0b0: 接收 FIFO1 的填充级别低于水位线 0b1: 接收 FIFO1 的填充级别达到水位线	RW
4	RF1N	接收 FIFO1 新消息	0b0: 没有新消息写入接收 FIFO1 0b1: 有新消息写入接收 FIFO1	RW
3	RF0L	接收 FIFO0 消息丢失	0b0: 接收 FIFO0 消息未丢失 0b1: 接收 FIFO0 消息有丢失; 对空间大小为 0 的接收 FIFO0 进行写操作时, 该位也会置位	RW
2	RF0F	接收 FIFO0 已满	0b0: 接收 FIFO0 未满 0b1: 接收 FIFO0 已满	RW
1	RF0W	接收 FIFO0 到达水位线	0b0: 接收 FIFO0 的填充级别低于水位线 0b1: 接收 FIFO0 的填充级别达到水位线	RW
0	RF0N	接收 FIFO0 新消息	0b0: 没有新消息写入接收 FIFO0 0b1: 有新消息写入接收 FIFO0	RW

37.4.16 MCAN 中断使能寄存器 (MCAN_IE)

中断使能寄存器中的设置，决定中断寄存器 (MCAN_IR) 中的哪些状态变化会指示在中断线上。

Offset	Bit Position																															
0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	BEUE	BECE	DRXE	TOOE	MRAFE	TSWE	TEFLE	TEFFE	TEFWE	TEFNE	TTEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE	
Access		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	ARAE	访问保留地址中断使能	0b0: 禁止中断 0b1: 使能中断	RW
28	PEDE	数据阶段协议错误中断使能	0b0: 禁止中断 0b1: 使能中断	RW
27	PEAE	仲裁阶段协议错误中断使能	0b0: 禁止中断 0b1: 使能中断	RW
26	WDIE	看门狗中断使能	0b0: 禁止中断 0b1: 使能中断	RW
25	BOE	Bus_Off 状态中断使能	0b0: 禁止中断 0b1: 使能中断	RW
24	EWE	警告状态中断使能	0b0: 禁止中断 0b1: 使能中断	RW
23	EPE	错误被动中断使能	0b0: 禁止中断 0b1: 使能中断	RW
22	ELOE	错误记录溢出中断使能	0b0: 禁止中断 0b1: 使能中断	RW
21	BEUE	位错误未纠正中断使能	0b0: 禁止中断 0b1: 使能中断	RW
20	BECE	位错误已纠正中断使能	0b0: 禁止中断 0b1: 使能中断	RW
19	DRXE	消息存储到专用接收缓冲区中断使能	0b0: 禁止中断 0b1: 使能中断	RW
18	TOOE	发生超时中断使能	0b0: 禁止中断 0b1: 使能中断	RW
17	MRAFE	消息 RAM 访问失败中断使能	0b0: 禁止中断 0b1: 使能中断	RW
16	TSWE	时间戳回卷中断使能	0b0: 禁止中断 0b1: 使能中断	RW

位/位域	标记	位名	功能描述	读写
15	TEFLE	发送事件 FIFO 元素丢失中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
14	TEFFE	发送事件 FIFO 已满中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
13	TEFWE	发送事件 FIFO 到达水位线中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
12	TEFNE	发送事件 FIFO 新条目中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
11	TFEE	发送 FIFO 为空中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
10	TCFE	发送取消已完成中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
9	TCE	发送完成中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
8	HPME	高优先级消息中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
7	RF1LE	接收 FIFO1 消息丢失中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
6	RF1FE	接收 FIFO1 已满中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
5	RF1WE	接收 FIFO1 到达水位线中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
4	RF1NE	接收 FIFO1 新消息中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
3	RF0LE	接收 FIFO0 消息丢失中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
2	RF0FE	接收 FIFO0 已满中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
1	RF0WE	接收 FIFO0 到达水位线中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW
0	RF0NE	接收 FIFO0 新消息中断使能	● 0b0: 禁止中断 ● 0b1: 使能中断	RW

37.4.17 MCAN 中断线选择寄存器 (MCAN_ILS)

中断线选择寄存器，将中断寄存器 (MCAN_IR) 中特定中断标志生成的中断分配到两条中断线之一。要产生中断，必须通过 MCAN_ILE.EINT0 和 MCAN_ILE.EINT1 使能相应的中断线。

Offset	Bit Position																																																												
0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																													
Reset	0x00000000																																																												
Name	Reserved																																																												
Access		RW	ARAL	RW	PEDL	RW	PEAL	RW	WDIL	RW	BOL	RW	EWL	RW	EPL	RW	ELOL	RW	BEUL	RW	BECL	RW	DRXL	RW	TOOL	RW	MRAFL	RW	TSWL	RW	TEFLL	RW	TEFFL	RW	TEFWL	RW	TEFNL	RW	TEFEL	RW	TCFL	RW	TCL	RW	HPML	RW	RFILL	RW	RFIFL	RW	RFIWL	RW	RFINL	RW	RFOLL	RW	RF0FL	RW	RF0WL	RW	RF0NL

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	ARAL	访问保留地址中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
28	PEDL	数据阶段协议错误中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
27	PEAL	仲裁阶段协议错误中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
26	WDIL	看门狗中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
25	BOL	Bus_Off 状态中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
24	EWL	警告状态中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
23	EPL	错误被动中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
22	ELOL	错误记录溢出中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
21	BEUL	位错误未纠正中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
20	BECL	位错误已纠正中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
19	DRXL	消息存储到专用接收缓冲区中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
18	TOOL	发生超时中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
17	MRAFL	消息 RAM 访问失败中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW
16	TSWL	时间戳回卷中断线	0b0: 选择中断线 0 0b1: 选择中断线 1	RW

位/位域	标记	位名	功能描述	读写
15	TEFLL	发送事件 FIFO 元素丢失中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
14	TEFFL	发送事件 FIFO 已满中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
13	TEFWL	发送事件 FIFO 到达水位线中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
12	TEFNL	发送事件 FIFO 新条目中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
11	TFEL	发送 FIFO 为空中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
10	TCFL	发送取消已完成中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
9	TCL	发送完成中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
8	HPML	高优先级消息中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
7	RF1LL	接收 FIFO1 消息丢失中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
6	RF1FL	接收 FIFO1 已满中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
5	RF1WL	接收 FIFO1 到达水位线中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
4	RF1NL	接收 FIFO1 新消息中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
3	RF0LL	接收 FIFO0 消息丢失中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
2	RF0FL	接收 FIFO0 已满中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
1	RF0WL	接收 FIFO0 到达水位线中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW
0	RF0NL	接收 FIFO0 新消息中断线	● 0b0: 选择中断线 0 ● 0b1: 选择中断线 1	RW

37.4.18 MCAN 中断线使能寄存器（MCAN_ILE）

连接到 CPU 的两条中断线中的每一条，都可以通过设置 EINT0 和 EINT1 分别使能或禁止。

Offset	Bit Position																															
0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														EINT1	EINT0
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	EINT1	中断线 1 使能	● 0b0：禁止 ● 0b1：使能	RW
0	EINT0	中断线 0 使能	● 0b0：禁止 ● 0b1：使能	RW

37.4.19 MCAN 全局过滤器配置寄存器 (MCAN_GFC)

用于消息 ID 过滤的全局设置。全局过滤器配置，控制标准消息和扩展消息的过滤路径，请参见“[标准消息 ID 过滤](#)”和“[扩展消息 ID 过滤](#)”。

Offset	Bit Position																																			
0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																												ANFS		ANFE		RRFS		RRFE	
Access																																				

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5:4	ANFS	接受不匹配的标准帧	定义如何处理，已接收的与过滤器列表任何元素都不匹配的 11 位标准 ID 消息： ● 0b00：在接收 FIFO0 中接受不匹配的标准帧 ● 0b01：在接收 FIFO1 中接受不匹配的标准帧 ● 0b10：拒绝不匹配的标准帧 ● 0b11：拒绝不匹配的标准帧	RWA
3:2	ANFE	接受不匹配的扩展帧	定义如何处理，已接收的与过滤器列表任何元素都不匹配的 29 位扩展 ID 消息： ● 0b00：在接收 FIFO0 中接受不匹配的扩展帧 ● 0b01：在接收 FIFO1 中接受不匹配的扩展帧 ● 0b10：拒绝不匹配的扩展帧 ● 0b11：拒绝不匹配的扩展帧	RWA
1	RRFS	拒绝远程标准帧	● 0b0：过滤 11 位标准 ID 的远程帧 ● 0b1：拒绝所有 11 位标准 ID 的远程帧	RWA
0	RRFE	拒绝远程扩展帧	● 0b0：过滤 29 位扩展 ID 的远程帧 ● 0b1：拒绝所有 29 位扩展 ID 的远程帧	RWA

37.4.20 MCAN 标准 ID 过滤器配置寄存器 (MCAN_SIDFC)

用于 11 位标准 ID 过滤器的设置。标准 ID 过滤器配置，控制标准消息的过滤路径请参见“[标准消息 ID 过滤](#)”。

Offset	Bit Position																															
0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								LSS								FLSSA[15:2]								Reserved							
Access									RWA								RWA															

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	LSS	标准 ID 过滤器列表大小	0: 无标准 ID 过滤器 1~128: 标准 ID 过滤器元素数量 >128: 大于 128 的设定值被解析为 128	RWA
15:2	FLSSA[15:2]	标准 ID 过滤器列表起始地址	设置标准 ID 过滤器列表的起始地址 (32 位字地址, 请参见“ 消息 RAM 配置 ”)	RWA
1:0	Reserved	保留	-	-

37.4.21 MCAN 扩展 ID 过滤器配置寄存器 (MCAN_XIDFC)

用于 29 位扩展 ID 过滤器的设置。扩展 ID 过滤器配置，控制扩展消息的过滤路径请参见“[扩展消息 ID 过滤](#)”。

Offset	Bit Position																															
0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								LES								FLESA[15:2]								Reserved							
Access									RWA								RWA															

位/位域	标记	位名	功能描述	读写
31:23	Reserved	保留	-	-
22:16	LES	标准 ID 过滤器列表大小	0: 无扩展 ID 过滤器 1~64: 扩展 ID 过滤器元素数量 >64: 大于 64 的设定值被解析为 64	RWA
15:2	FLESA[15:2]	扩展 ID 过滤器列表起始地址	设置扩展 ID 过滤器列表的起始地址 (32 位字地址, 请参见“ 消息 RAM 配置 ”)	RWA
1:0	Reserved	保留	-	-

37.4.22 MCAN 扩展 ID 与掩码寄存器 (MCAN_XIDAM)

Offset	Bit Position																															
0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x1FFFFFFF																															
Name	Reserved		EIDM																													
Access																																

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:0	EIDM	扩展 ID 掩码	对于扩展帧的接收过滤，会将 EIDM 与已接收帧的 ID 进行与运算。用于屏蔽 SAE J1939 中的 29 位 ID。所有位的复位值都被设置为 1，掩码无效。	RWA

37.4.23 MCAN 高优先级消息状态寄存器 (MCAN_HPMS)

配置为生成优先级事件的消息 ID 过滤器元素，每次匹配时都会更新此寄存器。这可用于监测收到的高优先级消息的状态，并可对这些消息进行快速访问。

Offset	Bit Position																															
0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved															RO	FLST	RO FIDX					RO	MSI	RO BIDX							
Access																RO		RO						RO		RO						

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15	FLST	过滤器列表	指示匹配的过滤器列表： 0b0：标准过滤器列表 0b1：扩展过滤器列表	RO
14:8	FIDX	过滤器索引	匹配的过滤器元素的索引。范围为 0 到 MCAN_SIDFC.LSS 减 1 或 MCAN_XIDFC.LSE 减 1	RO
7:6	MSI	消息存储器指示符	0b00：未选择 FIFO 0b01：FIFO 消息丢失 0b10：消息存储在 FIFO0 中 0b11：消息存储在 FIFO1 中	RO
5:0	BIDX	缓存区索引	存储消息的接收 FIFO 元素的索引。仅当 MSI[1]=1 时有效	RO

37.4.24 MCAN 新数据 1 寄存器 (MCAN_NDAT1)

Offset	Bit Position																															
0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16	ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	NDn (n=31~0)	新数据	对应保存接收缓冲区 31~0 的新数据标志： ● 0b0：接收缓冲区未更新 ● 0b1：接收缓冲区通过新消息更新 当接收帧更新接收缓冲区时，对应的标志位会置 1。在主机将标志位清零之前，标志位保持置 1 状态。通过向对应位写 1 将标志位清零，写 0 无效。硬复位会将寄存器清零。  说明 每个 Bit 对应一个缓冲区。	RW

37.4.25 MCAN 新数据 2 寄存器 (MCAN_NDAT2)

Offset	Bit Position																															
0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48	ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	NDn (n=63~32)	新数据	对应保存接收缓冲区 63~32 的新数据标志： ● 0b0：接收缓冲区未更新 ● 0b1：接收缓冲区通过新消息更新 当接收帧更新接收缓冲区时，对应的标志位会置 1。在主机将标志位清零之前，标志位保持置 1 状态。通过向对应位写 1 将标志位清零，写 0 无效。硬复位会将寄存器清零。  说明 每个 Bit 对应一个缓冲区。	RW

37.4.26 MCAN 接收 FIFO0 配置寄存器 (MCAN_RXF0C)

Offset	Bit Position																															
0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name									Reserved								F0SA[15:2]															Reserved
	F0OM	F0WM								F0S																						
Access	RWA	RWA								RWA							RWA															

位/位域	标记	位名	功能描述	读写
31	F0OM	接收 FIFO0 工作模式	0b0: 接收 FIFO0 为阻塞模式 0b1: 接收 FIFO0 为覆盖模式	RWA
30:24	F0WM	接收 FIFO0 水位线	0: 禁止接收 FIFO0 水位线中断 1~64: 接收 FIFO0 水位线中断 (MCAN_IR.RF0W) 的级别 >64: 禁止接收 FIFO0 水位线中断	RWA
23	Reserved	保留	-	-
22:16	F0S	接收 FIFO0 大小	0: 无接收 FIFO0 1~64: 接收 FIFO0 元素数量 >64: 大于 64 的设定值被解析为 64 接收 FIFO0 元素的索引为 0 到 F0S 减 1	RWA
15:2	F0SA[15:2]	接收 FIFO0 起始地址	消息 RAM 中接收 FIFO0 的起始地址 (32 位字地址, 请参见“ 消息 RAM 配置 ”)	RWA
1:0	Reserved	保留	-	-

37.4.27 MCAN 接收 FIFO0 状态寄存器 (MCAN_RXF0S)

Offset	Bit Position																															
0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved						RF0L	F0F	Reserved	F0PI						Reserved	F0GI						Reserved	F0FL								
Access							RO	RO		RO							RO							RO								

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	RF0L	接收 FIFO0 消息丢失	该位是中断标志 MCAN_IR.RF0L 的副本。当 MCAN_IR.RF0L 被复位时，该位也被复位： 0b0：接收 FIFO0 无消息丢失 0b1：接收 FIFO0 有消息丢失；对空间大小为 0 的接收 FIFO0 进行写操作时，该位也会置位  说明 当 MCAN_RXF0C.F0OM=1 时，覆盖最早的消息，不会置位该标志位。	RO
24	F0F	接收 FIFO0 满	0b0：接收 FIFO0 未满 0b1：接收 FIFO0 已满	RO
23:22	Reserved	保留	-	-
21:16	F0PI	接收 FIFO0 放入索引	接收 FIFO0 写入索引指针，有效范围是 0~63	RO
15:14	Reserved	保留	-	-
13:8	F0GI	接收 FIFO0 获取索引	接收 FIFO0 读取索引指针，有效范围是 0~63	RO
7	Reserved	保留	-	-
6:0	F0FL	接收 FIFO0 填充级别	接收 FIFO0 中存储的元素数量，有效范围是 0~64	RO

37.4.28 MCAN 接收 FIFO0 确认寄存器 (MCAN_RXF0A)

Offset	Bit Position																															
0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								F0AI							
Access																																

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5:0	F0AI	接收 FIFO0 确认索引	主机从接收 FIFO0 读取消息或消息序列后，必须将从接收 FIFO0 中读取的最后一个元素的缓冲区索引写入 F0AI。此操作会将接收 FIFO0 获取索引 MCAN_RXF0S.F0GI 设为 F0AI 加 1，并会更新 FIFO0 填充级别 MCAN_RXF0S.F0FL	RW

37.4.29 MCAN 接收缓冲区配置寄存器 (MCAN_RXBC)

Offset	Bit Position																																
0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																RBSA[15:2]																Reserved
Access																	RWA																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:2	RBSA[15:2]	接收缓冲区的起始地址	配置消息 RAM 中接收缓冲区的起始地址 (32 位地址)，也用于引用调试消息 A、B、C	RWA
1:0	Reserved	保留	-	-

37.4.30 MCAN 接收 FIFO1 配置寄存器 (MCAN_RXF1C)

Offset	Bit Position																															
0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	F1OM	F1WM							Reserved	F1S							F1SA[15:2]															Reserved
Access	RWA	RWA								RWA							RWA															

位/位域	标记	位名	功能描述	读写
31	F1OM	接收 FIFO1 工作模式	0b0: 接收 FIFO1 为阻塞模式 0b1: 接收 FIFO1 为覆盖模式	RWA
30:24	F1WM	接收 FIFO1 水位线	0: 禁止接收 FIFO1 水位线中断 1~64: 接收 FIFO1 水位线中断 (MCAN_IR.RF1W) 的级别 >64: 禁止接收 FIFO1 水位线中断	RWA
23	Reserved	保留	-	-
22:16	F1S	接收 FIFO1 大小	0: 无接收 FIFO1 1~64: 接收 FIFO1 元素数量 >64: 大于 64 的设定值被解析为 64 接收 FIFO1 元素的索引为 0 到 F1S 减 1	RWA
15:2	F1SA[15:2]	接收 FIFO1 起始地址	消息 RAM 中接收 FIFO1 的起始地址 (32 位字地址, 请参见“ 消息 RAM 配置 ”)	RWA
1:0	Reserved	保留	-	-

37.4.31 MCAN 接收 FIFO1 状态寄存器 (MCAN_RXF1S)

Offset	Bit Position																															
0xB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DMS	Reserved				RF1L	F1F	Reserved				F1PI				Reserved	F1GI				Reserved	F1FL										
Access	RO					RO	RO					RO					RO					RO										

位/位域	标记	位名	功能描述	读写
31:30	DMS	调试消息状态	0b00: 空闲状态, 等待接收调试消息 0b01: 已接收调试消息 A 0b10: 已接收调试消息 A、B 0b11: 已接收调试消息 A、B、C	RO
29:26	Reserved	保留	-	-
25	RF1L	接收 FIFO1 消息丢失	该位是中断标志 MCAN_IR.RF1L 的副本。当 MCAN_IR.RF1L 被复位时, 该位也被复位: 0b0: 接收 FIFO1 无消息丢失 0b1: 接收 FIFO1 有消息丢失; 对空间大小为 0 的接收 FIFO1 进行写操作时, 该位也会置位  说明 当 MCAN_RXF1C.F1OM=1 时, 覆盖最早的消息, 不会置位该标志位。	RO
24	F1F	接收 FIFO1 满	0b0: 接收 FIFO1 未滿 0b1: 接收 FIFO1 已滿	RO
23:22	Reserved	保留	-	-
21:16	F1PI	接收 FIFO1 放入索引	接收 FIFO1 写入索引指针, 有效范围是 0~63	RO
15:14	Reserved	保留	-	-
13:8	F1GI	接收 FIFO1 获取索引	接收 FIFO1 读取索引指针, 有效范围是 0~63	RO
7	Reserved	保留	-	-
6:0	F1FL	接收 FIFO1 填充级别	接收 FIFO1 中存储的元素数量, 有效范围是 0~64	RO

37.4.32 MCAN 接收 FIFO1 确认寄存器（MCAN_RXF1A）

Offset	Bit Position																															
0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								F1AI							
Access																																

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5:0	F1AI	接收 FIFO1 确认索引	主机从接收 FIFO1 读取消息或消息序列后，必须将从接收 FIFO1 中读取的最后一个元素的缓冲区索引写入 F1AI。此操作会将接收 FIFO1 获取索引 MCAN_RXF1S.F1GI 设为 F1AI 加 1，并会更新 FIFO1 填充级别 MCAN_RXF1S.F1FL	RW

37.4.33 MCAN 接收缓冲区和 FIFO 元素大小配置寄存器 (MCAN_RXESC)

配置接收缓冲区和接收 FIFO 元素的数据字节数。大于 8 字节的数据场 (Data field) 仅适用于 CAN FD 模式。

Offset	Bit Position																															
0xBC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																						RWA RBDS		Reserved	RWA F1DS		Reserved	RWA F0DS			
Access																							RWA			RWA			RWA			

位/位域	标记	位名	功能描述	读写
31:11	Reserved	保留	-	-
10:8	RBDS	接收缓冲区数据场大小	0b000: 8 字节 0b001: 12 字节 0b010: 16 字节 0b011: 20 字节 0b100: 24 字节 0b101: 32 字节 0b110: 48 字节 0b111: 64 字节	RWA
7	Reserved	保留	-	-
6:4	F1DS	接收 FIFO1 数据场大小	0b000: 8 字节 0b001: 12 字节 0b010: 16 字节 0b011: 20 字节 0b100: 24 字节 0b101: 32 字节 0b110: 48 字节 0b111: 64 字节	RWA
3	Reserved	保留	-	-
2:0	F0DS	接收 FIFO0 数据场大小	0b000: 8 字节 0b001: 12 字节 0b010: 16 字节 0b011: 20 字节 0b100: 24 字节 0b101: 32 字节 0b110: 48 字节 0b111: 64 字节	RWA

37.4.34 MCAN 发送缓冲区配置寄存器 (MCAN_TXBC)

仅当 MCAN_CCCR 寄存器的 CCE 位和 INIT 位均被置 1 时，才能对该寄存器进行写访问。

Offset	Bit Position																																
0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved	TFQM	TFQS					Reserved	NDTB							TBSA[15:2]																	Reserved
Access		RWA	RWA						RWA							RWA																	

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30	TFQM	发送 FIFO/队列模式	● 0b0：发送 FIFO 模式 ● 0b1：发送队列模式	RWA
29:24	TFQS	发送 FIFO/队列大小	● 0：无发送 FIFO 或队列 ● 1~32：发送 FIFO 队列的缓冲区大小 ● >32：大于 32 的设定值被解析为 32	RWA
23:22	Reserved	保留	-	-
21:16	NDTB	专用发送缓冲区大小	● 0：无专用发送缓冲区 ● 1~32：专用发送缓冲区大小 ● >32：大于 32 的设定值被解析为 32	RWA
15:2	TBSA[15:2]	发送缓冲区起始地址	消息 RAM 中发送缓冲区部分的起始地址（32 位地址，请参见“ 消息 RAM 配置 ”）	RWA
1:0	Reserved	保留	-	-

 **说明**

TFQS 和 NDTB 之和不得大于 32。MCAN 没有对该设置错误的监测。消息 RAM 中的发送缓冲区部分从专用发送缓冲区开始。

37.4.35 MCAN 发送 FIFO/队列状态寄存器（MCAN_TXFQS）

发送 FIFO 或队列的状态与寄存器 MCAN_TXBRP 中挂起的发送请求有关。因此，由于正在运行的发送扫描（TXBRP 尚未更新），添加或取消请求的效果可能会延迟。

Offset	Bit Position																																
0xC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved										TFQF	TFQPI				Reserved				TFGI				Reserved		TFFL							
Access											RO	RO								RO						RO							

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21	TFQF	发送 FIFO/队列满	0b0：发送 FIFO 或队列未满 0b1：发送 FIFO 或队列已满	RO
20:16	TFQPI	发送 FIFO/队列放入索引	发送 FIFO 或队列的放入索引指针，有效范围是 0~31	RO
15:13	Reserved	保留	-	-
12:8	TFGI	发送 FIFO 获取索引	发送 FIFO 的读取索引指针，有效范围是 0~31。若已配置为发送队列模式（MCAN_TXBC.TFQM=1），则该位读取值为 0	RO
7:6	Reserved	保留	-	-
5:0	TFFL	发送 FIFO 空闲级别	从 TFGI 开始的连续空闲的发送 FIFO 元素数量，有效范围是 0~32。若已配置为发送队列模式（MCAN_TXBC.TFQM=1），则该位读取值为 0	RO

 **说明**

在专用发送缓冲区与发送 FIFO 或发送队列组合配置的情况下，放入索引和获取索引是指，从第一个专用发送缓冲区开始的发送缓冲区编号。

例如：对于有 12 个专用发送缓冲区和 20 个发送 FIFO 缓冲区的组合配置，放入索引 15 指向发送 FIFO 的第 4 个缓冲区。

37.4.36 MCAN 发送缓冲区元素大小配置寄存器（MCAN_TXESC）

配置发送缓冲区元素的数据字节数。大于 8 字节的数据场（Data field）仅适用于 CAN FD 模式。

Offset	Bit Position																															
0xC8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															TBDS
Access																																

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	TBDS	发送缓冲区数据场大小	● 0b000: 8 字节 ● 0b001: 12 字节 ● 0b010: 16 字节 ● 0b011: 20 字节 ● 0b100: 24 字节 ● 0b101: 32 字节 ● 0b110: 48 字节 ● 0b111: 64 字节	RW

 **说明**

如果发送缓冲区元素的数据长度（由 DLC 配置），大于 TBDS 配置的数据长度，则发送缓冲区中未定义的数据字节将会以 0xCC（填充字节）发送。

37.4.37 MCAN 发送缓冲区请求挂起寄存器 (MCAN_TXBRP)

Offset	Bit Position																															
0xCC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16	TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
Access	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:0	TRPn (n=31~0)	发送请求挂起标志	对应发送缓冲区 31~0 的发送请求挂起位： ● 0b0：无发送请求挂起 ● 0b1：发送请求挂起 这些位通过寄存器 MCAN_TXBAR 置位；在请求的发送已完成，或请求的发送已通过寄存器 MCAN_TXBCR 取消后复位。 只有 MCAN_TXBC 配置了的发送缓冲区对应的 TXBRP 位才会置位。TXBRP 位置位后会启动发送扫描（参见“发送处理”），以检查优先级最高的挂起的发送请求（即消息 ID 最小的发送缓冲区）。 取消发送请求会复位相应的 TXBRP 位。如果请求取消时发送已经开始，则无论发送是否成功，在发送结束时 TXBRP 位都会被复位。TXBRP 位复位后，对应的取消请求位也会立即复位。 请求取消后，以下情况通过寄存器 MCAN_TXBCF 指示取消已完成： ● 发送成功及相应的 MCAN_TXBTO 位置位后 ● 取消时发送尚未开始 ● 发送因仲裁失败而中止 ● 帧发送期间发生错误 在 DAR 模式下，所有发送在失败后都会自动取消。所有失败的发送，对应的 MCAN_TXBCF 位都会置位。  说明 每个 Bit 对应一个缓冲区。 在发送扫描期间置位的 TXBRP 位，在本次发送扫描期间被忽略，在此期间，如果请求取消这些发送，则发送被立即取消，对应的 TXBRP 位复位。	RO

37.4.38 MCAN 发送缓冲区添加请求寄存器 (MCAN_TXBAR)

Offset	Bit Position																															
0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16	AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	ARn (n=31~0)	添加请求	对应发送缓冲区 31~0 的添加请求位： 写 1 置位相应的添加请求位，写 0 无效 读取： ● 0b0：未添加发送请求 ● 0b1：已添加发送请求 主机可一次设置多个发送缓冲区的发送请求。只有 MCAN_TXBC 配置了的发送缓冲区对应的 TXBAR 位才会置位。没有运行发送扫描时，这些位立即复位，否则将保持置位到发送扫描完成。  说明 每个 Bit 对应一个缓冲区。 如果添加请求的发送缓冲区已有发送请求挂起 (MCAN_TXBRP 寄存器相应位已置位)，该添加请求将被忽略。	RW

37.4.39 MCAN 发送缓冲区取消请求寄存器 (MCAN_TXBCR)

Offset	Bit Position																															
0xD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16	CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	CRn (n=31~0)	取消请求	对应发送缓冲区 31~0 的取消请求位： 写 1 置位相应的取消请求位，写 0 无效 读取： ● 0b0：无取消请求挂起 ● 0b1：有取消请求挂起 主机可一次设置多个发送缓冲区的取消请求。只有 MCAN_TXBC 配置了的发送缓冲区对应的 TXBCR 位才会置位。这些位保持置位直到对应的 MCAN_TXBRP 位复位。  说明 每个 Bit 对应一个缓冲区。	RW

37.4.40 MCAN 发送缓冲区发送已发生寄存器 (MCAN_TXBTO)

Offset	Bit Position																																
0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16	TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0	
Access	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:0	TOn (n=31~0)	发送已发生	对应发送缓冲区 31~0 的发送已发生标志位： ● 0b0：未进行发送 ● 0b1：已进行发送 当发送缓冲区发送完成，MCAN_TXBRP 相应位清零时，对应的发送已发生标志位置位。向 MCAN_TXBAR 相应位写 1 请求新的发送时，对应的发送已发生标志位复位。  说明 每个 Bit 对应一个缓冲区。	RO

37.4.41 MCAN 发送缓冲区取消已完成寄存器 (MCAN_TXBCF)

Offset	Bit Position																																
0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16	CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0	
Access	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:0	CFn (n=31~0)	取消已完成	对应发送缓冲区 31~0 的取消已完成标志位： ● 0b0：无发送缓冲区取消 ● 0b1：发送缓冲区已取消 当通过 MCAN_TXBCR 请求的取消已完成，MCAN_TXBRP 相应位清零时，对应的取消已完成标志位置位。如果 MCAN_TXBRP 相应位在取消时没有置位，取消已完成标志位会立即置位。向 MCAN_TXBAR 相应位写 1 请求新的发送时，对应的取消已完成标志位复位。  说明 每个 Bit 对应一个缓冲区。	RO

37.4.42 MCAN 发送缓冲区发送中断使能寄存器 (MCAN_TXBTIE)

Offset	Bit Position																															
0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16	TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	TIE _n (n=31~0)	发送中断使能	对应发送缓冲区 31~0 的发送中断使能位： 0b0：禁止发送中断 0b1：使能发送中断  说明 每个 Bit 对应一个缓冲区。	RW

37.4.43 MCAN 发送缓冲区取消已完成中断使能寄存器 (MCAN_TXBCIE)

Offset	Bit Position																															
0xE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16	CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	CFIE _n (n=31~0)	取消已完成中断使能	对应发送缓冲区 31~0 的取消已完成中断使能位： 0b0：禁止取消已完成中断 0b1：使能取消已完成中断  说明 每个 Bit 对应一个缓冲区。	RW

37.4.44 MCAN 发送事件 FIFO 配置寄存器 (MCAN_TXEFC)

Offset	Bit Position																															
0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	EFWM						Reserved	EFS						EFSA[15:2]										Reserved							
Access		RWA							RWA						RWA																	

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29:24	EFWM	发送事件 FIFO 水位线	● 0: 禁止水位线中断 ● 1~32: 发送事件 FIFO 水位线中断 (MCAN_IR.TEFW) 的级别 ● >32: 禁止水位线中断	RWA
23:22	Reserved	保留	-	-
21:16	EFS	发送事件 FIFO 大小	● 0: 禁止发送事件 FIFO ● 1~32: 发送事件 FIFO 的元素数量 ● >32: 大于 32 的设定值被解析为 32 发送事件 FIFO 元素的索引为 0 到 EFS 减 1	RWA
15:2	EFSA[15:2]	发送事件 FIFO 起始地址	消息 RAM 中发送事件 FIFO 的起始地址 (32 位地址, 请参见“ 消息 RAM 配置 ”)	RWA
1:0	Reserved	保留	-	-

37.4.45 MCAN 发送事件 FIFO 状态寄存器 (MCAN_TXEFS)

Offset	Bit Position																																																			
0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
Reset	0x00000000																																																			
Name	Reserved						TEFL	EFF	Reserved						EFPI						Reserved						EFGI						Reserved						EFFL													
Access							RO	TEFL	RO	EFF							RO												RO												RO						EFFL					

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	TEFL	发送事件 FIFO 元素丢失	该位是中断标志 MCAN_IR.TEFL 的副本。当 MCAN_IR.TEFL 被复位时，该位也被复位： 0b0：发送事件 FIFO 无元素丢失 0b1：发送事件 FIFO 有元素丢失；对空间大小为 0 的发送事件 FIFO 进行写操作时，该位也会置位	RO
24	EFF	发送事件 FIFO 已满	0：发送事件 FIFO 未滿 1：发送事件 FIFO 已滿	RO
23:21	Reserved	保留	-	-
20:16	EFPI	发送事件 FIFO 放入索引	发送事件 FIFO 写入索引指针，有效范围是 0~31	RO
15:13	Reserved	保留	-	-
12:8	EFGI	发送事件 FIFO 获取索引	发送事件 FIFO 读取索引指针，有效范围是 0~31	RO
7:6	Reserved	保留	-	-
5:0	EFFL	发送事件 FIFO 填充级别	发送事件 FIFO 中存储的元素个数，有效范围是 0~32	RO

37.4.46 MCAN 发送事件 FIFO 确认寄存器（MCAN_TXEFA）

Offset	Bit Position																															
0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										EFAI					
Access																																

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4:0	EFAI	发送事件 FIFO 确认索引	主机从发送事件 FIFO 读取元素或元素序列后，必须将从事件 FIFO 中读取的最后一个元素的索引写入 EFAI。此操作会将发送事件 FIFO 获取索引 MCAN_TXEFS.EFGI 设为 EFAI 加 1，并会更新事件 FIFO 填充级别 MCAN_TXEFS.EFFL	RW

37.4.47 MCAN 消息 RAM 校验控制寄存器 (MCANRAM_CKCR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															ECCMOD
Access																																RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1:0	ECCMOD	消息 RAM 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断	RW

 **说明**

1. 使用消息 RAM 的 ECC 功能时, 须在读写访问消息 RAM 之前配置 ECCMOD, 然后对所用消息 RAM 空间以字为单位进行初始化。
2. 当 ECCMOD 为 0b10 或 0b11 时, 若 MCAN1/MCAN2 访问消息 RAM 产生 1-bit 错误, 则 MCAN1/MCAN2 的 MCAN_IR.BEC 置位, 若相应中断打开, MCAN1/MCAN2 会产生中断。
3. 当 ECCMOD 为 0b01、0b10 或 0b11 时, 若 MCAN1/MCAN2 访问消息 RAM 产生 2-bit 错误, 则 MCAN1/MCAN2 的 MCAN_IR.BEU 置位, 若相应中断打开, MCAN1/MCAN2 会产生中断。

37.4.48 MCAN 消息 RAM 校验状态寄存器 (MCANRAM_CKSR)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														RAM_2ERR	RAM_1ERR
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RAM_2ERR	消息 RAM 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: 0b0: 无 2-bit ECC 错误发生 0b1: 有 2-bit ECC 错误发生	RW
0	RAM_1ERR	消息 RAM 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: 0b0: 无 1-bit ECC 错误发生 0b1: 有 1-bit ECC 错误发生	RW

37.4.49 MCAN 消息 RAM 错误注入使能寄存器 (MCANRAM_EIEN)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															EIEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	0b0: 禁止 0b1: 使能	RW

37.4.50 MCAN 消息 RAM 错误注入位使能寄存器 0 (MCANRAM_EIBIT0)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制消息 RAM 输出数据的 31~0 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

37.4.51 MCAN 消息 RAM 错误注入位使能寄存器 1 (MCANRAM_EIBIT1)

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制消息 RAM 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

37.4.52 MCAN 消息 RAM 校验错误地址寄存器（MCANRAM_ECCERRADDR）

Offset	Bit Position																															
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				RO ECCERRADDR											
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	ECCERRADDR	校验错误地址	记录消息 RAM 发生 1-bit 或 2-bit ECC 错误时的访问地址（相对于消息 RAM 基地址的偏移地址）	RO

37.5 注意事项

37.5.1 CAN 总线抗干扰措施

CAN 总线广泛应用于汽车、工业控制等行业，如果 CAN 应用现场电磁环境比较恶劣，存在电路不平衡性、空间电磁场、电网进线等因素，会导致 CAN 总线因辐射、传导干扰而产生大量通信噪声，致使总线错误帧增加、重发频繁，正确数据不能及时到达等情况，或者导致 CAN 协议未定义的错误，总线异常等不可预期的情况，严重影响数据通信质量。所以实际应用中应该致力于消除噪声干扰，保障 CAN 总线网络稳定工作。

以下是几类常用的 CAN 总线抗干扰措施（包括但不限于）：

- 1. 增加 CAN 总线接口电气隔离
- 2. 共收发器的信号地
- 3. 使用屏蔽双绞线缆并正确接地
- 4. 提高 CAN 传输线双绞程度
- 5. 增加信号保护器
- 6. 改进网络拓扑
- 7. 应用层软件抗干扰机制

37.5.2 CAN 控制器噪声制约

在 CAN 总线网络中应确保通信的位时间满足标准协议的要求，若引入不满足位时间宽度的噪声干扰，可能引起 MCAN 控制器异常动作。

38 以太网 MAC 控制器 (ETHMAC)

38.1 简介

以太网 MAC 控制器 (ETHMAC) 用于在以太网网络中按照 IEEE802.3-2002 标准发送和接收数据，有多种应用领域，如交换机、网络接口卡等。该 MAC 控制器支持与外部物理层 (PHY) 相连的两个工业标准接口：介质独立接口 (MII) (在 IEEE802.3 规范中定义) 和简化介质独立接口 (RMII)。

主要遵循以下协议规范：

- IEEE802.3-2002，用于以太网 MAC
- IEEE1588-2008 标准，用于规定联网时钟同步
- AMBA2.0，用于 AHB 主/从端口
- RMII 接口规范

38.2 主要特性

38.2.1 ETH_MAC 特性

ETH_MAC 的基本功能特性如下：

- 支持外部 PHY 接口实现 10/100Mbps 数据传输速率
- 支持 MII 或 RMII 接口与外部快速以太网 PHY 进行通信
- 为应用程序提供单独的发送、接收和控制接口
- 使用 SMI 接口配置和管理最多 32 个 PHY 设备
- 支持检测远程唤醒帧和 AMD Magic Packet™ 帧
- 支持通过 MII 进行内部回送的 LoopBack 模式
- 支持全双工和半双工操作：
 - ▶ 支持适用于半双工操作的 CSMA/CD 协议
 - ▶ 支持适用于半双工操作的背压流量控制
 - ▶ 支持适用于全双工操作的 IEEE802.3x 流量控制
 - ▶ 全双工操作时可以将接收的暂停控制帧转发到应用



说明

本章节中所述“应用”，是指在 MAC 将接收到的帧通过 RxDMA 传送到系统存储器，供上层的网络层、传输层使用；或者将上层网络层、传输层的数据从系统控制器通过 TxDMA 传送到 MAC，上层的网络层、传输层称之为应用。

- ▶ 全双工操作中如果流量控制输入信号消失，将自动发送零时间片暂停帧
- 以太网帧处理：
 - ▶ 接收以太网帧时，自动去除长度字段小于 1536 的帧的 PAD 和 FCS 字段
 - ▶ 接收以太网帧时，自动去除类型帧的 FCS 字段
 - ▶ 接收以太网帧时，自动进行接收帧的 CRC 计算
 - ▶ 发送以太网帧时，对 SA 进行插入、替换
 - ▶ 发送以太网帧时，对 CRC 进行插入、替换
 - ▶ 发送以太网帧时，对 VLAN 帧标识进行插入、替换、删除
 - ▶ 发送以太网帧时，对小于 60 字节的帧自动生成 PAD 填充
 - ▶ 发送以太网帧时，处理冲突帧的自动重新发送

- ▶ 发送以太网帧时，支持可编程帧间隔（40-96 位时间，以 8 为步长）
- ▶ 接收或发送以太网帧时，对帧长度可进行编程，支持高达 16KB 的巨型帧
- 以太网帧地址过滤：
 - ▶ 支持高达五个 48 位完美（DA）地址过滤器，对每个字节进行掩码操作
 - ▶ 支持高达四个 48 位完美（SA）地址过滤器，对每个字节进行掩码操作
 - ▶ 支持 64 位 Hash 滤波器，适用于单播和多播目标地址过滤
 - ▶ 可传送所有多播地址帧
 - ▶ 支持混合模式，可传送所有帧，无需为网络监视进行过滤
 - ▶ 传送所有传入数据包时（每次过滤时）均附有一份状态报告
- VLAN 标识符过滤：
 - ▶ 支持一组 VLAN 标记过滤器
 - ▶ 支持 12 位/16 位 VLAN 标记选择
 - ▶ 支持 VLAN 标记 Hash 滤波方式
- LAY3 IP 报地址过滤：
 - ▶ 支持一组 LAY3 IP 报地址过滤器，对每个字节进行掩码操作
 - ▶ 支持支持 IPv4 报、IPv6 报的地址过滤选择
- LAY4 TCP/UDP 报端口过滤：
 - ▶ 支持一组 LAY4 TCP/UDP 报端口过滤器
- 远程唤醒帧过滤模式：
 - ▶ 支持四组远程唤醒帧过滤器，控制四组掩码操作
- COE 引擎：
 - ▶ 在接收帧时，可选择丢弃无 TCP/UDP 字段的帧
 - ▶ 对接收到的 IPv4 报的 Header Checksum 字段进行校验
 - ▶ 对接收到的 TCP 报、UDP 报、ICMP 报的 Checksum 字段进行校验
 - ▶ 对发送的 IPv4 报进行 Checksum 计算，将计算结果插入到 Header Checksum 字段
 - ▶ 对发送的 TCP 报、UDP 报、ICMP 报进行 Checksum 计算，将计算结果插入到 Checksum 字段
- 支持 MMC 计数器进行网络统计
- FIFO 处理：
 - ▶ 具有一个可编程阈值的 2KB 发送 FIFO（Tx FIFO）和一个可配置阈值的 2KB 接收 FIFO（Rx FIFO）
 - ▶ Rx FIFO 和 Tx FIFO 均支持存储转发模式
 - ▶ 软件控制刷新 Tx FIFO
 - ▶ Rx FIFO 在存储转发模式下，可选择过滤所有的错误帧，不将这些错误帧转发给应用
 - ▶ Rx FIFO 可根据填充（阈值可配置）级别自动生成要发送给 MAC 内核的暂停控制帧或背压信号
 - ▶ Rx FIFO 中丢失或损坏的帧可进行数据统计
 - ▶ Rx FIFO 进行多帧存储时，通过在 EOF 传输后向 Rx FIFO 插入接收状态矢量，从而使得 Rx FIFO 无需存储这些帧的接收状态

38.2.2 ETH_PTP 特性

ETH_PTP 的基本功能特性如下：

- 支持接收帧和发送帧进行时间戳快照，插入时间戳
- 支持对系统时间进行粗校准和细校准
- 支持 2 组目标时间设定，系统时间大于目标时间时会触发中断

- 支持 2 组 PPS 输出，可以作为事件输出来控制其它模块的动作

38.2.3 ETH_DMA 特性

ETH_DMA 的基本功能特性如下：

- 支持软件设定 AHB Master 接口的 AHB 突发类型（固定或不确定突发）
- AHB Slave 接口中支持所有 AHB 突发类型
- AHB Master 接口可选择地址对齐突发传输
- 支持 RxDMA 和 TxDMA 引擎单独编程突发长度，以充分利用总线
- 支持对数据缓冲区进行字节对齐寻址
- 描述符特性：
 - ▶ 支持环式和链式两种描述符链接方式
 - ▶ 采用描述符架构可以在 CPU 几乎不干预的情况传输大型数据块
 - ▶ 每个描述符可传输高达 8KB 的数据
- TxDMA 和 RxDMA 均支持存储转发模式
- 报告正常工作和传输错误时的综合状态
- 可编程中断选项，根据不同的应用场景选择不同的中断
- 按帧控制发送/接收完成中断
- 接收引擎和发送引擎间采用循环调度仲裁或固定优先级仲裁

38.3 功能说明

以太网 MAC 控制器遵循局域网 (LAN) 的 IEEE802.3 国际标准, 将 CSMA/CD (带有冲突检测的载波侦听多路访问) 方式作为基本通信方式, 它包括一个带 MII 和 RMII 接口的 MAC802.3 (介质访问控制) 控制器和一个专用 DMA 控制器。

38.3.1 基本框图

下图为以太网 MAC 控制器的架构框图。

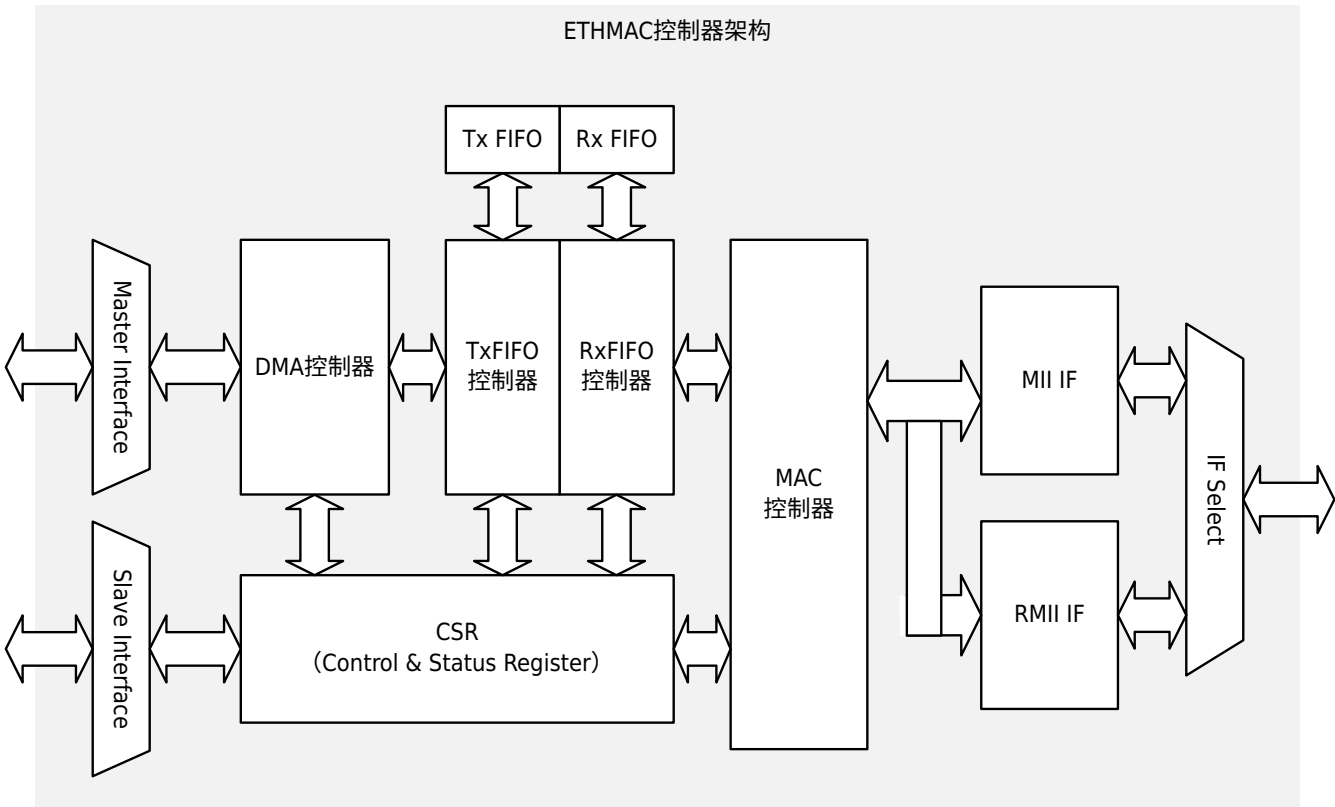


图 38-1 以太网 MAC 控制器架构图

从图中可以看出, 以太网 MAC 控制器的内核部分包括 AHB Master 接口、AHB Slave 接口、Rx FIFO、Tx FIFO 和 DMA 控制器、MAC 控制器、接口控制逻辑等。DMA 控制器通过 AHB 主从接口将 MAC 内核和系统存储器相连。

AHB Master 接口用于数据传输, AHB Slave 接口用于该控制器的基本寄存器访问配置等。

在发送数据时, 首先将数据从系统存储器由 DMA 控制器送至 Tx FIFO 进行缓冲, 经 MAC 控制器的一系列处理 (COE 控制、PTP 控制、MMC 控制等) 后通过 MII 接口或 RMII 接口送至外部 PHY。

在接收数据时, 经过 MII 接口或 RMII 接口接收进来的数据, 经 MAC 控制器的一系列处理 (COE 控制、PTP 控制、MMC 控制等) 后送至 Rx FIFO, DMA 控制器再将 Rx FIFO 的数据传送到系统存储器。

38.3.2 接口说明

以太网 MAC 控制器支持 MII、RMII、SMI 接口类型, 其中用户根据实际需求选择 MII 接口和 RMII 接口中的一种作为数据交换的接口, SMI 接口独立存在, 用于和 PHY 间进行基本配置相关的数据交换。

通过对接口控制寄存器 (ETH_MAC_IFCONFR) 的 IFSEL 位的设定, 可选择以太网控制其使用 MII 接口还是 RMII 接口, 在对 ETHMAC 的其它特性配置之前, 该位必须优先设定。

38.3.2.1 MII 接口

介质独立接口 (MII) 定义了 10Mbps 和 100Mbps 的数据传输速率下 MAC 子层与 PHY 之间的互连。下图是 MII 接口的基本连接图。

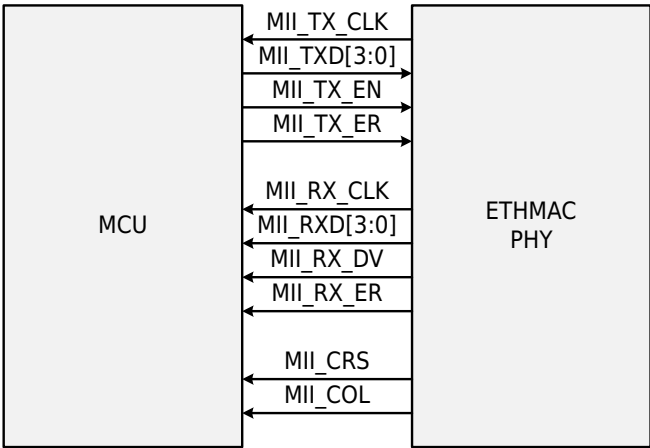


图 38-2 MII 接口连接图

下表介绍了 MII 接口各信号作用。

表 38-1 MII 接口信号说明

接口信号名	功能描述
MII_TX_CLK	发送时钟信号。 该信号提供进行 TX 数据传输时的参考时序。速率为 10Mbps 时为 2.5MHz；速率为 100Mbps 时为 25MHz。
MII_RX_CLK	接收时钟信号。 该信号提供进行 RX 数据传输时的参考时序。速率为 10Mbps 时为 2.5MHz；速率为 100Mbps 时为 25MHz。
MII_TX_EN	发送使能信号。 该信号表示 MAC 当前正对 MII 发送数据。
MII_TX_ER	发送错误信号。
MII_TXD[3:0]	发送数据信号。 该信号是 4 个一组的数据信号，由 MAC 子层同步驱动，在 MII_TX_EN 信号有效时才为有效信号。MII_TXD[0]为最低有效位，MII_TXD[3]为最高有效位。MII_TX_EN 信号无效时，发送数据不会对 PHY 产生任何影响。
MII_RX_DV	接收数据有效信号。 该信号表示 PHY 当前正对 MII 发送数据。为正确地接收帧，该信号必须在时间范围上涵盖要接收的帧，其开始时间不得迟于 SFD 字段出现的时间。
MII_RX_ER	接收错误信号。 该信号必须保持一个或多个发送时钟周期，从而向 MAC 子层指示在帧的某处检测到错误。该错误条件必须通过 RX_DV 信号验证（请参见表 38-3）。
MII_RXD[3:0]	接收数据信号。 该信号是 4 个一组的数据信号，由外部 PHY 同步驱动，在 MII_RX_DV 信号有效时才为有效信号。MII_RXD[0]为最低有效位，MII_RXD[3]为最高有效位。MII_RX_DV 信号无效、MII_RX_ER 信号使能时，特定的 MII_RXD[3:0]值用于传输来自 PHY 的特定信息（请参见表 38-3）。

接口信号名	功能描述
MII_CRS	载波侦听信号。 当发送或接收介质处于非空闲状态时，由 PHY 使能该信号；发送和接收介质均处于空闲状态时，由 PHY 禁止该信号。PHY 必须确保 MII_CRS 信号在冲突条件下保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。
MII_COL	冲突检测信号。 检测到介质上存在冲突后，PHY 必须立即使能冲突检测信号，并且只要存在冲突条件，冲突检测信号必须保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。

下表介绍发送数据时的相关信号编码状态。

表 38-2 发送数据时信号状态说明

MII_TX_EN	MII_TX_ER	MII_TXD[3:0]	状态描述
0b0	0b0	0b0000~0b1111	正常帧间，无数据发送
0b0	0b1	0b0000~0b1111	无意义
0b1	0b0	0b0000~0b1111	正常发送数据
0b1	0b1	0b0000~0b1111	发送数据出现错误

下表介绍接收数据时的相关信号编码状态。

表 38-3 接收数据时信号状态说明

MII_RX_DV	MII_RX_ER	MII_RXD[3:0]	状态描述
0b0	0b0	0b0000~0b1111	正常帧间，无数据接收
0b0	0b1	0b0000	正常帧间
		0b0001~0b1101	无意义
		0b1110	错误载波检测
		0b1111	无意义
0b1	0b0	0b0000~0b1111	正常接收数据
0b1	0b1	0b0000~0b1111	接收数据出现错误

38.3.2.2 RMII 接口

精简介质独立接口 (RMII) 规范降低了 10/100Mbps 下微控制器以太网 MAC 控制器与外部 PHY 间的端口数。它所使用的端口数由 MII 的 16 个减少为 7 个，具体如下图所示。

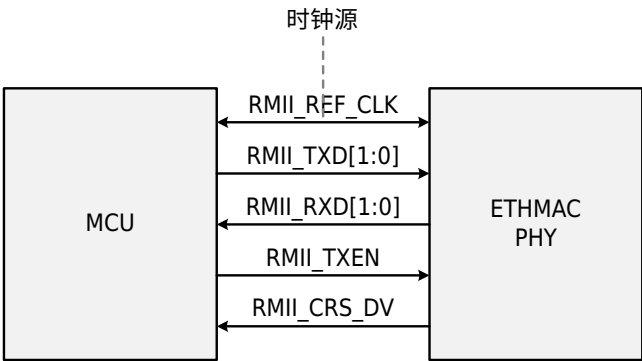


图 38-3 RMII 接口连接图

从连接图可以看出，RMII 接口和 MII 接口主要有以下几个不同点：

- 1. 发送时钟和接收时钟统一为参考时钟 RMII_REF_CLK，该时钟必须为 50MHz 时钟
- 2. 数据线由 4 根变为 2 根
- 3. MII_COL 信号丢弃，MII_RX_DV 和 MII_CRSDV 信号合并为 RMII_CRSDV 信号
- 4. MII_TX_ER、MII_RX_ER 信号丢弃

38.3.2.3 SMI 接口

SMI 接口可通过以太网 MAC 控制器对 PHY 实现站点管理。MAC 控制器通过时钟和数据 2 根线访问任意的 PHY 寄存器。该接口最多支持访问 32 个 PHY。下图是 SMI 接口的连接图。

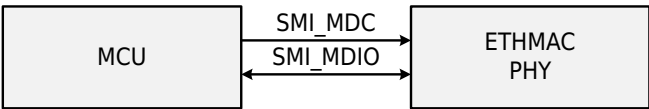


图 38-4 SMI 接口连接图

下表介绍了 SMI 接口各信号作用。

表 38-4 SMI 接口信号说明

接口信号名	功能描述
SMI_MDC	周期性时钟，提供以最大频率 2.5MHz 传输数据时的参考时序。MDC 的最短高电平和最短低电平必须均为 160ns。MDC 的最小周期必须为 400ns。在空闲状态下，SMI 将 MDC 时钟信号驱动为低电平。
SMI_MDIO	数据输入/输出比特流，用于通过 MDC 时钟信号与 PHY 设备进行数据信息传输。

MAC 控制器可以从 32 个 PHY 中选择一个 PHY，然后从选中的 PHY 所包含的 32 个寄存器中选择一个寄存器，发送控制数据或接收状态信息。任意给定时间内只能对一个 PHY 中的一个寄存器进行寻址。

SMI 的帧格式如下表所示，在与 PHY 进行数据交换的过程中遵循以下格式：

表 38-5 SMI 帧格式

	SMI 帧格式描述							
	Head (32-bit)	Start	Operation	PhyAdd	RegAdd	TA	Data (16-bit)	Idle
Read	0b1...1	0b01	0b10	0bxxxx	0bxxxx	Z0	0bxx...xx	Z
Write	0b1...1	0b01	0b01	0bxxxx	0bxxxx	0b10	0bxx...xx	Z

38.3.2.4 以太网端口配置

依据上述 MII、RMII、SMI 3 种类型的数据接口，将以太网 MAC 控制器的端口功能分配与各类型接口的对应关系整理如下表所示：

表 38-6 ETHMAC 端口功能分配

ETH 端口名	接口功能信号
ETH_MII_RMII_RXCLK	MII_RX_CLK/RMII_REF_CLK
ETH_MII_TXCLK	MII_TX_CLK
ETH_MII_RMII_TXEN	MII_TX_EN/RMII_TX_EN
ETH_MII_TXER	MII_TX_ER
ETH_MII_RMII_TXD1~0	MII_TXD[0]/RMII_TXD[0]
	MII_TXD[1]/RMII_TXD[1]
ETH_MII_TXD3~2	MII_TXD[2]
	MII_TXD[3]
ETH_MII_RMII_RXDV	MII_RX_DV/RMII_CRS_DV
ETH_MII_RXER	MII_RX_ER
ETH_MII_RMII_RXD1~0	MII_RXD[0]/RMII_RXD[0]
	MII_RXD[1]/RMII_RXD[1]
ETH_MII_RXD3~2	MII_RXD[2]
	MII_RXD[3]
ETH_MII_CRS	MII_CRS
ETH_MII_COL	MII_COL
ETH_SMI_MDC	SMI_MDC
ETH_SMI_MDIO	SMI_MDIO

38.3.3 ETH_MAC 功能

38.3.3.1 ETH_MAC 功能概述

MAC 子层执行以下与数据链路控制步骤相关的功能：

- 数据封装（发送和接收）
 - ▶ 组帧（帧边界定界、帧同步）
 - ▶ 寻址（处理源地址和目标地址）
 - ▶ 错误检测
- 介质访问管理
 - ▶ 介质分配（冲突避免）
 - ▶ 竞争解决（冲突处理）

MAC 子层主要有两个工作模式：

- 半双工模式：站点使用 CSMA/CD 算法争用物理介质
- 全双工模式：满足以下条件时，无需解决竞争问题（CSMA/CD 算法不是必需的）便可同时发送和接收数据：
 - ▶ 物理介质能够支持同步发送和接收
 - ▶ 正好有 2 个站点与 LAN 相连
 - ▶ 两个站均配置为全双工工作模式

38.3.3.2 MAC 802.3 帧格式

MAC 的数据通信可使用两种帧格式：

- 基本 MAC 帧格式
- 带 VLAN 标识的 MAC 帧格式（对基本 MAC 帧格式的扩展）

下图是两种帧格式的具体结构。

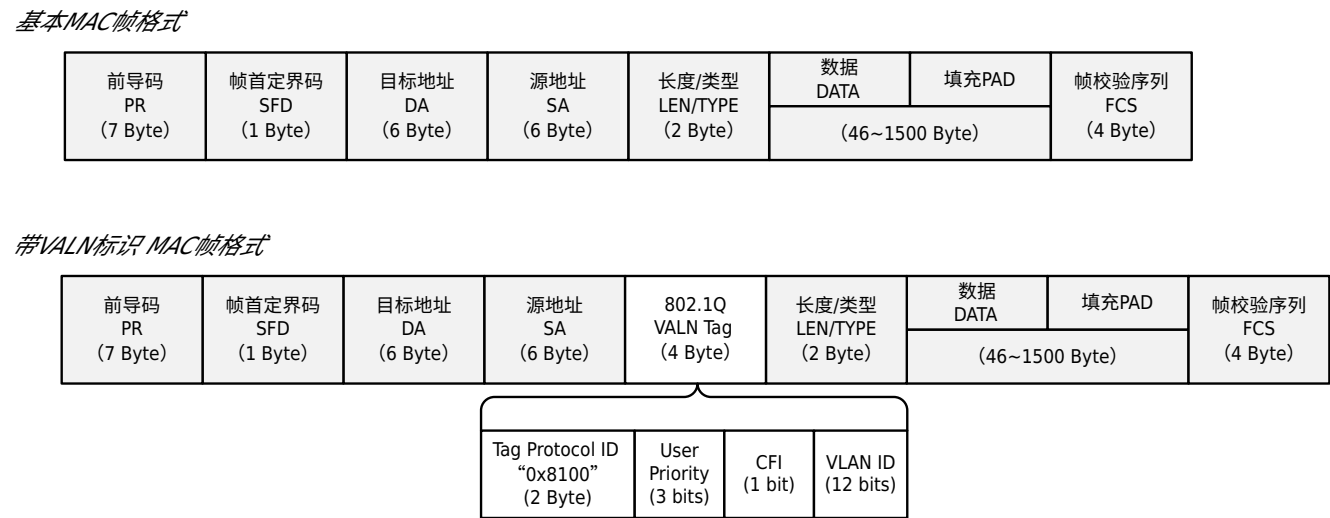


图 38-5 MAC 帧结构

除了帧校验序列 FCS，以太网 MAC 控制器发送每个字节时都按照低位先出的次序进行传输。
CRC 计算包括帧数据的所有字节除去前导码和帧首界定码域。以太网帧 32CRC 生成多项式为 0x04C11DB7，且此多项式用于以太网模块中所有的 32 位 CRC 计算，如下式所示：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

通过以下条件之一定义一个帧为无效 MAC 帧：

- 帧长度与长度/类型字段指定的预期值不一致。如果长度/类型字段包含类型值，则认为帧长度与此字段一致（没有无效帧）
- 帧长度不是字节的整数倍（额外位）
- 根据传入帧计算出的 CRC 值与包含的 FCS 不匹配

38.3.3.3 MAC 帧发送动作

TxDMA 控制发送路径的所有事务。从系统存储器读取的以太网帧由 TxDMA 送入 Tx FIFO，然后传输至 MAC 内核。帧传输结束时，从 MAC 内核获取发送状态并传回 TxDMA。
以太网 MAC 控制器搭载 2KB 的 Tx FIFO。Tx FIFO 填充级别将指示给 DMA，以便 TxDMA 可通过 AHB 接口在所需的系统存储器启动数据传送。来自 AHB Master 接口的数据将送入 Tx FIFO。
检测到 SOF 时，MAC 接收数据并开始向 MII 发送。在应用程序启动发送后，向 MII 发送帧数据所需的时间是可变的，具体取决于 IFG 延迟、发送报头/SFD 的时间以及半双工模式的任意回退延迟等延迟因素。EOF 传输到 MAC 内核后，内核将完成正常的发送，然后将发送的状态返回给 TxDMA。如果在发送过程中发生常规冲

突（在半双工模式下），MAC 内核将使发送状态有效，然后接受并丢弃所有后续数据，直至收到下一个 SOF。检测到来自 MAC 的重试请求（在状态中）时，应从 SOF 重新发送同一帧。

如果发送期间未连续提供数据，MAC 将发出下溢状态。在帧的正常传输期间，如果 MAC 在未获得前一帧的 EOF 的情况下接收到 SOF，则将忽略该 SOF 并将新的帧视为前一帧的延续。

向 MAC 内核输送数据有两种操作模式：

- 在阈值模式下，只要 FIFO 中的字节数超过配置的阈值（或在超过阈值前写入帧结束），数据就准备好并转发到 MAC 内核。该阈值可使用 ETH_DMA_OPRMODR 的 TTC 位配置。
- 在存储转发模式下，仅当在 FIFO 中存储完整的帧后，才会向 MAC 内核输送帧。如果 Tx FIFO 的大小小于要发送的以太网帧，则在 Tx FIFO 接近填满时向 MAC 内核输送帧。

应用可通过将 FTF 位（ETH_DMA_OPRMODR 寄存器[20]）置 1 来清空 Tx FIFO 的所有内容。此位自行清零，并将 FIFO 指针初始化为默认状态。如果向 MAC 内核传输帧时将该位置 1，则传输将停止，此时 Tx FIFO 被视为空，MAC 发送器将出现下溢事件并且相应的状态字将转发给 TxDMA。

发送基本协议

MAC 控制以太网帧的发送操作。它执行下列功能以满足 IEEE802.3/802.3z 规范。包括：

- 生成报头和 SFD
- 在半双工模式下生成 Jam 阻塞信号
- 控制 Jabber 超时
- 控制半双工模式下的流量（背压）
- 生成发送帧状态
- 包含符合 IEEE1588 的时间戳快照逻辑

当请求发送新的帧时，MAC 将发送报头和 SFD，紧接着发送数据。报头定义为“10101010”样式的 7 个字节，SFD 定义为“10101011”样式的 1 个字节。

冲突窗口定义为 1 个时隙（对于 10/100Mbps 以太网，为 512 个位时间）。阻塞信号生成仅适用于半双工模式，不适用于全双工模式。在 MII 模式下，如果在开始传输帧到 CRC 字段结束之间的任何时间发生冲突，MAC 将在 MII 上发送 0x55555555 的 32 位 Jam 阻塞信号，通知所有其它站已发生冲突。如果在报头发送阶段发生冲突，MAC 将完成报头和 SFD 的发送，然后发送 Jam 阻塞信号。

MAC 内核使用一个 Jabber 定时器，用于在传输的字节超过 2048 字节时切断以太网帧的发送。在半双工模式下，MAC 使用延迟机制进行流量控制（背压）。当应用请求停止接收帧时，如果已使能发送流量控制，则只要 MAC 检测到接收帧，就会发送一个 32 字节的 Jam 信号。这会导致冲突并使远程站回退。应用程序通过将 ETH_MAC_FLOCTLR 寄存器中的 BPA 位置 1 来请求流量控制。如果应用请求发送帧，则即使激活背压功能，也将按调度计划发送。如果背压功能长时间保持激活（发生的连续冲突事件超过 16 个），则远程站将由于冲突过多而中止发送。如果针对发送帧使能 IEEE1588 时间戳功能，则将 SFD 置于发送 MII 总线时会获取系统时间的时间快照。

调度算法程序

MAC 负责调度 MII 上的帧发送。通过 IFG 设定或截断二进制指数回退算法来保持两个发送帧之间的帧间隔。在满足 IFG 和回退条件后，MAC 使能发送。

IFG 设定可确保两个发送帧之间的空闲期段，即配置的帧间隔（ETH_MAC_CONFIGR 寄存器中的 IFG 位）。如果要发送的帧在配置的 IFG 时间之前到达，则 MII 会等待来自 MAC 的使能信号，然后再开始发送。只要 MII 的载波信号进入无效状态，MAC 就会启动 IFG 计数器。在 IFG 计数值与设定值相等时，MAC 将以全双工模式使能发送。

在半双工模式下，当 IFG 配置为 96 个位时间时，MAC 将遵循 IEEE802.3 规范指定的顺从规则。如果在 IFG 间隔的前三分之二时间内（对于所有 IFG 值都为 64 位时间）检测到载波，MAC 将复位其 IFG 计数器。如果在 IFG 间隔的后三分之一时间内检测到载波，MAC 将继续执行 IFG 计数并在 IFG 间隔结束后使能发送器。

在半双工模式下，实施截断二进制指数回退算法，具体由 MAC 配置寄存器 (ETH_MAC_CONFIGR) 的 BL 位设定控制。

冲突期间重发

在半双工模式下，向 MAC 传输帧时，可能在 MAC 线接口上发生冲突事件。MAC 甚至会在接收到帧结束状态之前就指示重试，然后将使能重新发送并再次将帧从 Tx FIFO 中送出。

当超过 96 个字节送到 MAC 内核后，Tx FIFO 控制器将释放该空间，使 DMA 可送入更多数据。这意味着超过阈值后或 MAC 内核指示延迟冲突事件时，无法重新发送。

TxFIFO 刷新

MAC 控制器可通过操作模式寄存器 (ETH_DMA_OPRMODR) 中的 FTF 位来清空 Tx FIFO。清空操作是立即操作，即使 Tx FIFO 正在向 MAC 内核传输帧，Tx FIFO 和相应的指针也会清零到初始状态。这将导致 MAC 发送器中生成下溢事件，并且帧发送将中止，此时帧的状态将同时标记下溢和清空事件 (TDES0 的 FFF 位和 UDE 位)。清空操作期间，没有数据从 TxDMA 传输到 Tx FIFO，TxDMA 根据清空的帧数 (包括局部帧)，将相应数量的传输发送状态字传输到应用。完全清空的帧的帧清空状态位 (TDES0 的 FFF 位) 将置 1。当 TxDMA 接受所有已清空的帧的状态字后，清空操作完成，FTF 位随后将清零。此时，将接受来自 TxDMA 的新帧。清空操作完成后，将丢弃所有提交的数据，除非数据以 SOF 标记开头。

CRC 处理和 PAD 生成

当从应用接收的字节数少于 60 (DA+SA+LT+数据) 时，可通过发送描述符 TDES0 的 DPAD 位复位，向发送帧附加零 (PAD)，使数据长度正好为 46 字节，以满足 IEEE802.3 的最小数据字段要求，也可选择不附加任何填充值。

MAC 还会计算帧检查序列 (FCS) 字段的循环冗余校验 (CRC)，根据发送描述符 TDES0 的 DCRC 位和 CRCR 位的设定可选择将校验值插入或替换到正在发送的数据中，具体请参考“[常规型 Tx 描述符](#)”和“[增强型 Tx 描述符](#)”。



说明

CRC 替换功能使能时，必须使能增强型 Tx 描述符。

如果将 MAC 编程为不将 CRC 值附加到以太网帧的末尾，则不发送计算出的 CRC。但当 TDES0 的 DPAD 位为 0 时，小于 60 字节 (DA+SA+LT+数据) 的帧会在数据后附加填充 PAD，CRC 结果也将自动附加在填充帧的末尾。

源地址 SA 处理

从应用过来的数据如果不含源地址字段，MAC 可对该字段进行插入；如果含有源地址字段，MAC 可选择用 MAC 地址寄存器 0~1 中设定的值对该字段进行替换。具体的对 SA 字段处理方式可以通过 MAC 配置寄存器 (ETH_MAC_CONFIGR) 的 SAIRC 位或描述符 TDES1 的 SAIRC 位进行控制。



说明

在通过描述符方式使能该功能时，必须使能增强型 Tx 描述符。

VLAN 标识处理

MAC 还可以对从应用来的 VLAN 帧的标识符字段进行插入、替换、删除处理。具体的处理方式可以通过 VLAN 标记发送控制寄存器 (ETH_MAC_VTACTLR) 的 VLANC 位或描述符 TDES0 的 VLANC 位进行控制。



说明

在通过描述符方式使能该功能时，必须使能增强型 Tx 描述符。

发送状态字

在向 MAC 内核传输以太网帧结束时，以及内核完成帧的发送后，发送状态会通过发送描述符提供给应用。发送状态的详细说明参见 TDES0 中的位[17:0]。如果使能 IEEE1588 时间戳功能，将返回特定帧的 64 位时间戳以及发送状态。

发送暂停帧

在全双工模式下，当发送流量控制使能位 (ETH_MAC_FLOCTL 中的 TFE 位) 置 1 时，MAC 将生成暂停帧并需要根据需要发送暂停帧。暂停帧与计算出的 CRC 附加在一起一并发送。可以通过两种方式启动暂停帧的生成。

- 通过将 ETH_MAC_FLOCTL 寄存器中的 FCA 位置 1 来请求流量控制，则 MAC 将生成并发送单个暂停帧。生成的帧中的暂停时间值为 ETH_MAC_FLOCTL 中编程的暂停时间值。要在先前发送的暂停帧中指定的时间之前延长或结束暂停时间，用户必须在应用程序中以适当的值编程 ETH_MAC_FLOCTL 寄存器中的 PAUSET 暂停时间值，然后请求另一次暂停帧发送。
- Rx FIFO 填满时并且应用程序已请求流量控制，则 MAC 将生成并发送暂停帧。生成的帧中的暂停时间值为 ETH_MAC_FLOCTL 中编程的暂停时间值。如果在此暂停时间结束前，Rx FIFO 在可配置的时隙数 (ETH_MAC_FLOCTL 中的 PLT 位) 期间保持填满状态，将发送第二个暂停帧。只要 Rx FIFO 保持填满状态，该过程将一直重复下去。如果在暂停时间结束之前不再满足此条件，MAC 将发送一个暂停时间为零的暂停帧，向远程端口表明 Rx FIFO 已准备好接收新数据帧。

COE 引擎

在网络互联过程中，基本的通信协议 (例如 TCP 和 UDP) 都实施校验和，并配备校验和字段，这有助于确定通过网络发送的数据的完整性。由于以太网最广泛的用途是通过 IP 数据报封装 TCP 和 UDP，因此以太网 MAC 控制器具有校验和减荷功能 (COE 引擎)，该功能支持发送路径中的校验和计算、插入以及接收路径中的校验和计算、检测。本节介绍 COE 引擎在发送中的操作，30.4.1.3.6 节介绍 COE 引擎在接收中的操作。COE 引擎通过完整的帧来计算 TCP、UDP 或 ICMP 的校验和，然后将其插入报头的校验和字段。由于此要求，仅当 Tx FIFO 配置为存储转发模式 (ETH_DMA_OPRMODR 寄存器中的 TSF 位置 1) 时，才使能此功能。如果内核配置为阈值模式，则将绕过 COE 引擎。

必须确保 Tx FIFO 足够深，使其能存储一个完整的帧，以便将该帧传输到 MAC 内核发送器。如果 FIFO 的深度小于输入以太网帧的大小，则将绕过 TCP/UDP/ICMP 校验和插入功能，仅支持 IPv4 报头校验和插入功能，即使在存储转发模式下也是如此。

发送校验和减荷功能的控制可通过将 CIC 位 (TDES0 中的[23:22]) 设定来实现每个帧的校验和控制。

● IP 报头校验和

在 IPv4 数据报中，报头字段的完整性由 16 位头校验和字段 (IPv4 数据报的第 11 字节和第 12 字节) 指示。当以太网帧的类型字段值为 0x0800 且 IP 数据报的版本字段值为 0x4 时，COE 引擎将检测到 IPv4 数据报。计算期间，将忽略输入帧的校验和字段并将其替换为计算出的值。IPv6 报头没有校验和字段，因此，COE 不修改 IPv6 报头字段。IPv4 报头校验和计算的结果由发送状态中的 IP 报头错误状态位 (TDES0 的 IHE 位) 指示。只要以太网类型字段的值和 IP 报头版本字段的值不一致，或当以太网帧没有足够的数据 (如 IP 报头长度字段所指示) 时，此状态位将置 1。

因此，总结下来，在以下情况下会发生 IP 报头错误：

1. 对于 IPv4 数据报：

- ▶ 以太网类型为 0x0800，但 IP 报头版本字段不等于 0x4
- ▶ IPv4 报头长度字段指示小于 0x5 (20 字节) 的值
- ▶ 总的帧长度小于 IPv4 报头长度字段给定的值

2. 对于 IPv6 数据报：

- ▶ 以太网类型为 0x86DD，但 IP 报头版本字段不等于 0x6
- ▶ 帧在 IPv6 报头 (40 字节) 之前结束，或已完全接收到扩展报头 (如扩展报头中相应的报头长度字段中给定)

如果以太网类型字段指示为 IPv4 有效负载，即使 COE 引擎测到 IP 报头错误，也会插入 IPv4 报头校验和。

● TCP/UDP/ICMP 校验和

TCP/UDP/ICMP 校验和功能会对 IPv4 或 IPv6 报头（包括扩展报头）进行处理，并确定封装的有效负载是 TCP、UDP 还是 ICMP。

在下面两种情况下，COE 引擎会无效：

- 1. 非 TCP、非 UDP 或非 ICMP/ICMPv6 的有效负载。
- 2. 分段的 IP 帧（IPv4 或 IPv6）、带安全功能（例如，验证报头或封装的安全有效负载）的 IP 帧和带路由报头的 IPv6 帧。

在 COE 引擎有效的情况下，它会计算 TCP、UDP 或 ICMP 有效负载的校验和，然后将其插入报头中相应的字段。它可工作在以下两种方式：

- 1. TCP、UDP 或 ICMPv6 伪报头并未包含在校验和计算中，并假定其存在于输入帧的校验和字段中。校验和字段包含在校验和计算中，然后替换为最终计算出的校验和。
- 2. TCP、UDP 或 ICMPv6 伪报头包含在校验和计算中，将忽略校验和字段，并使用最终计算出的值覆盖校验和字段。

**说明**

对于 ICMP-over-IPv4 数据包，由于没有为其定义伪报头，因此在这两种模式下 ICMP 数据包中的校验和字段都必须始终为 0x0000。如果不等于 0x0000，可能向数据包插入不正确的校验和。

此操作的结果由发送描述符中的有效负载错误状态位（TDES0 的 TPCE 位）指示。当检测到下列情况之一时，TPCE 位置 1：

- 1. 在存储转发模式下，帧已转发到 MAC 发送器，但帧结束未写入 Rx FIFO
- 2. 在接收到 IP 报头中的有效负载长度字段指示的字节数前，数据包已结束

**说明**

当数据包的长度大于指示的有效负载长度时，将字节作为填充字节忽略，并且不报告错误。检测到第一种类型的错误时，不修改 TCP、UDP 或 ICMP 报头。对于第二种错误类型，计算的校验和仍将插入相应的报头字段。

发送时序

在 RMII 接口时，来自 MII 的每个半字节都在 RMII 上发送，一次发送双位，双位的发送顺序如下图所示。首先发送位序较低的位（D0 和 D1），再发送位序较高的位（D2 和 D3）。

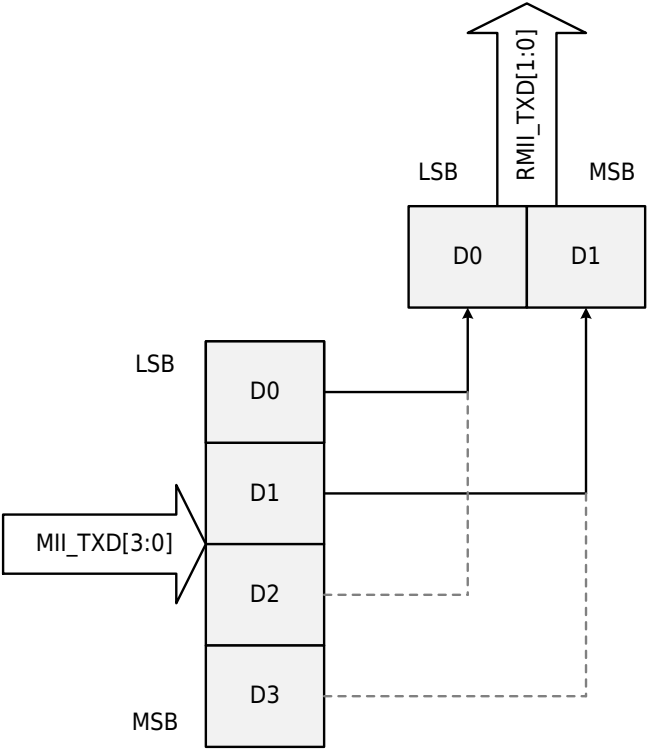


图 38-6 MII/RMII 发送位序

以下为基本的发送时序图。

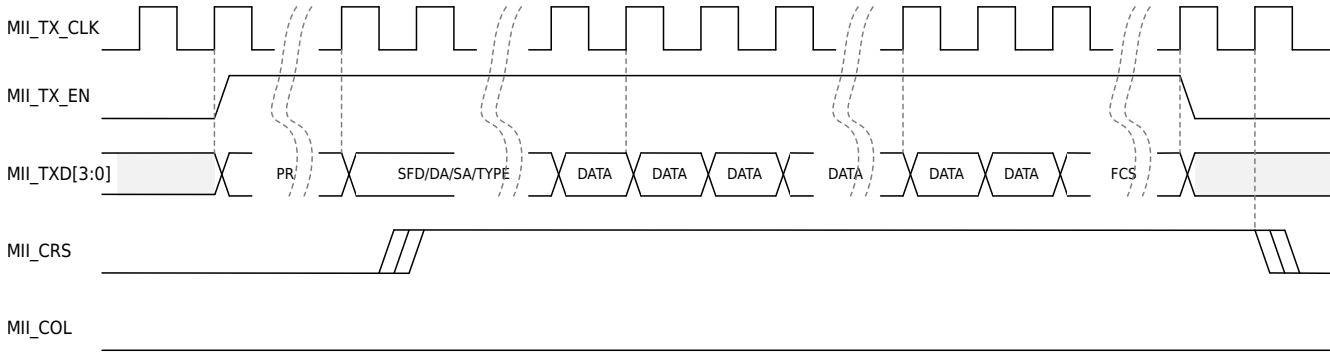


图 38-7 无冲突发送图

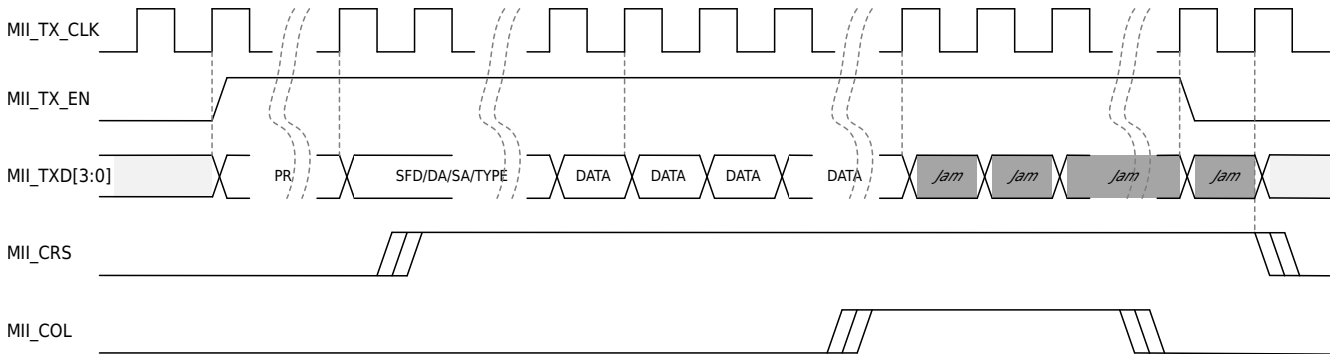


图 38-8 有冲突发送图

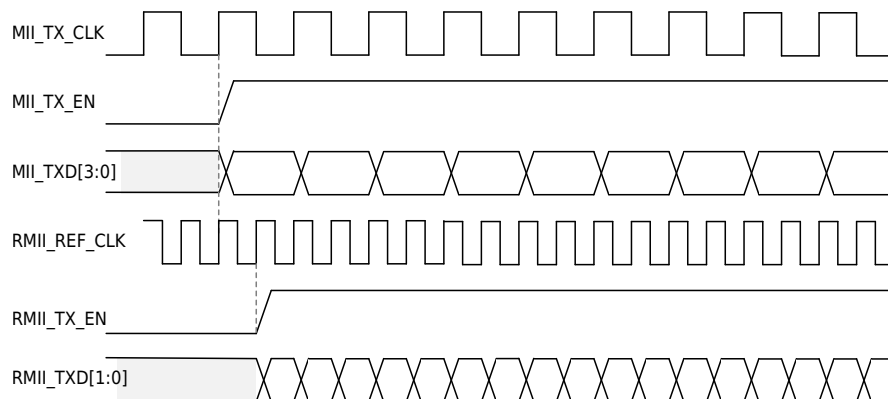


图 38-9 MII/RMII 模式下的发送图

38.3.3.4 MAC 帧接收动作

MAC 控制器接收的帧将送入 Rx FIFO，在接收达到一定数量条件时，就会指示 RxDMA 进行数据传送，RxDMA 可向 AHB 接口发起预配置的突发传输。

以太网 MAC 控制器搭载 2KB 的 Rx FIFO。RxDMA 将 Rx FIFO 中的数据输送到系统存储器有两种操作模式：在阈值模式下，当 Rx FIFO 接收收到的字节数大于配置的接收阈值时（使用 ETH_DMA_OPRMODR 寄存器中的 RTC 位配置）或完整的数据包时，数据将发起输送，其可用性将通知给 RxDMA。RxDMA 向 AHB 接口发起传输后，数据传输将从 Rx FIFO 持续进行，直到传输完成整个数据包。完成 EOF 帧的传输后，接收状态字将返回并发送到 RxDMA 控制器。在该模式下，某些错误帧不会被丢弃，因为在帧结束时接收到错误状态，而此时已从 Rx FIFO 将数据传送进系统存储器。

在存储转发模式下，当帧完全写入 Rx FIFO 后才可读出帧。在此模式下，将丢弃所有错误帧（如果内核配置为执行此操作），这样只会读出有效帧并将其转发到应用。

接收数据后状态立即可用，因此只要 Rx FIFO 未滿，就可以向 Rx FIFO 中存储帧。

当 MAC 在 MII 上检测到 SFD 时，将启动接收操作。MAC 内核将去除报头和 SFD，然后再继续处理帧。检查报头字段以进行过滤，FCS 字段用于验证帧的 CRC。如果帧未通过地址滤波器，则在内核中丢弃该帧。

接收基本协议

MAC 接收数据时，首先去除接收的帧的报头和 SFD。检测到 SFD 后，开始向 Rx FIFO 发送以太网帧数据，从 SFD 后面的第一个字节（目标地址）开始发送。如果使能 IEEE1588 时间戳功能，则在 MII 上检测到任何帧的 SFD 时，都将获取系统时间的快照。除非 MAC 滤出并丢弃帧，否则此时间戳将传递给应用。

如果接收的帧长度/类型字段小于 0x600 并且为 MAC 编程了自动去除 CRC/PAD 选项，则 MAC 将向 Rx FIFO 发送帧数据（数据量不超过长度/类型字段中指定的数量），然后开始丢弃字节（包括 FCS 字段）。如果长度/类型字段大于或等于 0x600，则不管编程的自动 CRC 去除选项的值如何，MAC 都会向 Rx FIFO 发送所有接收到的以太网帧数据。

默认情况下，使能 MAC 看门狗定时器，即超过 2048 个字节（DA+SA+LT+数据+PAD+FCS）的帧会被切断。可通过对 MAC 配置寄存器（ETH_MAC_CONFIGR）中的看门狗禁止（MWD）位编程来禁止此功能。但是，即使禁止看门狗定时器，仍将切断大于 16KB 的帧并给出看门狗超时状态。

接收错误处理

如果在从 MAC 接收 EOF 数据之前 Rx FIFO 已滿，则将声明上溢并丢弃整个帧，同时帧丢失统计寄存器（ETH_DMA_RFRNCNTR）中的上溢计数器将递增。由于上溢，状态将指示这是一个部分帧。如果使能相应功能（ETH_DMA_OPRMODR 中的 FEF 和 FUF 位），Rx FIFO 可过滤错误帧和过小帧。

在阈值模式下，从 Rx FIFO 读取帧 SOF 时，可获取当前帧的基本信息，通过与 Rx 描述符的基本设定比对，则可判定该帧是否为错误帧，并丢弃整个错误帧；在存储转发模式下，则可在数据传送之前过滤并丢弃所有错误帧。

CRC 计算和 PAD 去除

MAC 内核将计算接收的帧（包括目标地址字段到 FCS 字段）的 32 位 CRC，并检查接收帧中的任何 CRC 错误。不管是否自动去除 CRC/PAD（通过 ETH_MAC_CONFIGR 寄存器 ACS 位控制），MAC 都将接收整个帧来计算所接收帧的 CRC 校验，并将校验的结果通过接收描述符 RDES0 的 CRE 位反馈给应用。

接收状态字

以太网帧接收结束时，MAC 输出接收状态通过 RxDMA 送到接收描述符 RDES0 中以供应用参考查询。

接收暂停帧

在帧发送过程中，MAC 将检测接收暂停帧并暂停帧发送，暂停时间为接收的暂停帧内指定的延迟（仅限全双工模式）。可通过 ETH_MAC_FLOCTL 中的 RFE 位使能或禁止暂停帧检测功能。使能接收流量控制后，将开始监视接收帧的目标地址是否与控制帧的多播地址（0x0180C2000001）匹配。如果检测到匹配（接收的帧的目标地址与保留的控制帧的目标地址匹配），MAC 将根据 ETH_MAC_FLTCTL 中的 PCF 位来决定是否将接收的控制帧传输到应用。

MAC 还将对接收到的控制帧的类型、操作码和暂停定时器字段进行解码。如果状态的字节计数指示 64 个字节，并且不存在任何 CRC 错误，则 MAC 发送器将暂停任何数据帧的发送，暂停时间为解码的暂停时间值乘以时隙（对于 10/100Mbps 模式，均为 64 字节时间）。同时，如果检测到另一个零暂停时间值的暂停帧，MAC 将复位暂停时间并管理新的暂停请求。如果接收到的控制帧与类型字段（0x8808）、操作码（0x00001）以及字节长度（64 字节）均不匹配，或存在 CRC 错误，则 MAC 不会生成暂停。

对于具有多播目标地址的暂停帧，MAC 将根据地址匹配来过滤帧；对于具有单播目标地址的暂停帧，MAC 将根据 DA 字段是否与 MAC 地址寄存器 0 的内容匹配以及 ETH_MAC_FLOCTL 中的 UNP 位是否置 1（检测具有单播目标地址的暂停帧）来进行过滤。ETH_MAC_FLTCTL 寄存器中的 PCF 位可对控制帧的过滤以及地址过滤进行控制。

COE 引擎

以太网 MAC 控制器支持对接收的以太网帧中的 IPv4 和 IPv6 帧进行检测和处理，以确保数据完整性。通过将 ETH_MAC_CONFIGR 寄存器中的 IPCO 位置 1 来使能接收 COE 引擎。MAC 通过检查所接收的以太网帧的类型字段中是否存在值 0x0800 或 0x86DD，来识别 IPv4 或 IPv6 帧。此识别方法也适用于带 VLAN 标记的帧。

接收 COE 引擎可计算 IPv4 报头校验和，并检查其是否与接收的 IPv4 报头校验和匹配。如果指示的有效负载类型（以太网类型字段）与 IP 报头版本之间有任何不匹配，或接收的帧的字节数小于 IPv4 报头的长度字段中指示的数量（IPv4 或 IPv6 报头中的可用字节数少于 20），IP 报头错误位将置 1。

接收 COE 引擎还可以识别接收的 IP 数据报（IPv4 或 IPv6）中的 TCP、UDP 或 ICMP 有效负载，并正确计算此类有效负载的校验和字段，包括用于校验和计算的 TCP/UDP/ICMPv6 伪报头字节，并检查接收的校验和字段是否与计算的值匹配。此操作的结果由接收描述符 RDES4 中的 IP 有效负载错误位给出。如果 TCP、UDP 或 ICMP 有效负载的长度与 IP 报头中给出的预期有效负载长度不匹配，此状态位也将置 1。

COE 引擎将绕过分段 IP 数据报的有效负载、带安全功能的 IP 数据报、IPv6 路由报头以及除 TCP、UDP 或 ICMP 以外的有效负载。

接收时序

在 RMII 接口时，MAC 内部的每个半字节都从接收自 RMII 的双位发送到 MII，半字节发送顺序如下图所示。先接收位序较低的位（D0 和 D1），再接收位序较高的位（D2 和 D3）。

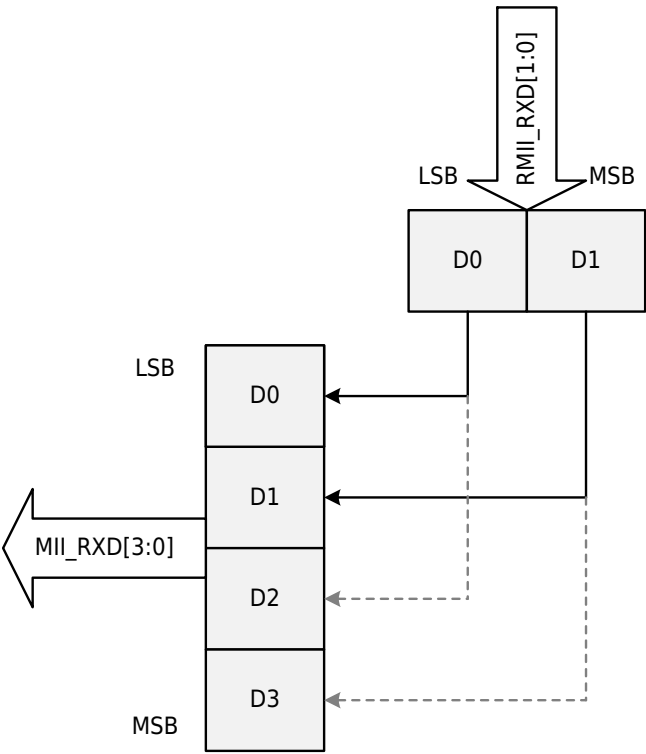


图 38-10 MII/RMII 接收位序

以下为基本的发送时序图。

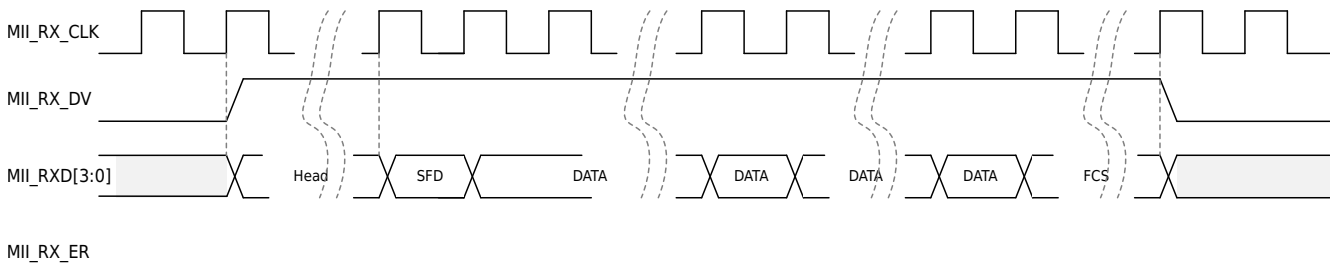


图 38-11 无错误发送图

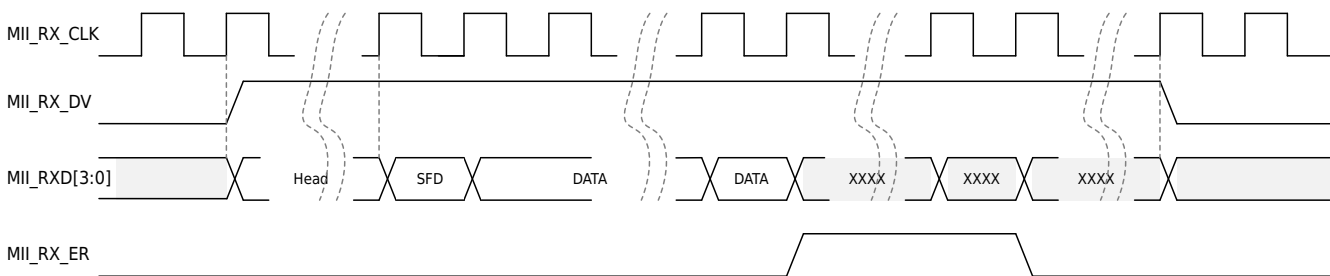


图 38-12 有错误发送图

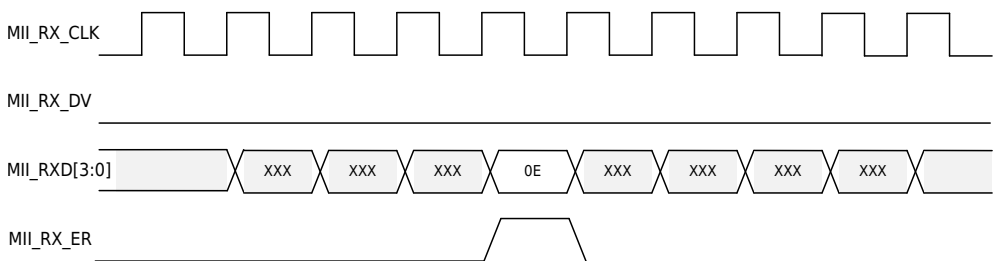


图 38-13 假载波指示下的接收图

38.3.3.5 MAC 帧过滤

MAC 可以对接收到的各种帧进行过滤以选择应用需要的帧，帧过滤方式包括以下内容：

1. MAC 帧的目标地址、源地址过滤
2. VLAN 帧的标识符过滤
3. LAY3 IP 报的源地址、目标地址过滤
4. LAY4 TCP、UDP 报的源端口、目标端口过滤

各帧过滤方式的基本说明分别在下面的章节介绍：

MAC 源地址过滤

MAC 可根据接收的帧的源地址字段来执行完美过滤。默认情况下，MAC 将 SA 字段与 MAC 地址寄存器中编程的值进行比较。可通过将 MAC 地址寄存器中的位 30 (SA) 置 1，来将 MAC 地址寄存器 1~4 配置为 SA 字段进行比较。如果帧过滤控制寄存器 (ETH_MAC_FLTCTLR) 中的 SAF 位置 1，则 MAC 将丢弃未通过 SA 过滤的帧。否则，SA 过滤的结果将通过接收状态字中的状态位给出（参见接收描述符字 RDES0）。

注：SAF 位置 1 时，对 SA 过滤和 DA 过滤的结果进行与运算，以决定是否需要转发帧。这意味着任何一个过滤未通过都将丢弃帧。两个过滤必须都通过，才能将帧转发到应用。

ETHMAC 控制器将 MAC 地址寄存器 1~4 中地址值的各个字节与接收的相应 SA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节，从而实现 SA 的组地址过滤。

对于源地址过滤，可在最终输出时选择反转过滤匹配结果，即在 SA 地址比较匹配时认定为过滤失败。该功能由帧过滤控制寄存器 (ETH_MAC_FLTCTLR) 的 SAIF 位控制。

MAC 目标地址过滤-单播

MAC 支持多达 5 个用于单播过滤的 MAC 目标地址。如果选择完美过滤（帧过滤控制寄存器中的 HUC 位设为 0），MAC 将接收的单播地址的所有 48 位目标地址与编程的 MAC 地址进行比较来确定是否匹配。默认情况下，始终使能 MAC 地址寄存器 0，其它 MAC 地址寄存器 1~4 则通过单独的使能位进行选择。

ETHMAC 控制器将 MAC 地址寄存器 1~4 的各个字节与接收的相应 DA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节，从而实现 DA 的组地址过滤。

MAC 目标地址过滤-多播

通过将帧过滤控制寄存器中的 PMF 位置 1，将 MAC 编程为通过所有多播帧。如果 PAM 位复位，MAC 将根据帧过滤寄存器中的 HMC 位执行对多播地址的过滤。在完美过滤模式下，将多播地址与编程的 MAC 地址寄存器 1~4 进行比较。该过滤方式也支持组地址过滤。

MAC 目标地址过滤-广播

在默认模式下，MAC 不过滤任何广播帧。但是，如果将帧过滤寄存器中的 DBF 位置 1 来将 MAC 编程为拒绝所有广播帧，则会丢弃任何广播帧。

对于目标地址过滤（单播、多播、广播），也可在最终输出时选择反转过滤匹配结果，即在 DA 地址比较匹配时认定为过滤失败。该功能由帧过滤控制寄存器 (ETH_MAC_FLTCTLR) 的 DAIF 位控制，同样适用于 Hash 过滤结果。

MAC 目标地址 Hash 过滤

对于单播目标地址和多播目标地址的过滤也支持 Hash 过滤方式。在 Hash 过滤模式（HUC 位/HMC 位置 1）下，MAC 将使用 64 位 Hash 表执行对单播地址或多播地址的不完美过滤，即 Hash 过滤。对于 Hash 过滤，MAC 将使用接收的目标地址的 6 个高 CRC 位来索引 Hash 表的内容。值为 000000 时，选取所选寄存器中的位 0；值为 111111 时，选取散列表寄存器中的位 63。如果相应位（由 6 位 CRC 指示）已置 1，将认为单播帧或多播帧已通过 Hash 过滤，否则认为帧未能通过 Hash 过滤。

**说明**

CRC 是使用下列多项式编码的 32 位值：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

VLAN 标识符过滤

当接收帧为 VLAN 帧时，MAC 可通过设定帧过滤控制寄存器 (ETH_MAC_FLTCTLR) 的 VTFE 位来使能对 VLAN 标识符字段的过滤。过滤用比较值设置在 VLAN 标记接收过滤寄存器 (ETH_MAC_VTAFLTR) 的 VLFLT 位，同时可通过设定该寄存器的 VTAL 位来选择对全 VLAN 标识符字段进行比较过滤还是仅比较低 12 位标识符。

对于 VLAN 标识符字段的过滤，也可在最终输出时选择反转过滤匹配结果，即在标识符字段比较匹配时认定为过滤失败。该功能由 VLAN 标记接收过滤寄存器 (ETH_MAC_VTAFLTR) 的 VTIM 位控制，同样适用于 Hash 过滤结果。

VLAN 标识符 Hash 过滤

对于 VLAN 标识符的过滤也可以选择 Hash 过滤方式。在 Hash 过滤模式 (ETH_MAC_VTAFLTR 的 VTHM 位置 1) 下，MAC 将使用 16 位 Hash 表执行对 VLAN 标识符的不完美过滤，即 Hash 过滤。对于 Hash 过滤，MAC 将使用 VLAN 标识符的 CRC 位来索引 Hash 表的内容。值为 1 时，表明通过 VLAN 标识符过滤；值为 0 时，表明未通过 VLAN 标识符过滤。

**说明**

CRC 是使用下列多项式编码的 32 位值：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

L3 源地址&目标地址过滤

对接收到的 LAY3 层的 IPv4 报和 IPv6 报也支持源地址和目标地址的过滤。通过 LAY3LAY4 控制寄存器的 L3SAM 位和 L3DAM 位可分别使能源地址过滤与目标地址过滤。源地址或目标地址的比较值在 LAY3 地址寄存器 (L3ADDR0~3) 中设定，具体请参考该寄存器说明。

LAY3 层 IP 地址的过滤可以选择高位屏蔽功能，以实现组地址的过滤。源地址和目标地址的高位屏蔽过滤控制分别由 LAY3LAY4 控制寄存器 (ETH_MAC_L34CTLR) 的 HSBM、HDBM 位设定控制。

对于 LAY3 层 IP 报的地址过滤，也可在最终输出时选择反转过滤匹配结果，即在地址字段比较匹配时认定为过滤失败。该功能由 LAY3LAY4 控制寄存器 (ETH_MAC_L34CTLR) 的 L3SAIM、L3DAIM 位控制。

L4 源端口&目标端口过滤

对接收到的 LAY4 层的 TCP 报和 UDP 报也支持源端口和目标端口的过滤。通过 LAY3LAY4 控制寄存器的 L4SPM 位和 L4DPM 位可分别使能源端口过滤与目标端口过滤。源端口或目标端口的比较值在 LAY4 端口寄存器 (L4PORTR) 中设定，具体请参考该寄存器说明。

对于 LAY4 层 TCP 报、UDP 报的端口过滤，也可在最终输出时选择反转过滤匹配结果，即在端口字段比较匹配时认定为过滤失败。该功能由 LAY3LAY4 控制寄存器 (ETH_MAC_L34CTLR) 的 L4SPIM、L3DPIM 位控制。

38.3.3.6 MAC Loopback 模式

MAC 支持对发送到其接收器的帧进行回送。默认情况下，禁止 MAC 回送功能，可通过编程 MAC 控制寄存器 (ETH_MAC_CONFIGR) 中的 LM 位使能该功能。

38.3.3.7 MAC 管理计数器 (MMC)

MAC 管理计数器 (MMC) 搭载一组寄存器来收集有关已接收帧和已发送帧的统计信息。其中包括一个用于控制各寄存器行为的控制寄存器 (ETH_MMC_MMCCTLR)、两个包含当前接收和发送统计状态的 32 位寄存器 (ETH_MMC_REVSTSR 和 ETH_MMC_TRSSTSR)、两个包含接收和发送中断控制的 32 位寄存器

(ETH_MMCRITCTLR 和 ETH_MMCTITCTLR)，以及各种帧类型的统计寄存器。这些统计寄存器均可从应用中访问，寄存器章节中详细描述了这些统计寄存器的用途和意义，请参考 ETH_MMCR 寄存器章节。MAC 管理计数器 (MMC) 针对通过地址过滤的接收帧更新相应的统计寄存器，已丢弃帧的统计信息不会进行更新，除非丢弃的帧为小于 6 字节的矮帧 (DA 字节不能完全接收)。

如果帧成功发送，则将发送的帧视为“好帧”。即，如果帧发送过程没有因为以下错误而中止，则认为发送的帧是“好帧”：

- Jabber 超时
- 无载波/载波丢失
- 延迟冲突
- 帧下溢
- 过度延迟
- 过度冲突

如果帧接收成功，则将接收的帧视为“好帧”。即，如果接收帧不存在以下错误，则认为接收帧是“好帧”：

- CRC 错误
- 矮帧 (短于 64 字节)
- 对齐错误 (仅限 10/100 Mb/s)
- 长度错误 (仅限非类型帧)
- 超出范围 (仅限非类型帧，超过最大大小)
- MII_RXER 输入错误



说明

最大帧大小取决于帧类型，如下：

- 无标识符帧的最大字节数：1518
- VLAN 帧的最大字节数：1522

38.3.3.8 MAC 电源管理 (PMT)

ETHMAC 控制器支持电源管理 (PMT) 机制。可通过远程唤醒帧使能位以及魔术数据包使能位使能 PMT 模块。这些使能位 (WKEN 和 MPEN) 位于 ETH_MAC_PMTCTLR 寄存器中，可由应用编程。在使能 Power Down 模式 (ETH_MAC_PMTCTLR.PWDN=1) 时，MAC 将丢弃所有接收到的帧并且不会将这些帧转发给应用。仅当使能相应检测位且接收到对应的魔术数据包或远程唤醒帧时，MAC 控制器才会退出 Power Down 模式。

远程唤醒帧过滤

ETHMAC 控制器有 8 个唤醒帧过滤寄存器。要对每个寄存器执行写操作，需要逐个值加载远程唤醒帧过滤寄存器 (ETH_MAC_RTWKFFR)。连续加载远程唤醒帧过滤寄存器八次，便可对远程唤醒帧过滤寄存器加载所需的值。读操作与写操作相同，要读取八个值，必须读唤醒帧过滤寄存器八次后，才能到达最后一个寄存器。每次读/写操作都会将唤醒帧过滤寄存器指针指向下一个过滤寄存器。通过读取 PMT 控制状态寄存器 (ETH_MAC_PMTCTLR) 的 RTWKPT 位可以知道当前的读写操作在哪个过滤寄存器上，同时可通过对该寄存器的 RTWKFR 位置 1 复位位内部的指针，复位之后可对该远程唤醒帧过滤寄存器重新进行操作。下图是 ETHMAC 内部的 8 个远程唤醒帧过滤寄存器。

ETH_MAC_RTWKFFR 0	FLT0							
	Byte Mask							
ETH_MAC_RTWKFFR 1	FLT1							
	Byte Mask							
ETH_MAC_RTWKFFR 2	FLT2							
	Byte Mask							
ETH_MAC_RTWKFFR 3	FLT3							
	Byte Mask							
ETH_MAC_RTWKFFR 4	-	FLT3 Command	-	FLT2 Command	-	FLT1 Command	-	FLT0 Command
ETH_MAC_RTWKFFR 5	FLT3 Offset		FLT2 Offset		FLT1 Offset		FLT0 Offset	
ETH_MAC_RTWKFFR 6	FLT1 CRC16				FLT0 CRC16			
	FLT3 CRC16				FLT2 CRC16			

图 38-14 远程唤醒帧过滤寄存器

上图中各部分的功能说明如下：

- 过滤器 i 字节掩码 (Byte Mask)
该寄存器定义过滤器 i 检测帧的哪些字节来确定帧是否为唤醒帧 (i=0~3)。MSB (第 31 位) 必须为零, 位 j[30:0]为字节掩码。如果将字节掩码的位 j (字节数) 置 1, 则传入帧的过滤器 i 偏移+j 由 CRC 模块处理; 否则将忽略过滤器 i 偏移+j。
- 过滤器 i 命令 (Command)
该 4 位命令控制过滤器 i 操作 (i=0~3)。Bit3 为地址类型选择, 置位时只检测多播帧, 反之只检测单播帧; Bit2 和 Bit1 为保留位; Bit0 为过滤器 i 的使能位, 置位时使能过滤器 i, 反之禁止过滤器 i。
- 过滤器 i 偏移 (Offset)
该寄存器定义过滤器 i 要检测的帧的偏移 (i=0~3)。该 8 位模式偏移是要检测的过滤器 i 第一个字节的偏移。允许的最小值为 12, 表示帧的第 13 个字节 (偏移值 0 表示帧的第一个字节)。
- 过滤器 i CRC-16 (CRC16)
该寄存器包含预先写入的 CRC-16 值 (i=0~3), 用于与帧数据 (过滤器 i Offset 且 Byte Mask 对应之后) 计算的 CRC-16 值进行比较。
CRC-16 的生成多项式为: $G(x) = x^{16} + x^{15} + x^2 + 1$ 。

远程唤醒帧检测

当 MAC 处于 Power Down 模式并已将 PMT 控制状态寄存器 (ETH_MAC_PMTCTLR) 中的远程唤醒位 WKEN 置 1 时, MAC 可在接收到远程唤醒帧后恢复正常工作。PMT 支持四个可编程过滤器, 这些过滤器可由应用配置为不同的接收帧模式。如果传入帧通过过滤器的地址过滤, 并且过滤器 CRC-16 与传入的检测模式相匹配, 则可接收唤醒帧。只需要检查唤醒帧是否存在长度错误、FCS 错误、帧尾错误、MII 错误、冲突, 确保其不是矮帧, 即使唤醒帧长度超过 512 字节, 但只要帧的 CRC 值有效, 帧便有效。

ETH_MAC_PMTCTLR 寄存器中的唤醒帧检测状态位 (WKFR) 会针对每个接收到的远程唤醒帧进行更新。此外, 还会生成 PMT 中断 (如果已使能) 来指示已接收到远程唤醒帧。

魔术数据包检测

魔术数据包是基于 AMD 公司的魔术数据包技术对网络上处于睡眠模式下的器件上电。MAC 接收称为魔术数据包的特定信息包，此信息包的目标地址是网络上的节点。MAC 控制器只对目标地址为本器件或多播地址的魔术数据包进行检查，以确定这些数据包是否满足唤醒要求。对通过地址过滤（单播或多播）的魔术数据包进行检测，确定其是否符合远程唤醒数据格式，即 6 个字节的数据所有位为全“1”的数据包加上重复出现 16 次的 MAC 地址。

应用通过向 ETH_MAC_PMTCTLR 寄存器中的 MPEN 位写入 1 来使能魔术数据包唤醒。PMT 模块持续监测目标地址为对应于指定魔术数据包模式的节点的每个帧。检查每个接收帧的目标地址和源地址字段之后是否为 0xFFFFFFFF 形式的同步数据流。之后，PMT 模块检查帧是否存在重复 16 次且没有任何断开或中断的 MAC 地址。如果重复 16 次的地址中存在中断，则会再次扫描传入帧中是否存在 0xFFFFFFFF 形式。这 16 次重复可位于帧中的任何位置，但必须在同步数据流后面（0xFFFFFFFF）。只要检测到重复 16 次的 MAC 地址，器件就可以接收多播帧。例如，如果节点 MAC 地址为 0x001122334455，则 MAC 扫描的数据序列为：

```
目标地址 源地址 ... FFFF FFFF FFFF
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
... CRC
```

时，认为接收到了一个魔术唤醒包。ETH_MAC_PMTCTLR 寄存器中的魔术数据包检测会针对接收到的魔术数据包进行更新。此外，还会生成 PMT 中断（如果已使能）来指示已接收到魔术数据包。

掉电期间系统状态

- 1. 在 Power Down 模式期间应保持 MAC 接收器使能状态，原因是此时要对魔术数据包/远程唤醒帧进行相关的接收检测，通过将 ETH_MAC_CONFIGR 寄存器中的 RE 位置 1 来使能 MAC 接收器。
- 2. 在 Power Down 模式期间应保持 MAC 发送器关闭状态。通过将 ETH_MAC_CONFIGR 寄存器中的 TE 位清零来关闭 MAC 发送器。
- 3. 在 Power Down 模式期间应禁止以太网 DMA，原因是此时无需将魔术数据包/网络唤醒帧复制到系统存储器。通过将 ETH_DMA_OPRMODR 寄存器中的 STT 位和 STR 位清零来关闭 TxDMA 和 RxDMA。

推荐的掉电和唤醒顺序如下：

- 步骤 1. 禁止 TxDMA，并等待所有之前的帧发送完成；
- 步骤 2. 通过将 ETH_MAC_CONFIGR 寄存器中的 RE 位和 TE 位清零来禁止 MAC 发送器和 MAC 接收器；
- 步骤 3. 等待 RxDMA 清空 Rx FIFO 中的所有帧；
- 步骤 4. 禁止 RxDMA；
- 步骤 5. 通过将 ETH_MAC_PMTCTLR 寄存器中的 MPEN/WKEN 位置 1 来使能魔术数据包/远程唤醒帧检测；
- 步骤 6. 通过将 ETH_MAC_PMTCTLR 寄存器中的 PWDN 位置 1 来使能 MAC 掉电模式；
- 步骤 7. 通过将 ETH_MAC_CONFIGR 寄存器中的 RE 位置 1 来使能 MAC 接收器；
- 步骤 8. 等待接收魔术数据包和远程唤醒帧；
- 步骤 9. 接收到有效唤醒帧时，以太网控制器退出掉电模式；
- 步骤 10. 读取 ETH_MAC_PMTCTLR 以清零 PMT 事件标志，使能 MAC 发送器、TxDMA 和 RxDMA。

38.3.4 ETH_PTP 功能

38.3.4.1 ETH_PTP 功能概述

IEEE1588 标准定义了一种协议，此协议支持使用网络通信、局域计算和分布式对象等技术实现的测量和控制系统中的精密时钟同步。该协议适用于利用支持多播消息传送的局域网（包括但不限于以太网）进行通信的

系统。该协议用于对非均匀系统进行同步，这类系统包含固有精度、分辨率和稳定性都不断变化的时钟。该协议支持亚秒范围的系统级同步精度，并且需要极少的网络和本地时钟计算资源。这种基于消息的协议称为精密时间协议（PTP），它通过 UDP/IP 传送。系统或网络归类为主节点和从节点，用于分配时序/时钟信息。该协议通过交换 PTP 报文来同步从节点与主节点，如下图所示。

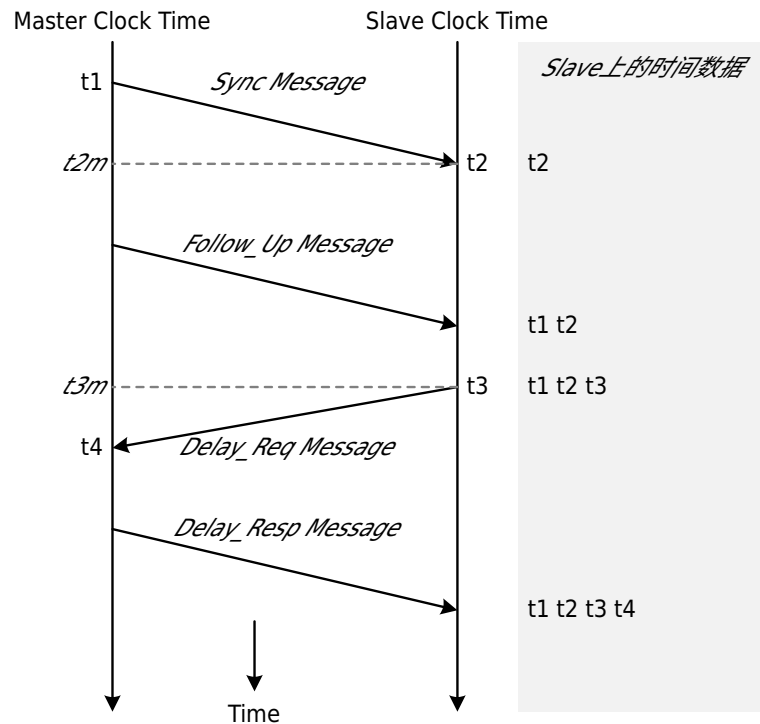


图 38-15 时钟同步图

1. 主节点向其所有节点广播 PTP 同步报文。同步报文包含主节点的参考时间信息。报文离开主节点系统的时间为 t1。对于以太网端口，必须通过 MII 接口来捕获该时间；
2. 从节点接收同步报文并利用其参考时序捕获准确时间 t2；
3. 主节点随后会向从节点发送一个包含 t1 信息的 Follow_Up 报文以备后用；
4. 从节点向主节点发送一个 Delay_Req 报文，标出该帧离开 MII 的准确时间 t3；
5. 主节点接收该报文，捕获报文进入其系统的准确时间 t4；
6. 主节点将 Delay_Resp 报文中的 t4 信息发送给从节点；
7. 从节点使用 t1、t2、t3 和 t4 这四个值来同步其本地参考时序与主节点的参考时序。

虽然大多数协议都在 UDP 层之上通过软件实现，但需要硬件完成对通过 MII 接口来捕获特定 PTP 包进入或离开以太网端口时的准确时间。

38.3.4.2 PTP 参考时序源

根据 IEEE1588 规范的定义，要获取时间快照，内核需要一个 64 位格式的参考时间（分成两个 32 位通道，高 32 位表示时间的秒数，低 32 位表示时间的亚秒数）。
PTP 参考时钟输入用于在内部生成参考时间（亦称作系统时间）以及捕获时间戳。该参考时钟的频率必须大于等于时间戳计数器的分辨率。主节点与各从节点之间的同步精度目标约为 100ns。精度取决于 PTP 参考时钟输入周期、振荡器的特性（漂移）以及同步过程的频率。
由于从 Tx 和 Rx 时钟输入域到 PTP 参考时钟域的同步，时间戳锁存值的不确定度为 1 个参考时钟周期。如果加上分辨率导致的不确定度，则会增加一半的时间戳周期。

38.3.4.3 PTP 报文类型

时间戳功能使能后，可通过时间戳控制寄存器（ETH_PTP_TSPCTLR）的 TSPMTSEL 位来控制对哪一个或哪一类报文进行时间戳快照。具体的对应关系如下表所示。

表 38-7 时间戳快照目标报文

TSPMTSEL[3:0]	时间戳使能的报文类型
0b00X0	SYNC、Follow_Up、Delay_Req、Delay_Resp
0b0001	SYNC
0b0011	Delay_Req
0b01X0	SYNC、Follow_Up、Delay_Req、Delay_RespPdelay_Req、Pdelay_Resp、Pdelay_Resp_Follow_Up
0b0101	SYNC、Pdelay_Req、Pdelay_Resp
0b0111	Delay_Req、Pdelay_Req、Pdelay_Resp
0b10XX	SYNC、Delay_Req
0b11XX	Pdelay_Req、Pdelay_Resp

38.3.4.4 PTP 帧发送功能

在 MII 上输出帧的 SFD 时，将捕获时间戳。对于需要捕获时间戳的帧，可对每个发送帧进行标记以指示是否有必要捕获该帧的时间戳。

无需处理发送帧即可识别 PTP 帧，可通过发送描述符中的控制位执行帧控制。

捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的发送状态一同发送回相应的发送描述符内，从而自动将时间戳与特定 PTP 帧相连。64 位时间戳信息会写回 TDES6 和 TDES7 字段，其中 TDES6 会保持时间戳的 32 个最低有效位。

38.3.4.5 PTP 帧接收功能

使能 IEEE1588 时间戳功能时，以太网 MAC 将捕获 MII 上接收到的所有帧的时间戳。MAC 会在完成帧接收过程时提供时间戳。

捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的接收状态一同发送回相应的接收描述符内。64 位时间戳信息会写回 RDES6 和 RDES7 字段，其中 RDES6 会保持时间戳的 32 个最低有效位。

38.3.4.6 PTP 系统时间校准

使用 PTP 输入参考时钟更新 64 位 PTP 时间。该 PTP 时间可用作时钟源，以获取 MII 上发送或接收的以太网帧的快照（时间戳）。可使用粗校准方式或细校准方式对系统时间定时器进行初始化或校准。

使用粗校准方式时，初始值或偏移值会写入时间戳更新寄存器。对于初始化，会将时间戳更新寄存器中的值写入系统时间计数器；对于系统时间校准，会将偏移值（时间戳更新寄存器）加到系统时间中或从系统时间中减去。

使用细校准方式时，从参考时钟频率相对于主时钟（如 IEEE1588 中定义）的偏移会在一段时间内进行校准，而不像粗校准方式中那样，在单个时钟周期内进行校准。校准时间越长，越有助于保持线性时间，并且不会导致各 PTP 同步消息间隔之间的参考时间发生剧烈变化（或者大型抖动）。在此方法中，会使用一个累加器对基本加数寄存器（ETH_PTP_TSPADDR）中的内容求和，如下图所示。累加器生成的算数进位将用作使系统时间计数器递增的脉冲。累加器和基本加数寄存器均为 32 位寄存器。此处，累加器用作高精度频率乘法器或除法器。

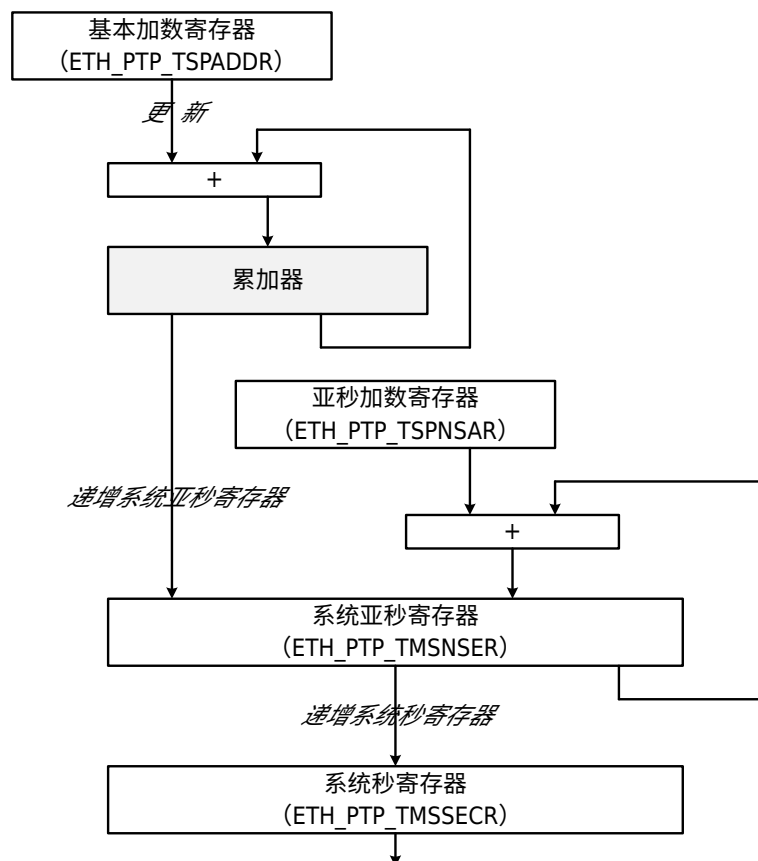


图 38-16 系统时间校准

系统时间更新逻辑需要使用 50MHz 的时钟频率，以达到 20ns 的精度。分频比是指参考时钟频率与所需时钟频率的比率。如果参考时钟频率为 66MHz，则通过计算可知分频比为 $66\text{MHz}/50\text{MHz}=1.32$ ，基本加数寄存器中设置的默认加数值为 $2^{32}/1.32$ ，相当于 $0xC1F07C1F$ ；如果参考时钟向下偏移降至 65MHz，则分频比为 $65\text{MHz}/50\text{MHz}=1.3$ ，基本加数寄存器中要设置的值为 $2^{32}/1.3$ ，相当于 $0xC4EC4EC4$ ；如果时钟向上偏移升至 67MHz，则必须将基本加数寄存器设置为 $0xBF0B7672$ 。当时钟偏移为零时，应编程的默认加法值为 $0xC1F07C1F$ ($2^{32}/1.32$)。

用于亚秒加数寄存器递增的值设定为十进制数 43。这可使系统时间的精度达到 20ns（换句话说，增量步长时间为 20ns）。

软件必须根据同步报文对频率偏移进行计算，并相应更新基本加数寄存器。首先，使用基本加数寄存器中的 FreqCompensationValue0 设置从时钟。该值的计算公式如下：

$$\text{FreqCompensationValue0} = 2^{32} / \text{FreqDivisionRatio}$$

如果起初假定 MasterToSlaveDelay 对于连续的同步消息是相同的，则必须应用下述算法。数个同步周期之后，频率会锁定。从时钟随后可确定 MasterToSlaveDelay 的精确值，并使用新值重新与主时钟同步。

该算法如下：

- 在 MasterSyncTime (n) 时刻，主时钟向从时钟发送同步报文。从时钟在其本地时钟为 SlaveClockTime (n) 时接收到该报文，并用如下公式计算 MasterClockTime (n)：

$$\text{MasterClockTime (n)} = \text{MasterSyncTime (n)} + \text{MasterToSlaveDelay (n)}$$
- 当前同步周期的主时钟计数 MasterClockCount (n) 的计算公式如下：

$$\text{MasterClockCount (n)} = \text{MasterClockTime (n)} - \text{MasterClockTime (n-1)}$$
 （假定 MasterToSlaveDelay 对于同步周期 n 和 n-1 是相同的）
- 当前同步周期的从时钟计数 SlaveClockCount (n) 的计算公式如下：

$$\text{SlaveClockCount (n)} = \text{SlaveClockTime (n)} - \text{SlaveClockTime (n-1)}$$

- 当前同步周期的主从时钟计数差值 SlaveClockCount (n) 的计算公式如下：

$$\text{ClockDiffCount (n)} = \text{MasterClockCount (n)} - \text{SlaveClockCount (n)}$$
- 从时钟的分频系数 FreqScaleFactor (n) 的计算公式如下：

$$\text{FreqScaleFactor (n)} = (\text{MasterClockCount (n)} + \text{ClockDiffCount (n)}) / \text{SlaveClockCount (n)}$$
- 加数寄存器的频率补偿值 FreqCompensationValue (n) 的计算公式如下：

$$\text{FreqCompensationValue (n)} = \text{FreqScaleFactor (n)} \times \text{FreqCompensationValue (n-1)}$$

理论上，该算法可在一个同步周期内实现锁定，但是由于网络传播延迟和工作条件会不断变化，因此可能需要多个周期。

该算法可进行自校准：如果出于某些原因导致最初通过主时钟设置的从时钟不正确，则该算法会花费更多同步周期将其校准。

38.3.4.7 PTP 系统时间生成初始化

通过将时间戳控制寄存器 (ETH_PTP_TSPCTLR) 中的 TSPEN 位置 1 来使能时间戳功能。但之后必须对时间戳计数器进行初始化才能启动时间戳操作。设定顺序如下：

- 步骤 1. 通过将 ETH_MAC_INTMSKR 寄存器中的 TSPIM 位置 1 以屏蔽时间戳触发中断
- 步骤 2. 编程 ETH_PTP_TSPCTLR 寄存器 TSPEN 位以使能时间戳
- 步骤 3. 根据 PTP 时钟频率编程亚秒加数寄存器 (ETH_PTP_TAPNSAR)
- 步骤 4. 编程基本加数寄存器 (ETH_PTP_TSPADDR) 并将时间戳控制寄存器的 TSPADUP 位置 1 (基本加数寄存器更新)
- 步骤 5. 轮询时间戳控制寄存器，确认 TSPADUP 位清零
- 步骤 6. 要选择精密校准方式时，将时间戳控制寄存器的 TSPUPSEL 位置 1
- 步骤 7. 用适当的时间值编程时间戳更新秒寄存器和时间戳更新亚秒寄存器
- 步骤 8. 将时间戳控制寄存器 TSPINI 位置 1 (时间戳初始化)
- 步骤 9. 用时间戳更新寄存器中写入的值初始化时间戳计数器后，时间戳计数器便开始运行
- 步骤 10. 使能 MAC 接收器和发送器以使时间戳功能正常运行

38.3.4.8 PTP PPS 输出

PTP 模块可以将生成的内部时钟通过 PPS (PPS0) 的方式输出到 ETH_PTP_PPS 端口上 (连续输出模式)，用于网络系统的所有节点的时钟同步，同时有两个 PPS (PPS0 和 PPS1) 的输出 (单次输出模式)，可分别作为事件输出控制其它模块。

在 PPS 连续输出模式时，系统时间发生秒进位 (亚秒时间溢出) 后，PPS0 通道可生成一系列特定的脉冲或时钟，供外部网络节点时钟同步。具体输出脉冲或输出时钟频率的设定由 PPS 输出控制寄存器

(ETH_PTP_PPSCCLR) 的 PPSFRE0 位控制。只有 PPS0 通道支持连续输出模式。

在 PPS 单次输出模式时，系统时间若等于目标时间寄存器 0 或目标时间寄存器 1 设定的时间后，在 PPS0 或 PPS1 通道上可分别产生一个脉冲。只有当设定 PPS 输出控制寄存器 (ETH_PTP_PPSCCLR) 的 TT0SEL 位或 TT1SEL 位后，该功能才会被使能；当设定 PPS 输出控制寄存器 (ETH_PTP_PPSCCLR) 的 PPSFRE0 位或 PPSFRE1 位后，该功能的才会被激活。在脉冲产生之后，若想再次产生脉冲输出，需重新设定目标时间寄存器 0/1 和 PPS 输出控制寄存器的 PPSFRE0/1 位。

例如，用户根据需求，可将单次输出模式下的 PPS0 或 PPS1 的事件输出与 Timer2 联动，具体的实现方式请参考《通用定时器 (Timer2)》和《中断控制器 (INTC)》章节。这种与 Timer2 的联动主要有以下两个作用：

1. 当系统时间等于目标时间时，MAC 提供的触发中断会引起已知延迟，从而导致命令执行时间出现不确定性。在中断触发至命令执行期间可通过 Timer2 的计时功能获得一个准确的中断处理延迟时间。
2. 系统时间的偏差校准。PPS 输出一个固定时间的脉冲，通过 Timer2 的脉宽测量功能对该固定时间脉冲进行检测，并根据 Timer2 的测量结果反馈调整 PPS 的系统时间，实现对 PTP 内部系统时间的校准。

38.3.5 ETH_DMA 功能

38.3.5.1 ETH_DMA 功能概述

ETHMAC 控制器 DMA 具有独立的发送和接收引擎。发送引擎将数据从系统存储器传送到 Tx FIFO，而接收引擎将数据从 Rx FIFO 传送到系统存储器。DMA 可以在 CPU 完全不干预的情况下，通过描述符有效地将数据在 FIFO 和系统存储器之间传送。该 DMA 控制器可经过应用编程，在完成帧发送和接收操作时以及其它正常/错误条件下产生 CPU 中断。因此可以看出，ETHMAC 控制器 DMA 和 MCU 内核之间通过两种数据结构进行通信：

- 控制和状态寄存器
- 描述符列表和数据缓冲区

DMA 既可将 MAC 接收到的数据帧传送系统的数据缓冲区，也可以将系统数据缓冲区中的数据发送给 MAC，描述符中包含指向这些数据缓冲区的指针。DMA 共有两个描述符列表：一个用于接收的 Rx 描述符列表，一个用于发送的 Tx 描述符列表。Rx 描述符列表和 Tx 描述符列表的基地址分别通过描述符列表寄存器 (ETH_DMA_RXDLADR 和 ETH_DMA_TXDLADR) 设定。该描述符列表是一种前向链表（无论是隐式还是显式），最后一个描述符会指回第一个描述符以构成环形结构。描述符列表位于系统的物理存储空间。每个描述符最多可指向两个缓冲区，这样便能使用两个物理寻址的缓冲区替代存储器中两个连续的缓冲区。通过配置 Rx 描述符和 Tx 描述符 (RDES1[14]和 TDES0[20]) 中链接的第二个地址来完成描述符的显式链接。

数据缓冲区同样位于系统的物理存储空间，通常由整个帧或部分帧组成，但不会超过单个帧。数据缓冲区中仅包含数据，数据通信的状态保存在描述符中。检测到帧结束时，DMA 会跳到下一个帧缓冲区，数据链接可实现跨越多个数据缓冲区存储帧，可以使能或禁止数据链接。描述符有两种结构方式：环形结构与链接结构，其实现方式如下图所示。

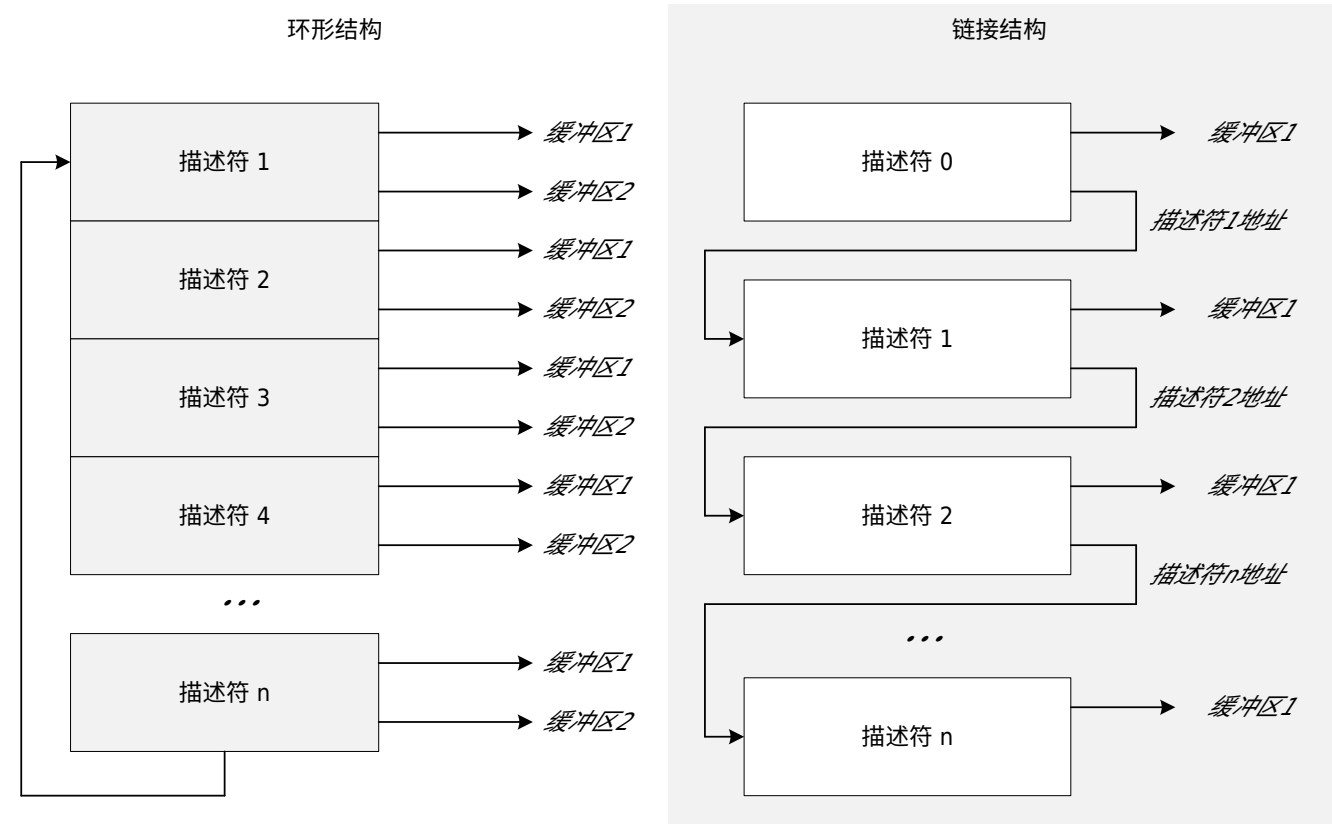


图 38-17 描述符结构

38.3.5.2 DMA 初始化

DMA 的初始化步骤如下：

- 步骤 1. 对 ETH_DMA_BUSMODR 执行写操作以设置总线访问参数；
- 步骤 2. 对 ETH_DMA_INTENAR 寄存器执行写操作以屏蔽不必要的中断源；
- 步骤 3. 创建描述符列表，对 ETH_DMA_RXDLADR 和 ETH_DMA_TXDLADR 寄存器执行写操作，为 DMA 提供各列表的起始地址；
- 步骤 4. 对 MAC 寄存器执行写操作设定所需的过滤选项；
- 步骤 5. 对 ETH_MAC_CONFIGR 寄存器执行写操作以配置和使能发送与接收工作模式；
- 步骤 6. 对 ETH_DMA_OPRMODR 寄存器执行写操作，将 STT 位和 STR 位置 1 以启动发送和接收；
- 步骤 7. 发送与接收引擎进入运行状态，并尝试从相应描述符列表中获取描述符，这两个引擎随后开始处理接收和发送操作。发送和接收处理过程彼此独立，可单独进行启动或停止。

总线突发访问

DMA 会尝试在 AHB Master 接口上执行固定长度的突发传送（当 ETH_DMA_BUSMODR 中的 FBST 位进行相应配置时）。突发的最大长度由 TPBL 或 RPBL 字段（ETH_DMA_BUSMODR [13:8]和[22:17]）指示和限制。对于要读取的 16 个字节，接收和发送描述符始终采用可能的最大突发大小进行访问。

仅当 Tx FIFO 中的空间足以容纳配置的突发或帧结束之前的字节数时（当帧短于配置的突发长度时），TxDMA 才会启动数据传送。TxDMA 会向 AHB Master 接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和 SINGLE 的最佳搭配来传送数据。否则（非固定长度突发）会使用 INCR（未定义长度）与 SINGLE 传送数据。

仅当 Rx FIFO 中有足够的数据用于配置的突发时，或者在 Rx FIFO 中检测到帧结束时（当帧短于配置的突发长度时），RxDMA 才会启动数据传送。RxDMA 会向 AHB Master 接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和 SINGLE 的最佳搭配来传送数据，如果在 AHB 接口上的固定突发结束前已到达帧结束，将执行空传送以完成固定长度的突发传送。否则（非固定长度突发）会使用 INCR（未定义长度）和 SINGLE 传送数据。

当 AHB 接口配置为地址对齐的节拍（ETH_DMA_BUSMODR.AAL=1）时，两个 DMA 引擎会确保 AHB 启动的第一次突发传送小于或者等于已配置 PBL 的大小。这样，后续的所有节拍都会从与已配置 PBL 对齐的地址开始。由于 AHB 接口多于 INCR16 的传送，DMA 只能对齐节拍最大为 16（PBL>16）的地址。

数据缓冲区对齐

发送和接收数据缓冲区在起始地址对齐方面没有任何限制。缓冲区的起始地址可与四个字节中的任意一个对齐。但是，DMA 始终在地址与总线宽度对齐时启动传输，并且在不需要字节通道上传输空数据。这通常发生在以太网帧的开始或结束传送期间。

● 缓冲区读操作示例：

如果发送缓冲区地址为 0x00000FF2，并且需要传送 15 个字节，则 TxDMA 将从地址 0x00000FF0 读取 5 个全字，但在将数据传送到 Tx FIFO 时，丢弃或忽略额外的字节（前两个字节）。同样地，还将忽略最后一次传送的最后 3 个字节。TxDMA 始终确保向 Tx FIFO 传送的是全 32 位数据，除非是帧结束。

● 缓冲区写操作示例：

如果接收缓冲区地址为 0x00000FF2，并且需要传送接收到的帧的 16 个字节，则 RxDMA 将从地址 0x00000FF0 开始写入 5 个全 32 位数据。但是，第一次传送的前 2 个字节与第三次传送的最后 2 个字节将包含空数据。

缓冲区大小计算

以太网控制器 DMA 不会更新发送和接收描述符中的缓冲区大小字段，只更新描述符的状态字段。应用必须用驱动程序计算缓冲区的大小。

TxDMA 会向 MAC 内核传送准确的字节数（由 TDES1 中的缓冲区大小字段指示）。如果将描述符标记为第一个描述符（将 TDES0 中的 TFS 位置 1），则 DMA 会将缓冲区的第一次传送标记为帧起始。如果将描述符标记为最后一个描述符（将 TDES0 中的 TLS 位置 1），则 DMA 会将数据缓冲区的最后一次传送标记为帧结束。

RxDMA 持续将数据传送至缓冲区，直到缓冲区已满或接收到帧结束为止。当描述符的 RFS 位置 1 时，如果未将描述符标记为最后一个描述符（RDES0 中的 RLS 位），则该描述符对应的缓冲区会填满，有效数据量由 RDES1 中的缓冲区大小指示；如果将描述符标记为最后一个描述符，则缓冲区不会填满，有效数据量将通过缓冲区大小字段减去数据缓冲指针偏移（RDES0 中的 FRAL 位）得到的结果表示，当数据缓冲区指针与数据总线宽度对齐时，偏移为零。



说明

即使当接收缓冲区的起始地址与系统数据总线宽度未对齐时，系统也应分配一个大小与系统总线宽度对齐的接收缓冲区。例如，如果系统分配一个起始地址为 0x1000、大小为 1024 字节 (1 KB) 的接收缓冲区，则软件可将接收描述符中的缓冲区起始地址编程为具有 0x1002 偏移。RxDMA 将帧写入该缓冲区，其中前两个单元 (0x1000 和 0x1001) 中为空数据。实际帧从单元 0x1002 开始写入。因此，尽管已将缓冲区大小编程为 1024 字节，但由于存在起始偏移地址，因此该缓冲区的实际有用空间为 1022 字节。

DMA 仲裁器

以太网控制器 DMA 内的仲裁器会在发送和接收通道对 AHB Master 接口进行的访问之间进行仲裁。可以使用两类仲裁：循环优先级和固定优先级。如果选择循环优先级仲裁（ETH_DMA_BUSMODR.DMAA=0），仲裁器会在发送和接收 DMA 同时请求访问时，按照 ETH_DMA_BUSMODR.TXPR 及 PTAT 位设置的优先方式和比率分配数据总线；如果选择固定优先仲裁（ETH_DMA_BUSMODR.DMAA=1），仲裁器会在发送和接收 DMA 同时请求访问时，按照 ETH_DMA_BUSMODR.TXPR 位设置的优先方式分配数据总线。

38.3.5.3 DMA 错误响应

对于由 DMA 通道发起的任何数据传送，如果从机给出错误响应，则相应 DMA 将停止所有操作并更新状态寄存器（ETH_DMA_DMASTR）中的错误位和致命总线错误位。此时，该 DMA 控制器只能在软复位或硬复位外设以及重新初始化 DMA 之后才能恢复操作。

38.3.5.4 DMA 发送配置

TxDMA 操作

在连续帧传送过程中，TxDMA 根据操作模式寄存器 ETH_DMA_OPRMODR 的 OSF 位的设定不同，对应的第二帧处理时序不同。当 OSF 位置 1 时，发送过程完成第一个帧的传送后，会立即轮询第二个帧的发送描述符列表，如果第二个帧有效，发送过程会在写入第一个帧的状态信息前发送第二帧。这种发送过程无需关闭第一个帧的状态描述符便可同时获取两个帧的方式称之为 OSF 模式，反之为非 OSF 模式。

TxDMA 引擎在 ETH_DMA_OPRMODR.OSF=0 下的操作顺序如下：

- 步骤 1.** 在数据缓冲区设置好相应的以太网帧数据后，用户配置发送描述符（TDES0-TDES3）并将 OWN 位（TDES0[31]）置 1；
- 步骤 2.** STT 位（ETH_DMA_OPRMODR 寄存器[13]）置 1 后，DMA 立即进入运行状态；
- 步骤 3.** 在运行状态下，DMA 为需要传送的帧轮询发送描述符列表。轮询启动后，DMA 会以连续的描述符环形顺序或链接顺序持续进行。如果 DMA 检测到标记为 CPU 所有的描述符（TDES0.OWN=0），或者发生错误条件，则会挂起传送并将发送缓冲区不可用位（ETH_DMA_DMASTR 寄存器 TUS 位）与正常中断汇总使能位（ETH_DMA_DMASTR 寄存器 AIS 位）置 1。发送引擎继续执行步骤 9；
- 步骤 4.** 如果获得的描述符标记为 DMA 所有（TDES0.OWN=1），则 DMA 将根据取得的描述符对发送数据缓冲区地址进行解码；
- 步骤 5.** DMA 从系统存储器中获取发送数据并传送这些数据；
- 步骤 6.** 如果以太网帧保存在多个描述符的数据缓冲区中，则 DMA 将关闭中间描述符并获取下一个描述符。重复执行步骤 3、4 和 5，直到完成以太网帧数据的传送；
- 步骤 7.** 完成帧传送后，如果已为帧使能 IEEE1588 时间戳（按发送状态中的指示），则时间戳值将写入发送描述符（TDES6 和 TDES7），随后状态信息将写入该发送描述符的各状态位（TDES0）。此步骤中会清零 OWN 位，此时该描述符变为由 CPU 所有；

步骤 8. 帧发送完成后，在其最后一个描述符位 (TDES0.TLS) 置 1 时，DMA 动作状态寄存器的 TIS 位将置 1。随后 DMA 引擎返回步骤 3；

步骤 9. 在挂起状态下，DMA 会在接收到发送轮询要求时尝试重新获取描述符（返回步骤 3），并将下溢中断状态位清零。

非 OSF 模式下，TxDMA 的动作流程图如下图所示。

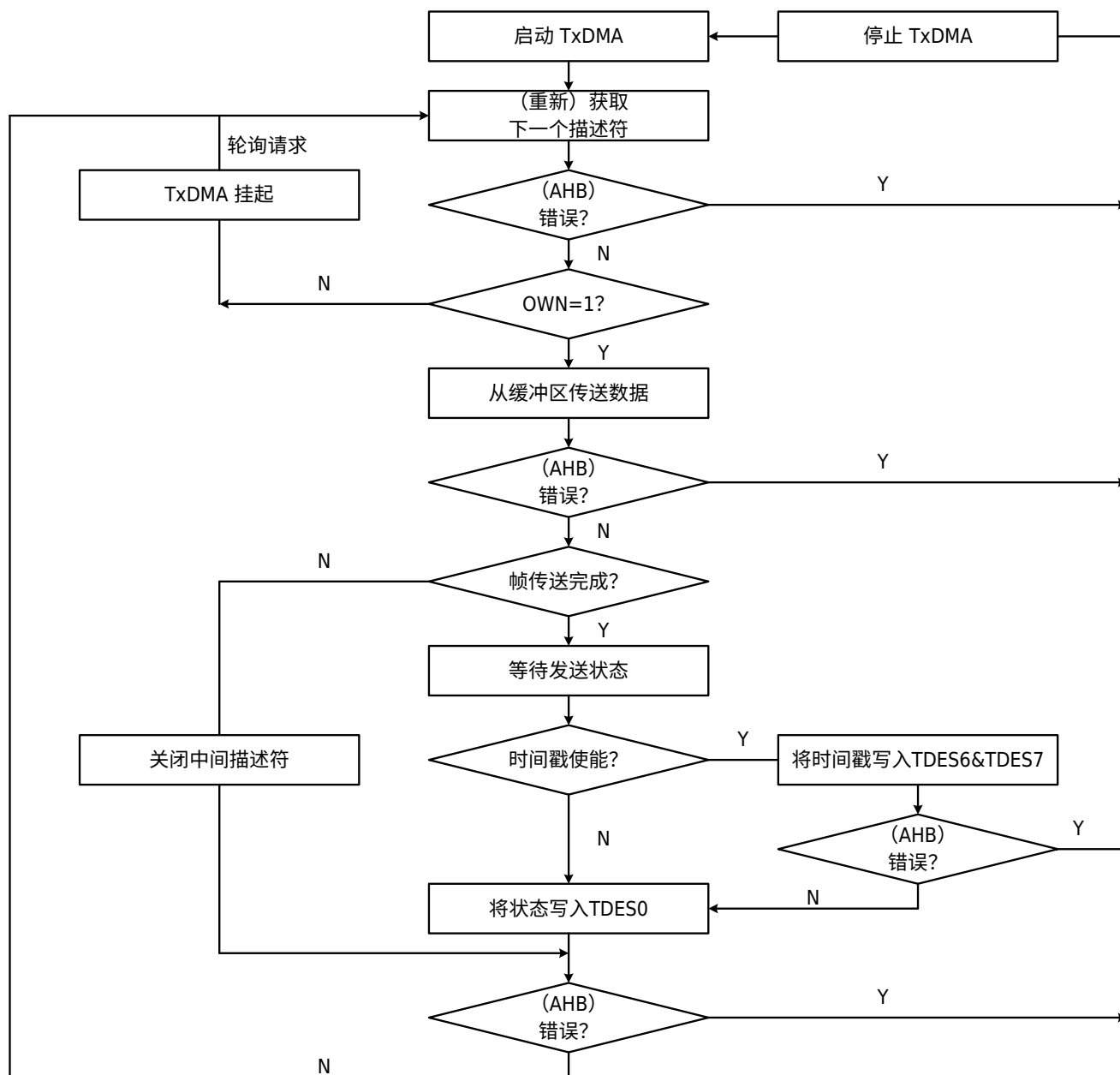


图 38-18 TxDMA 动作流程（非 OSF 模式）

TxDMA 引擎在 ETH_DMA_OPRMODR.OSF=1 下的操作顺序如下：

步骤 1. DMA 的操作过程如非 OSF 模式下 TxDMA 的步骤 1~6 中所述；

步骤 2. 无需关闭前一帧的最后一个描述符，DMA 便可获取下一个描述符；

步骤 3. 如果 DMA 拥有所需描述符，便会对该描述符中的发送缓冲区地址进行解码。如果 DMA 未拥有描述符，则会进入挂起模式并跳到步骤 7；

步骤 4. DMA 从系统存储器中取得发送帧并发送该帧，直到帧数据发送结束，如果该帧拆分到多个描述符中，则会同时关闭中间描述符；

- 步骤 5.** DMA 等待前一个帧的发送状态和时间戳。当状态可用时，如果捕获到时间戳（由状态位指示），DMA 会将这些时间戳写入 TDES6 和 TDES7。之后，OWN 位清零，DMA 将状态写入相应 TDES0，进而关闭描述符。如果没有为前一个帧使能时间戳，DMA 不会更改 TDES6 和 TDES7 的内容；
- 步骤 6.** 如果已使能时间戳，发送中断将置 1，DMA 获取下一个描述符，然后继续执行步骤 3（状态正常时）。如果上一个发送状态显示下溢错误，则 DMA 会进入挂起模式（步骤 7）；
- 步骤 7.** 在挂起模式下，如果 DMA 接收到挂起状态和时间戳，则会将时间戳（如果已为当前帧使能时间戳）写入 TDES6 和 TDES7，随后将状态写入相应的 TDES0。之后，将相关中断置 1 并返回挂起模式；
- 步骤 8.** 只有在接收到发送轮询要求（ETH_DMA_TXPOLLR 寄存器）后，DMA 才会退出挂起模式并进入运行状态（转到步骤 1 或步骤 2，具体取决于挂起状态）。

OSF 模式下，TxDMA 的动作流程图如下图所示。

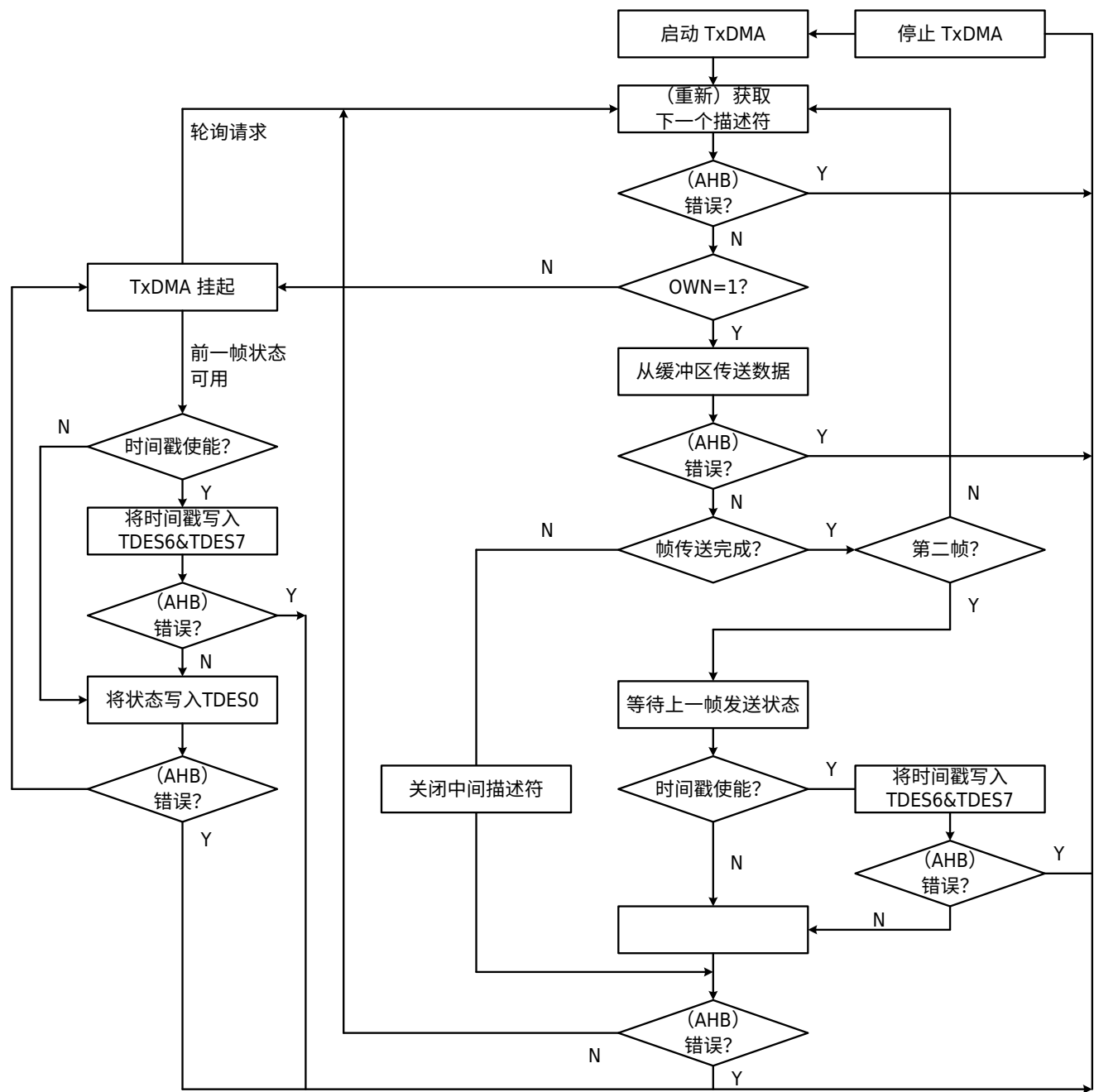


图 38-19 TxDMA 动作流程 (OSF 模式)

发送帧处理

系统数据缓冲区须包含完整的以太网帧，不包括报头、PAD 和 FCS 字段，DA 字段、SA 字段和类型/长度字段须包含在有效数据中。如果发送描述符指示 MAC 内核必须禁止插入 CRC 或填充字节，则缓冲区必须具有包括 CRC 字节的完整以太网帧（不包括报头）。帧可以采用数据链接形式，也可以跨越多个缓冲区。帧必须由第一个描述符位（TDES0.TFS）和最后一个描述符位（TDES0.TLS）定界。传送启动时，必须将第一个描述符的 TDES0.TFS 置 1，之后帧数据便从存储器缓冲区传送到 Tx FIFO。同时，如果当前帧的 TDES0.TLS 为 0，发送过程将尝试获取下一个描述符。TDES0.TLS 为 0，则指示中间缓冲区，TDES0.TLS 为 1，则指示帧的最后一个缓冲区。当帧的最后一个缓冲区完成传送后，DMA 会将最终状态信息写回描述符的（TDES0）相关状态位。此时，如果 TDES0.IOC 位置 1，则发送状态寄存器的 TIS 位将置 1，并获取下一个描述符，然后重复执行该过程。

根据 ETH_DMA_OPRMODR 寄存器 TSF 位的设定，实际的帧传送过程在 Tx FIFO 达到可编程的发送阈值时（ETH_DMA_OPRMODR 寄存器 TTC 位），或者 Tx FIFO 中包含完整的帧时启动传送过程。DMA 完成帧的传送后会释放描述符（清零 TDES0.OWN 位）。

发送轮询挂起

可通过以下任意条件暂停发送轮询：

- DMA 检测到 CPU 所有的描述符（TDES0[31]=0），并且 ETH_DMA_DMASTR 寄存器的发送缓冲区不可用标志 TUS 置 1。如要恢复，驱动程序必须将描述符的所有权交给 DMA，然后发出轮询查询要求命令。
- 检测到由下溢导致的传送错误时，将中止帧的传送。相应的发送描述符（TDES0）位将置 1。

如果 DMA 由于第一个条件而进入挂起状态，则发送状态寄存器的正常中断汇总位与发送缓冲区不可用位（ETH_DMA_DMASTR 寄存器位 NIS 和 TUS 位）均置 1；如果发生第二个条件，异常中断汇总位与发送下溢位（ETH_DMA_DMASTR 寄存器位 AIS 位和 UNS 位）都将置 1，并且信息将写入发送描述符，从而导致挂起。两种情况下，发送列表中的位置都会保留。保留的位置是 DMA 关闭的最后一个描述符后面的描述符位置。纠正挂起原因后，驱动程序必须明确发出发送轮询要求命令。

常规型 Tx 描述符

常规型 Tx 描述符结构由四个 32 位字组成，如下表所示，分别定义为 TDES0、TDES1、TDES2 和 TDES3。

表 38-8 常规型 Tx 描述符

描述符结构	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDES0	OWN	IOC	TLS	TFS	DCRC	DPAD	TTSE	CRCR	CIC		TER	TSAC	VLANC		TTSS	IHE	ETSUM	JTE	FFF	TPCE	LOCE	NCE	TLCE	ECE	VLF	COC				EDE	UDE	DEE
TDES1	SAIRC		TBS2												Reserved		TBS1															
TDES2	TBAP1																															
TDES3	TBAP2																															

下面分别对 TDES0~TDES3 的具体位做说明。

● TDES0 的各位功能如下：

位/位域	标记	位名	功能描述
31	OWN	所有关系位	► 0b0：该描述符为 CPU 所有 ► 0b1：该描述符为 DMA 所有 DMA 在完成帧发送时或在描述符分配的缓冲区全部读取完成后将该位清零。 属于同一个帧的所有后续描述符置 1 后，也应该将帧的第一个描述符的 OWN 位置 1。
30	IOC	完成时中断	该位置 1 时，当前帧发送完毕后，动作状态寄存器的 TIS 位置 1  说明 该位仅在最后一个描述符位 (TDES0.TLS) 置 1 的情况下有效。
29	TLS	最后一个描述符	该位指示此描述符包含帧的最后一个缓冲区
28	TFS	第一个描述符	该位指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0，则第二个缓冲区将包含帧的帧头；如果第二个缓冲区的大小为 0，则下一个描述符将包含帧的帧头
27	DCRC	无效 CRC	该位置 1 时，MAC 不会将循环冗余校验 (CRC) 附加到所发送帧的末尾  说明 该位仅在第一个描述符位 (TDES0.TFS) 置 1 的情况下有效。
26	DPAD	无效 PAD	► 0b0：DMA 会自动为不足 64 字节的帧添加补位项和 CRC，不管该描述符的 DCRC 位的设定 ► 0b1：MAC 不会自动为不足 64 字节的帧添加补位项  说明 该位仅在第一个描述符位 (TDES0.TFS) 置 1 的情况下有效。
25	TTSE	时间戳使能	该位置 1 时，将针对描述符所描述的发送帧激活 IEEE1588 硬件时间戳功能  说明 该位仅在第一个描述符位 (TDES0.TFS) 置 1 的情况下有效。
24	CRCR	CRC 替换控制	该位置 1 时，MAC 将计算好的 CRC 值替换即将发送的发送帧的 FCS 字段  说明 只有当该描述符的 TDES0.DCRC 位置 1 时，该位有效。
23:22	CIC	Checksum 插入控制	这些位控制校验和的计算与插入，如下所示： ► 0b00：禁止插入校验和 ► 0b01：仅使能 IP 报头校验和的计算与插入 ► 0b10：使能 IP 报头校验和以及 TCP/UDP/ICMP 校验和的计算与插入，但不会在硬件中计算伪报头校验和 ► 0b11：使能 IP 报头校验和以及 TCP/UDP/ICMP 校验和的计算与插入，并在硬件中计算伪报头校验和

位/位域	标记	位名	功能描述
23:22	CIC	Checksum 插入控制	 说明 该位仅在最后一个描述符位 (TDES0.TLS) 置 1 的情况下有效。
21	TER	环发送结束	该位指示描述符列表已到达其最后一个描述符, DMA 会返回描述符列表的首地址, 形成一个描述符环
20	TSAC	第二个地址链接	该位置 1 时, 描述符中的第二个地址是下一个描述符地址, 而非第二个缓冲区地址, 此时该描述符的 TBS2 位为无效  说明 TER 位优先级高于 TSAC 位。
19:18	VLANC	VLAN 插入控制	▶ 0b00: 对 VLAN 帧不做处理 ▶ 0b01: 在发送前将 VLAN 帧的标记和类型字段删除 ▶ 0b10: 将 VLAN 标记发送控制寄存器 (ETH_MAC_VTACTLR) 中的 VLAN 标记值插入到发送帧中 ▶ 0b11: 用 VLAN 标记发送控制寄存器 (ETH_MAC_VTACTLR) 中的 VLAN 标记值替换原发送帧中的 VLAN 标识符
17	TTSS	发送时间戳状态	该位指示已为当前发送帧捕获到时间戳, 置 1 时 TDES6 和 TDES7 将保存发送帧捕获到的时间戳值  说明 该位仅在最后一个描述符位 (TDES0.TLS) 置 1 的情况下有效。
16	IHE	IP Header 错误	该位置 1 时指示 MAC 发送器在 IP 数据报头中检测到错误  说明 当 COE 引擎检测出一个 IP Header 错误时, 它仍然会将计算好的 Checksum 值插入到 IPv4 Header Checksum 字段。
15	ETSUM	发送错误汇总	该描述符的 Bit[16]、Bit[14]、Bit[13]、Bit[12]、Bit[11]、Bit[10]、Bit[9]、Bit[8]、Bit[2]、Bit[1]位中有一个被置位, 该位也随之被置位
14	JTE	Jabber 超时错误	该位置 1 时, 指示 MAC 发送器经历了 Jabber 超时  说明 只有在 MAC 配置寄存器 (ETH_MAC_CONFIGR) 的 MJD 位未置 1 时, 该位才会置 1。
13	FFF	帧刷新	该位指示 DMA 已依照 CPU 发出的软件刷新命令刷新帧
12	TPCE	有效负载错误	该位指示 COE 引擎在 TCP、UDP 或 ICMP 报有效负载中检测到错误, 且不会更新原帧中的 Checksum 字段
11	LOCE	载波丢失错误	该位指示帧发送期间丢失载波, 即, 帧发送期间有一个或多个发送时钟周期的 MII_CRS 信号无效  说明 该位仅对在 MAC 处于半双工模式时实现无冲突发送的帧有效。
10	NCE	无载波错误	该位指示在发送期间未由 PHY 触发载波侦听信号

位/位域	标记	位名	功能描述
9	TLCE	延迟冲突错误	该位指示帧发送过程因冲突窗口（MII 模式下为 64 个字节时间，含报头）后出现冲突而中止  说明 如果该描述符的 UDE 错误位置 1，则该位无效。
8	ECE	过度冲突错误	该位指示尝试发送当前帧时因出现 16 个连续冲突而中止发送  说明 如果 MAC 配置寄存器（ETH_MAC_CONFIGR）中的 DRTY（禁止重试）位置 1，则此位会在出现首个冲突后置 1 并且帧发送过程将中止。
7	VLF	VLAN 帧	该位指示所发送的帧为 VLAN 帧
6:3	COC	冲突计数	该位指示发送帧之前出现的冲突个数  说明 过度冲突位（TDES0.ECE）置 1 时，该计数无效。
2	EDE	过度延迟错误	该位指示因出现超过 24288 个位时间的过度延迟而中止发送  说明 该位在 MAC 配置寄存器（ETH_MAC_CONFIGR）中的延迟检查（DC）位置 1 时有效。
1	UDE	下溢错误	该位指示 MAC 因发送缓冲区的数据未及时到达而中止了帧发送下溢错误表示 DMA 在帧发送期间遇到发送缓冲区为空的情况
0	DEE	延迟错误	该位指示 MAC 在发送前因存在载波而延迟  说明 该位仅在半双工模式下有效。

● TDES1 的各位功能如下：

位/位域	标记	位名	功能描述
31:29	SAIRC	SA 插入替换控制	对发送帧的 SA 字段进行如下处理： ▶ 0b001：将 MAC 地址寄存器 0 中的地址值作为 SA 地址插入到发送帧中 ▶ 0b101：将 MAC 地址寄存器 1 中的地址值作为 SA 地址插入到发送帧中 ▶ 0b010：将 MAC 地址寄存器 0 中的地址值作为 SA 地址替换发送帧中的 SA 字段 ▶ 0b110：将 MAC 地址寄存器 1 中的地址值作为 SA 地址替换发送帧中的 SA 字段 ▶ 其他：禁止设置
28:16	TBS2	缓冲区 2 大小	该位以字节为单位指示第二个数据缓冲区的大小  说明 该位在 TDES0.TSAC 位置 1 时无效。
15:13	Reserved	保留	-

位/位域	标记	位名	功能描述
12:0	TBS1	缓冲区 1 大小	该位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0，DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于该描述符的 TSAC 的设置

● TDES2 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	TBAP1	发送缓冲区 1 地址	该位向 DMA 指示数据在存储器中的位置，当软件为 DMA 提供此描述符（TDES0 中的 OWN 位置 1）时，这些位将指示缓冲区 1 的物理地址

● TDES3 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	TBAP2	发送缓冲区 2 地址/下一个描述符地址	该位向 DMA 指示数据在存储器中的位置，当软件为 DMA 提供此描述符（TDES0 中的 OWN 位置 1）并且使用描述符环结构时，这些位将指示缓冲区 2 的物理地址；如果 TDES0.TSAC 位置 1，则该地址包含下一个描述符所在物理寄存器的指针
			 说明 只有在 TDES0.TSAC 位置 1 时，缓冲区地址指针才必须与总线宽度相符。

增强型 Tx 描述符

当时间戳功能有效（ETH_PTP_TSPCTLR.TSPEN=1）时或者 Checksum Offload 功能有效（ETH_MAC_CONFIGR.IPCO=1）时，应使用增强型 Tx 描述符。

增强型 Tx 描述符结构由 8 个 32 位字组成，如下表所示，分别定义为 TDES0、TDES1、TDES2、TDES3、TDES4、TDES5、TDES6 和 TDES7。其中 TDES0~TDES3 的功能与常规型相同。

表 38-13 增强型 Tx 描述符

描述符结构	Bit Position																																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TDES0	OWN	IOC	TLS	TFS	DCRC	DPAD	TTSE	CRCR		CIC		TER	TSAC		VLANC		TTSS	IHE	ETSUM	JTE	FFF	TPCE	LOCE	NCE	TLCE	ECE	VLF	COC			EDE	UDE	DEE
TDES1	SAIRC		TBS2													Reserved		TBS1															
TDES2																	TBAP1																
TDES3																	TBAP2																
TDES4	Reserved																																
TDES5	Reserved																																

描述符结构	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDES6																	TTSL															
TDES7																	TTSH															

下面分别对 TDES0~TDES7 的具体位做说明。

- TDES0 的各位功能：与常规型描述符 TDES0 的功能相同。
- TDES1 的各位功能：与常规型描述符 TDES1 的功能相同。
- TDES2 的各位功能：与常规型描述符 TDES2 的功能相同。
- TDES3 的各位功能：与常规型描述符 TDES3 的功能相同。
- TDES4、TDES5：Reserved
- TDES6 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	TTSL	时间戳低位	DMA 将相应发送帧所捕获的时间戳的低 32 位更新进该字段  说明 在描述符中的最后一个描述符位 (TDES0.TLS) 置 1 且发送时间戳状态位 (TDES0.TTSS) 置 1 时，该字段才包含时间戳。

- TDES7 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	TTSH	时间戳高位	DMA 将相应发送帧所捕获的时间戳的高 32 位更新进该字段  说明 在描述符中的最后一个描述符位 (TDES0.TLS) 置 1 且发送时间戳状态位 (TDES0.TTSS) 置 1 时，该字段才包含时间戳。

38.3.5.5 DMA 接收配置

RxDMA 操作

RxDMA 引擎的操作顺序如下：

- 步骤 1. 用户设置接收描述符 (RDES0-RDES3) 并将 OWN 位 (RDES0[31]) 置 1；
- 步骤 2. 操作模式寄存器 ETH_DMA_OPRMODR.STR 位置 1 后，DMA 进入运行状态。在运行状态下，DMA 轮询接收描述符列表，尝试获取空闲描述符。如果获得的描述符不空闲（由 CPU 所拥有），则 DMA 进入挂起状态并跳转到步骤 9；
- 步骤 3. DMA 将所获取描述符中的接收数据缓冲区地址解码；
- 步骤 4. 处理传入帧并将其放入所获取描述符的数据缓冲区；
- 步骤 5. 当缓冲区已满或帧传输完成时，接收引擎将获取下一个描述符；
- 步骤 6. 如果当前的帧传输完成，DMA 将继续执行第 7 步。如果 DMA 未拥有下一个接收描述符且帧传输尚未完成（尚未传输 EOF 字段），则 DMA 会将 RDES0 中的描述符错误位 DPE 置 1（除非禁止刷新）。DMA 关闭当前描述符（将 OWN 位清零）并通过将最后一个描述符位 (RDES0.RLS) 清零来将其标记为中间描述符（如果禁止刷新，则将其标记为最后一个描述符），随后继续执行第 8 步；如果 DMA 已拥有下一个描述符，但当前的帧传输尚未完成，则 DMA 将关闭当前描述符作为中间值并返回第 4 步；

- 步骤 7.** 如果已使能 IEEE1588 时间戳功能，DMA 会将时间戳（如果可用）写入当前描述符的 RDES6 和 RDES7。DMA 随后获取所接收帧的状态并将该状态字写入当前描述符的 RDES0，同时 OWN 位清零且最后一个描述符位置 1；
- 步骤 8.** 接收引擎检查最新描述符的 OWN 位。如果 CPU 拥有该描述符（OWN 位为 0），则接收缓冲区不可用位（ETH_DMA_DMASTR.RUS）置 1 且 DMA 接收引擎进入挂起状态（第 9 步）；如果 DMA 拥有该描述符，引擎将返回第 4 步并等待下一个帧；
- 步骤 9.** 在接收引擎进入挂起状态前，将从 Rx FIFO 中刷新部分帧（可使用 ETH_DMA_OPRMODR 寄存器的 DFRF 位控制刷新）；
- 步骤 10.** 当收到接收轮询要求命令或者可以从 Rx FIFO 获得下一个帧的起点时，RxDMA 将退出挂起状态。引擎继续执行第 2 步并重新获取下一个描述符。

RxDMA 的动作流程图如下图所示。

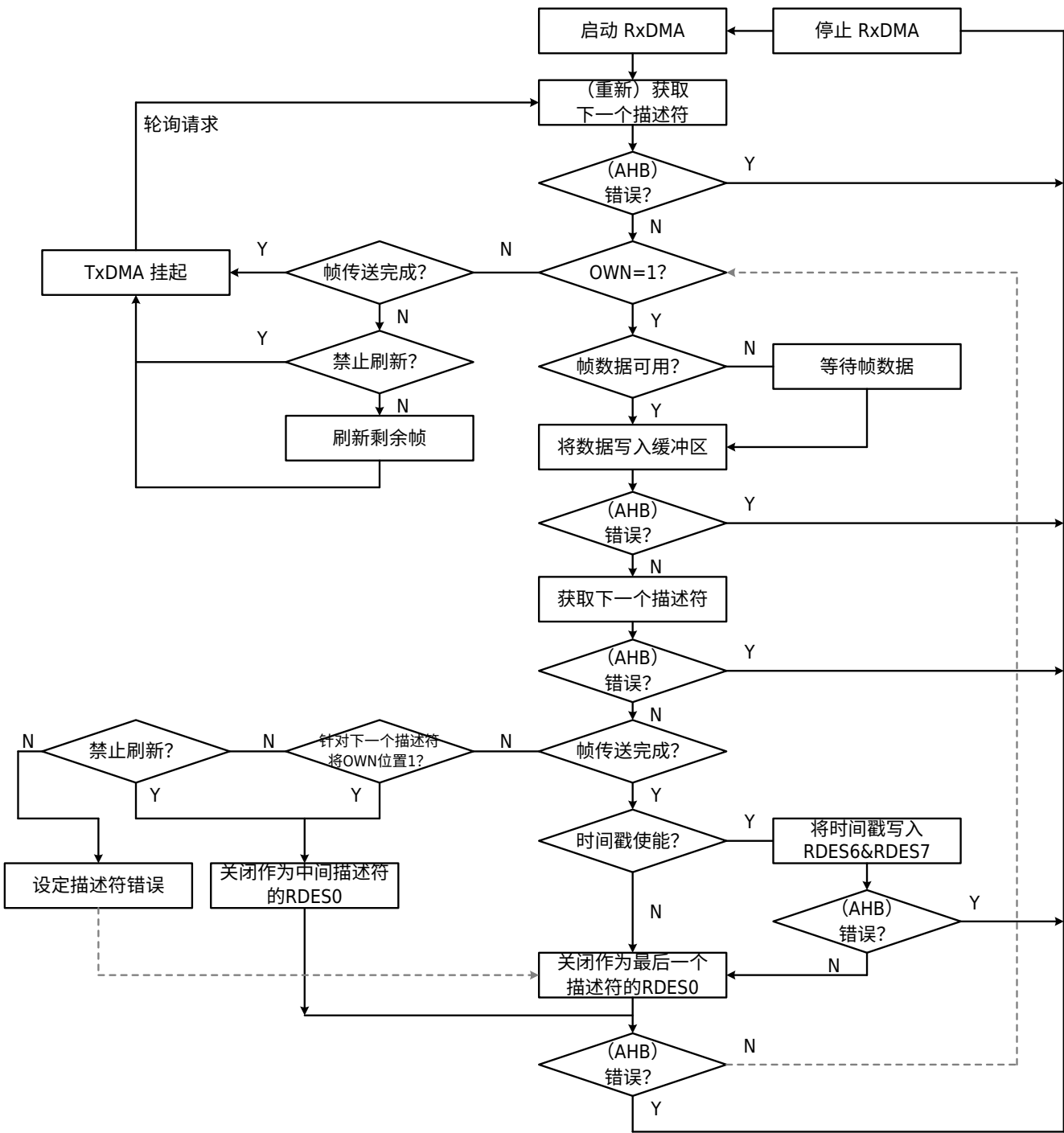


图 38-20 RxDMA 动作流程

RxDMA 的接收状态直到完成时间戳回写并准备将状态回写到描述符时才会确认。如果使能时间戳功能，则当无法获得帧的有效时间戳值时（例如，在写入时间戳前 Rx FIFO 已满），DMA 会向 RDES6 和 RDES7 的所有位写入 1；如果未使能时间戳功能，RDES6 和 RDES7 保持不变。

接收帧获取

接收引擎始终会尝试为即将传入的帧获取一个额外的描述符。只要满足以下任一条件，即尝试获取描述符：

- DMA 进入运行状态后，接收启动/停止位 (ETH_DMA_OPRMODR.STR) 立即置 1
- 在当前传输的帧结束前，当前描述符的数据缓冲区已满
- MAC 控制器已完成数据帧接收，但当前的接收描述符尚未关闭
- 接收过程因描述符由 CPU 所拥有 (RDES0.OWM=0) 而挂起，并且接收到新帧
- 已发出接收轮询要求命令

接收帧处理

当帧通过地址过滤且其大小大于或等于为 Rx FIFO 所设置的可配置阈值字节数时，或者在存储转发模式下将整个帧写入 Rx FIFO 时，MAC 才会将接收的帧传输到系统存储器；当帧未通过地址过滤时，将丢弃该帧（除非接收所有位 ETH_MAC_FLTCTLR.RA 置 1）。不足 64 字节的帧由于会发生冲突或过早结束，因而可从 Rx FIFO 中清除。接收 64（可配置阈值）个字节后，DMA 开始将帧数据传输到当前描述符所指定的接收缓冲区中。DMA AHB 接口准备好接收数据传输后，如果 DMA 当前未从存储器获取发送数据，则将第一个描述符位 (RDES0.RFS) 置 1，以分隔该帧。数据缓冲区已满或者帧的末段已传输到接收缓冲区时，OWN (RDES0) 位将复位为 0，此时释放描述符。如果帧包含在单个描述符中，则最后一个描述符位 (RDES0.RLS) 和第一个描述符位 (RDES0.RFS) 均置 1。

DMA 获取下一个帧的描述符时，将最后一个描述符位 (FDES0.RLS) 置 1，并释放前一个帧描述符中的状态位，然后将接收中断位 (ETH_DMA_DMASTR 寄存器 RIS) 置 1。该过程将重复执行，直至 DMA 遇到被标记为由 CPU 所有的描述符。这种情况时，接收过程将 ETH_DMA_DMASTR 寄存器中接收缓冲区不可用位 RUS 置 1，随后进入挂起状态，但其在接收列表中的位置仍然保留。

接收过程挂起

如果在接收过程处于挂起状态时有新的接收帧到达，则 DMA 将重新获取系统存储器中的当前描述符。如果该描述符现在由 DMA 所拥有，接收过程将重新进入运行状态并开始接收帧；如果该描述符仍由主机所拥有，则默认情况下，DMA 将丢弃 Rx FIFO 顶部的当前帧并将丢失帧计数器递增。如果 Rx FIFO 中存储了多个帧，将重复执行该过程。

若将 DMA 工作模式寄存器 (ETH_DMA_OPRMODR) 的 DFRF 位置 1 后，可避免丢弃或刷新 Rx FIFO 顶部的帧。在这种情况下，接收过程将动作状态寄存器 (ETH_DMA_DMASTR) 的接收缓冲区不可用位 RUS 置 1 并返回到挂起状态。

常规型 Rx 描述符

常规型 Rx 描述符结构由四个 32 位字组成，如下表所示，分别定义为 RDES0、RDES1、RDES2 和 RDES3。

表 38-16 常规型 Rx 描述符

描述符结构	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDES0	OWN	DAF	FRAL														ERSUM	DPE	SAF	LEE	OVE	VLAT	RFS	RLS	IPE/TSPA/GF	RLCE	FRAT	WTE	REE	DBE	CRE	DAS/ESA
RDES1	DIC	Reserved	RBS2														RER	RSAC	Reserved	RBS1												
RDES2																	RBAP1															
RDES3																	RBAP2															

下面分别对 RDES0~RDES3 的具体位做说明。

● RDES0 的各位功能如下：

位/位域	标记	位名	功能描述
31	OWN	所有关系位	► 0b0：该描述符为 CPU 所有 ► 0b1：该描述符为 DMA 所有 DMA 在帧接收完成或此描述符的关联缓冲区已满时将该位清零。
30	DAF	目标地址过滤失败	该位指示帧未能通过 MAC 内核中的 DA 过滤
29:16	FRAL	帧长统计	该位指示传输到主机存储器中的接收帧字节长度（含 CRC） 只有在最后一个描述符位（RDES0.RLS）置 1，并且描述符错误位（RDES0.DPE）和上溢错误位（RDES0.OVE）复位时，该字段才有效。 当最后一个描述符位和错误汇总位均未置 1 时，该字段指示为当前帧传输的累计字节数。
15	ERSUM	接收错误汇总	该描述符的 Bit[14]、Bit[11]、Bit[7]、Bit[6]、Bit[4]、Bit[3]、Bit[1]位及 REDS4 描述符的 Bit[4]、Bit[3]位中有一个被置位，该位也随之被置位
14	DPE	描述符错误	该位指示某个帧因超过当前描述符缓冲区的大小而被截断以及 DMA 未拥有下一个描述符  说明 该位只有在最后一个描述符（RDES0.RLS）置 1 时才有效。
13	SAF	源地址过滤失败	该位指示帧的 SA 字段未能通过 MAC 内核中的 SA 过滤

位/位域	标记	位名	功能描述
12	LEE	长度错误	该位指示接收帧的实际长度与长度/类型字段的值不符  说明 ▶ 该位仅在帧类型位 (RDES0.FRAT) 复位后有效。 ▶ 在 CRC 错误位 (RDES0.CRE) 有效时, 该位无效。
11	OVE	上溢错误	该位指示接收帧因缓冲区上溢而损坏
10	VLAT	VLAN 标识符	该位指示描述符所指向的帧是由 MAC 内核标记的 VLAN 帧
9	RFS	第一个描述符	该位指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0, 则第二个缓冲区将包含帧的帧头; 如果第二个缓冲区的大小为 0, 则下一个描述符将包含帧的帧头
8	RLS	最后一个描述符	该位指示此描述符指向的缓冲区为帧的最后一个缓冲区
7	IPE/TSPA/GF	COE 错误/时间戳/巨帧	当 Checksum Offload 功能有效时: ▶ 该位指示 IPv4 或 IPv6 报头中存在错误 导致此错误的原因可能是: 1. 以太网类型字段与 IP 报头版本字段值不一致 2. IPv4 中报头的校验和不匹配 3. 以太网帧缺少所需的 IP 报头字节数 当时间戳功能有效时: ▶ 该位指示有一个时间戳快照被记录在了 RDES6 和 RDES7 中  说明 该位只在最后一个描述符 (RDES0.RLS) 置 1 时有效。 以上两个均无效时: ▶ 该位指示 MAC 接收到一个巨帧
6	RLCE	延迟冲突错误	该位指示在以半双工模式接收帧时发生了延迟冲突
5	FRAT	帧类型	▶ 0b0: MAC 接收到以太网帧 ▶ 0b1: MAC 接收到 PTP 帧  说明 当接收到一个矮帧时, 该位无效。
4	WTE	看门狗错误	该位指示接收看门狗计时器在接收当前帧时超时, 且当前帧在看门狗超时后被截断
3	REE	接收错误	该位指示在帧接收期间, PHY 发出 RX_DV 信号时, 因为发出了 RX_ER 信号
2	DBE	Dribble 位错误	该位指示接收的帧具有非整数倍数的字节 (奇数半字节)  说明 该位仅在 MII 模式下有效。
1	CRE	CRC 错误	该位指示接收的帧发生循环冗余校验 (CRC) 错误  说明 该位只在最后一个描述符 (RDES0.RLS) 置 1 有效。

位/位域	标记	位名	功能描述
0	DAS/ESA	地址过滤成功/状态位扩展	在 COE 功能和 PTP 功能无效时: ▶ 0b0: 接收帧通过了 MAC 地址寄存器 0 的 DA 地址过滤 ▶ 0b1: 接收帧通过了 MAC 地址寄存器 1~5 的 DA 地址过滤  说明 当该描述符的 DAF 位置位时, 该位无效。 在 COE 功能或 PTP 功能有效时: ▶ 该位指示 RDES4 描述符有效  说明 该位只在最后一个描述符 (RDES0.RLS) 置 1 时有效。

● RDES1 的各位功能如下:

位/位域	标记	位名	功能描述
31	DIC	完成时禁止中断	该位置 1 时, 在接收帧在由该描述符所指示的缓冲区中结束时禁止引发主机的中断, 同时也阻止动作状态寄存器 (ETH_DMA_DMASTR) 的 RIS 位置 1  说明 该位仅在最后一个描述符位 (RDES0.RLS) 置 1 的情况下有效。
30:29	Reserved	保留	-
28:16	RBS2	缓冲区 2 大小	该位以字节为单位指示第二个数据缓冲区的大小  说明 该位在该描述符的 RSAC 位置 1 时无效。
15	RER	环接收结束	该位指示描述符列表已到达其最后一个描述符, DMA 会返回描述符列表的首地址, 形成一个描述符环
14	RSAC	第二个地址链接	该位置 1 时, 描述符中的第二个地址是下一个描述符地址, 而非第二个缓冲区地址, 此时该描述符的 RBS2 位为无效  说明 RER 位优先级高于 RSAC 位。
13	Reserved	保留	-
12:0	RBS1	缓冲区 1 大小	该位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0, DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符, 具体取决于该描述符的 RSAC 的设置

● RDES2 的各位功能如下:

位/位域	标记	位名	功能描述
31:0	RBAP1	接收缓冲区 1 地址	该位向 DMA 指示数据在存储器中的位置, 当软件为 DMA 提供此描述符 (RDES0 中的 OWN 位置 1) 时, 这些位将指示缓冲区 1 的物理地址

● RDES3 的各位功能如下:

位/位域	标记	位名	功能描述
31:0	RBAP2	接收缓冲区 2 地址/下一个描述符地址	该位向 DMA 指示数据在存储器中的位置，当软件为 DMA 提供此描述符（RDES0 中的 OWN 位置 1）并且使用描述符环结构时，这些位将指示缓冲区 2 的物理地址；如果 RDES1.RSAC 位置 1，则该地址包含下一个描述符所在物理寄存器的指针  说明 只有在 RDES1.RSAC 位置 1 时，缓冲区地址指针才必须与总线宽度相符。

增强型 Rx 描述符

当时间戳功能有效（ETH_PTP_TSPCTLR.TSPEN=1）时或 Checksum Offload 功能有效（ETH_MAC_CONFIGR.IPCO=1）时，应使用增强型 Rx 描述符。

增强型 Tx 描述符结构由 8 个 32 位字组成，如下表所示，分别定义为 RDES0、RDES1、RDES2、RDES3、RDES4、RDES5、RDES6 和 RDES7。其中 RDES0~RDES3 的功能与常规型相同。

表 38-21 增强型 Rx 描述符

描述符结构	Bit Position																																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RDES0	OWN	DAF	FRAL														ERSUM	DPE	SAF	LEE	OVE	VLAT	RFS	RLS	IPE/TSPA/GF	RLCE	FRAT	WTE	REE	DBE	CRE	DAS/ESA	
RDES1	DIC	Reserved	RBS2														RER	RSAC	Reserved	RBS1													
RDES2	RBAP1																RBAP2																
RDES3	RBAP1																RBAP2																
RDES4	Reserved				L4FMS		L3FMS		Reserved								TSPD	PTPV	PTPFT	MTP				IPv6DR	IPv4DR	IPCB	IPPE	IPHE	IPPT				
RDES5	Reserved																																
RDES6	RTSL																RTSH																
RDES7	RTSL																RTSH																

下面分别对 RDES0~RDES7 的具体位做说明。

- RDES0 的各位功能：与常规型描述符 RDES0 的功能相同。
- RDES1 的各位功能：与常规型描述符 RDES1 的功能相同。
- RDES2 的各位功能：与常规型描述符 RDES2 的功能相同。
- RDES3 的各位功能：与常规型描述符 RDES3 的功能相同。

● RDES4 的各位功能如下：

位/位域	标记	位名	功能描述
31:26	Reserved	保留	-
25	L4FMS	L4 过滤成功	该位指示接收帧通过了 L4 端口过滤
24	L3FMS	L3 过滤成功	该位指示接收帧通过了 L3 地址过滤
23:15	Reserved	保留	-
14	TSPD	时间戳丢弃	该位指示由于 Rx FIFO 溢出，虽进行了时间戳快照但是丢弃了时间戳
13	PTPV	PTP 版本	该位指示接收到的 PTP 帧的版本类型 ▶ 0b0：IEEE1588v1 版本的 PTP 帧 ▶ 0b1：IEEE1588v2 版本的 PTP 帧
12	PTPFT	PTP 帧类型	该位指示接收到的 PTP 帧类型： ▶ 0b0：Over Ethernet 类型的 PTP 帧 ▶ 0b1：Over UDP-IPv4 或 Over UDP-IPv6 类型的 PTP 帧  说明 具体为 Over UDP-IPv4 还是 Over UDP-IPv6 可通过该描述符的 IPv6FR、IPv4FR 位获取。
11:8	MTP	报文类型	该位指示接收到的报文类型： ▶ 0b0000：未接收到任何 PTP 消息 ▶ 0b0001：SYNC（所有时钟类型） ▶ 0b0010：Follow_Up（所有时钟类型） ▶ 0b0011：Delay_Req（所有时钟类型） ▶ 0b0100：Delay_Resp（所有时钟类型） ▶ 0b0101：Pdelay_Req（在点对点透明时钟中） ▶ 0b0110：Pdelay_Resp（在点对点透明时钟中） ▶ 0b0111：Pdelay_Resp_Follow_Up（在点对点透明时钟中） ▶ 0b1000：Announce ▶ 0b1001：Management ▶ 0b1010：Signaling ▶ 0b1111：默认报文类型 ▶ 其他：Reserved
7	IPv6DR	接收到 IPv6 报	该位指示接收到 IPv6 报
6	IPv4DR	接收到 IPv4 报	该位指示接收到 IPv4 报
5	IPCB	Checksum Offload 直通	该位表示绕过 COE 引擎
4	IPPE	IP 有效负载错误	以下情况时，该位置 1： 1. 当内核计算的 16 位 IP 有效负载校验和（即 TCP、UDP 或 ICMP 校验和）与接收帧中对应的校验和字段不匹配 2. 当 TCP、UDP 或 ICMP 段长度与 IP 报头字段中的有效负载长度值不匹配
3	IPHE	IP 头错误	该位指示由内核计算的 16 位 IPv4 报头校验和与接收的校验和字节不匹配，或 IP 报头版本字段与以太网类型值不一致

位/位域	标记	位名	功能描述
2:0	IPPT	有效负载类型	该位表示 IP 数据报中封装的有效负载类型，当 IP 报头出错或存在分段的 IP 时，这些位为“000” ▶ 0b000：未知，或未处理 IP 有效负载 ▶ 0b001：UDP ▶ 0b010：TCP ▶ 0b011：ICMP ▶ 0b1xx：Reserved

- RDES5：Reserved
- RDES6 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	RTSL	时间戳地位	DMA 将相应接收帧所捕获的低 32 位更新进该字段。 说明 只有在描述符中最后一个描述符位（RDES0.RLS）置 1 时，该字段才包含时间戳。

- RDES7 的各位功能如下：

位/位域	标记	位名	功能描述
31:0	RTSH	时间戳高位	DMA 将相应接收帧所捕获的高 32 位更新进该字段。 说明 只有在描述符中最后一个描述符位（RDES0.RLS）置 1 时，该字段才包含时间戳。

38.3.6 中断

38.3.6.1 ETHMAC 中断概述

下图是以太网 MAC 控制器中的中断实现方案。从图中可以看出，ETHMAC 将 DMA 的正常事件（NIS）、DMA 的异常事件（AIS）、PMT 事件（PMTIS）、MMC 事件（MMC**IS）、PTP 事件（TSPIS）等通过中断的使能或屏蔽控制汇总成一个中断事件输出至中断模块，所以系统在中断响应时，需要通过读取相应的状态信息来确定是哪种类型的事件产生了中断。

同时，ETHMAC 的 PMT 模块还有一个非屏蔽的中断输出。在 PMT 事件产生时，不管中断屏蔽寄存器（ETH_MAC_INTMSKR）的 PMTIM 位是否置 1，都会产生 PMT 非屏蔽中断。

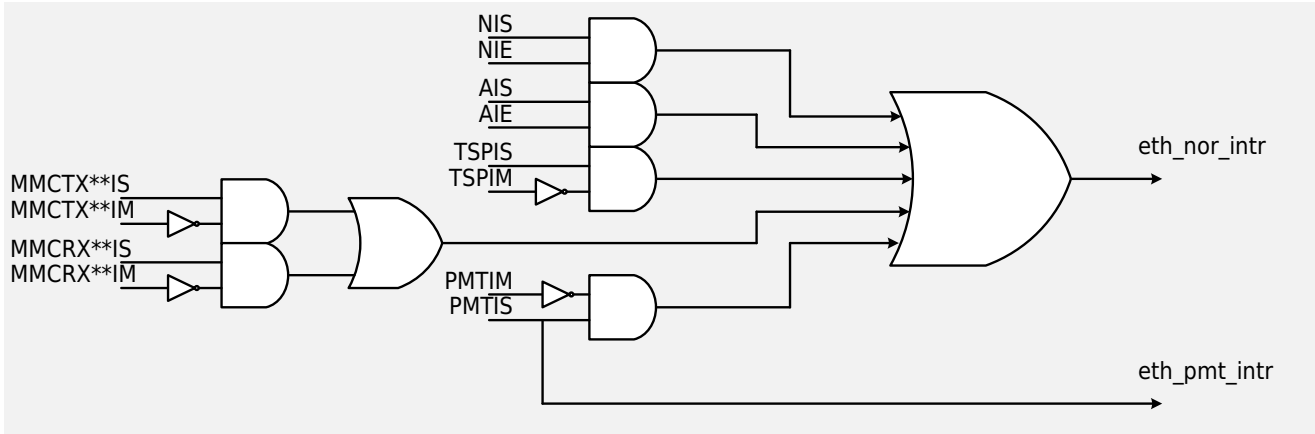


图 38-21 ETHMAC 中断方案

38.3.6.2 DMA 中断

DMA 在发送数据和接收数据过程中产生的各种正常事件会触发普通中断、各种异常事件会触发异常中断。如下图所示。

各具体事件的中断控制可通过 DMA 中断使能寄存器 (ETH_DMA_INTENAR) 的各个位控制。

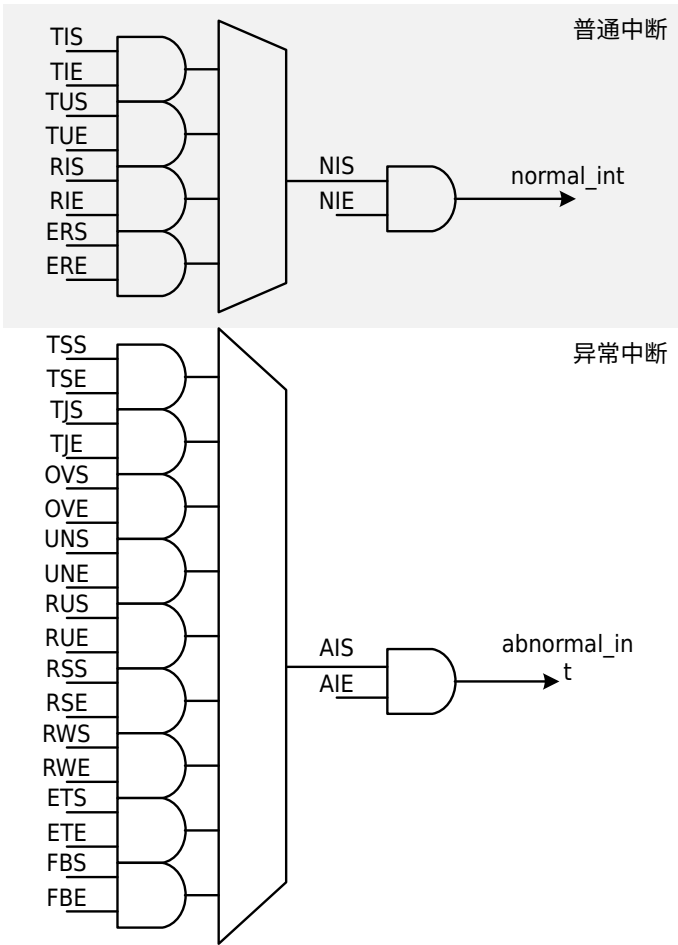


图 38-22 DMA 中断组成

38.3.6.3 PMT 中断

MAC 在接收到远程唤醒帧和魔术数据包时可触发生成 PMT 中断（可屏蔽中断和非屏蔽中断）。其中可屏蔽中断的生成由 MAC 中断屏蔽寄存器 (ETH_MAC_INTMSKR) 的 PMTIM 位控制。

38.3.6.4 PTP 中断

在 PTP 功能有效时，系统时间到达时间戳目标时间寄存器 0~1 中的设定值时可触发生成 PTP 中断。该中断由 PPS 输出控制寄存器 (ETH_PTP_PPCTLR) 的 TT0SEL、TT1SEL 位，ETH_PTP 时间戳控制寄存器 (ETH_PTP_TSPCTLR) 的 TSPINT 位和 MAC 中断屏蔽寄存器 (ETH_MAC_INTMSKR) 的 TSPIM 位控制。

38.3.6.5 MMC 中断

在 MMC 中各项发送帧统计数和接收帧统计数溢出时可触发生成 MMC 中断。该中断由 MMC 的 Rx 中断控制寄存器 (ETH_MMC_RITCTLR) 和 Tx 中断控制寄存器 (ETH_MMC_TITCTLR) 的各个位控制。

38.4 寄存器

38.4.1 寄存器总表

ETHMAC 公共寄存器基地址：0x40055400

表 38-25 ETHMAC 公共寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0010	ETH_MAC_IFCONFR	ETH_MAC 接口选择寄存器	32

ETHMAC 基地址：0x40060000

表 38-26 以太网 MAC 控制器 (ETHMAC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	ETH_MAC_CONFIGR	ETH_MAC MAC 配置寄存器	32
0x0018	ETH_MAC_FLOCTLR	ETH_MAC MAC 流控寄存器	32
0x0024	ETH_MAC_MACSTSR	ETH_MAC MAC 状态寄存器	32
0x0038	ETH_MAC_INTSTSR	ETH_MAC 中断状态寄存器	32
0x003C	ETH_MAC_INTMSKR	ETH_MAC 中断屏蔽寄存器	32
0x0010	ETH_MAC_SMIADDR	ETH_MAC SMI 地址寄存器	32
0x0014	ETH_MAC_SMIDATR	ETH_MAC SMI 数据寄存器	32
0x002C	ETH_MAC_PMTCTLR	ETH_MAC PMT 控制状态寄存器	32
0x0028	ETH_MAC_RTWKFFR	ETH_MAC 远程唤醒帧过滤寄存器	32
0x0004	ETH_MAC_FLTCTLR	ETH_MAC MAC 帧过滤控制寄存器	32
0x0040	ETH_MAC_MACADHR0	ETH_MAC MAC 地址高位寄存器 0	32
0x0044	ETH_MAC_MACADLR0	ETH_MAC MAC 地址低位寄存器 0	32
0x0048	ETH_MAC_MACADHR1	ETH_MAC MAC 地址高位寄存器 1	32
0x004C	ETH_MAC_MACADLR1	ETH_MAC MAC 地址低位寄存器 1	32
0x0050	ETH_MAC_MACADHR2	ETH_MAC MAC 地址高位寄存器 2	32
0x0054	ETH_MAC_MACADLR2	ETH_MAC MAC 地址低位寄存器 2	32
0x0058	ETH_MAC_MACADHR3	ETH_MAC MAC 地址高位寄存器 3	32
0x005C	ETH_MAC_MACADLR3	ETH_MAC MAC 地址低位寄存器 3	32
0x0060	ETH_MAC_MACADHR4	ETH_MAC MAC 地址高位寄存器 4	32
0x0064	ETH_MAC_MACADLR4	ETH_MAC MAC 地址低位寄存器 4	32
0x0008	ETH_MAC_HASHTHR	ETH_MAC MAC Hash 表高位寄存器	32
0x000C	ETH_MAC_HASHTLR	ETH_MAC MAC Hash 表低位寄存器	32

偏移地址	寄存器	描述	位宽
0x0584	ETH_MAC_VTACTLR	ETH_MAC VLAN 标记发送控制寄存器	32
0x001C	ETH_MAC_VTAFLTR	ETH_MAC VLAN 标记接收过滤寄存器	32
0x0588	ETH_MAC_VLAHTBR	ETH_MAC VLAN Hash 表寄存器	32
0x0400	ETH_MAC_L34CTLR	ETH_MAC LAY3LAY4 控制寄存器	32
0x0404	ETH_MAC_L4PORTR	ETH_MAC LAY4 端口寄存器	32
0x0410	ETH_MAC_L3ADDR0	ETH_MAC LAY3 地址寄存器 0	32
0x0414	ETH_MAC_L3ADDR1	ETH_MAC LAY3 地址寄存器 1	32
0x0418	ETH_MAC_L3ADDR2	ETH_MAC LAY3 地址寄存器 2	32
0x041C	ETH_MAC_L3ADDR3	ETH_MAC LAY3 地址寄存器 3	32
0x0700	ETH_PTP_TSPCTLR	ETH_PTP 时间戳控制寄存器	32
0x0728	ETH_PTP_TSPSTSR	ETH_PTP 时间戳状态寄存器	32
0x0718	ETH_PTP_TSPADDR	ETH_PTP 时间戳基本加数寄存器	32
0x0704	ETH_PTP_TSPNSAR	ETH_PTP 时间戳亚秒加数寄存器	32
0x0708	ETH_PTP_TMSSECR	ETH_PTP 时间戳系统秒寄存器	32
0x070C	ETH_PTP_TMSNSER	ETH_PTP 时间戳系统亚秒寄存器	32
0x0710	ETH_PTP_TMUSECR	ETH_PTP 时间戳更新秒寄存器	32
0x0714	ETH_PTP_TMUNSER	ETH_PTP 时间戳更新亚秒寄存器	32
0x071C	ETH_PTP_TMTSECR0	ETH_PTP 时间戳目标秒寄存器 0	32
0x0720	ETH_PTP_TMTNSER0	ETH_PTP 时间戳目标亚秒寄存器 0	32
0x0780	ETH_PTP_TMTSECR1	ETH_PTP 时间戳目标秒寄存器 1	32
0x0784	ETH_PTP_TMTNSER1	ETH_PTP 时间戳目标亚秒寄存器 1	32
0x072C	ETH_PTP_PPSCTLR	ETH_PTP PPS 输出控制寄存器	32
0x1000	ETH_DMA_BUSMODR	ETH_DMA 总线模式寄存器	32
0x1018	ETH_DMA_OPRMODR	ETH_DMA 操作模式寄存器	32
0x1014	ETH_DMA_DMASTR	ETH_DMA 动作状态寄存器	32
0x101C	ETH_DMA_INTENAR	ETH_DMA 中断使能寄存器	32
0x1020	ETH_DMA_RFRCNTR	ETH_DMA 帧丢失统计寄存器	32
0x1024	ETH_DMA_REVWDTR	ETH_DMA 看门狗定时寄存器	32
0x1004	ETH_DMA_TXPOLLR	ETH_DMA 发送轮询要求寄存器	32
0x1008	ETH_DMA_RXPOLLR	ETH_DMA 接收轮询要求寄存器	32

偏移地址	寄存器	描述	位宽
0x1010	ETH_DMA_TXDLADR	ETH_DMA 发送描述符列表地址寄存器	32
0x100C	ETH_DMA_RXDLADR	ETH_DMA 接收描述符列表地址寄存器	32
0x1048	ETH_DMA_CHTXDER	ETH_DMA 当前主机发送描述符寄存器	32
0x104C	ETH_DMA_CHRXDER	ETH_DMA 当前主机接收描述符寄存器	32
0x1050	ETH_DMA_CHTXBFR	ETH_DMA 当前主机发送缓冲区寄存器	32
0x1054	ETH_DMA_CHRXBFR	ETH_DMA 当前主机接收缓冲区寄存器	32
0x0100	ETH_MMC_MMCCTLR	ETH_MMC MMC 控制寄存器	32
0x0104	ETH_MMC_REVSTSR	ETH_MMC Rx 统计状态寄存器	32
0x0108	ETH_MMC_TRSSTSR	ETH_MMC Tx 统计状态寄存器	32
0x010C	ETH_MMC_RITCTLR	ETH_MMC Rx 中断控制寄存器	32
0x0110	ETH_MMC_TITCTLR	ETH_MMC Tx 中断控制寄存器	32
0x01C4	ETH_MMC_RXUNGFR	ETH_MMC Rx 单播好帧统计寄存器	32
0x0190	ETH_MMC_RXMUGFR	ETH_MMC Rx 多播好帧统计寄存器	32
0x018C	ETH_MMC_RXBRGFR	ETH_MMC Rx 广播好帧统计寄存器	32
0x0194	ETH_MMC_RXCREFR	ETH_MMC Rx CRC 错误帧统计寄存器	32
0x0198	ETH_MMC_RXALEFR	ETH_MMC Rx 对齐错误帧统计寄存器	32
0x019C	ETH_MMC_RXRUEFR	ETH_MMC Rx 矮帧错误帧统计寄存器	32
0x01C8	ETH_MMC_RXLEEFR	ETH_MMC Rx 长度错误帧统计寄存器	32
0x01CC	ETH_MMC_RXOREFR	ETH_MMC Rx 超范围错误帧统计寄存器	32
0x0168	ETH_MMC_TXUNGFR	ETH_MMC Tx 单播好帧统计寄存器	32
0x0120	ETH_MMC_TXMUGFR	ETH_MMC Tx 多播好帧统计寄存器	32
0x011C	ETH_MMC_TXBRGFR	ETH_MMC Tx 广播好帧统计寄存器	32
0x0154	ETH_MMC_TXDEEFR	ETH_MMC Tx 延迟错误帧统计寄存器	32
0x0160	ETH_MMC_TXCAEFR	ETH_MMC Tx 载波错误帧统计寄存器	32
0x0158	ETH_MMC_TXLCEFR	ETH_MMC Tx 延迟冲突错误帧统计寄存器	32
0x015C	ETH_MMC_TXECEFR	ETH_MMC Tx 过度冲突错误帧统计寄存器	32
0x016C	ETH_MMC_TXEDEFR	ETH_MMC Tx 过度延迟错误帧统计寄存器	32

38.4.2 ETH_MAC 寄存器

38.4.2.1 ETH_MAC 接口选择寄存器 (ETH_MAC_IFCONFR)

Offset	Bit Position																																	
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																										TCKINV		RCKINV		Reserved		IFSEL	
Access																											RW		RW				RW	

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	TCKINV	TX 时钟取反	0b0: 从 ETH_MII_TXCLK 端口输入的时钟保持原极性 0b1: 从 ETH_MII_TXCLK 端口输入的时钟极性取反	RW
4	RCKINV	RX 时钟取反	0b0: 从 ETH_MII_RMII_RXCLK 端口输入的时钟保持原极性 0b1: 从 ETH_MII_RMII_RXCLK 端口输入的时钟极性取反	RW
3:1	Reserved	保留	-	-
0	IFSEL	接口选择位	0b0: MII 接口 0b1: RMII 接口	RW

38.4.2.2 ETH_MAC MAC 配置寄存器 (ETH_MAC_CONFIGR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008000																															
Name	Reserved	SAIRC			Reserved	CST	Reserved	MWD	MJB	Reserved	IFG				DCRS	Reserved	FES	DO	LM	DM	IPCO	DRTY	Reserved	ACS	BL		DC	TE	RE	Reserved		
Access		RW				RW		RW	RW			RW			RW		RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW			

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:28	SAIRC	源地址插入及替换	0b010: 将 MAC 地址寄存器 0 中的地址值作为 SA 地址插入到发送帧中 0b011: 将 MAC 地址寄存器 0 中的地址值作为 SA 地址替换发送帧中的 SA 字段 0b110: 将 MAC 地址寄存器 1 中的地址值作为 SA 地址插入到发送帧中 0b111: 将 MAC 地址寄存器 1 中的地址值作为 SA 地址替换发送帧中的 SA 字段 0bx0x: 发送帧中 SA 字段地址的处理由 Tx 描述符 TDES1 的 SAIRC 位控制	RW
27:26	Reserved	保留	-	-
25	CST	TYPE 帧 FCS 去除	0b0: 该功能无效 0b1: 在将帧转发到应用之前, 去除并丢弃所有 TYPE (类型字段大于 0x0600) 帧的最后 4 个字节 (FCS 字段)	RW
24	Reserved	保留	-	-
23	MWD	屏蔽看门狗	0b0: MAC 允许接收的帧不超过 2048 字节, 并会截断超过此限制接收的任何字节 0b1: MAC 禁止接收器上的看门狗定时器, 并可接收多达 16384 字节的帧	RW
22	MJB	屏蔽 Jabber	0b0: 发送期间应用发送超过 2048 字节的数据, MAC 会截断发送器 0b1: MAC 禁止发送器上的 Jabber 定时器, 并可发送多达 16384 字节的帧	RW
21:20	Reserved	保留	-	-
19:17	IFG	帧间间隙	发送期间帧间的最小间隙: 0b000: 96 位时间 0b001: 88 位时间 0b010: 80 位时间 0b111: 40 位时间  说明 在半双工模式下, 最小 IFG 仅能配置为 64 位时间 (IFG=100), 不考虑更低的值。	RW

位/位域	标记	位名	功能描述	读写
16	DCRS	屏蔽载波侦听	0b0: 半双工模式下帧发送期间的 MII_CRS 信号有效时, MAC 发送器会生成载波侦听错误, 甚至中止发送 0b1: 半双工模式下帧发送期间的 MII_CRS 信号被忽略, 不会因在此发送期间载波丢失或无载波而生成错误  说明 该位只在半双工模式时有效, 全双工模式时无效。	RW
15	Reserved	保留	-	-
14	FES	速度选择	0b0: 10Mbps 0b1: 100Mbps	RW
13	DO	屏蔽接收自身包	0b0: 在半双工模式下, MAC 接收来自 PHY 的所有包 0b1: 在半双工模式下, MAC 禁止接收帧  说明 该位只在半双工模式时有效, 全双工模式时无效。	RW
12	LM	Loopback 模式	0b0: Loopback 模式无效 0b1: Loopback 模式有效	RW
11	DM	全双工模式	0b0: 半双工模式 0b1: 全双工模式	RW
10	IPCO	Checksum Offload 功能	0b0: IP Checksum Offload 功能无效 0b1: IP Checksum Offload 功能有效	RW
9	DRTY	屏蔽重试	0b0: 在 MII 模式下发生冲突时, MAC 会尝试根据该寄存器的 BL 位的设置进行重试 0b1: 在 MII 模式下发生冲突时, MAC 会忽略当前帧发送并以发送帧状态下过度冲突错误报告帧中止, 即 MAC 仅尝试一次发送  说明 该位只在半双工模式时有效, 全双工模式时无效。	RW
8	Reserved	保留	-	-
7	ACS	自动去除 PAD/FCS	0b0: MAC 不管接收到的帧的长度, 传送所有接收帧到应用 0b1: MAC 在长度字段值小于 1536 字节时自动去除接收帧上的 PAD/FCS 字段, 在长度字段大于或等于 1536 字节时不去除 PAD/FCS 字段	RW

位/位域	标记	位名	功能描述	读写
6:5	BL	后退限制	后退限制决定发生冲突后重试期间 MAC 在重新安排一次发送之前等待的随机整数 (r) 个时间片延迟 (512 位时间)： ● 0b00: $k=\min(n,10)$ ● 0b01: $k=\min(n,8)$ ● 0b10: $k=\min(n,4)$ ● 0b11: $k=\min(n,1)$ 其中 n=重新发送尝试的次数。随机整数 r 的取值范围为：$0 \leq r < 2^k$  说明 该位只在半双工模式时有效，全双工模式时无效。	RW
4	DC	延迟检查	● 0b0: 禁止延迟检查功能。MAC 发生延迟，直到 CRS 信号变为无效信号 ● 0b1: 使能延迟检查功能。当发送状态机在延迟超过 24288 位时间时，MAC 将指示帧中止状态，同时在发送帧状态中将过度延迟错误位置 1  说明 1. 延迟的产生：当发送器准备好发送但因 MII 上含有有效 CRS（载波侦听）信号而被阻止时延迟计数就开始。延迟时间是非累积性的，如果发送器延迟 10000 位时间，则重新执行一遍发送、冲突、后退，然后在完成后退后复位延迟计数器为 0 并重新开始计数。 2. 该位只在半双工模式时有效，全双工模式时无效。	RW
3	TE	发送使能	● 0b0: 发送无效 ● 0b1: 发送使能	RW
2	RE	接收使能	● 0b0: 接收无效 ● 0b1: 接收使能	RW
1:0	Reserved	保留	-	-

38.4.2.3 ETH_MAC MAC 流控寄存器 (ETH_MAC_FLOCTLR)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																						
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																							
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																						
Name	PAUSET																Reserved								DZPQ	Reserved	PLT	UNP	RFE	TFE	FCA BPA																																																																																																																																																																																																																																																																																																								
Access																										RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:16	PAUSET	暂停时间	该位表示发送控制帧中暂停时间字段要使用的值	RW
15:8	Reserved	保留	-	-
7	DZPQ	屏蔽零时间片帧	0b0: 当流控制信号有效时, 正常生成零时间片暂停控制帧 0b1: 当流控制信号有效时, 禁止生成零时间片暂停控制帧	RW
6	Reserved	保留	-	-
5:4	PLT	暂停阈值下限	配置暂停定时器的阈值, 达到该值时会自动重新传输 PAUSE 帧: 0b00: 暂停时间减去 4 个时隙 0b01: 暂停时间减去 28 个时隙 0b10: 暂停时间减去 144 个时隙 0b11: 暂停时间减去 256 个时隙  说明 - 时隙定义为 MII 接口每发送 512 位 (64 字节) 所需的时间。 - 该阈值应始终小于暂停时间。例如, 如果暂停时间 PAUSET=0x100 (256 个时隙), 而 PLT=0b01, 则在第一个 PAUSE 帧发送完成后, 第二个 PAUSE 帧会在第 228(256-28)个时隙启动时自动发送。	RW
3	UNP	单播暂停帧检测	0b0: MAC 仅检测具有 802.3x 标准中指定的唯一多播地址的暂停帧 0b1: MAC 除了检测具有唯一多播地址的暂停帧外, 还会检测具有 ETH_MAC_MACADHR0 和 ETH_MAC_MACADLR0 寄存器所指定的站单播地址的暂停帧	RW
2	RFE	接收流控制使能	0b0: 禁止暂停帧的解码功能 0b1: MAC 对接收到的暂停帧进行解码, 并禁止其在指定的暂停时间内发送	RW
1	TFE	发送流控制使能	全双工模式时: 0b0: 禁止 MAC 传送任何暂停帧 0b1: 使能流控制操作, MAC 发送暂停帧 半双工模式时: 0b0: 禁止背压功能 0b1: 使能背压操作	RW

位/位域	标记	位名	功能描述	读写
0	FCA_BPA	流控制状态	向流控制寄存器写入数据前此位应读为 0 全双工模式时： 此位置 1 时 MAC 内核启动暂停控制帧。在控制帧发送过程中，此位保持置 1 以指示帧发送正在进行中。当暂停控制帧发送完成后，MAC 会将此位复位为 0。此位清零后，才可以对流控制寄存器执行写操作 半双工模式时： 此位置 1 时，MAC 内核将置位背压功能。在背压操作期间，当 MAC 接收到新帧时，发送器会开始发送一个导致冲突的 Jam 信号	RW

38.4.2.4 ETH_MAC MAC 状态寄存器 (ETH_MAC_MACSTSR)

Offset	Bit Position																																		
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	Reserved							TFF	TFNE	Reserved	TFWA	TFRS		MTP	MTS		MTEA	Reserved							RFFL	Reserved		RFRS		RFWA	Reserved		MRS		MREA
Access								RO	RO		RO	RO		RO	RO		RO								RO			RO		RO			RO		RO

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	TFF	Tx FIFO 满	0b0: Tx FIFO 未滿 0b1: Tx FIFO 已滿, 无法再接收发送的帧	RO
24	TFNE	Tx FIFO 非空	0b0: Tx FIFO 空 0b1: Tx FIFO 非空, 还有未发送的帧	RO
23	Reserved	保留	-	-
22	TFWA	Tx FIFO 写有效	0b0: Tx FIFO 写无效 0b1: Tx FIFO 写控制器有效且正在将数据传输到 Tx FIFO	RO
21:20	TFRS	Tx FIFO 读状态	0b00: 空闲态 0b01: 读状态 (数据从 Tx FIFO 往 MAC 发送器传送) 0b10: 等待来自 MAC 发送器的 Tx Status 0b11: 写入收到的 Tx Status 或刷新 Tx FIFO	RO
19	MTP	MAC 发送器暂停	0b0: MAC 发送器处于发送数据状态 0b1: MAC 发送器处于暂停状态 (全双工模式时), 不安排任何帧发送	RO
18:17	MTS	MAC 发送器状态	0b00: 空闲态 0b01: 等待前一个帧的状态或 IFG/回退阶段结束 0b10: 生成并发送暂停控制帧 (在全双工模式下) 0b11: 传输要发送的输入帧	RO
16	MTEA	MAC MII 发送引擎状态	0b0: MAC MII 发送引擎处于空闲态 0b1: MAC MII 发送引擎正在主动发送数据而未处于空闲状态	RO
15:10	Reserved	保留	-	-
9:8	RFFL	Rx FIFO 填充级别	0b00: Rx FIFO 为空 0b01: Rx FIFO 填充级别低于流控制, 取消激活阈值 0b10: Rx FIFO 填充级别高于流控制, 激活阈值 0b11: Rx FIFO 已滿	RO
7	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
6:5	RFRS	Rx FIFO 读状态	● 0b00: 空闲态 ● 0b01: 读取帧数据 ● 0b10: 读取帧状态或时间戳 ● 0b11: 刷新帧数据和状态	RO
4	RFWA	Rx FIFO 写有效	● 0b0: Rx FIFO 写无效 ● 0b1: Rx FIFO 写有效，并正在将收到的帧传输到 Rx FIFO	RO
3	Reserved	保留	-	-
2:1	MRS	MAC 接收器状态	● 0b00: 空闲 ● 0b01: MAC 接收器中的 FIFO 写控制器在工作 ● 0b10: MAC 接收器中的 FIFO 读控制器在工作 ● 0b11: MAC 接收器中的 FIFO 读和写控制器在工作	RO
0	MREA	MAC MII 接收引擎状态	● 0b0: MAC MII 接收引擎处于空闲态 ● 0b1: MAC MII 接收引擎正在主动接收数据而未处于空闲状态	RO

38.4.2.5 ETH_MAC 中断状态寄存器 (ETH_MAC_INTSTSR)

Offset	Bit Position																																				
0x0038	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved																							RO TSPIS		Reserved		RO MMCTXIS		RO MMCRXIS		RO MMCIS		RO PMTIS		Reserved	
Access																								RO				RO		RO		RO		RO			

位/位域	标记	位名	功能描述	读写
31:10	Reserved	保留	-	-
9	TSPIS	PTP 时间戳中断	当系统时间值等于或大于目标时间寄存器中指定的值时，或者系统时间溢出时，此位置 1  说明 该位在读取 ETH_PTP_TSPSTSR 寄存器的 Bit0 位时清零。	RO
8:7	Reserved	保留	-	-
6	MMCTXIS	MMC 发送统计中断	ETH_MMC_TITCTLR 寄存器中生成中断，此位就变为高电平  说明 该位在 ETH_MMC_TITCTLR 寄存器中所述所有中断清零时清零。	RO
5	MMCRXIS	MMC 接收统计中断	ETH_MMC_RITCTLR 寄存器中生成中断，此位就变为高电平  说明 该位在 ETH_MMC_RITCTLR 寄存器中所述所有中断清零时清零。	RO
4	MMCIS	MMC 统计中断	该寄存器的 MMCTXIS 或 MMCRXIS 位有高电平，此位就变为高电平  说明 该位在该寄存器的 Bit6~Bit5 均变为 0 时清零。	RO
3	PMTIS	PMT 中断	在 Power Down 模式下接收到魔术数据包或远程唤醒帧，此位就变为高电平  说明 该位在 ETH_MAC_PMTCTLR 寄存器的 WKFR 和 MPFR 位均变为 0 时清零。	RO
2:0	Reserved	保留	-	-

38.4.2.6 ETH_MAC 中断屏蔽寄存器 (ETH_MAC_INTMSKR)

Offset	Bit Position																															
0x003C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																						TSPIM	Reserved				PMTIM	Reserved			
Access																							RW					RW				

位/位域	标记	位名	功能描述	读写
31:10	Reserved	保留	-	-
9	TSPIM	PTP 时间戳中断屏蔽	● 0b0: PTP 时间戳中断不屏蔽 ● 0b1: PTP 时间戳中断屏蔽	RW
8:4	Reserved	保留	-	-
3	PMTIM	PMT 中断屏蔽	● 0b0: PMT 中断不屏蔽 ● 0b1: PMT 中断屏蔽	RW
2:0	Reserved	保留	-	-

38.4.2.7 ETH_MAC SMI 地址寄存器 (ETH_MAC_SMIADDR)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SMIA				SMIR				SMIC				SMIW		SMIB	
Access																																

位/位域	标记	位名	功能描述	读写																		
31:16	Reserved	保留	-	-																		
15:11	SMIA	物理层地址	指示要访问 32 个可能的 PHY 器件中的哪一个	RW																		
10:6	SMIR	PHY 寄存器	指示在所选 PHY 器件中要选择寄存器	RW																		
5:2	SMIC	MDC 时钟	基于系统频率（PCLK1）决定 MDC 时钟频率： <table><tr><th>选项</th><th>系统频率</th><th>MDC 时钟</th></tr><tr><td>0b0000</td><td>60-100MHz</td><td>系统频率/42</td></tr><tr><td>0b0001</td><td>100-120MHz</td><td>系统频率/62</td></tr><tr><td>0b0010</td><td>20-35MHz</td><td>系统频率/16</td></tr><tr><td>0b0011</td><td>35-60MHz</td><td>系统频率/26</td></tr><tr><td>其他</td><td>-</td><td>Reserved</td></tr></table>	选项	系统频率	MDC 时钟	0b0000	60-100MHz	系统频率/42	0b0001	100-120MHz	系统频率/62	0b0010	20-35MHz	系统频率/16	0b0011	35-60MHz	系统频率/26	其他	-	Reserved	RW
选项	系统频率	MDC 时钟																				
0b0000	60-100MHz	系统频率/42																				
0b0001	100-120MHz	系统频率/62																				
0b0010	20-35MHz	系统频率/16																				
0b0011	35-60MHz	系统频率/26																				
其他	-	Reserved																				
1	SMIW	SMI 读写	● 0b0：SMI 读操作 ● 0b1：SMI 写操作	RW																		
0	SMIB	SMI 访问忙	● 向 ETH_MAC_SMIADDR 和 ETH_MAC_SMIDATR 写入前，此位应读取逻辑 0 ● 向 ETH_MAC_SMIADDR 写入过程中，此位也必须复位为 0 ● 在 PHY 寄存器访问过程中，此位由应用程序设为 1，指示读或写访问正在进行中 在对 PHY 进行写操作过程中，ETH_MAC_SMIDATR 应始终有效，直到 MAC 将此位清零；在对 PHY 进行读操作过程中，ETH_MAC_SMIDATR 始终无效，直到 MAC 将此位清零。此位清零后，才可以向 ETH_MAC_SMIADDR 写入值	RW																		

38.4.2.8 ETH_MAC SMI 数据寄存器 (ETH_MAC_SMIDATR)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SMID															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	SMID	SMI 数据	站点管理读操作之后从 PHY 中读取的 16 位数据值，或站点管理写操作之前要写入 PHY 的 16 位数据值	RW

38.4.2.9 ETH_MAC PMT 控制状态寄存器 (ETH_MAC_PMTCTLR)

Offset	Bit Position																																		
0x002C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	RTWKFR	Reserved				RTWKPT				Reserved												RTWKTR	GLBU		Reserved		WKFR	MPFR		Reserved		WKEN	MPEN		PWDN
Access	RW					RO																RW	RW			RO	RO				RW	RW	RW		

位/位域	标记	位名	功能描述	读写
31	RTWKFR	远程唤醒帧过滤寄存器指针复位	0b0: 远程唤醒帧过滤寄存器指针不复位 0b1: 远程唤醒帧过滤寄存器指针复位  说明 复位完成后该位自动清零。	RW
30:27	Reserved	保留	-	-
26:24	RTWKPT	远程唤醒帧过滤寄存器指针位置	该位表明远程唤醒帧过滤寄存器指针在第几个寄存器上 (该值为 0~7)	RO
23:11	Reserved	保留	-	-
10	RTWKTR	远程唤醒帧传送	该位置 1 后, MAC 不再接收除魔术数据包和远程唤醒帧以外的包, 并将接收到的远程唤醒帧转发给应用  说明 - 该位在接收到魔术数据包或远程唤醒帧之后会自动变为 0; 也可以在接收到魔术数据包或远程唤醒帧之前软件复位。 - 当该寄存器的 GLBU 位或 WKEN 位或 MPEN 位为 1、同时 PWDN 位为 0 时, 该位必须设定为 1。 - 在 PWDN 位为 1 时, 该位设定无效。	RW
9	GLBU	全局单播	0b0: 将 MAC (DAF) 地址过滤的任意单播帧视为无效 0b1: 将 MAC (DAF) 地址过滤的任意单播帧视为唤醒帧	RW
8:7	Reserved	保留	-	-
6	WKFR	接收远程唤醒帧	0b0: 未接收到远程唤醒帧 0b1: 接收到远程唤醒帧  说明 该位在读过之后自动清零。	RO
5	MPFR	接收魔术数据包	0b0: 未接收到魔术数据包 0b1: 接收到魔术数据包  说明 该位在读过之后自动清零。	RO
4:3	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
2	WKEN	远程唤醒帧使能	0b0: 不接收远程唤醒帧 0b1: 接收远程唤醒帧, 触发 PMT 事件	RW
1	MPEN	魔术数据包使能	0b0: 不接收魔术数据包 0b1: 接收魔术数据包, 触发 PMT 事件	RW
0	PWDN	Power Down 模式使能	0b0: Active 模式 0b1: Power Down 模式 所有接收到的帧都丢弃, 只有接收到魔术数据包或远程唤醒帧时, 此位会自动清零, 同时退出 Power Down 模式; 该位清零后收到的帧会转发给应用。  说明 只有 WKEN 位或 MPEN 位置 1 时, 才可以将此位置 1。	RW

38.4.2.10 ETH_MAC 远程唤醒帧过滤寄存器 (ETH_MAC_RTWKFFR)

Offset	Bit Position																															
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	WKUPFRMFT																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	WKUPFRMFT	远程唤醒帧过滤值	应用将通过此地址对远程唤醒帧过滤寄存器进行写/读操作。详细信息请参见“ 远程唤醒帧过滤 ”	RW

38.4.2.11 ETH_MAC MAC 帧过滤控制寄存器 (ETH_MAC_FLTCTLR)

Offset	Bit Position																																			
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	RA	Reserved										DNTU	IPFE	Reserved			VTFE	Reserved						HPF	SAF	SAIF	PCF		DBF	PMF	DAIF	HMC	HUC	PR		
Access	RW											RW	RW				RW							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31	RA	接收所有	0b0: MAC 接收器将通过地址过滤的接收帧传送给应用 0b1: MAC 接收器将所有的接收帧传送到应用, 不管它们是否已通过地址过滤	RW
30:22	Reserved	保留	-	-
21	DNTU	丢弃无 TCP/UDP 报	0b0: MAC 接收器将所有接收的 IP 报传送到应用, 不管它们是否有 TCP/UDP 字段 0b1: MAC 接收器丢弃所有无 TCP 或 UDP 字段的 IP 报	RW
20	IPFE	L3&L4 过滤使能	0b0: MAC 接收器将所有接收的 IP 报传送到应用, 不管它们是否已通过 L3&L4 过滤 0b1: MAC 接收器丢弃所有与 L3&L4 设定的过滤值比较不匹配的 IP 报, 只处理通过 L3&L4 过滤的 IP 报	RW
19:17	Reserved	保留	-	-
16	VTFE	VLAN 标记过滤使能	0b0: MAC 接收器将所有接收的 VLAN 帧传送到应用, 不管它们是否已通过 VLAN 标记过滤 0b1: MAC 接收器丢弃与 VLAN 标记设定的过滤值比较不匹配的 VLAN 帧	RW
15:11	Reserved	保留	-	-
10	HPF	Hash 过滤或完美过滤选择	0b0: 若该寄存器的 HUC 位或 HMC 位置 1, 则地址过滤器只传送与 Hash 过滤器匹配的帧 0b1: 若该寄存器的 HUC 位或 HMC 位置 1, 则地址过滤器传送与完美过滤器或 Hash 过滤器匹配的帧	RW
9	SAF	源地址过滤	0b0: MAC 接收器将所有接收的帧传送到应用, 不管它们是否已通过源地址过滤 0b1: MAC 接收器丢弃与源地址设定的过滤值比较不匹配的帧	RW
8	SAIF	源地址反转过滤	0b0: MAC 接收器接收到的帧 SA 字段与设定的过滤值不匹配时, 认定为过滤失败 0b1: MAC 接收器接收到的帧 SA 字段与设定的过滤值匹配时, 认定为过滤失败	RW

位/位域	标记	位名	功能描述	读写
7:6	PCF	传送控制帧	对控制帧（包括单播暂停帧和多播暂停帧）的转发做如下操作： ● 0b00：MAC 阻止所有控制帧到达应用 ● 0b01：MAC 将除了暂停控制帧以外的所有控制帧转发到应用，即使这些帧的地址过滤失败 ● 0b10：MAC 将所有控制帧转发到应用，即使这些帧的地址过滤失败 ● 0b11：MAC 转发通过地址过滤的控制帧  说明 关于暂停控制帧的几点说明： 1. 在全双工模式时，暂停控制帧的处理由流控寄存器 (ETH_MAC_FLOCTL) 的 RFE 位决定。 2. 在流控寄存器 (ETH_MAC_FLOCTL) 的 UNP 位置 1 时，接收到的帧的目标地址和 MAC 地址寄存器 0 设定的值匹配时，该帧被认为是单播暂停帧。 3. 接收到的帧的 TYPE 段值为 0x8808 或 OP CODE 段值为 0x0001 时，该帧被认为是暂停控制帧。	RW
5	DBF	屏蔽广播帧	● 0b0：地址过滤器不对接收到的广播帧进行过滤 ● 0b1：地址过滤器对所有传入的广播帧进行过滤  说明 该位置 1 时，其优先级高于其它过滤设定。	RW
4	PMF	传送所有多播帧	● 0b0：对接收到的带多播目标地址（目标地址字段的第一位是 1）的帧进行地址过滤，过滤方式由该寄存器的 HMC 位决定 ● 0b1：对接收到的带多播目标地址（目标地址字段的第一位是 1）的帧不进行地址过滤	RW
3	DAIF	目标地址反转过滤	● 0b0：MAC 接收器接收到的单播帧或多播帧的目标地址与设定的过滤值不匹配时，认定为过滤失败 ● 0b1：MAC 接收器接收到的单播帧或多播帧的目标地址与设定的过滤值匹配时，认定为过滤失败	RW
2	HMC	Hash 多播控制	● 0b0：MAC 对多播帧执行完美目标地址过滤，即将 DA 字段与设定的过滤值进行比较 ● 0b1：MAC 对多播帧执行 Hash 目标地址过滤	RW
1	HUC	Hash 单播控制	● 0b0：MAC 对单播帧执行完美目标地址过滤，即将 DA 字段与设定的过滤值进行比较 ● 0b1：MAC 对单播帧执行 Hash 目标地址过滤	RW
0	PR	混合模式	● 0b0：地址过滤器进行正常的地址过滤 ● 0b1：地址过滤器不管目标或源地址，都传送所有传入的帧	RW

38.4.2.12 ETH_MAC MAC 地址高位寄存器 0 (ETH_MAC_MACADHR0)

Offset	Bit Position																																
0x0040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x8000FFFF																																
Name		Reserved																ASDDRH0															
Access	RO																	RW															

位/位域	标记	位名	功能描述	读写
31	AE0	地址使能 0	● 0b1：使用 MAC 地址 0 进行目标地址完美过滤  说明 该位始终为 1。	RO
30:16	Reserved	保留	-	-
15:0	ASDDRH0	高位地址值 0	MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中	RW

38.4.2.13 ETH_MAC MAC 地址低位寄存器 0 (ETH_MAC_MACADLR0)

Offset	Bit Position																															
0x0044	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDRLO																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	ADDRLO	低位地址值 0	MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中	RW

38.4.2.14 ETH_MAC MAC 地址高位寄存器 1 (ETH_MAC_MACADHR1)

Offset	Bit Position																																	
0x0048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x0000FFFF																																	
Name			MBC1						Reserved								ASDDRH1																	
	AE1	SA1																																
Access	RW	RW	RW														RW																	

位/位域	标记	位名	功能描述	读写
31	AE1	地址使能 1	0b0: 忽略 MAC 地址 1 的地址过滤 0b1: 使用 MAC 地址 1 进行完美过滤	RW
30	SA1	地址选择 1	0b0: 用 MAC 地址 1 对接收帧的 DA 进行过滤 0b1: 用 MAC 地址 1 对接收帧的 SA 进行过滤	RW
29:24	MBC1	字节屏蔽 1	地址过滤时屏蔽对应的字节 - Bit29: 屏蔽 ADDR1[15:8]位 - Bit28: 屏蔽 ADDR1[7:0]位 - Bit27: 屏蔽 ADDRL1[31:24]位 - Bit26: 屏蔽 ADDRL1[23:16]位 - Bit25: 屏蔽 ADDRL1[15:8]位 - Bit24: 屏蔽 ADDRL1[7:0]位	RW
23:16	Reserved	保留	-	-
15:0	ASDDRH1	高位地址值 1	MAC 使用此字段过滤所接收的帧	RW

38.4.2.15 ETH_MAC MAC 地址低位寄存器 1 (ETH_MAC_MACADLR1)

Offset	Bit Position																															
0x004C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDR L1																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	ADDRL1	低位地址值 1	MAC 使用此字段过滤所接收的帧	RW

38.4.2.16 ETH_MAC MAC 地址高位寄存器 2 (ETH_MAC_MACADHR2)

Offset	Bit Position																															
0x0050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name			MBC2						Reserved								ASDDRH2															
	AE2	SA2																														
Access	RW	RW	RW														RW															

位/位域	标记	位名	功能描述	读写
31	AE2	地址使能 2	0b0: 忽略 MAC 地址 2 的地址过滤 0b1: 使用 MAC 地址 2 进行完美过滤	RW
30	SA2	地址选择 2	0b0: 用 MAC 地址 2 对接收帧的 DA 进行过滤 0b1: 用 MAC 地址 2 对接收帧的 SA 进行过滤	RW
29:24	MBC2	字节屏蔽 2	地址过滤时屏蔽对应的字节 - Bit29: 屏蔽 ADDR2[15:8]位 - Bit28: 屏蔽 ADDR2[7:0]位 - Bit27: 屏蔽 ADDRL2[31:24]位 - Bit26: 屏蔽 ADDRL2[23:16]位 - Bit25: 屏蔽 ADDRL2[15:8]位 - Bit24: 屏蔽 ADDRL2[7:0]位	RW
23:16	Reserved	保留	-	-
15:0	ASDDRH2	高位地址值 2	MAC 使用此字段过滤所接收的帧	RW

38.4.2.17 ETH_MAC MAC 地址低位寄存器 2 (ETH_MAC_MACADLR2)

Offset	Bit Position																															
0x0054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDR12																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	ADDRL2	低位地址值 2	MAC 使用此字段过滤所接收的帧	RW

38.4.2.18 ETH_MAC MAC 地址高位寄存器 3 (ETH_MAC_MACADHR3)

Offset	Bit Position																															
0x0058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name			MBC3						Reserved								ASDDRH3															
	AE3	SA3																														
Access	RW	RW	RW														RW															

位/位域	标记	位名	功能描述	读写
31	AE3	地址使能 3	0b0: 忽略 MAC 地址 3 的地址过滤 0b1: 使用 MAC 地址 3 进行完美过滤	RW
30	SA3	地址选择 3	0b0: 用 MAC 地址 3 对接收帧的 DA 进行过滤 0b1: 用 MAC 地址 3 对接收帧的 SA 进行过滤	RW
29:24	MBC3	字节屏蔽 3	地址过滤时屏蔽对应的字节 - Bit29: 屏蔽 ADDR3[15:8]位 - Bit28: 屏蔽 ADDR3[7:0]位 - Bit27: 屏蔽 ADDRL3[31:24]位 - Bit26: 屏蔽 ADDRL3[23:16]位 - Bit25: 屏蔽 ADDRL3[15:8]位 - Bit24: 屏蔽 ADDRL3[7:0]位	RW
23:16	Reserved	保留	-	-
15:0	ASDDRH3	高位地址值 3	MAC 使用此字段过滤所接收的帧	RW

38.4.2.19 ETH_MAC MAC 地址低位寄存器 3 (ETH_MAC_MACADLR3)

Offset	Bit Position																															
0x005C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDRL3																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	ADDRL3	低位地址值 3	MAC 使用此字段过滤所接收的帧	RW

38.4.2.20 ETH_MAC MAC 地址高位寄存器 4 (ETH_MAC_MACADHR4)

Offset	Bit Position																															
0x0060	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name			MBC4						Reserved								ASDDRH4															
	AE4	SA4																														
Access	RW	RW	RW														RW															

位/位域	标记	位名	功能描述	读写
31	AE4	地址使能 4	0b0: 忽略 MAC 地址 4 的地址过滤 0b1: 使用 MAC 地址 4 进行完美过滤	RW
30	SA4	地址选择 4	0b0: 用 MAC 地址 4 对接收帧的 DA 进行过滤 0b1: 用 MAC 地址 4 对接收帧的 SA 进行过滤	RW
29:24	MBC4	字节屏蔽 4	地址过滤时屏蔽对应的字节 - Bit29: 屏蔽 ADDR4[15:8]位 - Bit28: 屏蔽 ADDR4[7:0]位 - Bit27: 屏蔽 ADDRL4[31:24]位 - Bit26: 屏蔽 ADDRL4[23:16]位 - Bit25: 屏蔽 ADDRL4[15:8]位 - Bit24: 屏蔽 ADDRL4[7:0]位	RW
23:16	Reserved	保留	-	-
15:0	ASDDRH4	高位地址值 4	MAC 使用此字段过滤所接收的帧	RW

38.4.2.21 ETH_MAC MAC 地址低位寄存器 4 (ETH_MAC_MACADLR4)

Offset	Bit Position																															
0x0064	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDRL4																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	ADDRL4	低位地址值 4	MAC 使用此字段过滤所接收的帧	RW

38.4.2.22 ETH_MAC MAC Hash 表高位寄存器 (ETH_MAC_HASHTHR)

Offset	Bit Position																															
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	HTH																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	HTH	Hash Table 高位值	Hash Table 的高 32 位值	RW

38.4.2.23 ETH_MAC MAC Hash 表低位寄存器 (ETH_MAC_HASHTLR)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	HTL																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	HTL	Hash Table 低位值	Hash Table 的低 32 位值	RW

38.4.2.24 ETH_MAC VLAN 标记发送控制寄存器 (ETH_MAC_VTACTLR)

Offset	Bit Position																															
0x0584	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														VLANs	VLANc	VLANv															
Access															RW	RW	RW															

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18	VLANs	VLAN 标记选择	0b0: 在发送 VLAN 帧时, 对标记字段的处理由 Tx 描述符的 TDES0 的 VLANc 控制位设定决定 0b1: 在发送 VLAN 帧时, 对标记字段的处理由该寄存器的 VLANc 控制位设定决定	RW
17:16	VLANc	VLAN 标记控制	在发送 VLAN 帧时, 对标记字段的处理: 0b00: 不做操作, 直接发送 0b01: 将 VLAN 帧的 TYPE 字段和标记字段删除 0b10: 在插入 TYPE 字段之后把该寄存器的 VLANv 位的值插入作为标记字段, 不管原帧中是否含有标记字段 0b11: 用该寄存器的 VLANv 位的值替换原帧中的标记字段	RW
15:0	VLANv	VLAN 标记值	用户定义 VLAN 标记字段值, 用于在发送帧时插入或替换标记字段	RW

38.4.2.25 ETH_MAC VLAN 标记接收过滤寄存器 (ETH_MAC_VTAFLTR)

Offset	Bit Position																															
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												VTHM	Reserved	VTIM	VTAL	VLFLT															
Access													RW		RW	RW	RW															

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19	VTHM	VLAN 标记 Hash 表使能	0b0: VLAN 标记 Hash 过滤功能无效 0b1: VLAN 标记 Hash 过滤功能使能	RW
18	Reserved	保留	-	-
17	VTIM	VLAN 标记反转过滤	0b0: VLAN 帧的标记字段与该寄存器 VLFLT 位设定的过滤值不匹配时, 认定为过滤失败 0b1: VLAN 帧的标记字段与该寄存器 VLFLT 位设定的过滤值匹配时, 认定为过滤失败	RW
16	VTAL	12BIT VLAN 比较	0b0: VLAN 标记字段的全 16 位与 VLFLT 设定值比较 0b1: VLAN 标记字段的低 12 位与 VLFLT 设定值比较  说明 该位的设定也适用于 Hash 过滤方式。	RW
15:0	VLFLT	VLAN 标记过滤值	此字段包含用于过滤 VLAN 帧的标记字段的比较值, 该位与接收的 VLAN 帧的第 15 和第 16 字节进行比较	RW

38.4.2.26 ETH_MAC VLAN Hash 表寄存器 (ETH_MAC_VLAHTBR)

Offset	Bit Position																															
0x0588	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																VLHT															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	VLHT	VLAN Hash Table	VLAN Hash Table 值	RW

38.4.2.27 ETH_MAC_LAY3LAY4 控制寄存器 (ETH_MAC_L34CTLR)

Offset	Bit Position																																						
0x0400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x00000000																																						
Name	Reserved												L4DPIM	L4DPM	L4SPIM	L4SPM	Reserved	L4PEN	L3HDBM				L3HSBM				L3DAIM	L3DAM	L3SAIM	L3SAM	Reserved	L3PEN							
Access													RW	RW	RW	RW		RW	RW				RW				L3HSBM												

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21	L4DPIM	L4 目标端口反转过滤	0b0: L4 的目标端口字段与设定的过滤值不匹配时, 认定为过滤失败 0b1: L4 的目标端口字段与设定的过滤值匹配时, 认定为过滤失败  说明 当该寄存器的 L4DPM 位有效时, 该位设定才有效。	RW
20	L4DPM	L4 目标端口过滤	0b0: 忽略 L4 的目标端口字段的过滤 0b1: 对 L4 的目标端口字段进行过滤	RW
19	L4SPIM	L4 源端口反转过滤	0b0: L4 的源端口字段与设定的过滤值不匹配时, 认定为过滤失败 0b1: L4 的源端口字段与设定的过滤值匹配时, 认定为过滤失败  说明 当该寄存器的 L4SPM 位有效时, 该位设定才有效。	RW
18	L4SPM	L4 源端口过滤	0b0: 忽略 L4 的源端口字段的过滤 0b1: 对 L4 的源端口字段进行过滤	RW
17	Reserved	保留	-	-
16	L4PEN	L4 协议使能	0b0: 对 L4 的 TCP 协议帧进行端口过滤 0b1: 对 L4 的 UDP 协议帧进行端口过滤	RW
15:11	L3HDBM	L3IP 目标地址高位屏蔽	对 L3 目标地址过滤时的位屏蔽设定: IPv4 时: 0b00000: 全目标地址过滤 0b00001: 忽略目标地址的最低位 LSB[0]的过滤 0b00010: 忽略目标地址的最低位 LSB[1:0]的过滤 0b00011: 忽略目标地址的最低位 LSB[2:0]的过滤 0b11111: 忽略目标地址的最低位 LSB[30:0]的过滤 IPv6 时: - L3HDBM 的 Bit2~1 作为 L3HSBM 的 Bit7~6	RW

位/位域	标记	位名	功能描述	读写
15:11	L3HDBM	L3IP 目标地址高位屏蔽	 说明 该寄存器的 L3DAM 有效时，该位才有效。	RW
10:6	L3HSBM	L3IP 源地址高位屏蔽	对源地址过滤时的位屏蔽设定： IPv4 时： ● 0b000000：全源地址过滤 ● 0b000001：忽略源地址的最低位 LSB[0]的过滤 ● 0b000010：忽略源地址的最低位 LSB[1:0]的过滤 ● 0b000011：忽略源地址的最低位 LSB[2:0]的过滤 ● ● 0b111111：忽略源地址的最低位 LSB[30:0]的过滤 IPv6 时： ● 0b00000000：全源或目标地址过滤 ● 0b00000001：忽略源或目标地址的最低位 LSB[0]的过滤 ● 0b00000010：忽略源或目标地址的最低位 LSB[1:0]的过滤 ● 0b00000011：忽略源或目标地址的最低位 LSB[2:0]的过滤 ● ● 0b11111111：忽略源或目标地址的最低位 LSB[126:0]的过滤  说明 1. 该寄存器的 L3SAM 或 L3DAM 有效时，该位才有效。 2. 选择 IPv6 功能，且 L3DAM 和 L3SAM 均使能时，源地址和目标地址的过滤控制共用一组过滤地址，且该高位屏蔽控制位也共用。	RW
5	L3DAIM	L3 目标地址反转过滤	● 0b0：L3 的 IP 报的目标地址字段与设定的过滤值不匹配时，认定为过滤失败 ● 0b1：L3 的 IP 报的目标地址字段与设定的过滤值匹配时，认定为过滤失败  说明 当该寄存器的 L3DAM 位有效时，该位设定才有效。	RW
4	L3DAM	L3 目标地址过滤	● 0b0：忽略 L3 的 IP 报的目标地址字段的过滤 ● 0b1：对 L3 的 IP 报的目标地址字段进行过滤	RW
3	L3SAIM	L3 源地址反转过滤	● 0b0：L3 的 IP 报的源地址字段与设定的过滤值不匹配时，认定为过滤失败 ● 0b1：L3 的 IP 报的源地址字段与设定的过滤值匹配时，认定为过滤失败  说明 当该寄存器的 L3SAM 位有效时，该位设定才有效。	RW
2	L3SAM	L3 源地址过滤	● 0b0：忽略 L3 的 IP 报的源地址字段的过滤 ● 0b1：对 L3 的 IP 报的源地址字段进行过滤	RW
1	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
0	L3PEN	L3 协议使能	0b0: 对 L3 的 IPv4 报进行地址过滤 0b1: 对 L3 的 IPv6 报进行地址过滤	RW

38.4.2.28 ETH_MAC LAY4 端口寄存器 (ETH_MAC_L4PORTR)

Offset	Bit Position																															
0x0404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	L4DPVAL																L4SPVAL															
Access	RW																RW															

位/位域	标记	位名	功能描述	读写
31:16	L4DPVAL	目标端口过滤值	-L4 的目标端口过滤值	RW
15:0	L4SPVAL	源端口过滤值	L4 的源端口过滤值	RW

38.4.2.29 ETH_MAC LAY3 地址寄存器 0 (ETH_MAC_L3ADDRR0)

Offset	Bit Position																															
0x0410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	L3ADDR0																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	L3ADDR0	地址过滤值 0	L3 的地址过滤值: - L3PEN=0&L3SAM=1 时, 该寄存器为 IPv4 报的 L3 的源地址过滤值 - L3PEN=1&L3SAM=1 时, 该寄存器为 IPv6 报的 L3 的源地址[31:0]过滤值 - L3PEN=1&L3DAM=1 时, 该寄存器为 IPv6 报的 L3 的目标地址[31:0]位过滤值	RW

38.4.2.30 ETH_MAC LAY3 地址寄存器 1 (ETH_MAC_L3ADDRR1)

Offset	Bit Position																															
0x0414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	L3ADDR1																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	L3ADDR1	地址过滤值 1	L3 的地址过滤值： ● L3PEN=0&L3DAM=1 时，该寄存器为 IPv4 报的 L3 的目标地址过滤值 ● L3PEN=1&L3SAM=1 时，该寄存器为 IPv6 报的 L3 的源地址[63:32]过滤值 ● L3PEN=1&L3DAM=1 时，该寄存器为 IPv6 报的 L3 的目标地址[63:32]位过滤值	RW

38.4.2.31 ETH_MAC LAY3 地址寄存器 2 (ETH_MAC_L3ADDRR2)

Offset	Bit Position																															
0x0418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	L3ADDR2																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	L3ADDR2	地址过滤值 2	L3 的地址过滤值： ● L3PEN=1&L3SAM=1 时，该寄存器为 IPv6 报的 L3 的源地址[95:64]过滤值 ● L3PEN=1&L3DAM=1 时，该寄存器为 IPv6 报的 L3 的目标地址[95:64]位过滤值	RW

38.4.2.32 ETH_MAC LAY3 地址寄存器 3 (ETH_MAC_L3ADDRR3)

Offset	Bit Position																															
0x041C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	L3ADDR3																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	L3ADDR3	地址过滤值 3	L3 的地址过滤值： ● L3PEN=1&L3SAM=1 时，该寄存器为 IPv6 报的 L3 的源地址[127:96]过滤值 ● L3PEN=1&L3DAM=1 时，该寄存器为 IPv6 报的 L3 的目标地址[127:96]位过滤值	RW

38.4.3 ETH_PTP 寄存器

38.4.3.1 ETH_PTP 时间戳控制寄存器 (ETH_PTP_TSPCTLR)

Offset	Bit Position																																						
0x0700	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x00002000																																						
Name	Reserved														TSPADF	TSPMTSEL				TSPOVIPV4	TSPOVIPV6	TSPOVETH	TSPVER	TSPSSR	TSPEALL	Reserved	TSPADUP	TSPINT	TSPUP	TSPINI	TSPUPSEL	TSPEN							
Access															RW	RW				RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18	TSPADF	PTP 帧过滤	PTP 帧直接通过以太网发送时，若该位置 1，则对该 PTP 帧进行目标地址过滤	RW
17:14	TSPMTSEL	PTP 报文类型选择	选择要拍摄时间戳快照的报文类型，具体请参见“ PTP 报文类型 ”	RW
13	TSPOVIPV4	接收 Over UDP-IPv4 封装类型的帧	0b0：忽略 Over UDP-IPv4 封装形式的 PTP 帧 0b1：接收 Over UDP-IPv4 封装形式的 PTP 帧	RW
12	TSPOVIPV6	接收 Over UDP-IPv6 封装类型的帧	0b0：忽略 Over UDP-IPv6 封装形式的 PTP 帧 0b1：接收 Over UDP-IPv6 封装形式的 PTP 帧	RW
11	TSPOVETH	接收 Over Ethernet 封装类型的帧	0b0：忽略 Over Ethernet 封装形式的 PTP 帧 0b1：接收 Over Ethernet 封装形式的 PTP 帧	RW
10	TSPVER	PTP 报文处理版本	0b0：用 IEEE1588v1 版本格式对接收到的 PTP 帧处理 0b1：用 IEEE1588v2 版本格式对接收到的 PTP 帧处理	RW
9	TSPSSR	亚秒计数进制	0b0：亚秒寄存器的值达到 0x7FFFFFFFH，时间戳秒计数器增加 1 秒，亚秒寄存器归零重新计数 0b1：亚秒寄存器的值达到 0x3B9AC9FFH，时间戳秒寄存器增加 1 秒，亚秒寄存器归零重新计数  说明 0x3B9AC9FF=999999999 亚秒增量必须根据 PTP 参考时钟频率和此位的值正确进行编程。	RW
8	TSPEALL	使能所有帧	该位置 1 时，对接收的所有帧都拍摄时间戳快照	RW
7:6	Reserved	保留	-	-
5	TSPADUP	计数寄存器更新	该位置 1 时，时间戳加数寄存器 (ETH_PTP_TSPADDR) 的值会被更新到 PTP 进行精密校准，更新结束时此位会自动清零  说明 该寄存器位必须在读为 0 后，才可以将其置 1。	RW

位/位域	标记	位名	功能描述	读写
4	TSPINT	时间戳中断使能	如果系统时间大于在目标时间寄存器中写入的值，将产生时戳中断  说明 当 PPS0 产生中断 (ETH_PTP_TSPSTSR.TSTAR0 为 1) 后，该位自动清零，时间戳中断功能被禁用。如仍需使用该中断功能，需要在应用程序中将该位置 1。	RW
3	TSPUP	时间戳更新	该位置 1 时，系统时间将通过在时间戳更新秒寄存器 (ETH_PTP_TMUSECR) 和时间戳更新亚秒寄存器 (ETH_PTP_TMUNSER) 中指定的值进行更新 (加入/减去)。更新结束时此位会自动清零  说明 该寄存器位必须在自己和 TSPINI 两位均读为 0 后，才可以将其置 1。	RW
2	TSPINI	时间戳初始化	该位置 1 时，系统时间将通过在时间戳更新秒寄存器 (ETH_PTP_TMUSECR) 和时间戳更新亚秒寄存器 (ETH_PTP_TMUNSER) 中指定的值进行初始化 (覆盖)。初始化结束时此位会自动清零  说明 该寄存器位必须在读为 0 后，才可以将其置 1。	RW
1	TSPUPSEL	时间戳校准方式	● 0b0：粗校准方式 ● 0b1：细校准方式	RW
0	TSPEN	PTP 功能使能	● 0b0：时间戳功能无效，MAC 不会为发送和接收帧添加时间戳 ● 0b1：时间戳功能有效，针对发送和接收帧添加时间戳  说明 该位一旦置 1，需要对系统时间进行初始化。	RW

38.4.3.2 ETH_PTP 时间戳状态寄存器 (ETH_PTP_TSPSTSR)

Offset	Bit Position																															
0x0728	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										TSERR1	TSTAR1	TSERR0	Reserved	TSTAR0	TSOVF
Access																											RC	RC	RC		RC	RC

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	TSERR1	目标时间 1 错误	0b0: 要写入目标时间寄存器 1 的时间正确 0b1: 要写入目标时间寄存器 1 的时间已经过去, 该写入值错误	RC
4	TSTAR1	PPS1 中断标志位	0b0: 未产生中断 0b1: 产生中断 置位条件: - 系统时间达到目标时间寄存器 1 指示的时间 (需使能中断并选择中断输出事件)	RC
3	TSERR0	目标时间 0 错误	0b0: 要写入目标时间寄存器 0 的时间正确 0b1: 要写入目标时间寄存器 0 的时间已经过去, 该写入值错误	RC
2	Reserved	保留	-	-
1	TSTAR0	PPS0 中断标志位	0b0: 未产生中断 0b1: 产生中断 置位条件: - 系统时间达到目标时间寄存器 0 指示的时间 (需使能中断并选择中断输出事件)	RC
0	TSOVF	系统时间溢出	0b0: 系统时间未溢出 0b1: 系统时间溢出	RC

38.4.3.3 ETH_PTP 时间戳基本加数寄存器 (ETH_PTP_TSPADDR)

Offset	Bit Position																															
0x0718	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TSPADD																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TSPADDR	基本加数值	该寄存器由软件来重新线性调节时钟频率，使其与主时钟频率匹配。在每个时钟周期将该寄存器的值添加到 32 位累加器中  说明 该寄存器只在细校准模式下有效。	RW

38.4.3.4 ETH_PTP 时间戳亚秒加数寄存器 (ETH_PTP_TSPNSAR)

Offset	Bit Position																															
0x0704	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TSPNSEADD															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	TSPNSEADD	亚秒加数值	该寄存器包含系统亚秒寄存器递增时使用的 8 位值 - 在粗校准模式下，每个时钟周期都将该寄存器的值添加到系统时间 - 在细校准模式下，只在 32 位累加器溢出时将该寄存器的值添加到系统时间	RW

38.4.3.5 ETH_PTP 时间戳系统秒寄存器 (ETH_PTP_TMSSECR)

[illegible]

位/位域	标记	位名	功能描述	读写
31:0	TSPSYSSEC	系统秒时间	该位表示系统时间的秒时间	RO

38.4.3.6 ETH PTP 时间戳系统亚秒寄存器 (ETH PTP TMSNSR)

[illegible]

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:0	TSPSYNSEC	系统亚秒时间	该位表示系统时间的亚秒时间	RO

38.4.3.7 ETH PTP 时间戳更新秒寄存器 (ETH PTP TMUSECR)

[illegible]

位/位域	标记	位名	功能描述	读写
31:0	TSPUPSEC	系统秒更新值	该位表示要更新进系统时间的秒时间	RW

38.4.3.8 ETH_PTP 时间戳更新亚秒寄存器 (ETH_PTP_TMUNSER)

Offset	Bit Position																															
0x0714	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TSPUPNS	TSPUPNSE																														
Access	RW	RW																														

位/位域	标记	位名	功能描述	读写
31	TSPUPNS	更新方式	时间更新正负号： ● 0b0：从系统时间寄存器的值中减去更新寄存器的值 ● 0b1：在系统时间寄存器的值中加上更新寄存器的值	RW
30:0	TSPUPNSE	系统亚秒更新值	该位表示要更新进系统时间的亚秒时间	RW

38.4.3.9 ETH_PTP 时间戳目标秒寄存器 0 (ETH_PTP_TMTSECR0)

Offset	Bit Position																															
0x071C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TSPTAGSEC0																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TSPTAGSEC0	目标秒时间 0	系统时间超过或等于该寄存器和目标亚秒寄存器 0 的设定值时，发生中断事件或 PPS 事件	RW

38.4.3.10 ETH_PTP 时间戳目标亚秒寄存器 0 (ETH_PTP_TMTNSER0)

Offset	Bit Position																															
0x0720	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	TSPTAGNSEC0																														
Access		RW																														

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:0	TSPTAGNSEC0	目标亚秒时间 0	系统时间超过或等于该寄存器和目标秒寄存器 0 的设定值时，发生中断事件或 PPS 事件	RW

38.4.3.11 ETH_PTP 时间戳目标秒寄存器 1 (ETH_PTP_TMTSECR1)

Offset	Bit Position																															
0x0780	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TSPTAGSEC1																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TSPTAGSEC1	目标秒时间 1	系统时间超过或等于该寄存器和目标亚秒寄存器 1 的设定值时，发生中断事件或 PPS 事件	RW

38.4.3.12 ETH_PTP 时间戳目标亚秒寄存器 1 (ETH_PTP_TMTNSER1)

Offset	Bit Position																															
0x0784	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	TSPTAGNSEC1																														
Access		RW																														

位/位域	标记	位名	功能描述	读写
31	Reserved	保留	-	-
30:0	TSPTAGNSEC1	目标亚秒时间 1	系统时间超过或等于该寄存器和目标秒寄存器 1 的设定值时，发生中断事件或 PPS 事件	RW

38.4.3.13 ETH_PTP PPS 输出控制寄存器 (ETH_PTP_PPSCTRL)

Offset	Bit Position																															
0x072C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TT1SEL	Reserved		PPSFRE1		Reserved		TT0SEL		PPSOMD		PPSFRE0				
Access																	RW			RW				RW		RW		RW				

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14:13	TT1SEL	目标时间 1 功能选择	0b00: 目标寄存器 1 仅用于中断输出事件 0b01: 无意义, 请不要设定 0b10: 目标寄存器 1 用于中断输出事件和 PPS1 单次输出事件 0b11: 目标寄存器 1 仅用于 PPS1 单次输出事件	RW
12:11	Reserved	保留	-	-
10:8	PPSFRE1	PPS1 输出频率	读出恒为 0 写入: 0b001: 当系统时间等于目标秒寄存器 1 时, 产生一次脉冲, 之后不再产生脉冲 - 其他: 禁止设置	RW
7	Reserved	保留	-	-
6:5	TT0SEL	目标时间 0 功能选择	0b00: 目标寄存器 0 仅用于中断输出事件 0b01: 无意义, 请不要设定 0b10: 目标寄存器 0 用于中断输出事件和 PPS0 单次输出事件 0b11: 目标寄存器 0 仅用于 PPS0 单次输出事件	RW
4	PPSOMD	PPS 输出模式	0b0: 连续输出模式 0b1: 单次输出模式  说明 PPS0 支持上述两种模式, PPS1 仅支持单次输出模式。	RW
3:0	PPSFRE0	PPS0 输出频率	连续输出模式时: 当系统时间发生秒进位 (亚秒时间溢出) 时, PPS0 根据下面的设定产生不同形式的输出: 0b0000: 连续输出间隔为 1Hz 的脉冲 0b0001: 连续输出频率为 2Hz 的时钟 0b0010: 连续输出频率为 4Hz 的时钟 0b0011: 连续输出频率为 8Hz 的时钟 0b0100: 连续输出频率为 16Hz 的时钟 0b1111: 连续输出频率为 32768Hz 的时钟	RW

位/位域	标记	位名	功能描述	读写
3:0	PPSFRE0	PPS0 输出频率	说明 连续输出模式时，时间戳控制寄存器的 TSPSSR 位须设定为 0。 单次输出模式时：读出恒为 0 写入： 0b0001：当系统时间等于目标秒寄存器 0 时，产生一次脉冲，之后不再产生脉冲 - 其他：禁止设置	RW

38.4.4 ETH_DMA 寄存器

38.4.4.1 ETH_DMA 总线模式寄存器 (ETH_DMA_BUSMODR)

Offset	Bit Position																															
0x1000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00020101																															
Name	Reserved				TXPR	MBST	AAL	M8PBL	SPBL	RPBL				FBST	PRAT	TPBL				DSEN	DSL				DMAA	SWR						
Access					RW	RW	RW	RW	RW	RW				RW	RW	RW				RW	RW				RW	RW						

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27	TXPR	传送优先级	0b0: RxDMA 事务优先级高于 TxDMA 事务 0b1: TxDMA 事务优先级高于 RxDMA 事务	R/W
26	MBST	混合突发传送	该位置 1 且 FBST 位为 0 时, AHB Master 口将以 INCR 启动所有长度大于 16 的突发传输, 以 SINGLE 和 INCRx 启动长度等于或小于 16 的突发传输 (x=4/8/16)	R/W
25	AAL	地址对齐	该位置 1 时: 1. 若 FBST=1 时, AHB 接口会生成与起始地址 LS 位对齐的所有突发 2. 若 FBST=0 时, 则第一个突发 (访问数据缓冲器的起始地址) 不对齐, 但后续的突发与地址对齐	R/W
24	M8PBL	8 倍 PBL	该位置 1 时, 会将编程的 PBL 值 (RPBL 位和 TPBL 位) 乘以 8 倍, 因此, DMA 根据 PBL 值以最大 8、16、32、64、128 和 256 个节拍传输数据	R/W
23	SPBL	独立 PBL 控制	0b0: 该寄存器的 TPBL 设定用于 TxDMA 事务和 RxDMA 事务 0b1: 该寄存器的 TPBL 设定只用于 TxDMA 事务, RPBL 设定用于 RxDMA 事务	R/W
22:17	RPBL	Rx 可编程突发长度	和该寄存器的 TPBL 位的意义相同, 仅用于 RxDMA 事务  说明 该位只在该寄存器的 SPBL 位有效时才有用。	R/W
16	FBST	固定突发传送	0b0: DMA Master 接口只使用 SINGLE 和 INCR 访问类型 0b1: DMA Master 接口使用 SINGLE 和 INCR4、INCR8、INCR16 访问类型	R/W

位/位域	标记	位名	功能描述	读写
15:14	PRAT	优先级比	Rx 和 Tx 对 BUS 的抢占优先级比： ● 0b00: 1:1 ● 0b11: 2:1 ● 0b10: 3:1 ● 0b11: 4:1  说明 1. 该位只在该寄存器的 DMAA 位为 0 时有效。 2. Tx:Rx 或 Rx:Tx 的选择由该寄存器的 TXPR 位决定。	R/W
13:8	TPBL	可编程突发长度	该位指示要在一个 TxDMA 事务或 RxDMA 事务中传输的最大节拍数，这是在单个块读/写操作中使用的最大值。DMA 每次在主机总线上开始突发传输时，始终尝试按 PBL 中指定的方式进行突发。允许使用值 1、2、4、8、16 和 32 对 PBL 进行编程，任何其它值都会产生未定义的行为  说明 PBL 值有以下限制： 1. 可能的最大节拍数（PBL）受 Tx FIFO 和 Rx FIFO 大小的限制 2. 请不要编程超出范围的 PBL 值	R/W
7	DSEN	增强型描述符	增强型描述符格式使能： ● 0b0: 常规型描述符 ● 0b1: 增强型描述符  说明 在 PTP 功能有效 (ETH_PTP_TSPCTLR.TSPEN=1) 或校验和减荷功能有效 (ETH_MAC_CONFIGR.IPCO=1) 时，必须使用增强型描述符。	R/W
6:2	DSL	描述符跳跃长度	该位指定两个未链接描述符之间跳过的 Word 数，地址从当前描述符结束处开始跳到下一个描述符起始处。当 DSL 值等于零时，在环形模式下，DMA 会将描述符表视为连续的	R/W
1	DMAA	DMA 仲裁	● 0b0: 循环优先级，仲裁方式由该寄存器的 PRAT 位和 TXPR 位决定 ● 0b1: 固定优先级，优先方式由该寄存器的 TXPR 位决定	R/W
0	SWR	软件复位	该位置 1 时，DMA 控制器复位所有的 MAC 子系统的内部寄存器和逻辑。在所有内核时钟域完成复位操作后，该位自动清零	R/W

38.4.4.2 ETH_DMA 操作模式寄存器 (ETH_DMA_OPRMODR)

Offset	Bit Position																																							
0x1018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved				DTCOE		RSF		DFRF		Reserved		TSF		FTF		Reserved		TTC		STT		Reserved				FEF		FUF		DGF		RTC		OSF		STR		Reserved	
Access					RW		RW		RW				RW		RW				RW		RW						RW		RW		RW		RW		RW		RW			

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26	DTCOE	禁止丢弃 TCP/IP 校验和错误帧	0b0: 如果该寄存器的 FEF 位置位 1, 则丢弃所有错误帧 0b1: 如果接收帧中仅存在由接收校验和减荷引擎检测出来的错误, 则内核不会丢弃该帧  说明 这类帧在 MAC 接收到的以太网帧中没有任何错误 (包括 CRC 错误), 而仅在封装的有效负载中有错误。	RW
25	RSF	接收存储转发	0b0: Rx FIFO 中的帧达到该寄存器的 RTC 位指定的值时, 才进行 Rx FIFO 的读动作 0b1: Rx FIFO 写入完整帧后, 就执行读动作, 同时忽略 RTC 位	RW
24	DFRF	禁止刷新接收帧	该位置 1 时, RxDMA 不会因为接收描述符/缓冲区不可用而刷新任何帧, 详细描述请参见“ 接收过程挂起 ”	RW
23:22	Reserved	保留	-	-
21	TSF	发送存储转发	0b0: Tx FIFO 中的帧达到该寄存器的 TTC 位指定的值时, 才进行 Tx FIFO 的读动作, 即 MAC 发送才启动 0b1: Tx FIFO 中有一个完整帧, 则 MAC 发送会启动, 同时忽略 TTC 位  说明 该位只有在已停止传输时才能更改。	RW
20	FTF	刷新 Tx FIFO	该位置 1 时, Tx FIFO 复位为默认值, 此时 Tx FIFO 中的所有数据均会丢失/刷新。刷新操作结束时该位自动清零  说明 此位清零之前不得对工作模式寄存器执行写操作。	RW
19:17	Reserved	保留	-	-
16:14	TTC	发送阈值	该位用于控制 Tx FIFO 的阈值级别。当 Tx FIFO 中的帧大小大于阈值时启动 DMA 发送。此外, 长度小于阈值的完整帧会自动传输: 0b000: 64 0b001: 128 0b010: 192 0b011: 256	RW

位/位域	标记	位名	功能描述	读写
16:14	TTC	发送阈值	0b100: 40 0b101: 32 0b110: 24 0b111: 16  说明 该位只有在 TSF 位为 0 时才使用。	RW
13	STT	发送启动停止	0b0: 发送过程在完成发送当前帧的任务之后进入停止状态, 并保存发送列表中的下一个描述符位置, 该描述符位置在重启发送后会成为当前位置 0b1: 启动发送过程, DMA 检查发送列表中的当前位置, 即发送描述符地址列表寄存器 (ETH_DMA_TXDLADR) 设定的地址, 或者上一次停止发送时的保留位置尝试获取描述符, 来查找待发送的帧 如果当前描述符不属于 TxDMA, 则发送过程会进入挂起状态; 如果该命令在设置 ETH_DMA_TXDLADR 寄存器之前发出, 则 TxDMA 行为无法预知。  说明 启动发送命令只有在传输已停止时才有效; 停止发送命令只有在当前帧发送过程结束或发送过程处于挂起状态时才有效。	RW
12:8	Reserved	保留	-	-
7	FEF	转发错误帧	0b0: Rx FIFO 会丢弃带错误状态 (CRC 错误、冲突错误、巨帧、看门狗超时、上溢) 的帧 如果某个帧的起始字节指针已传输到读控制器端 (阈值模式下), 则不会丢弃该帧; 如果总线上未传输帧的起始字节, 则 Rx FIFO 会丢弃此错误帧 0b1: 除矮帧错误帧之外的所有帧都会转发	RW
6	FUF	转发过小好帧	0b0: Rx FIFO 丢弃所有不足 64 字节的帧, 除非因接收阈值下限更低 (例如 RTC=01) 而导致此类帧已传输 0b1: Rx FIFO 转发包含 PAD 字节和 CRC 的过小帧 (无错误但长度不足 64 字节的帧)	RW
5	DGF	丢弃巨帧	0b0: Rx FIFO 不丢弃巨帧 0b1: Rx FIFO 丢弃巨帧	RW
4:3	RTC	接收阈值	该位用于控制 Rx FIFO 的阈值级别。当 Rx FIFO 中的帧大小大于阈值时启动 DMA 传输 (请求)。此外, 长度小于阈值的完成帧会自动传输: 0b00: 64 0b01: 32 0b10: 96 0b11: 128  说明 该位只有在 RSF 位为 0 时才使用。	RW

位/位域	标记	位名	功能描述	读写
2	OSF	处理第二帧	该位置 1 时会命令 DMA 处理第二个发送数据帧，即使尚未获得首个帧的状态	RW
1	STR	接收启动停止	● 0b0：RxDMA 会在传输当前帧之后停止操作，并保存接收列表中的下一个描述符位置，该描述符位置在接收过程重启后会成为当前位置 ● 0b1：启动接收过程，DMA 尝试从接收列表中的当前位置，即，接收描述符地址列表寄存器（ETH_DMA_RXDLADR）设定的地址，或者上一次停止接收时的保留位置尝试获取描述符，并处理传入帧 如果 RxDMA 未占据任何描述符，则接收过程会进入挂起状态；如果该命令在设置 ETH_DMA_RXDLADR 寄存器之前发出，则 RxDMA 行为无法预知。  说明 启动接收命令只有在已停止接收时才有效；停止接收命令只有在当前接收过程处于运行（等待接收数据包）或挂起状态时才有效。	RW
0	Reserved	保留	-	-

38.4.4.3 ETH_DMA 动作状态寄存器 (ETH_DMA_DMASTSR)

Offset	Bit Position																																
0x1014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved		PTPS	PMTS	MMCS	Reserved		EBUS			TSTS			RSTS		NIS	AIS	ERS	FBS		Reserved		ETS	RWS	RSS	RUS	RIS	UNS	OVS	TJS	TUS	TSS	TIS
Access			RO	RO	RO			RO			RO			RO		RW	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29	PTPS	PTP 事件状态	0b0: PTP 模块无时间戳事件产生 0b1: PTP 模块有时间戳事件发生  说明 置位该位的事件状态清零时该位自动清零。	RO
28	PMTS	PMT 事件状态	0b0: PMT 模块无唤醒事件发生 0b1: PMT 模块有唤醒事件发生  说明 置位该位的事件状态清零时该位自动清零。	RO
27	MMCS	MMC 事件状态	0b0: MMC 模块无帧统计事件发生 0b1: MMC 模块有帧统计事件发生  说明 置位该位的事件状态清零时该位自动清零。	RO
26	Reserved	保留	-	-
25:23	EBUS	错误位状态	该位指示导致总线错误 (AHB 接口上的错误响应) 的错误类型, 不会产生中断 0b000: RxDMA 写数据时发生错误 0b011: TxDMA 读数据时发生错误 0b100: RxDMA 写描述符时发生错误 0b101: TxDMA 写描述符时发生错误 0b110: RxDMA 读描述符时发生错误 0b111: TxDMA 读描述符时发生错误 - 其他: Reserved  说明 该位仅在该寄存器的 FBS 位置 1 时有效。	RO
22:20	TSTS	发送状态	该位指示 TxDMA FSM 的状态, 不会产生中断 0b000: 停止, 发出复位或停止发送命令 0b001: 运行中, 正在获取发送传输描述符 0b010: 运行中, 正在等待状态 0b011: 运行中, 正在读取主机存储器缓冲区中的数据并将其加入发送缓冲区 (Tx FIFO) 队列 0b100: 时间戳写入	RO

位/位域	标记	位名	功能描述	读写
22:20	TSTS	发送状态	● 0b101: Reserved ● 0b110: 挂起, 发送描述符不可用或发送缓冲区下溢 ● 0b111: 运行中, 正在关闭发送描述符	RO
19:17	RSTS	接收状态	该位指示 RxDMA FSM 的状态, 不会产生中断。 ● 0b000: 停止, 发出复位或停止接收命令 ● 0b001: 运行中, 正在获取接收传输描述符 ● 0b010: Reserved ● 0b011: 运行中, 正在等待接收数据包 ● 0b100: 挂起, 接收描述符不可用 ● 0b101: 运行中, 正在关闭接收描述符 ● 0b110: 时间戳写入 ● 0b111: 运行中, 将接收数据包的数据从接收缓冲区传输到主机存储器	RO
16	NIS	普通中断状态汇总	下列任何一个条件满足, 该位就被置 1: ● TIS=1 && ETH_DMA_INTENAR.TIE=1 ● TUS=1 && ETH_DMA_INTENAR.TUE=1 ● RIS=1 && ETH_DMA_INTENAR.RIE=1 ● ERS=1 && ETH_DMA_INTENAR.ERE=1  说明 1. 只有未屏蔽的中断使能位会影响该普通中断状态汇总位。 2. 每当导致该位置 1 的对应位被清零时, 该位也必须清零 (通过向此位写入 1)。	RW
15	AIS	异常中断状态汇总	下列任何一个条件满足, 该位就被置 1: ● TSS=1 && ETH_DMA_INTENAR.TSE=1 ● TJS=1 && ETH_DMA_INTENAR.TJE=1 ● OVS=1 && ETH_DMA_INTENAR.OVE=1 ● UNS=1 && ETH_DMA_INTENAR.UNE=1 ● RUS=1 && ETH_DMA_INTENAR.RUE=1 ● RSS=1 && ETH_DMA_INTENAR.RSE=1 ● RWS=1 && ETH_DMA_INTENAR.RWE=1 ● ETS=1 && ETH_DMA_INTENAR.ETE=1 ● FBS=1 && ETH_DMA_INTENAR.FBE=1  说明 1. 只有未屏蔽的中断使能位会影响该异常中断状态汇总位。 2. 每当导致该位置 1 的对应位被清零时, 该位也必须清零 (通过向此位写入 1)。	RW
14	ERS	提前接收状态	该位指示 DMA 已填满数据包的首个数据缓冲区  说明 可通过向该位写入 1 来清零该位, 也可在该寄存器的 RIS 为置位后自动清零。	RW

位/位域	标记	位名	功能描述	读写
13	FBS	致命总线错误状态	该位指示发生了总线错误，具体错误类型参见该寄存器的 EBUS 位，当该位置 1 后，对应的 DMA 引擎会禁止其所有的总线访问  说明 可通过向该位写入 1 来清零该位。	RW
12:11	Reserved	保留	-	-
10	ETS	提前发送状态	该位指示要发送的帧已完全传输到 Tx FIFO  说明 可通过向该位写入 1 来清零该位。	RW
9	RWS	接收看门狗状态	当接收到的帧的长度大于 2048 个字节时，该位置 1  说明 可通过向该位写入 1 来清零该位。	RW
8	RSS	接收停止状态	当接收过程进入停止状态时，该位置 1  说明 可通过向该位写入 1 来清零该位。	RW
7	RUS	接收缓冲不可用状态	该位指示接收列表中的下一个描述符由主机所拥有，DMA 无法获取，接收过程进入挂起状态 要恢复处理接收描述符，主机应更改描述符的拥有关系，然后发出接收轮询要求命令 如果未发出接收轮询要求命令，则当接收到下一个识别的输入帧时，接收过程会恢复 仅当上一个接收描述符由 DMA 所拥有时，该位置 1  说明 可通过向该位写入 1 来清零该位。	RW
6	RIS	接收状态	该位指示帧接收已完成，特定的帧状态信息已发布在描述符中，接收保持运行状态  说明 可通过向该位写入 1 来清零该位。	RW
5	UNS	发送下溢状态	该位指示在帧发送期间，发送缓冲区发生下溢，发送会进入挂起状态，且描述符中下溢错误 TDES0[1]标志位置 1  说明 可通过向该位写入 1 来清零该位。	RW
4	OVS	接收上溢状态	该位指示在帧接收期间，接收缓冲区发生上溢，如果部分帧已传输到应用，则描述符中上溢错误 RDES0[11]位置 1  说明 可通过向该位写入 1 来清零该位。	RW

位/位域	标记	位名	功能描述	读写
3	TJS	发送 Jabber 超时状态	该位指示发送 Jabber 定时器已过期，这意味着发送器过度有效，发送过程会中止并将其置于停止状态，且描述符中发送 Jabber 超时 TDES0[14]标志位置 1  说明 可通过向该位写入 1 来清零该位。	RW
2	TUS	发送缓冲不可用状态	该位指示发送列表中的下一个描述符由主机所拥有，DMA 无法获取，发送会进入挂起状态 要恢复处理发送描述符，主机应更改描述符的拥有关系，然后发出发送轮询要求命令  说明 可通过向该位写入 1 来清零该位。	RW
1	TSS	发送停止状态	当发送过程进入停止状态时，该位置 1  说明 可通过向该位写入 1 来清零该位。	RW
0	TIS	发送中断状态	该位指示帧发送已完成，特定的帧状态信息已发布在描述符中  说明 可通过向该位写入 1 来清零该位。	RW

38.4.4.4 ETH_DMA 中断使能寄存器 (ETH_DMA_INTENAR)

Offset	Bit Position																																					
0x101C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																	NIE	AIE	ERE	FBE		Reserved		ETE	RWE	RSE	RUE	RIE	UNE	OVE	TJE	TUE	TSE	TIE			
Access																	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	NIE	普通中断使能汇总	0b0: 普通事件中断无效 0b1: 普通事件中断有效  说明 普通事件是指动作状态寄存器 ETH_DMA_DMASTSR 的 Bit0、Bit2、Bit6、Bit14 位所指示的事件。	RW
15	AIE	异常中断使能汇总	0b0: 异常事件中断无效 0b1: 异常事件中断有效  说明 异常事件是指动作状态寄存器 ETH_DMA_DMASTSR 的 Bit1、Bit3、Bit4、Bit5、Bit7、Bit8、Bit9、Bit10、Bit13 位所指示的事件。	RW
14	ERE	提前接收中断使能	0b0: 提前接收中断无效 0b1: 提前接收中断使能  说明 该位在 NIE 位置 1 后有效。	RW
13	FBE	致命总线错误中断使能	0b0: 致命总线错误中断无效 0b1: 致命总线错误中断有效  说明 该位在 AIE 位置 1 后有效。	RW
12:11	Reserved	保留	-	-
10	ETE	提前发送中断使能	0b0: 提前发送中断无效 0b1: 提前发送中断有效  说明 该位在 AIE 位置 1 后有效。	RW
9	RWE	接收看门狗溢出中断使能	0b0: 接收看门狗溢出中断无效 0b1: 接收看门狗溢出中断有效  说明 该位在 AIE 位置 1 后有效。	RW

位/位域	标记	位名	功能描述	读写
8	RSE	接收停止中断使能	● 0b0: 接收停止中断无效 ● 0b1: 接收停止中断有效  说明 该位在 AIE 位置 1 后有效。	RW
7	RUE	接收缓冲不可用中断使能	● 0b0: 接收缓冲不可用中断无效 ● 0b1: 接收缓冲不可用中断有效  说明 该位在 AIE 位置 1 后有效。	RW
6	RIE	接收中断使能	● 0b0: 接收中断无效 ● 0b1: 接收中断有效  说明 该位在 NIE 位置 1 后有效。	RW
5	UNE	发送下溢中断使能	● 0b0: 发送下溢中断无效 ● 0b1: 发送下溢中断有效  说明 该位在 AIE 位置 1 后有效。	RW
4	OVE	接收上溢中断使能	● 0b0: 接收上溢中断无效 ● 0b1: 接收上溢中断有效  说明 该位在 AIE 位置 1 后有效。	RW
3	TJE	发送 Jabber 超时中断使能	● 0b0: 发送 Jabber 超时中断无效 ● 0b1: 发送 Jabber 超时中断有效  说明 该位在 AIE 位置 1 后有效。	RW
2	TUE	发送缓冲不可用中断使能	● 0b0: 发送缓冲不可用中断无效 ● 0b1: 发送缓冲不可用中断有效  说明 该位在 NIE 位置 1 后有效。	RW
1	TSE	发送停止中断使能	● 0b0: 发送停止中断无效 ● 0b1: 发送停止中断有效  说明 该位在 AIE 位置 1 后有效。	RW
0	TIE	发送中断使能	● 0b0: 发送中断无效 ● 0b1: 发送中断有效  说明 该位在 NIE 位置 1 后有效。	RW

38.4.4.5 ETH_DMA 帧丢失统计寄存器 (ETH_DMA_RFRCNTR)

Offset	Bit Position																															
0x1020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved		OVFOVF	OVFCNT										UNAOVF	UNACNT																	
Access			RO	RC										RO	RC																	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28	OVFOVF	上溢丢失帧计数溢出	该寄存器的 OVFCNT 位统计丢失帧计数器上溢时，该位置 1  说明 该位在读出该寄存器 OVFCNT 位后自动清零。	RO
27:17	OVFCNT	上溢丢失帧统计	该位指示因 Rx FIFO 上溢情况以及帧长度过短（不足 64 字节的好帧）而丢失的帧数量	RC
16	UNAOVF	缓冲不可用丢失帧计数溢出	该寄存器的 UNACNT 位统计丢失帧计数器上溢时，该位置 1  说明 该位在读出该寄存器 UNACNT 位后自动清零。	RO
15:0	UNACNT	缓冲不可用丢失帧统计	该位指示因缓冲区不可用（无可用的接收描述符）而丢失的帧数量	RC

38.4.4.6 ETH_DMA 看门狗定时寄存器 (ETH_DMA_REVWDTR)

Offset	Bit Position																															
0x1024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								RIWT							
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	RIWT	接收看门狗计数	该位指示系统时钟周期数乘以 256 之后的时间值，即看门狗定时器的设定时间值 看门狗定时器会在 RxDMA 结束帧传输之后由编程设定的值触发，此时，动作状态寄存器 ETH_DMA_DMASTR 的 RIS 位因相对应的描述符中 RDES1.RIS 位置 1 而未被置 1，直到看门狗定时器计时结束时，RIS 位才置 1 且看门狗定时器停止。当动作状态寄存器 ETH_DMA_DMASTR 的 RIS 位设为高电平时，看门狗定时器复位	RW

38.4.4.7 ETH_DMA 发送轮询要求寄存器 (ETH_DMA_TXPOLLR)

Offset	Bit Position																															
0x1004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TXPOLL																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TXPOLL	发送轮询要求发起	向该位写入任何值时，DMA 都会读取 ETH_DMA_CHTXDER 寄存器指向的当前描述符，如果该描述符不可用（由主机所有），则 TxDMA 会返回到挂起状态，并将 ETH_DMA_DMASTSRTUS 位进行置位；如果该描述符可用，则发送会继续进行 使用该寄存器来指示 TxDMA 轮询发送描述符列表，检查当前描述符是否为 DMA 所有。如果 TxDMA 处于挂起模式，则发出发送轮询要求命令将其唤醒；如果发送帧中出现下溢错误或 TxDMA 所拥有的描述符不可用，则 TxDMA 会进入挂起模式 用户可以随时发出此命令，当该命令开始重新获取主机存储器的当前描述符后，TxDMA 即会对其进行复位  说明 该位读出恒为 0。	RW

38.4.4.8 ETH_DMA 接收轮询要求寄存器 (ETH_DMA_RXPOLLR)

Offset	Bit Position																															
0x1008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RX POLL																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	RXPOLL	接收轮询要求发起	向该位写入任何值时，DMA 都会读取 ETH_DMA_CHRXDER 寄存器指向的当前描述符，如果该描述符不可用（由主机所有），则 RxDMA 会返回到挂起状态，且不会将 ETH_DMA_DMASTSR 寄存器 RUS 位进行置位；如果该描述符可用，则 RxDMA 将返回到活动状态 使用该寄存器来指示 RxDMA 轮询接收描述符列表，检查新描述符，用于将 RxDMA 从挂起状态唤醒，仅当 RxDMA 所拥有的描述符不可用时，它才会进入挂起状态  说明 该位读出恒为 0。	RW

38.4.4.9 ETH_DMA 发送描述符列表地址寄存器 (ETH_DMA_TXDLADR)

Offset	Bit Position																															
0x1010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TXDLAD																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TXDLAD	发送描述符首地址	该位表示发送描述符列表中的首个描述符的地址  说明 最低 2 位读出恒为 0。	RW

38.4.4.10 ETH_DMA 接收描述符列表地址寄存器 (ETH_DMA_RXDLADR)

Offset	Bit Position																															
0x100C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RXDLAD																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	RXDLAD	接收描述符首地址	该位表示接收描述符列表中的首个描述符的地址  说明 最低 2 位读出恒为 0。	RW

38.4.4.11 ETH_DMA 当前主机发送描述符寄存器 (ETH_DMA_CHTXDER)

Offset	Bit Position																															
0x1048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHTXDE																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	CHTXDE	发送描述符地址指针	该位指向 DMA 所读取的当前发送描述符的起始地址	RO

38.4.4.12 ETH DMA 当前主机接收描述符寄存器 (ETH DMA CHRXDER)

[illegible]

位/位域	标记	位名	功能描述	读写
31:0	CHRXDE	接收描述符地址指针	该位指向 DMA 所读取的当前接收描述符的起始地址	RO

38.4.4.13 ETH DMA 当前主机发送缓冲区寄存器 (ETH DMA CHTXBFR)

Offset	Bit Position																															
0x1050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RO CHTXBF																															
Access																																

位/位域	标记	位名	功能描述	读写
31:0	CHTXBF	发送缓冲区地址指针	该位指向 DMA 所读取的当前发送缓冲区地址	RO

38.4.4.14 ETH DMA 当前主机接收缓冲区寄存器 (ETH DMA CHRXBFR)

Offset	Bit Position																															
0x1054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CHRxBF																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	CHRXBF	接收缓冲区地址指针	该位指向 DMA 所读取的当前接收缓冲区地址	RO

38.4.5 ETH_MMC 寄存器

38.4.5.1 ETH_MMC MMC 控制寄存器 (ETH_MMC_MMCCTLR)

Offset	Bit Position																																					
0x0100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																										MCPSEL		MCPSET		MCF		ROR		COS		CRST	
Access																											RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	MCPSEL	计数器预设值选择	0b0: 将所有 MMC 统计寄存器的值预设“预设半值” (0x7FF0) 0b1: 将所有 MMC 统计寄存器的值预设“预设全值” (0xFFF0)  说明 - 预设全值是指 0x0000-0x0010=0xFFF0。 - 预设半值是指 0x8000-0x0010=0x7FF0。	RW
4	MCPSET	计数器预设值设定	该位置 1 后, 根据该寄存器的 MCPSEL 位的设定将所有的 MMC 统计寄存器预设全值或半值  说明 该位在置位后 1 个时钟后自动清零。	RW
3	MCF	计数器冻结	该位置 1 后, 所有 MMC 统计计数器的值被冻结, 不再因为接收或发送帧而变化	RW
2	ROR	计数器读复位	该位置 1 后, 所有 MMC 统计计数器在被读取之后便复位为零  说明 - 不管该寄存器的 MCF 位是否有效, 该位都可有效。 - 读取统计计数器的低 8 位时, 即视为读动作。	RW
1	COS	计数器计满停止	0b0: 计数器计到 FFFFH 后, 归零, 重新计数 0b1: 计数器计到 FFFFH 后, 停止, 不再计数	RW
0	CRST	计数器复位	置 1 后, 所有 MMC 统计计数器的值复位  说明 该位在置位后 1 个时钟后自动清零。	RW

38.4.5.2 ETH_MMC Rx 统计状态寄存器 (ETH_MMC_REVSTSR)

Offset	Bit Position																																											
0x0104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
Reset	0x00000000																																											
Name	Reserved												RO	RXOEIS	RO	RXLEIS	RO	RXUGIS	Reserved												RO	RXREIS	RO	RXAEIS	RO	RXCEIS	RO	RXMGIS	RO	RXBGIS	Reserved			
Access													RO		RO		RO														RO		RO		RO		RO		RO					

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19	RXOEIS	超范围错误帧统计状态	0b0: 接收到的超范围错误帧数未到半值或全值 0b1: 接收到的超范围错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
18	RXLEIS	长度错误帧统计状态	0b0: 接收到的长度错误帧数未到半值或全值 0b1: 接收到的长度错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
17	RXUGIS	单播好帧统计状态	0b0: 接收到的单播好帧数未到半值或全值 0b1: 接收到的单播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
16:8	Reserved	保留	-	-
7	RXREIS	矮帧错误帧统计状态	0b0: 接收到的矮帧错误帧数未到半值或全值 0b1: 接收到的矮帧错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
6	RXAEIS	对齐错误帧统计状态	0b0: 接收到的对齐错误帧数未到半值或全值 0b1: 接收到的对齐错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
5	RXCEIS	CRC 错误帧统计状态	0b0: 接收到的 CRC 错误帧数未到半值或全值 0b1: 接收到的 CRC 错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO

位/位域	标记	位名	功能描述	读写
4	RXMGIS	多播好帧统计状态	● 0b0: 接收到的多播好帧数未到半值或全值 ● 0b1: 接收到的多播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
3	RXBGIS	广播好帧统计状态	● 0b0: 接收到的广播好帧数未到半值或全值 ● 0b1: 接收到的广播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
2:0	Reserved	保留	-	-

38.4.5.3 ETH_MMC Tx 统计状态寄存器 (ETH_MMC_TRSSTSRR)

Offset	Bit Position																															
0x0108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								TXEDEIS	TXUGIS	Reserved	TXCAEIS	TXECEIS	TXLCEIS	TXDEEIS	Reserved								TXMGIS	TXBGIS	Reserved						
Access									RO	RO		RO	RO	RO	RO									RO	RO							

位/位域	标记	位名	功能描述	读写
31:23	Reserved	保留	-	-
22	TXEDES	过度延迟错误帧统计状态	0b0: 传送的过度延迟错误帧数未到半值或全值 0b1: 传送的过度延迟错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
21	TXUGIS	单播好帧统计状态	0b0: 传送的单播好帧数未到半值或全值 0b1: 传送的单播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
20	Reserved	保留	-	-
19	TXCAEIS	载波错误帧统计状态	0b0: 传送的载波错误帧数未到半值或全值 0b1: 传送的载波错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
18	TXECEIS	过度冲突错误帧统计状态	0b0: 传送的过度冲突错误帧数未到半值或全值 0b1: 传送的过度冲突错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
17	TXLCEIS	延迟冲突错误帧统计状态	0b0: 传送的延迟冲突错误帧数未到半值或全值 0b1: 传送的延迟冲突错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
16	TXDEEIS	延迟错误帧统计状态	0b0: 传送的延迟错误帧数未到半值或全值 0b1: 传送的延迟错误帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
15:4	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
3	TXMGIS	多播好帧统计状态	● 0b0: 传送的多播好帧数未到半值或全值 ● 0b1: 传送的多播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
2	TXBGIS	广播好帧统计状态	● 0b0: 传送的广播好帧数未到半值或全值 ● 0b1: 传送的广播好帧数达到半值或全值  说明 该位在读出对应 MMC 统计寄存器后自动清零。	RO
1:0	Reserved	保留	-	-

38.4.5.4 ETH_MMC Rx 中断控制寄存器 (ETH_MMC_RITCTLR)

Offset	Bit Position																																				
0x010C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved												RXOEIM	RXLEIM	RXUGIM	Reserved												RXREIM	RXAEIM	RXCEIM	RXMGIM	RXBGIM	Reserved				
Access													RW	RW	RW													RW	RW	RW	RW	RW					

位/位域	标记	位名	功能描述	读写
31:20	Reserved	保留	-	-
19	RXOEIM	超范围错误帧中断屏蔽	0b0: RXOEIS 置位时, 超范围错误帧 MMC 统计中断不屏蔽 0b1: RXOEIS 置位时, 超范围错误帧 MMC 统计中断屏蔽	RW
18	RXLEIM	长度错误帧中断屏蔽	0b0: RXLEIS 置位时, 长度错误帧 MMC 统计中断不屏蔽 0b1: RXLEIS 置位时, 长度错误帧 MMC 统计中断屏蔽	RW
17	RXUGIM	单播好帧中断屏蔽	0b0: RXUGIS 置位时, 单播好帧 MMC 统计中断不屏蔽 0b1: RXUGIS 置位时, 单播好帧 MMC 统计中断屏蔽	RW
16:8	Reserved	保留	-	-
7	RXREIM	矮帧错误帧中断屏蔽	0b0: RXREIS 置位时, 矮帧错误帧 MMC 统计中断不屏蔽 0b1: RXREIS 置位时, 矮帧错误帧 MMC 统计中断屏蔽	RW
6	RXAEIM	对齐错误帧中断屏蔽	0b0: RXAEIS 置位时, 对齐错误帧 MMC 统计中断不屏蔽 0b1: RXAEIS 置位时, 对齐错误帧 MMC 统计中断屏蔽	RW
5	RXCEIM	CRC 错误帧中断屏蔽	0b0: RXCEIS 置位时, CRC 错误帧 MMC 统计中断不屏蔽 0b1: RXCEIS 置位时, CRC 错误帧 MMC 统计中断屏蔽	RW
4	RXMGIM	多播好帧中断屏蔽	0b0: RXMGIS 置位时, 多播好帧 MMC 统计中断不屏蔽 0b1: RXMGIS 置位时, 多播好帧 MMC 统计中断屏蔽	RW
3	RXBGIM	广播好帧中断屏蔽	0b0: RXBGIS 置位时, 广播好帧 MMC 统计中断不屏蔽 0b1: RXBGIS 置位时, 广播好帧 MMC 统计中断屏蔽	RW
2:0	Reserved	保留	-	-

38.4.5.5 ETH_MMC Tx 中断控制寄存器 (ETH_MMC_TITCLR)

Offset	Bit Position																															
0x0110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								TXEDEIM	TXUGIM	Reserved	TXCAEIM	TXECEIM	TXLCEIM	TXDEEIM	Reserved								TXMGIM	TXBGIM	Reserved						
Access									RW	RW		RW	RW	RW	RW									RW	RW							

位/位域	标记	位名	功能描述	读写
31:23	Reserved	保留	-	-
22	TXEDEM	过度延迟错误帧中断屏蔽	0b0: TXEDEM 置位时, 过度延迟错误帧 MMC 统计中断不屏蔽 0b1: TXEDEM 置位时, 过度延迟错误帧 MMC 统计中断屏蔽	RW
21	TXUGIM	单播好帧中断屏蔽	0b0: TXUGIS 置位时, 单播好帧 MMC 统计中断不屏蔽 0b1: TXUGIS 置位时, 单播好帧 MMC 统计中断屏蔽	RW
20	Reserved	保留	-	-
19	TXCAEIM	载波错误帧中断屏蔽	0b0: TXCAEIS 置位时, 载波错误帧 MMC 统计中断不屏蔽 0b1: TXCAEIS 置位时, 载波错误帧 MMC 统计中断屏蔽	RW
18	TXECEIM	过度冲突错误帧中断屏蔽	0b0: TXECEIS 置位时, 过度冲突错误帧 MMC 统计中断不屏蔽 0b1: TXECEIS 置位时, 过度冲突错误帧 MMC 统计中断屏蔽	RW
17	TXLCEIM	延迟冲突错误帧中断屏蔽	0b0: TXLCEIS 置位时, 延迟冲突错误帧 MMC 统计中断不屏蔽 0b1: TXLCEIS 置位时, 延迟冲突错误帧 MMC 统计中断屏蔽	RW
16	TXDEEIM	延迟错误帧中断屏蔽	0b0: TXDEEIS 置位时, 延迟错误帧 MMC 统计中断不屏蔽 0b1: TXDEEIS 置位时, 延迟错误帧 MMC 统计中断屏蔽	RW
15:4	Reserved	保留	-	-
3	TXMGIM	多播好帧中断屏蔽	0b0: TXMGIS 置位时, 多播好帧 MMC 统计中断不屏蔽 0b1: TXMGIS 置位时, 多播好帧 MMC 统计中断屏蔽	RW
2	TXBGIM	广播好帧中断屏蔽	0b0: TXBGIS 置位时, 广播好帧 MMC 统计中断不屏蔽 0b1: TXBGIS 置位时, 广播好帧 MMC 统计中断屏蔽	RW
1:0	Reserved	保留	-	-

38.4.5.6 ETH_MMC Rx 单播好帧统计寄存器 (ETH_MMC_RXUNGFR)

Offset	Bit Position																															
0x01C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXUNCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXUNCNT	单播好帧数	接收到的单播好帧的数量	RO

38.4.5.7 ETH_MMC Rx 多播好帧统计寄存器 (ETH_MMC_RXMUGFR)

Offset	Bit Position																															
0x0190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXMUGCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXMUGCNT	多播好帧数	接收到的多播好帧的数量	RO

38.4.5.8 ETH_MMC Rx 广播好帧统计寄存器 (ETH_MMC_RXBRGFR)

Offset	Bit Position																															
0x018C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXBRCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXBRGCNT	广播好帧数	接收到的广播好帧的数量	RO

38.4.5.9 ETH_MMC Rx CRC 错误帧统计寄存器 (ETH_MMC_RXCREFR)

Offset	Bit Position																															
0x0194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXCRECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXCRECNT	CRC 错误帧数	接收到的 CRC 错误帧的数量	RO

38.4.5.10 ETH_MMC Rx 对齐错误帧统计寄存器 (ETH_MMC_RXALEFR)

Offset	Bit Position																															
0x0198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXALECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXALECNT	对齐错误帧数	接收到的对齐错误帧的数量	RO

38.4.5.11 ETH_MMC Rx 矮帧错误帧统计寄存器 (ETH_MMC_RXRUEFR)

Offset	Bit Position																															
0x019C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXRUECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXRUECNT	矮帧错误帧数	接收到的矮帧错误帧的数量	RO

38.4.5.12 ETH_MMC Rx 长度错误帧统计寄存器 (ETH_MMC_RXLEEFR)

Offset	Bit Position																															
0x01C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXLEECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXLEECNT	长度错误帧数	接收到的长度错误帧的数量	RO

38.4.5.13 ETH_MMC Rx 超范围错误帧统计寄存器 (ETH_MMC_RXOREFR)

Offset	Bit Position																															
0x01CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RXORECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	RXORECNT	超范围错误帧数	接收到的超范围错误帧的数量	RO

38.4.5.14 ETH_MMC Tx 单播好帧统计寄存器 (ETH_MMC_TXUNGFR)

Offset	Bit Position																															
0x0168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXUNCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXUNCNT	单播好帧数	已发送的单播好帧的数量	RO

38.4.5.15 ETH_MMC Tx 多播好帧统计寄存器 (ETH_MMC_TXMUGFR)

Offset	Bit Position																															
0x0120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXMUGCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXMUGCNT	多播好帧数	已发送的多播好帧的数量	RO

38.4.5.16 ETH_MMC Tx 广播好帧统计寄存器 (ETH_MMC_TXBRGFR)

Offset	Bit Position																															
0x011C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXBRCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXBRGCNT	广播好帧数	已发送的广播好帧的数量	RO

38.4.5.17 ETH_MMC Tx 延迟错误帧统计寄存器 (ETH_MMC_TXDEEFR)

Offset	Bit Position																															
0x0154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXDEECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXDEECNT	延迟错误帧数	已发送的延迟错误帧的数量	RO

38.4.5.18 ETH_MMC Tx 载波错误帧统计寄存器 (ETH_MMC_TXCAEFR)

Offset	Bit Position																															
0x0160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXCAECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXCAECNT	载波错误帧数	已发送的载波错误帧的数量	RO

38.4.5.19 ETH_MMC Tx 延迟冲突错误帧统计寄存器 (ETH_MMC_TXLCEFR)

Offset	Bit Position																															
0x0158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXLCECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXLCECNT	延迟冲突错误帧数	已发送的延迟冲突错误帧的数量	RO

38.4.5.20 ETH_MMC Tx 过度冲突错误帧统计寄存器 (ETH_MMC_TXECEFR)

Offset	Bit Position																															
0x015C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXECCNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXECECNT	过度冲突错误帧数	已发送的过度冲突错误帧的数量	RO

38.4.5.21 ETH_MMC Tx 过度延迟错误帧统计寄存器 (ETH_MMC_TXEDEFRR)

Offset	Bit Position																															
0x016C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																TXEDECNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	TXEDECNT	过度延迟错误帧数	已发送的过度延迟错误帧的数量	RO

39 SDIO 控制器 (SDIOC)

39.1 简介

SDIOC 提供了一个 SD 主机接口和一个 MMC 主机接口，用于和支持 SD2.0 协议的 SD 卡，SDIO 设备以及支持 eMMC4.2 协议的 MMC 设备进行通信。本产品带有 2 个 SDIO 控制器，能够同时与 2 个 SD/MMC/SDIO 设备进行通信。

39.2 主要特性

- 支持 SDSC，SDHC，SDXC 格式 SD 卡及 SDIO 设备
- 支持一线式 (1-bit) 和四线式 (4-bit) SD 总线
- 支持一线式 (1-bit)，四线式 (4-bit) 和八线式 (8-bit) MMC 总线
- SD 时钟最高 50MHz
- 具有卡识别和硬件写保护功能

39.3 功能说明

39.3.1 端口分配

表 39-1 端口分配表

端口名	IO	功能
SDIOx_CK	O	SD 时钟输出信号
SDIOx_CMD	I/O	SD 命令和应答信号
SDIOx_D0	I/O	SD 数据信号
SDIOx_D1	I/O	SD 数据信号
SDIOx_D2	I/O	SD 数据信号
SDIOx_D3	I/O	SD 数据信号
SDIOx_D4	I/O	SD 数据信号
SDIOx_D5	I/O	SD 数据信号
SDIOx_D6	I/O	SD 数据信号
SDIOx_D7	I/O	SD 数据信号
SDIOx_CD	I	SD 卡识别状态信号
SDIOx_WP	I	SD 写保护状态信号

 **说明**
x 表示单元编号，x 为 1~2，本章节后文提到 x 时均指单元编号，不再赘述。

上述端口根据 SD2.0 协议和 eMMC4.51 协议，SD 时钟输出信号(SDIOx_CK)用于在 SDIOC 和 SD/MMC 设备通信时输出 SD 时钟；SD 命令和应答信号(SDIOx_CMD)用于向设备输出 SD/MMC 命令以及接收设备发回的应答信息；SD 数据信号(SDIOx_Dy(y=0~7))用于 SDIOC 和设备在通信过程中收发数据，当采用一线式(1-bit)通信时，只有 SDIOx_D0 有效，SDIOx_Dy(y=1~7)保持高电平，采用四线式(4-bit)通信时，SDIOx_Dy(y=0~3)有效，SDIOx_Dy(y=4~7)保持高电平，采用八线式(8-bit)通信时，SDIOx_Dy(y=0~7)有效，八线式(8-bit)通信仅限于 MMC 设备通信。

39.3.2 基本访问方式

用户通过读写 SDIOC 寄存器的方式启动 SDIOC 同片外 SD/MMC 设备进行通信。由于写命令寄存器将触发 SDIOC 的发送命令动作，因此写命令寄存器必须最后进行。在此之前，用户需要通过传输模式寄存器 (TRANSMODE) 设定传输方式，通过参数寄存器 (ARG0, ARG1) 设定命令参数，并在设置命令寄存器 (CMD) 的时候同时设置正确的命令的编号 (command index)，类型，应答 (response) 类型等。一旦写命令寄存器执行后，SDIOC 将发送命令后，用户可以通过读取中断状态寄存器 (NORINTST, ERRINTST) 来查询是否接收完应答信息，有无错误信息等。当一个命令执行完成后，用户可以通过读取应答寄存器 (RESP0~7) 得到命令的应答。

39.3.3 数据传输

SDIOC 的数据缓冲寄存器用于 CPU/DMA 等主机设备和 SD 设备之间进行数据交换。SDIOC 内置了 FIFO 以加快数据交换的速度。当命令包含数据发送时，用户将数据依次写入数据缓冲寄存器 (BUF0, BUF1)，这些

数据将首先缓存在 SDIOC 的 FIFO 中，当写入的数据数量达到数据长度寄存器 (BLKSIZE) 的设定值或 512 字节时，SDIOC 将通过 SDIOx_Dy (y=0~7) 发送数据。在此期间 SDIOC 通过 FIFO 能够继续向数据缓冲寄存器写入数据。同样地，当命令包含数据接收时，SDIOC 通过 SDIOx_Dy (y=0~7) 接收数据，并将数据缓存在 SDIOC 的缓冲器 (BUFFER) 中，用户通过读取数据缓冲寄存器 (BUF0, BUF1) 得到数据。数据的格式请参考 SD 和 MMC 协议。

39.3.4 SD 时钟

SD 时钟 (SDIOx_CK) 由主机生成并输出给设备用于两者间的通信。SD 时钟由 SDIOC 模块的动作时钟 (PCLK1) 经由分频器分频产生。分频器的分频系数由时钟控制寄存器 (CLKCON) 设定，根据 SD2.0 协议的要求，在数据传输模式 (data transfer mode) 下最快应为 50MHz，因此用户需要根据 PCLK1 的频率设定合理的分频系数。

39.3.5 中断和 DMA 启动请求

SD 中断

SDIOC 提供两类中断，普通中断 (normal interrupt) 和错误中断 (error interrupt)。普通中断是 SDIOC 和 SD/MMC 卡通信的过程中产生各种事件时发生的中断。错误中断是 SDIOC 动作通信过程中发生各种错误时产生的中断。普通中断和错误中断分别具有各自的中断状态寄存器，中断状态使能寄存器，以及中断信号使能寄存器。中断状态寄存器用于表示中断产生的要因，中断状态使能寄存器用于使能中断状态寄存器的各个状态位，中断状态寄存器的各个状态位需要在中断状态使能寄存器使能位使能时有效，中断信号使能寄存器用于是否允许各个中断要因向 CPU 申请中断。

SDIO 中断

SDIO 设备在传输空闲时可以向主机发送卡中断 (card interrupt) 请求。使用 SDIO 中断时需要将中断状态使能寄存器 (NORINTSTEN) 的卡中断 (CINTEN) 使能，如果需要向 CPU 申请中断，还需将中断信号使能寄存器 (NORINTSGEN) 的卡中断 (CINTEN) 使能。SDIO 设备通过将 SDIOx_D1 数据线拉低向主机申请中断。SDIOC 检测到 SDIOx_D1 被拉低后将中断状态寄存器 (NORINTST) 的卡中断 (CINT) 置位，并根据设定向 CPU 申请中断。SDIO 设备判断中断处理完成后将释放 SDIOx_D1。

DMA 请求

SDIOC 在通信中的读写数据能够通过 DMA 来完成。使用 DMA 传送向设备写数据时，将 DMA 的一个通道的启动源设置为 SDIOC 的写请求，再将 DMA 的传输目标地址设置为数据缓冲寄存器 (BUF0, BUF1) 且为固定地址。当 SDIOC 发送完写数据命令后，如果此时数据 FIFO 空，则向 DMA 发出写请求信号，启动 DMA 向数据缓冲寄存器 (BUF0, BUF1) 写入数据，这些数据将依次进入数据 FIFO 中。当 FIFO 中数据达到数据长度寄存器 (BLKSIZE) 的设定值或 512 字节时，SDIOC 将通过 SDIOx_Dy (y=0~7) 发送数据。如果是多块写命令，SDIOC 将同时继续向 DMA 发送启动请求写入数据。使用 DMA 传送从设备读数据时，将 DMA 的另一个通道的启动源设置为 SDIOC 的读请求，再将 DMA 的传输源地址设置为数据缓冲寄存器 (BUF0, BUF1) 且为固定地址。当 SDIOC 发送完读数据命令后，设备将通过 SDIOx_Dy (y=0~7) 发送数据，SDIOC 的数据 FIFO 开始接收数据，当 FIFO 中数据达到数据长度寄存器 (BLKSIZE) 的设定值或 512 字节时，SDIOC 向 DMA 发出读请求信号，启动 DMA 从数据缓冲寄存器 (BUF0, BUF1) 读出数据。如果是多块读命令，SDIOC 将同时继续从设备读取下一个数据块。

卡的插入 (insert) 和移除 (removal)

SD/MMC/SDIO 设备的插入 (insert) 和移除 (removal) 由 SDIOx_CD 信号线识别。当卡槽内无设备时，卡槽会通过电阻将 SDIOx_CD 信号拉高。当有设备插入后，SDIOx_CD 信号将被拉低，设备被移除后将重新变为高。SDIOC 通过 SDIOx_CD 的电平来判断是否有设备。用户可以通过读取主机状态寄存器 (PSTAT) 的 CDL 位来判断是否有设备插入。SDIOC 在设备的插入和移除时能够产生相应的中断，通过中断状态使能寄存器 (NORINTSTEN) 和中断信号使能寄存器 (NORINTSGEN) 来使能。

39.4 操作示例

39.4.1 主机和设备初始化

39.4.1.1 主机初始化

背景信息

SDIOC 在使用时需要首先进行初始化设置。

操作步骤

- 步骤 1. 读取主机状态寄存器 (PSTAT)，查询时钟状态以及设备插入状态
- 步骤 2. 配置电源控制寄存器 (PWRCON)，使能 SDIOC
- 步骤 3. 配置时钟控制寄存器 (CLKCON)，使 SDIOC 输出 SD 时钟，并根据 PCLK1 的频率配置 SD 时钟分频以保证卡识别模式下 SD 时钟频率不超过 400kHz
- 步骤 4. 配置主机控制寄存器 (HOSTCON)，选择一线式 (1-bit) 模式并禁用高速模式 (high speed mode)
- 步骤 5. 根据设备特性配置超时控制寄存器 (TOUTCON)，使得主机在通信超时结束通信并报告错误
- 步骤 6. 配置普通和错误中断状态使能寄存器，中断信号使能寄存器以使 SDIOC 在需要时发生中断并将标志位置起

39.4.1.2 SD 卡初始化

背景信息

完成 SDIOC 初始化配置后，如果连接了 SD 卡，需要按照 SD 协议进行初始化。

操作步骤

- 步骤 1. 卡复位，向设备发送复位命令 CMD0，CMD0 没有应答信息 (response)，此时卡将进入卡识别模式
- 步骤 2. 确认卡的工作状态 (operation condition)，向设备发送 CMD8 并等待接收应答信息 (response)
- 步骤 3. 发送初始化命令 ACMD41 (先发送 CMD55，再发送 CMD41)，根据应答信息判断设备是否完成初始化，否则持续发送 ACMD41 直到初始化完成
- 步骤 4. 发送 CMD2 得到卡的 CID 信息
- 步骤 5. 发送 CMD3 得到卡的 RCA 信息

操作结果

此时卡将进入数据传输模式，初始化完成。

39.4.1.3 MMC 卡初始化

背景信息

完成 SDIOC 初始化配置后，如果连接了 MMC 卡，需要按照 MMC 协议进行初始化。

操作步骤

- 步骤 1. 卡复位，向设备发送复位命令 CMD0，CMD0 没有应答信息 (response)，此时卡将进入卡识别模式
- 步骤 2. 确认卡的工作状态 (operation condition)，向设备发送 CMD1 并等待接收应答信息 (response)，根据应答信息判断设备是否完成初始化，否则持续发送 CMD1 直到初始化完成
- 步骤 3. 发送 CMD2 得到卡的 CID 信息
- 步骤 4. 发送 CMD3 得到卡的 RCA 信息

操作结果

此时卡将进入数据传输模式，初始化完成。

39.4.1.4 SDIO 初始化

背景信息

成 SDIOC 初始化配置后，如果连接了 SDIO 设备，需要按照 SDIO 协议进行初始化。

操作步骤

- 步骤 1. 设备复位，向设备发送复位命令 CMD0，CMD0 没有应答信息 (response)
- 步骤 2. 确认 SDIO 设备的工作状态 (operation condition)，向设备发送 CMD5 并等待接收应答信息 (response)
- 步骤 3. 发送 CMD3 得到设备的 RCA 信息，初始化完成

39.4.2 SD/MMC 单数据块 (single block) 读写

背景信息

在 SD/MMC 卡进入数据传输模式后，SDIOC 就能够通过读写命令访问 SD/MMC 卡的数据。其中单数据块 (block) 读写的顺序如下：

操作步骤

- 步骤 1. 配置钟控制寄存器 (CLKCON)，使 SDIOC 输出 SD 时钟，并根据 PCLK1 的频率配置 SD 时钟分频使 SD 时钟频率不超过默认速度模式下的最大时钟速度 (default speed mode, $f_{pp} \leq 25\text{MHz}$)。
- 步骤 2. 发送 CMD7，SD/MMC 卡将进入数据传输状态。
- 步骤 3. 如有需要，对于 SD 卡，可配置主机控制寄存器 (HOSTCON) 设置主机总线宽度，并发送 ACMD6 设置 SD 总线宽度 (1-bit 或 4-bit)，对于 MMC 卡，可配置主机控制寄存器 (HOSTCON) 设置主机总线宽度，并通过 CMD6 (SWITCH) 命令改写 MMC 卡的 Ext_CSD 寄存器来设置总线宽度 (1-bit, 4-bit 或 8-bit)。
- 步骤 4. 如有需要，对于 SD 卡，可配置数据长度寄存器 (BLKSIZE) 设置数据块长度，并通过 CMD16 命令设置数据块的大小 (字节数)，配置范围为 1~512 字节。对于 SDHC/SDXC 以及 MMC 卡数据块的大小为固定 512 字节。
- 步骤 5. 如有需要，对于 SD 卡，可配置主机控制寄存器 (HOSTCON) 设置主机为高速模式 (high speed mode)，并发送 CMD6 将 SD 卡切换卡至高速模式 (high speed mode, $f_{pp} \leq 50\text{MHz}$)，切换成功后可将 SDIOx_CK 设置为最高 50MHz
- 步骤 6. 写数据时，首先配置传输模式寄存器 (TRANSMODE)，设置传输方式为写，单块传输。发送单 block 写命令 CMD24，如使用 CPU 写数据，在确认中断状态寄存器 (NORINTST) 的 BWR 位为 1 后，向数据缓冲寄存器 (BUF0, BUF1) 写入数据，如使用 DMA 写数据，则等待 DMA 传输完成，SDIOC 将通过数据信号 SDIOx_Dy 发送数据。发送完成后，中断状态寄存器 (NORINTST) 的 TC 将置 1 并产生中断请求，如果传输过程出现错误，则相应错误标志位会置起并产生中断请求。
- 步骤 7. 读数据时，首先配置传输模式寄存器 (TRANSMODE)，设置传输方式为读，单块传输。发送单 block 读命令 CMD17，SDIOC 将通过数据信号 SDIOx_Dy 接收数据。如使用 CPU 读数据，在确认中断状态寄存器 (NORINTST) 的 BRR 位为 1 后，从数据缓冲寄存器 (BUF0, BUF1) 读出数据，如使用 DMA 读数据，则等待 DMA 传输完成，读取完成后，中断状态寄存器 (NORINTST) 的 TC 将置 1 并产生中断请求，如果传输过程出现错误，则相应错误标志位会置起并产生中断请求。

39.4.3 SD/MMC 多数据块 (multi block) 读写

操作步骤

- 步骤 1. 配置钟控制寄存器 (CLKCON)，使 SDIOC 输出 SD 时钟，并根据 PCLK1 的频率配置 SD 时钟分频使 SD 时钟频率不超过默认速度模式下的最大时钟速度 (default speed mode, $f_{pp} \leq 25\text{MHz}$)。
- 步骤 2. 发送 CMD7，SD/MMC 卡将进入数据传输状态。

- 步骤 3.** 如有需要，对于 SD 卡，可配置主机控制寄存器 (HOSTCON) 设置主机总线宽度，并发送 ACMD6 设置 SD 总线宽度 (1-bit 或 4-bit)，对于 MMC 卡，可配置主机控制寄存器 (HOSTCON) 设置主机总线宽度，并通过 CMD6 (SWITCH) 命令改写 MMC 卡的 Ext_CSD 寄存器来设置总线宽度 (1-bit, 4-bit 或 8-bit)。
- 步骤 4.** 如有需要，对于 SD 卡，可配置数据长度寄存器 (BLKSIZE) 设置数据块长度，并通过 CMD16 命令设置数据块的大小 (字节数)，配置范围为 1~512 字节。对于 SDHC/SDXC 以及 MMC 卡数据块的大小为固定 512 字节。
- 步骤 5.** 配置传输模式寄存器 (TRANSMODE)，设置传输方式 (读/写)，多块传输。如果设置了允许数据块计数使能，则需要在数据块计数寄存器 (BLKCNT) 中设置需要传输的数据块数量，不设置数据块计数寄存器 (BLKCNT) 则不能启动多块传输。如果设置了禁止数据块计数使能，则不需要设置数据块计数寄存器 (BLKCNT)，此时可以进行不限数量的多块传输，当主机决定结束传输时，需要在最后一个数据块传输完成后向设备发送一个 CMD12 告知设备数据传输结束。
- 步骤 6.** 如有需要，对于 SD 卡，可配置主机控制寄存器 (HOSTCON) 设置主机为高速模式 (high speed mode)，并发送 CMD6 将 SD 卡切换卡至高速模式 (high speed mode, fpp≤50MHz)，切换成功后可将 SDIOx_CK 设置为最高 50MHz
- 步骤 7.** 写数据时，首先发送多 block 写命令 CMD25，如使用 CPU 写数据，在确认中断状态寄存器 (NORINTST) 的 BWR 位为 1 后，向数据缓冲寄存器 (BUF0, BUF1) 写入数据，发送过程中 BWR 将保持 0，发送结束后重新置 1，用户可以在此时写入第二个 block 的数据。如使用 DMA 写数据，则等待 DMA 传输完成，SDIOC 将通过数据信号 SDIOx_Dy 发送数据。中断状态寄存器 (NORINTST) 的 TC 将置 1 并产生中断请求，如果传输过程出现错误，则相应错误标志位会置起并产生中断请求。全部数据发送完毕后，如果设置了自动发送 CMD12，则 SDIOC 将自动发送一个 CMD12 以结束传输，否则需要手动向设备发送 CMD12 告知设备数据传输结束。
- 步骤 8.** 读数据时，发送多 block 读命令 CMD18，SDIOC 将通过数据信号 SDIOx_Dy 接收数据。如使用 CPU 读数据，在确认中断状态寄存器 (NORINTST) 的 BRR 位为 1 后，从数据缓冲寄存器 (BUF0, BUF1) 读出数据，接收过程中 BRR 将保持 0，发送结束后重新置 1，用户可以在此时读取第二个 block 的数据。如使用 DMA 读数据，则等待 DMA 传输完成，读取完成后，中断状态寄存器 (NORINTST) 的 TC 将置 1 并产生中断请求，如果传输过程出现错误，则相应错误标志位会置起并产生中断请求。全部数据接收完毕后，如果设置了自动发送 CMD12，则 SDIOC 将自动发送一个 CMD12 以结束传输，否则需要手动向设备发送 CMD12 告知设备数据传输结束。

39.4.4 传输终止 (abort)，挂起 (suspend) 和恢复 (resume)

当进行多块传输时，可以通过软件控制来终止 (abort) 或者挂起 (suspend)。无论是否设定了传输数据块的数量都可以执行终止 (abort) 操作。终止操作分异步终止和同步终止。异步终止操作需要在传输进行中，通过写命令寄存器 (CMD) 发送 CMD12 来终止传输，此时传输将立即终止。对于写操作，SD/MMC 卡将舍弃当前数据块数据并进入烧写模式 (program)，将之前接收到的数据块写入 FLASH 中。对于读操作，SD/MMC 将停止传输数据。同步终止是指传输进行中通过设置数据块间隔寄存器 (BLKGAP) 使传输在数据块间隔处停止传输，设置完成后，传输将在当前数据块传输结束后停止传输。此时需要发送 CMD12 以结束传输。

执行挂起 (suspend) 操作时，首先通过设置数据块间隔寄存器 (BLKGAP) 的 SABGR 位使传输在数据块间隔处停止传输，并且使能读等待 (read wait) 功能。设置完成后，传输将在当前数据块传输结束后停止传输。此时通过写命令寄存器 (CMD) 发送 CMD52 来将传输挂起。发送挂起命令后，需要继续通过 CMD52 读取 SDIO 的 CCCR 寄存器的 BS 位 (bus status)，如果 BS=0 代表传输已挂起，SD 总线转为空闲状态。传输被挂起后，主机可以对 SDIO 的其他功能执行操作。但是如果稍后进行恢复 (resume) 操作，需要在挂起后备份 SDIOC 寄存器 (偏移地址 0x00~0x0D)，执行完其他操作后，恢复这些寄存器设置。执行恢复 (resume) 操作时，首先清除数据块间隔寄存器 (BLKGAP) 使传输在数据块间隔处停止传输的设置，之后通过写命令寄存器 (CMD) 发送 CMD52 来恢复传输。发送恢复命令后，需要继续通过 CMD52 读取 SDIO 的

CCCR 寄存器的 DF 位 (data flag)，如果 DF=1 代表执行恢复后有数据需要继续传输，如果 DF=0 则代表没有数据需要传输，此时应写软件复位寄存器 (SFTRST) 复位数据线。

**注意**

挂起 (suspend) 和恢复 (resume) 操作需要 SDIO 设备和 combo card 设备支持此类操作以及读等待 (read wait) 操作。SDIO 的 CIA 区域 (common I/O area) 的访问不支持挂起 (suspend) 和恢复 (resume) 操作，只能使用读等待 (read wait) 功能。

39.4.5 读等待 (read wait)

读等待 (read wait) 允许主机在连续数据传输中插入 CMD52 以访问 SDIO 设备的其他功能。执行读等待 (read wait) 时首先通过设置数据块间隔寄存器 (BLKGAP) 的 SABGR 位使传输在数据块间隔处停止传输，并且使能读等待 (read wait) 功能。设置完成后，主机将在当前数据块传输结束后将 SDIOx_D2 数据线拉低。此时传输将暂停，主机可以在此时插入 CMD52 以访问其他不需要数据传输的功能。需要结束读等待 (read wait) 功能时，通过设置数据块间隔寄存器 (BLKGAP) 的 CR 并清除 SABGR 以恢复传输。

**注意**

读等待 (read wait) 需要 SDIO 设备和 combo card 设备支持，并且需要在四线式总线传输下进行。

39.4.6 唤醒 (wakeup)

当长时间不工作时，可将系统转入低功耗状态以降低功耗。在低功耗状态下可以通过 SD/MMC/SDIO 设备的插入 (insert) /移除 (removal) 和卡中断 (card interrupt) 来唤醒系统继续工作。使用唤醒功能时，需要将插入 (insert) /移除 (removal) 或卡中断 (card interrupt) 在中断状态使能寄存器 (NORINTSTEN) 和中断信号使能寄存器 (NORINTSGEN) 中相应的使能位使能。同时使能 SDIOx_CD/SDIOx_D1 端口相应的唤醒功能。配置完成后可将系统转入低功耗模式。当发生插入 (insert) /移除 (removal) 和卡中断 (card interrupt) 时，SDIOx_CD/SDIOx_D1 端口的唤醒功能将会唤醒系统，并发生插入 (insert) /移除 (removal) 和卡中断 (card interrupt) 中断。

使用插入 (insert) 唤醒停止模式 (stop mode) 流程

- 步骤 1. 配置 SDIOC 端口，选择 SDIOx_CD 为 PA10
- 步骤 2. 配置 PA10 的 PCR 寄存器，选择 EIRQ10 有效
- 步骤 3. 配置电源控制模块低功耗模式为停止模式，配置方法请参见《电源控制 (PWC)》章节相关描述
- 步骤 4. 配置中断控制模块 WUPEN 寄存器使能 EIRQ10 的唤醒功能
- 步骤 5. 配置中断控制模块的 EIRQCR10 寄存器，选择下降沿触发
- 步骤 6. 配置 PWRCON 使能 SDIOC
- 步骤 7. 配置中断状态使能寄存器 (NORINTSTEN) 和中断信号使能寄存器 (NORINTSGEN) 中插入 (insert) 相应的使能位使能
- 步骤 8. 配置中断源选择寄存器选择 SDIOC 中断为中断源，并使能中断
- 步骤 9. 执行 WFI 指令使系统进入停止模式
- 步骤 10. 当设备插入时，PA10 将被拉低，通过 EIRQ10 唤醒 CPU，并根据 SD 中断申请进入中断子程序
- 步骤 11. 清除中断状态寄存器 (NORINTST) 中插入 (insert) 状态，退出中断子程序，执行后续操作

使用插入 (insert) 唤醒掉电模式 (power down mode) 流程

- 步骤 1. 当设备插入时，PA10 将被拉低，通过 EIRQ10 唤醒系统重新上电配置 SDIOC 端口，选择 SDIOx_CD 为 PA10
- 步骤 2. 配置电源控制模块的 PWRC0 寄存器使 CPU 执行 WFI 命令后进入掉电模式
- 步骤 3. 配置电源控制模块低功耗模式为掉电模式，配置方法请参见《电源控制 (PWC)》章节相关描述
- 步骤 4. 配置电源控制模块的 PDWKES 寄存器，选择下降沿

步骤 5. 执行 WFI 指令使系统进入掉电模式

步骤 6. 当设备插入时，PA10 将被拉低，通过 WKUP2_2 唤醒系统重新上电

39.5 寄存器

SDIOC 模块根据 SD Host Controller Standard Specification 标准设计，因此寄存器也与该标准描述相同，未使用的地址和位改为保留位（Reserved）。

39.5.1 寄存器总表

SDIOC 系统控制寄存器基地址：0x40055400

表 39-2 SDIOC 系统控制寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x04	SDIOC_SYCTLREG	SDIOC 系统控制寄存器	32

- SDIOC1 基地址：0x40070000
- SDIOC2 基地址：0x40078400

表 39-3 SDIO 控制器 (SDIOC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x04	SDIOC_BLKSIZE	数据块长度寄存器	16
0x06	SDIOC_BLKCNT	数据块计数寄存器	16
0x08	SDIOC_ARG0	参数寄存器 0	16
0x0A	SDIOC_ARG1	参数寄存器 1	16
0x0C	SDIOC_TRANSMODE	传输模式寄存器	16
0x0E	SDIOC_CMD	命令寄存器	16
0x10	SDIOC_RESP0	应答寄存器 0	16
0x12	SDIOC_RESP1	应答寄存器 1	16
0x14	SDIOC_RESP2	应答寄存器 2	16
0x16	SDIOC_RESP3	应答寄存器 3	16
0x18	SDIOC_RESP4	应答寄存器 4	16
0x1A	SDIOC_RESP5	应答寄存器 5	16
0x1C	SDIOC_RESP6	应答寄存器 6	16
0x1E	SDIOC_RESP7	应答寄存器 7	16
0x20	SDIOC_BUF0	数据缓冲寄存器 0	16
0x22	SDIOC_BUF1	数据缓冲寄存器 1	16
0x24	SDIOC_PSTAT	主机状态寄存器	32
0x28	SDIOC_HOSTCON	主机控制寄存器	8
0x29	SDIOC_PWRCON	电源控制寄存器	8
0x2A	SDIOC_BLKGPON	数据块间隔控制寄存器	8

偏移地址	寄存器	描述	位宽
0x2C	SDIOC_CLKCON	时钟控制寄存器	16
0x2E	SDIOC_TOUTCON	超时控制寄存器	8
0x2F	SDIOC_SFTRST	软件复位寄存器	8
0x30	SDIOC_NORINTST	普通中断状态寄存器	16
0x32	SDIOC_ERRINTST	错误中断状态寄存器	16
0x34	SDIOC_NORINTSTEN	普通中断状态使能寄存器	16
0x36	SDIOC_ERRINTSTEN	错误中断状态使能寄存器	16
0x38	SDIOC_NORINTSGEN	普通中断信号使能寄存器	16
0x3A	SDIOC_ERRINTSGEN	错误中断信号使能寄存器	16
0x3C	SDIOC_ATCERRST	自动命令错误状态寄存器	16
0x50	SDIOC_FEA	强制自动命令错误状态控制寄存器	16
0x52	SDIOC_FEE	强制错误状态控制寄存器	16

39.5.2 数据块长度寄存器 (SDIOC_BLKSIZE)

Offset	Bit Position															
0x04	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved				TBS											
Access																

位/位域	标记	位名	功能描述	读写
15:12	Reserved	保留	-	-
11:0	TBS	数据块长度 (Transfer Block Size)	设定传输的数据块长度，以字节为单位，取值范围是 1~512	RW

39.5.3 数据块计数寄存器 (SDIOC_BLKCNT)

Offset	Bit Position															
0x06	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	BC															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	BC	数据块计数 (Block count)	设定传输数据块的数量，设定该寄存器需要在传输停止时进行，并且需要传输模式寄存器的数据块计数使能位 (BCE) 有效	RW

39.5.4 参数寄存器 0 (SDIOC_ARG0)

Offset	Bit Position															
0x08	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	ARG0															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	ARG0	命令参数	设定当前发送命令包含的参数，该寄存器为参数的低 16 位	RW

39.5.5 参数寄存器 1 (SDIOC_ARG1)

Offset	Bit Position															
0x0A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	ARG1															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	ARG1	命令参数	设定当前发送命令包含的参数，该寄存器为参数的高 16 位	RW

39.5.6 传输模式寄存器 (SDIOC_TRANSMODE)

Offset	Bit Position															
0x0C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved										MULB	DDIR	ATCEN	BCE	Reserved	
Access											RW	RW	RW	RW		

位/位域	标记	位名	功能描述	读写
15:6	Reserved	保留	-	-
5	MULB	多数据块	0b0: 当前传输为单数据块传输 (single block) 0b1: 当前传输为多数据块传输 (multi block)	RW
4	DDIR	数据传输方向	0b0: 写操作 (主机发送数据) 0b1: 读操作 (主机接收数据)	RW
3:2	ATCEN	自动命令使能	0b00: 不发送自动命令 0b01: 多块传输结束后自动发送 CMD12 0b10: 禁止设置 0b11: 禁止设置	RW
1	BCE	块计数使能	0b0: 禁用数据块计数使能 0b1: 允许数据块计数使能	RW
0	Reserved	保留	-	-

39.5.7 命令寄存器 (SDIOC_CMD)

Offset	Bit Position															
0x0E	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved		IDX				TYP				DAT	ICE	CCE	Reserved	RESTYP	
Access			RW				RW				RW	RW	RW		RW	

位/位域	标记	位名	功能描述	读写
15:14	Reserved	保留	-	-
13:8	IDX	命令编号	发送命令的编号	RW
7:6	TYP	命令类型	0b00: 普通命令 0b01: 挂起 (suspend) 命令 0b10: 恢复 (resume) 命令 0b11: 终止 (abort) 命令	RW
5	DAT	带数据命令	0b0: 当前命令仅使用 SDIOx_CMD 命令线 0b1: 当前命令需要使用 SDIOx_Dy 数据线	RW
4	ICE	编号检查	0b0: 不检查应答 (response) 中的命令编号 0b1: 检查应答 (response) 中的命令编号	RW
3	CCE	CRC 检查	0b0: 不检查应答 (response) 中的 CRC 校验码 0b1: 检查应答 (response) 中的 CRC 校验码	RW
2	Reserved	保留	-	-
1:0	RESTYP	应答类型	0b00: 该命令无应答 0b01: 命令具有长度为 136-bit 的应答 0b10: 命令具有长度为 48-bit 的应答 0b11: 命令具有长度为 48-bit 并且带有忙 (busy) 状态的应答	RW

39.5.8 应答寄存器 0 (SDIOC_RESP0)

Offset	Bit Position															
0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RESP0															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP0	应答信息	应答信息的 15~0 位	RO

39.5.9 应答寄存器 1 (SDIOC_RESP1)

Offset	Bit Position															
0x12	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP1															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP1	应答信息	应答信息的 31~16 位	RO

39.5.10 应答寄存器 2 (SDIOC_RESP2)

Offset	Bit Position															
0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP2															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP2	应答信息	应答信息的 47~32 位	RO

39.5.11 应答寄存器 3 (SDIOC_RESP3)

Offset	Bit Position															
0x16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP3															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP3	应答信息	应答信息的 63~48 位	RO

39.5.12 应答寄存器 4 (SDIOC_RESP4)

Offset	Bit Position															
0x18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP4															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP4	应答信息	应答信息的 79~64 位	RO

39.5.13 应答寄存器 5 (SDIOC_RESP5)

Offset	Bit Position															
0x1A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP5															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP5	应答信息	应答信息的 95~80 位	RO

39.5.14 应答寄存器 6 (SDIOC_RESP6)

Offset	Bit Position															
0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP6															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP6	应答信息	应答信息的 111~96 位	RO

39.5.15 应答寄存器 7 (SDIOC_RESP7)

Offset	Bit Position															
0x1E	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RO RESP7															
Access	RO															

位/位域	标记	位名	功能描述	读写
15:0	RESP7	应答信息	应答信息的 127~112 位	RO

39.5.16 数据缓冲寄存器 0 (SDIOC_BUF0)

Offset	Bit Position															
0x20	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RW BUF0															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	BUF0	数据缓冲	写入发送数据和读取接收数据，该寄存器为数据的低 16 位	RW

39.5.17 数据缓冲寄存器 1 (SDIOC_BUF1)

Offset	Bit Position															
0x22	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	RW BUF1															
Access	RW															

位/位域	标记	位名	功能描述	读写
15:0	BUF1	数据缓冲	写入发送数据和读取接收数据，该寄存器为数据的高 16 位	RW

39.5.18 主机状态寄存器 (SDIOC_PSTAT)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
Name	Reserved									DATL						WPL			CDL			CSS			CIN		Reserved						BRE			BWE			RTA			WTA		Reserved						DA			CID			CIC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
Access								RO		RO						RO			RO			RO			RO			RO								RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO			RO

位/位域	标记	位名	功能描述	读写
31:25	Reserved	保留	-	-
24	CMDL	命令线状态	命令线 (SDIOx_CMD) 的状态	RO
23:20	DATL	数据线状态	数据线 (SDIOx_Dy (y=0~3)) 的状态	RO
19	WPL	写保护线状态	写保护线 (SDIOx_WP) 的状态	RO
18	CDL	卡识别线状态	卡识别线 (SDIOx_CD) 的状态	RO
17	CSS	设备稳定状态	0b0: 卡识别线状态不稳定 0b1: 卡识别线状态稳定, 设备已插入或未插入	RO
16	CIN	设备插入状态	0b0: 无设备插入 0b1: 有设备插入	RO
15:12	Reserved	保留	-	-
11	BRE	数据缓冲满	0b0: 数据缓冲器没有足够的数据供读取 0b1: 数据缓冲器有足够的数据供读取	RO
10	BWE	数据缓冲空	0b0: 数据缓冲器不能写入数据 0b1: 数据缓冲器可以写入数据	RO
9	RTA	读操作状态	0b0: 没有正在进行的读操作 0b1: 有正在进行的读操作	RO
8	WTA	写操作状态	0b0: 没有正在进行的写操作 0b1: 有正在进行的写操作	RO
7:3	Reserved	保留	-	-
2	DA	数据线传输状态	0b0: 数据线空闲 0b1: 数据线正在传输数据	RO
1	CID	带数据命令抑制	0b0: 允许发送带数据命令 0b1: 禁止发送带数据命令	RO
0	CIC	命令抑制	0b0: 允许发送命令 0b1: 禁止发送命令	RO

39.5.19 主机控制寄存器 (SDIOC_HOSTCON)

Offset	Bit Position							
0x28	7	6	5	4	3	2	1	0
Reset	0x00							
Name	CDSS	CDTL	EXDW	Reserved		HSEN	DW	Reserved
Access	RW	RW	RW			RW	RW	

位/位域	标记	位名	功能描述	读写
7	CDSS	卡识别线选择	0b0: 选择真实的 SDIOx_CD 线反映卡识别状态 - PSTAT.CDL=0 表示卡移除 - PSTAT.CDL=1 表示卡插入 0b1: 选择卡识别测试信号反映卡识别状态	RW
6	CDTL	卡识别测试信号状态	0b0: 卡识别测试信号为低电平 (无设备插入) 0b1: 卡识别测试信号为高电平 (有设备插入)	RW
5	EXDW	扩展数据位宽	0b0: 数据线位宽使用 DW 位的设定 0b1: 数据线位宽为 8 位 (8-bit)	RW
4:3	Reserved	保留	-	-
2	HSEN	高速模式使能	0b0: 禁用高速模式 (high speed mode) 0b1: 启用高速模式	RW
1	DW	数据位宽选择	0b0: 数据线位宽为 1 位 (1-bit) 0b1: 数据线位宽为 4 位 (4-bit)	RW
0	Reserved	保留	-	-

39.5.20 电源控制寄存器 (SDIOC_PWRCON)

Offset	Bit Position							
0x29	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved							PWON
Access								RW

位/位域	标记	位名	功能描述	读写
7:1	Reserved	保留	-	-
0	PWON	SDIOC 使能	0b0: 禁用 SDIOC 0b1: 启用 SDIOC	RW

39.5.21 数据块间隙控制寄存器 (SDIOC_BLKGPCON)

Offset	Bit Position							
0x2A	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved				IABG	RWC	CR	SABGR
Access					RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:4	Reserved	保留	-	-
3	IABG	数据块间隙中断控制	0b0: 关闭数据块间隙期间接收 SDIO 设备中断 (card interrupt) 0b1: 开启数据块间隙期间接收 SDIO 设备中断 (card interrupt)	RW
2	RWC	读等待控制	0b0: 禁用读等待功能 (read wait) 0b1: 启用读等待功能 (read wait)	RW
1	CR	继续传输	0b0: 无任何效果 0b1: 解除因 SABGR 位置位而停止的传输	RW
0	SABGR	数据块间隙停止传输	0b0: 不在数据块间隙时停止传输 0b1: 在数据块间隙时停止传输	RW

39.5.22 时钟控制寄存器 (SDIOC_CLKCON)

Offset	Bit Position															
0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0002															
Name	FS								Reserved				CE		Reserved	ICE
Access	RW												RW			RW

位/位域	标记	位名	功能描述	读写
15:8	FS	分频选择	SDIOx_CK 时钟分频选择，基准时钟为 PCLK1 ● 0x00: PCLK1 ● 0x01: PCLK1 的 2 分频 ● 0x02: PCLK1 的 4 分频 ● 0x04: PCLK1 的 8 分频 ● 0x08: PCLK1 的 16 分频 ● 0x10: PCLK1 的 32 分频 ● 0x20: PCLK1 的 64 分频 ● 0x40: PCLK1 的 128 分频 ● 0x80: PCLK1 的 256 分频 ● 0x81: PCLK1 的 512 分频 ● 其他: 禁止设置	RW
7:3	Reserved	保留	-	-
2	CE	SDIOx_CK 输出控制	● 0b0: SDIOx_CK 停止输出 ● 0b1: SDIOx_CK 输出	RW
1	Reserved	保留	-	-
0	ICE	时钟使能	● 0b0: SDIOC 动作时钟开启 ● 0b1: SDIOC 动作时钟关闭	RW

39.5.23 超时控制寄存器 (SDIOC_TOUTCON)

Offset	Bit Position							
0x2E	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved				DTO			
Access					RW			

位/位域	标记	位名	功能描述	读写
7:4	Reserved	保留	-	-
3:0	DTO	数据超时时间	设定数据线 SDIOx_Dy (y=0~7) 超时判定的时间，单位为 PCLK1 的时钟周期 ● 0b0000: $PCLK1 \times 2^{13}$ ● 0b0001: $PCLK1 \times 2^{14}$ ● ● 0b1110: $PCLK1 \times 2^{27}$ ● 0b1111: 禁止设置	RW

39.5.24 软件复位寄存器 (SDIOC_SFTRST)

Offset	Bit Position							
0x2F	7	6	5	4	3	2	1	0
Reset	0x00							
Name	Reserved					RSTD	RSTC	RSTA
Access						RW	RW	RW

位/位域	标记	位名	功能描述	读写
7:3	Reserved	保留	-	-
2	RSTD	数据复位	复位所有数据相关的寄存器，包含以下寄存器位： BUF0, BUF1, PSTAT.BRE, PSTAT.BWE, PSTAT.RTA, PSTAT.WTA, PSTAT.DLA, PSTAT.CID, BLKGPCON.CR, BLKGPCON.SABGR, NORINTST.BRR, NORINTST.BWR, NORINTST.BGE, NORINTST.TC ● 0b0: 正常工作 ● 0b1: 执行复位	RW
1	RSTC	命令复位	复位所有命令相关的寄存器，包含以下寄存器位： PSTAT.CIC, NORINTST.CC ● 0b0: 正常工作 ● 0b1: 执行复位	RW
0	RSTA	全部复位	复位除卡识别功能外的全部 SDIOC 寄存器 ● 0b0: 正常工作 ● 0b1: 执行复位	RW

39.5.25 普通中断状态寄存器 (SDIOC_NORINTST)

Offset	Bit Position															
0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	EI	Reserved						CINT	CRM	CIST	BRR	BWR	Reserved	BGE	TC	CC
Access	RO							RO	RW	RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15	EI	错误中断	发生错误中断状态寄存器 (ERRINTST) 中任意错误时置位	RO
14:9	Reserved	保留	-	-
8	CINT	卡中断	SDIO 设备发出卡中断 (card interrupt) 申请时置位, SDIO 设备撤销申请后复位	RO
7	CRM	卡移除	设备移除 (removal) 时置位, 写 1 复位	RW
6	CIST	卡插入	设备插入 (insert) 时置位, 写 1 复位	RW
5	BRR	缓冲器可读	缓冲器内数据可被读取 (PSTAT.BRE=1) 时置位, 写 1 复位	RW
4	BWR	缓冲器可写	缓冲器可写入数据 (PSTAT.BWE=1) 时置位, 写 1 复位	RW
3	Reserved	保留	-	-
2	BGE	数据块间隙停止传输	当传输在数据块间隙停止时置位, 写 1 复位	RW
1	TC	传输完成	读写传输完成时置位, 写 1 复位	RW
0	CC	命令完成	无应答命令命令发送完成和有应答命令的应答接收完成后置位, 写 1 复位	RW

39.5.26 错误中断状态寄存器 (SDIOC_ERRINTST)

Offset	Bit Position															
0x32	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							ACE	Reserved	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
Access								RW		RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	ACE	自动发送命令错误	自动发送命令 (auto CMD) 发生错误时置位，错误的种类可以查询自动命令错误寄存器 (ATCERRST)，写 1 复位	RW
7	Reserved	保留	-	-
6	DEBE	数据停止位错误	当数据线在停止位检测到低电平时置位，写 1 复位	RW
5	DCE	数据 CRC 校验错误	当数据线发生 CRC 校验错误时置位，写 1 复位	RW
4	DTOE	数据超时错误	当发生数据超时时置位，写 1 复位，数据超时时间由超时控制寄存器 (TOUTCON) 设定	RW
3	CIE	命令编号错误	接收到的应答所包含的命令编号错误时置位，写 1 复位	RW
2	CEBE	命令停止位错误	当命令线在停止位检测到低电平时置位，写 1 复位	RW
1	CCE	命令 CRC 校验错误	当命令线发生 CRC 校验错误时置位，写 1 复位	RW
0	CTOE	命令超时错误	当命令发送后超过 64 个 SDIOx_CLK 周期未收到应答时置位，写 1 复位	RW

39.5.27 普通中断状态使能寄存器 (SDIOC_NORINTSTEN)

Offset	Bit Position															
0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							CINTEN	CRMEN	CISTEN	BRREN	BWREN	Reserved	BGEEN	TCEN	CCEN
Access								RW	RW	RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	CINTEN	卡中断状态使能	0b0: 禁止 NORINTST.CINT 置位 0b1: 允许 NORINTST.CINT 置位	RW
7	CRMEN	卡移除状态使能	0b0: 禁止 NORINTST.CRM 置位 0b1: 允许 NORINTST.CRM 置位	RW
6	CISTEN	卡插入状态使能	0b0: 禁止 NORINTST.CIST 置位 0b1: 允许 NORINTST.CIST 置位	RW
5	BRREN	缓冲器可读状态使能	0b0: 禁止 NORINTST.BRR 置位 0b1: 允许 NORINTST.BRR 置位	RW
4	BWREN	缓冲器可写状态使能	0b0: 禁止 NORINTST.BWR 置位 0b1: 允许 NORINTST.BWR 置位	RW
3	Reserved	保留	-	-
2	BGEEN	数据块间隙停止传输状态使能	0b0: 禁止 NORINTST.BGE 置位 0b1: 允许 NORINTST.BGE 置位	RW
1	TCEN	传输完成状态使能	0b0: 禁止 NORINTST.TC 置位 0b1: 允许 NORINTST.TC 置位	RW
0	CCEN	命令完成状态使能	0b0: 禁止 NORINTST.CC 置位 0b1: 允许 NORINTST.CC 置位	RW

39.5.28 错误中断状态使能寄存器 (SDIOC_ERRINTSTEN)

Offset	Bit Position															
0x36	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							ACEEN	Reserved	DEBEEN	DCEEN	DTOEEN	CIEEN	CEBEEN	CCEEN	CTOEEN
Access								RW		RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	ACEEN	自动发送命令错误状态使能	● 0b0: 禁止 ERRINTST.ACE 置位 ● 0b1: 允许 ERRINTST.ACE 置位	RW
7	Reserved	保留	-	-
6	DEBEEN	数据停止位错误状态使能	● 0b0: 禁止 ERRINTST.DEBE 置位 ● 0b1: 允许 ERRINTST.DEBE 置位	RW
5	DCEEN	数据 CRC 校验错误状态使能	● 0b0: 禁止 ERRINTST.DCE 置位 ● 0b1: 允许 ERRINTST.DCE 置位	RW
4	DTOEEN	数据超时错误状态使能	● 0b0: 禁止 ERRINTST.DTOE 置位 ● 0b1: 允许 ERRINTST.DTOE 置位	RW
3	CIEEN	命令编号错误状态使能	● 0b0: 禁止 ERRINTST.CIE 置位 ● 0b1: 允许 ERRINTST.CIE 置位	RW
2	CEBEEN	命令停止位错误状态使能	● 0b0: 禁止 ERRINTST.CEBE 置位 ● 0b1: 允许 ERRINTST.CEBE 置位	RW
1	CCEEN	命令 CRC 校验错误状态使能	● 0b0: 禁止 ERRINTST.CCE 置位 ● 0b1: 允许 ERRINTST.CCE 置位	RW
0	CTOEEN	命令超时错误状态使能	● 0b0: 禁止 ERRINTST.CTOE 置位 ● 0b1: 允许 ERRINTST.CTOE 置位	RW

39.5.29 普通中断信号使能寄存器 (SDIOC_NORINTSGEN)

Offset	Bit Position															
0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							CINTSEN	CRMSSEN	CISTSEN	BRRSEN	BWRSEN	Reserved	BGESEN	TCSEN	CCSEN
Access								RW	RW	RW	RW	RW		RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	CINTSEN	卡中断信号使能	0b0: 禁止 NORINTST.CINT 申请中断 0b1: 允许 NORINTST.CINT 申请中断	RW
7	CRMSSEN	卡移除信号使能	0b0: 禁止 NORINTST.CRM 申请中断 0b1: 允许 NORINTST.CRM 申请中断	RW
6	CISTSEN	卡插入信号使能	0b0: 禁止 NORINTST.CIST 申请中断 0b1: 允许 NORINTST.CIST 申请中断	RW
5	BRRSEN	缓冲器可读信号使能	0b0: 禁止 NORINTST.BRR 申请中断 0b1: 允许 NORINTST.BRR 申请中断	RW
4	BWRSEN	缓冲器可写信号使能	0b0: 禁止 NORINTST.BWR 申请中断 0b1: 允许 NORINTST.BWR 申请中断	RW
3	Reserved	保留	-	-
2	BGESEN	数据块间隙停止传输信号使能	0b0: 禁止 NORINTST.BGE 申请中断 0b1: 允许 NORINTST.BGE 申请中断	RW
1	TCSEN	传输完成信号使能	0b0: 禁止 NORINTST.TC 申请中断 0b1: 允许 NORINTST.TC 申请中断	RW
0	CCSEN	命令完成信号使能	0b0: 禁止 NORINTST.CC 申请中断 0b1: 允许 NORINTST.CC 申请中断	RW

39.5.30 错误中断信号使能寄存器 (SDIOC_ERRINTSGEN)

Offset	Bit Position															
0x3A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							ACESEN	Reserved	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESSEN	CTOESEN
Access								RW		RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	ACESEN	自动发送命令错误信号使能	0b0: 禁止 ERRINTST.ACE 申请中断 0b1: 允许 ERRINTST.ACE 申请中断	RW
7	Reserved	保留	-	-
6	DEBESEN	数据停止位错误信号使能	0b0: 禁止 ERRINTST.DEBE 申请中断 0b1: 允许 ERRINTST.DEBE 申请中断	RW
5	DCESEN	数据 CRC 校验错误信号使能	0b0: 禁止 ERRINTST.DCE 申请中断 0b1: 允许 ERRINTST.DCE 申请中断	RW
4	DTOESEN	数据超时错误信号使能	0b0: 禁止 ERRINTST.DTOE 申请中断 0b1: 允许 ERRINTST.DTOE 申请中断	RW
3	CIESEN	命令编号错误信号使能	0b0: 禁止 ERRINTST.CIE 申请中断 0b1: 允许 ERRINTST.CIE 申请中断	RW
2	CEBESEN	命令停止位错误信号使能	0b0: 禁止 ERRINTST.CEBE 申请中断 0b1: 允许 ERRINTST.CEBE 申请中断	RW
1	CCESSEN	命令 CRC 校验错误信号使能	0b0: 禁止 ERRINTST.CCE 申请中断 0b1: 允许 ERRINTST.CCE 申请中断	RW
0	CTOESEN	命令超时错误信号使能	0b0: 禁止 ERRINTST.CTOE 申请中断 0b1: 允许 ERRINTST.CTOE 申请中断	RW

39.5.31 自动命令错误状态寄存器 (SDIOC_ATCERRST)

Offset	Bit Position															
0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved								CMDE	Reserved		IE	EBE	CE	TOE	NE
Access									RO			RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
15:8	Reserved	保留	-	-
7	CMDE	未发送错误	当本寄存器 Bit4~Bit0 位对应错误发生导致其它命令未发送时置位	RO
6:5	Reserved	保留	-	-
4	IE	命令编号错误	接收到的应答所包含的自动命令的编号错误时置位	RO
3	EBE	停止位错误	当命令线在停止位检测到低电平时置位	RO
2	CE	数据超时错误	当命令线发生 CRC 校验错误时置位	RO
1	TOE	命令超时错误	当自动命令发送后超过 64 个 SDIOx_CLK 周期未收到应答时置位	RO
0	NE	未执行错误	当自动命令因其它原因未发送时置位	RO

39.5.32 强制自动命令错误状态控制寄存器 (SDIOC_FEA)

Offset	Bit Position															
0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved								FCMDE	Reserved		FIE	FEBE	FCE	FTOE	FNE
Access									RW			RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:8	Reserved	保留	-	-
7	FCMDE	强制未发送错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.CMDE 错误 读该寄存器位无任何效果	RW
6:5	Reserved	保留	-	-
4	FIE	强制命令编号错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.IE 错误 读该寄存器位无任何效果	RW
3	FEBE	强制停止位错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.EBE 错误 读该寄存器位无任何效果	RW
2	FCE	强制数据超时错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.CE 错误 读该寄存器位无任何效果	RW
1	FTOE	强制命令超时错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.TOE 错误 读该寄存器位无任何效果	RW
0	FNE	强制未执行错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ATCERRST.NE 错误 读该寄存器位无任何效果	RW

39.5.33 强制错误状态控制寄存器 (SDIOC_FEE)

Offset	Bit Position															
0x52	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000															
Name	Reserved							FACE	Reserved	FDEBE	FDCE	FDTOE	FCIE	FCEBE	FCCE	FCTOE
Access								RW		RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
15:9	Reserved	保留	-	-
8	FACE	强制自动发送命令错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.ACE 错误 读该寄存器位无任何效果	RW
7	Reserved	保留	-	-
6	FDEBE	强制数据停止位错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.DEBE 错误 读该寄存器位无任何效果	RW
5	FDCE	强制数据 CRC 校验错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.DCE 错误 读该寄存器位无任何效果	RW
4	FDTOE	强制数据超时错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.DTOE 错误 读该寄存器位无任何效果	RW
3	FCIE	强制命令编号错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.CIE 错误 读该寄存器位无任何效果	RW
2	FCEBE	强制命令停止位错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.CEBE 错误 读该寄存器位无任何效果	RW
1	FCCE	强制命令 CRC 校验错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.CCE 错误 读该寄存器位无任何效果	RW
0	FCTOE	强制命令超时错误	写入： ● 0b0：无任何效果 ● 0b1：强制发生 ERRINTST.CTOE 错误 读该寄存器位无任何效果	RW

39.5.34 系统控制寄存器 (SDIOC_SYCTLREG)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3	SELMMC2	SDIOC2 MMC 模式使能	● 0b0: SDIOC2 选择 SD 模式 ● 0b1: SDIOC2 选择 MMC 模式	RW
2	Reserved	保留	-	-
1	SELMMC1	SDIOC1 MMC 模式使能	● 0b0: SDIOC1 选择 SD 模式 ● 0b1: SDIOC1 选择 MMC 模式	RW
0	Reserved	保留	-	-

40 外部存储器控制器 (EXMC)

40.1 简介

外部存储器控制器 EXMC (External Memory Controller) 是一个用来访问各种片外存储器，实现数据交换的独立模块。EXMC 通过配置可以把内部的 AMBA 协议接口转换为各种类型的专用片外存储器通信协议接口，包括 SRAM、PSRAM、NOR Flash、NAND Flash 和 SDRAM 等。EXMC 内部划分为多个子模块，每个子模块支持特定的存储器类型，用户可以通过对子模块的寄存器配置来控制外部对应类型的存储器。

40.2 主要特性

支持的特性如下：

- 支持的外部存储器类型：SRAM、PSRAM、NOR Flash、NAND Flash、SDRAM
- 接口协议：支持 AHB 协议与各种外部存储器的接口转换
- 接口时序：各种类型存储器的接口时序可编程
- 总线位宽：
 - ▶ SMC 支持 8 位、16 位、32 位 MEM 总线宽度
 - ▶ DMC 支持 16 位、32 位 MEM 总线宽度
 - ▶ NFC 支持 8 位、16 位 MEM 总线宽度
- 总线复用：NOR Flash 和 PSRAM 支持地址线和数据线复用
- 自动分割：AHB 总线位宽和外部存储器位宽不匹配时，支持自动分割及字节选择控制
- 中断类型：
 - ▶ NFC ECC 计算完成中断
 - ▶ NFC ECC 错误中断
 - ▶ NFC 设备访问结束中断

40.3 功能说明

40.3.1 控制器架构

各种类型的外部存储器控制器独立生成对应协议的接口送至端口 MUX 逻辑。端口 MUX 逻辑将各种外部存储器的地址、数据、控制信号等共享在相同的端口上后再从芯片的端口上输出，此时 EXMC 控制器一次只能访问一个外部器件；也可将各种外部存储器的部分信号共享后同时访问不同的外部器件。

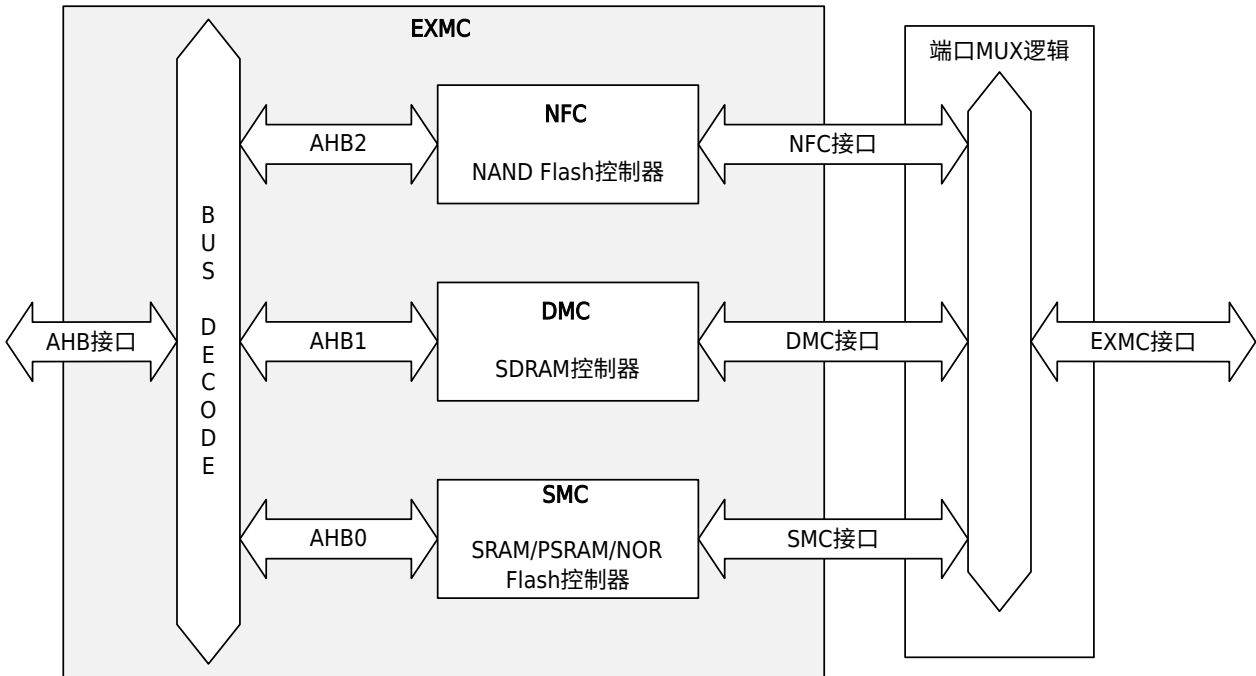


图 40-1 EXMC 架构图

 **说明**

架构图中将 SRAM/PSRAM/NOR Flash 控制器定义为 SMC（Static Memory Controller）、SDRAM 控制器定义为 DMC（Dynamic Memory Controller）、将 NAND Flash 控制器定义为 NFC（NAND Flash Memory Controller），后面提到 SMC、DMC、NFC 时，与对应的控制器有相同的含义。

40.3.2 基本访问规范

EXMC 是 AHB 总线协议和外部存储器协议的转换接口。在 SMC 和 DMC 的数据传输过程中，AHB 数据宽度和存储器数据宽度可能不相同。为了保证数据传输的一致性，读写访问需要遵从以下规范：

- 1. AHB 访问宽度等于存储器宽度，数据正常访问；
- 2. AHB 访问宽度大于存储器宽度，自动将 AHB 访问分割成几个连续的存储器数据宽度的传输；
- 3. AHB 访问宽度小于存储器宽度，通过字节选择控制信号 SMC_BLS[3:0]或 DMC_DQM[3:0]来访问对应的字节。

AHB 访问宽度和各种数据位宽的存储器之间的具体的访问方式参见下表。

表 40-1 AHB 访问宽度和存储器位宽对应访问方式表

RW	AHB 访问宽度	外部存储器位宽	说明
R	8	8	-
	16		转换成 2 次 EXMC 读操作
	32		转换成 4 次 EXMC 读操作
	8	16	-
	16		-
	32		转换成 2 次 EXMC 读操作
	8	32	-
	16		-
	32		-
W	8	8	使用字节控制信号 BLS[0]
	16		转换成 2 次 EXMC 写操作
	32		转换成 4 次 EXMC 写操作
	8	16	使用字节控制信号 BLS[1:0]或 DQM[1:0]
	16		-
	32		转换成 2 次 EXMC 写操作
	8	32	使用字节控制信号 BLS[3:0]或 DQM[3:0]
	16		使用字节控制信号 BLS[3:0]或 DQM[3:0]
	32		-



说明

上表仅适用于 SMC 和 DMC 的数据访问，其中 SMC 支持外部存储器位宽为 8/16/32，DMC 支持外部存储器位宽为 16/32。

40.3.3 地址映射

本产品中定义了总大小为 1GB 的外部存储器访问区间，用于不同类型的外部存储器和内部数据交换，包括 EXMC 和 QSPI 等。该 1GB 的空间按下图的方案分配，SRAM/PSRAM/NOR Flash (SMC) 的数据访问映射 512MB 的地址空间、SDRAM (DMC) 的数据访问映射 128MB 的地址空间、NAND Flash (NFC) 的控制和数据访问映射 1MB 的地址空间、QSPI 的控制和数据访问映射 128MB 的地址空间。

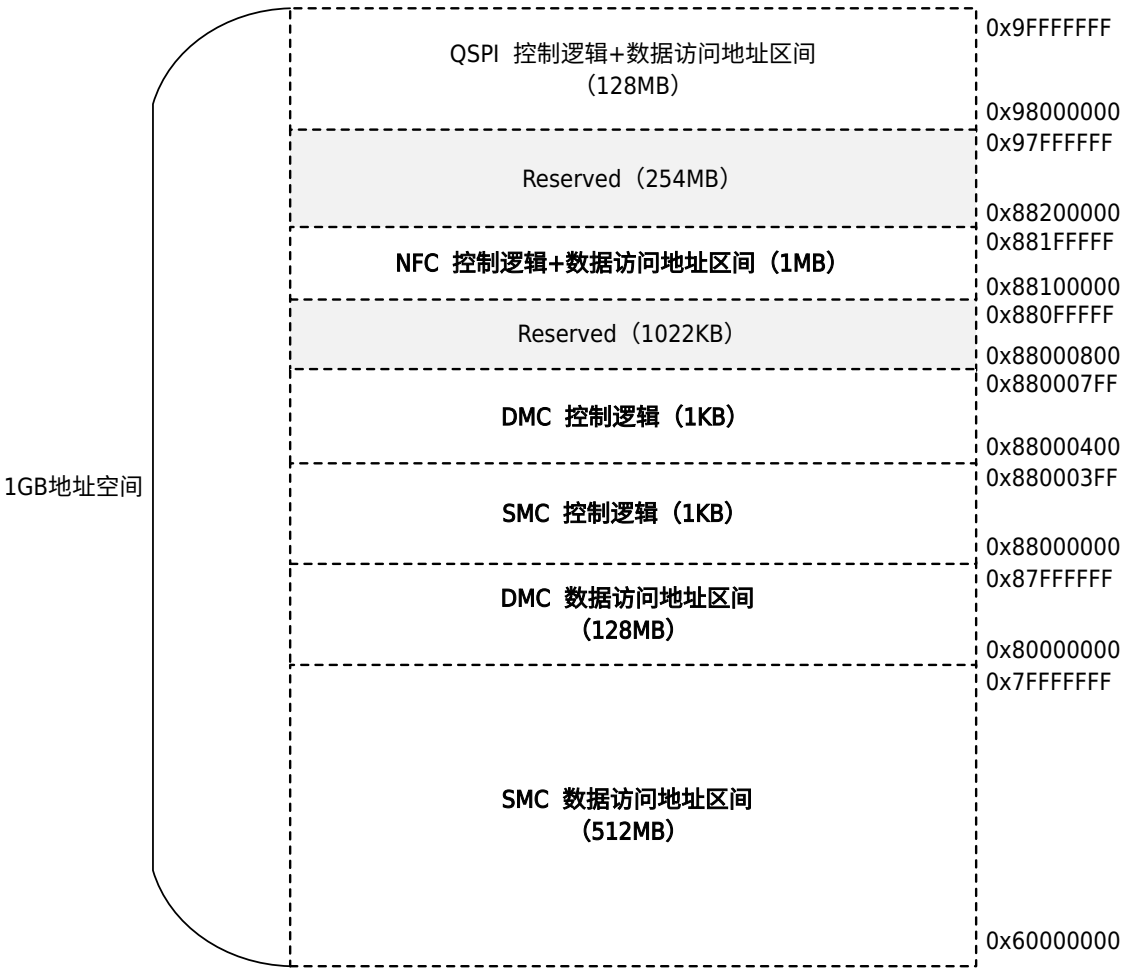


图 40-2 外部空间地址分配

说明

1. 地址空间为 0x98000000~0x9FFFFFFF 的访问控制由 QSPI 模块实现，具体请参考《四线式串行外设接口 (QSPI) 》章节相关描述。

2. DMC 的数据地址空间 0x80000000~0x87FFFFFF 不具有 bufferable 功能，如需提高 DMC 写效率，需要使用 ARM Cortex-M4 自带 MPU 将该地址区间设置为可 bufferable。

SMC 地址空间

SMC 的地址空间可划分为 4 个大小可编程的 Chip，通过片选控制寄存器 (SMC_CSCR0~1) 的设定可以划分各个 Chip 的空间大小。各个 Chip 对应的外部存储器对象可以独立地配置，并对应有不同的 CS (Chip Select) 信号输出至外部端口。下图是 SMC 的 Chip Select 划分，当 CPU 访问的地址空间在某一个 SMC 的 CS 空间时，其对应的 SMC_CS 信号输出就变为有效电平。

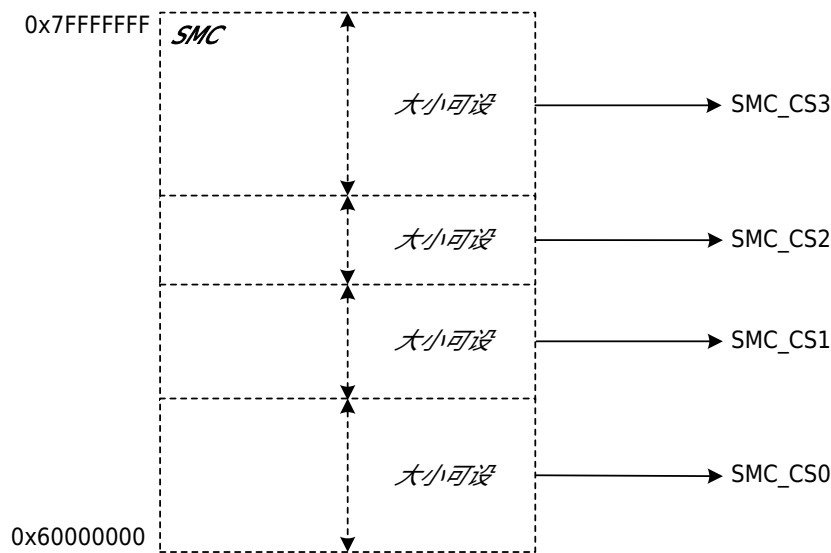


图 40-3 SMC 的地址空间划分



说明

SMC 的地址设定方法，请参见“[片选控制寄存器 \(SMC_CSCR1\)](#)”。

DMC 地址空间

DMC 的地址空间可划分为 4 个大小可编程的 Chip，通过片选控制寄存器 (DMC_CSCR0~3) 的设定可以划分各个 Chip 的空间大小。各个 Chip 对应的外部存储器对象可以独立地配置，并对应有不同的 CS (Chip Select) 信号输出至外部端口。下图是 DMC 的 Chip Select 划分，当 CPU 访问的地址空间在某一个 DMC 的 CS 空间时，其对应的 DMC_CS 信号输出就变为有效电平。

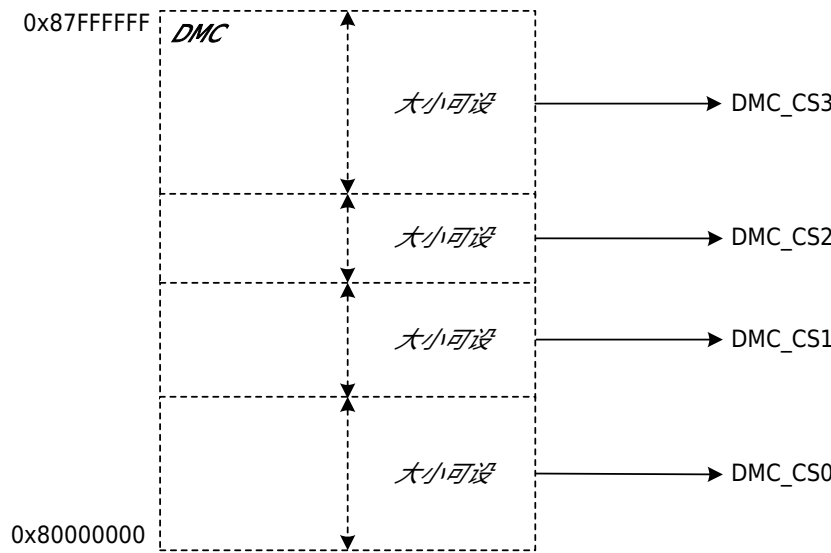


图 40-4 DMC 的地址空间划分



说明

DMC 的地址设定方法，请参见“[片选控制寄存器 \(DMC_CSCR0~3\)](#)”。

NFC 地址空间

NFC 的地址空间分为数据寄存器空间和控制寄存器空间。其中数据寄存器占用 32KB 空间，控制寄存器占用 32KB 空间，共计 64KB，剩余地址为 NFC 预留地址。下图表示了 NFC 的地址空间划分。

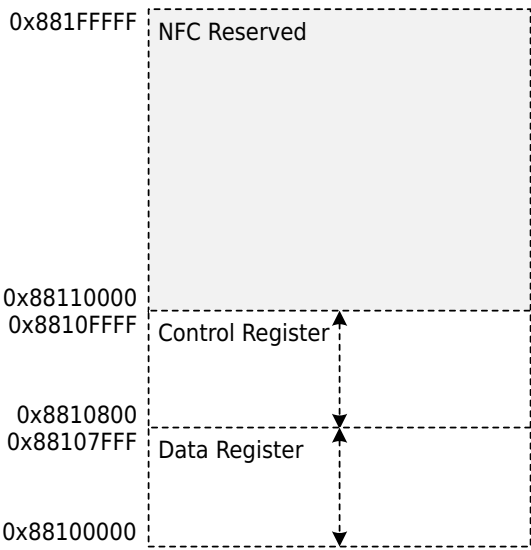


图 40-5 NFC 的地址空间划分

40.3.4 协议接口

SMC、DMC、NFC 对应不同类型的存储器，各自有不同类型的协议接口信号。EXMC 通过该接口协议实现和外部存储器的数据交换。以下为各种类型存储器所需要的协议接口说明。

SMC 协议接口

下表为 SRAM/PSRAM/NOR Flash 控制（SMC）访问所需的协议接口。

表 40-2 SMC 协议接口

协议接口名	方向	有效电平	功能描述
SMC_CLK	输出	高电平/低电平	SMC 的时钟输出
SMC_ADD[28:0]	输出	高电平/低电平	SMC 的地址输出
SMC_DATA[31:0]	输入/输出	高电平/低电平	SMC 的数据
SMC_WE	输出	低电平	SMC 的写使能
SMC_OE	输出	低电平	SMC 的输出使能
SMC_CS[3:0]	输出	低电平	SMC 的片选信号
SMC_BLS[3:0]	输出	低电平	SMC 的字节选通信号
SMC_ADV	输出	低电平	SMC 的地址锁存信号
SMC_CRE	输出	高电平	SMC 的 MdRegConfig 指示信号
SMC_WAIT	输入	低电平	SMC 的输入等待信号

DMC 协议接口

下表为 SDRAM 控制（DMC）访问所需的协议接口。

表 40-3 DMC 协议接口

协议接口名	方向	有效电平	功能描述
DMC_CLK	输出	-	DMC 的时钟输出
DMC_ADD[13:0]	输出	-	DMC 的地址输出
DMC_DATA[31:0]	输入/输出	-	DMC 的数据
DMC_CKE	输出	高电平	DMC 的 CLK 输出使能
DMC_AP	输出	高电平	DMC 的 Auto Precharge 命令指示信号（可不使用）
DMC_WE	输出	低电平	DMC 的写使能
DMC_RAS	输出	低电平	DMC 的行地址选通信号
DMC_CAS	输出	低电平	DMC 的列地址选通信号
DMC_BA[1:0]	输出	-	DMC 的 Bank 地址信号
DMC_CS[3:0]	输出	低电平	DMC 的片选信号
DMC_DQM[3:0]	输出	低电平	DMC 的字节选通信号

NFC 协议接口

下表为 NAND Flash 控制（NFC）访问所需的协议接口。

表 40-4 NFC 协议接口

协议接口名	方向	有效电平	功能描述
NFC_CLE	输出	高电平	NFC 的命令锁存信号
NFC_ALE	输出	高电平	NFC 的地址锁存信号
NFC_DATA[15:0]	输入/输出	-	NFC 的地址和数据
NFC_CE[7:0]	输出	低电平	NFC 的片选信号
NFC_WE	输出	低电平	NFC 的写使能
NFC_RE	输出	低电平	NFC 的读使能
NFC_WP	输出	低电平	NFC 的写保护信号
NFC_RB[7:0]	输入	低电平	NFC 的输入忙信号

EXMC 端口功能配置

基于上述三种类型外部存储器的接口信号表，将 EXMC 端口功能配置分为两种：一次只访问一个外部器件配置和同时访问多个外部器件配置。

一次访问一个外部器件时端口功能配置参考下表。

表 40-5 EXMC 端口功能分配 1

引脚名	方向	有效电平	功能描述		
			SMC	DMC	NFC
EXMC_CLK	输出	高电平/低电平	SMC_CLK	DMC_CLK	-
EXMC_ADD28~18	输出	高电平/低电平	SMC_ADD[28:18]	-	-
EXMC_ADD17_DMC_BA1_NFC_ALE	输出	高电平/低电平	SMC_ADD[17]	DMC_BA[1]	-
EXMC_ADD16_DMC_BA0_NFC_CLE	输出	高电平/低电平	SMC_ADD[16]	DMC_BA[0]	-
EXMC_ADD15_DMC_BA1	输出	高电平/低电平	SMC_ADD[15]	-	-
EXMC_ADD14_DMC_BA0	输出	高电平/低电平	SMC_ADD[14]	-	-
EXMC_ADD13~0	输出	高电平/低电平	SMC_ADD[15:0]	DMC_ADD[13:0]	-
EXMC_DATA31~0	输入/输出	高电平/低电平	SMC_DATA[31:16]	DMC_DATA[31:16]	-
	输入/输出	高电平/低电平	SMC_DATA[15:0]	DMC_DATA[15:0]	NFC_DATA[15:0]
EXMC_WE	输出	低电平	SMC_WE	DMC_WE	NFC_WE
EXMC_CE7~4	输出	低电平	SMC_BLS[3:0]	DMC_DQM[3:0]	NFC_CE[7:4]
EXMC_CE3~0	输出	低电平	SMC_CS[3:0]	DMC_CS[3:0]	NFC_CE[3:0]
EXMC_OE	输出	低电平	SMC_OE	DMC_RAS	NFC_RE
EXMC_BAA	输出	低电平	-	DMC_CAS	NFC_WP
EXMC_ADV	输出	低电平	SMC_ADV	-	-
EXMC_ALE	输出	高电平	SMC_CRE	DMC_CKE	NFC_ALE
EXMC_CLE	输出	高电平	-	DMC_AP	NFC_CLE
EXMC_RB7~1	输入	低电平	-	-	NFC_RB[7:1]
EXMC_RB0	输入	低电平	SMC_WAIT	-	NFC_RB[0]

同时访问多个外部器件功能配置如下表，该功能配置也可用于一次访问一个外部器件。



说明

1. NFC 不支持与 SMC/DMC 之一或两者同时使用。
2. 禁止通过 CPU 直接在 DMC 和 SMC 之间直接搬运数据，如果需要在 SMC 与 DMC 之间搬运数据可通过 DMA 搬运或者通过系统 RAM 中继搬运。

表 40-6 EXMC 端口功能分配 2

引脚名	方向	有效电平	功能描述		
			SMC	DMC	NFC
EXMC_DMC_CLK	输出	高电平/低电平	-	DMC_CLK	-
EXMC_SMC_CLK	输出	高电平/低电平	SMC_CLK	-	-
EXMC_ADD28~18	输出	高电平/低电平	SMC_ADD[28:18]	-	-
EXMC_ADD17_NFC_ALE	输出	高电平/低电平	SMC_ADD[17]	-	NFC_ALE
EXMC_ADD16_NFC_CLE	输出	高电平/低电平	SMC_ADD[16]	-	NFC_CLE
EXMC_ADD15_DMC_BA1	输出	高电平/低电平	SMC_ADD[15]	DMC_BA[1]	-
EXMC_ADD14_DMC_BA0	输出	高电平/低电平	SMC_ADD[14]	DMC_BA[0]	-
EXMC_ADD13~0	输出	高电平/低电平	SMC_ADD[13:0]	DMC_ADD[13:0]	-
EXMC_DATA31~0	输入/输出	高电平/低电平	SMC_DATA[31:16]	DMC_DATA[31:16]	-
	输入/输出	高电平/低电平	SMC_DATA[15:0]	DMC_DATA[15:0]	NFC_DATA[15:0]

引脚名	方向	有效电平	功能描述		
			SMC	DMC	NFC
EXMC_DMC_WE	输出	低电平	-	DMC_WE	-
EXMC_SMC_NFC_WE	输出	低电平	SMC_WE	-	NFC_WE
EXMC_SMC_OE_NFC_RE	输出	低电平	SMC_OE	-	NFC_RE
EXMC_SMC_DMC_BL S3~0	输出	低电平	SMC_BLS[3:0]	DMC_DQM[3:0]	-
EXMC_SMC_NFC_CS 3~0	输出	低电平	SMC_CS[3:0]	-	NFC_CE[3:0]
EXMC_DMC_CS3~0	输出	低电平	-	DMC_CS[3:0]	-
EXMC_NFC_CE7~4	输出	低电平	-	-	NFC_CE[7:4]
EXMC_DMC_RAS	输出	低电平	-	DMC_RAS	-
EXMC_SMC_BAA_NF C_WP	输出	低电平	-	-	NFC_WP
EXMC_DMC_CAS	输出	低电平	-	DMC_CAS	-
EXMC_SMC_ADV	输出	低电平	SMC_ADV	-	-
EXMC_DMC_CKE	输出	高电平	-	DMC_CKE	-
EXMC_SMC_CRE	输出	高电平	SMC_CRE	-	-
EXMC_DMC_AP	输出	高电平	-	DMC_AP	-
EXMC_RB7~1	输入	低电平	-	-	NFC_RB[7:1]
EXMC_RB0	输入	低电平	SMC_WAIT	-	NFC_RB[0]

40.3.5 SMC-SRAM/PSRAM/NOR Flash 控制器

40.3.5.1 SRAM/PSRAM/NOR Flash 介绍

SRAM/NOR Flash 都是具有静态存取功能的存储器，不需要刷新电路即能保存它内部存储的数据。其地址线的宽度决定了存储单元的数量，每一个存储单元上对应存储器存储宽度大小的空间用于存储数据，因此 SRAM/PSRAM/NOR Flash 存储器的容量大小是存储器数据宽度 $\times 2^{\text{地址线宽度}}$ 。

PSRAM 是假静态随机存储器，由于它具有 SRAM 的接口，因此能像 SRAM 一样，与系统内部通过 EXMC 的 SMC 模块实现数据交换。

40.3.5.2 SMC 基本功能

SMC 基本功能特性如下：

- 支持 8 位、16 位、32 位外部存储器数据带宽
- AHB 字、半字、字节访问
- 为每个存储器 Chip 提供独立的片选控制
- 字节选择信号输出
- 地址线、数据线复用
- 可编程的协议时序参数
- 可编程速率的自动刷新动作（PSRAM 时使用）
- 具有 2 个命令 FIFO
- 具有 4 个 32 位的写数据 FIFO
- 具有 4 个 32 位的读数据 FIFO
- 低功耗管理
- 支持 LCD 并行接口，兼容 8080/6800 模式

40.3.5.3 SMC 初始设定

SMC 在上电复位之后的状态图及状态之间的切换如下图所示。
初始复位时 SMC 为 Low Power 状态；通过状态控制寄存器（SMC_STCR0 和 SMC_STCR1）的设定，SMC 可以在 Ready 状态和 Low Power 状态之间进行切换，具体操作请参见“SMC 低功耗管理”。

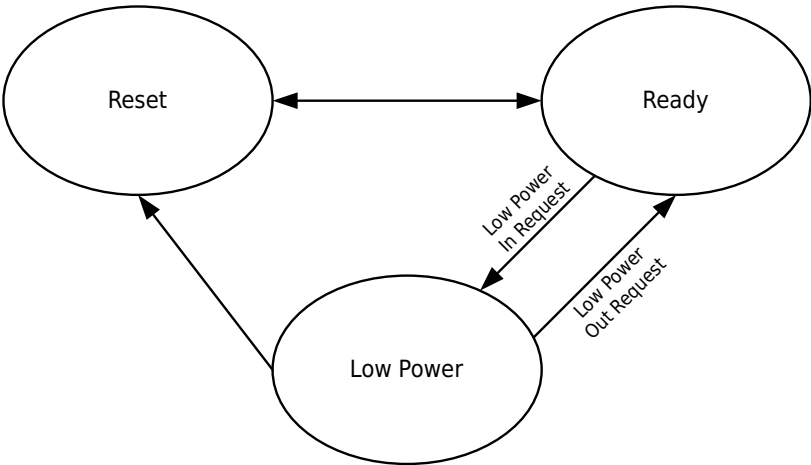
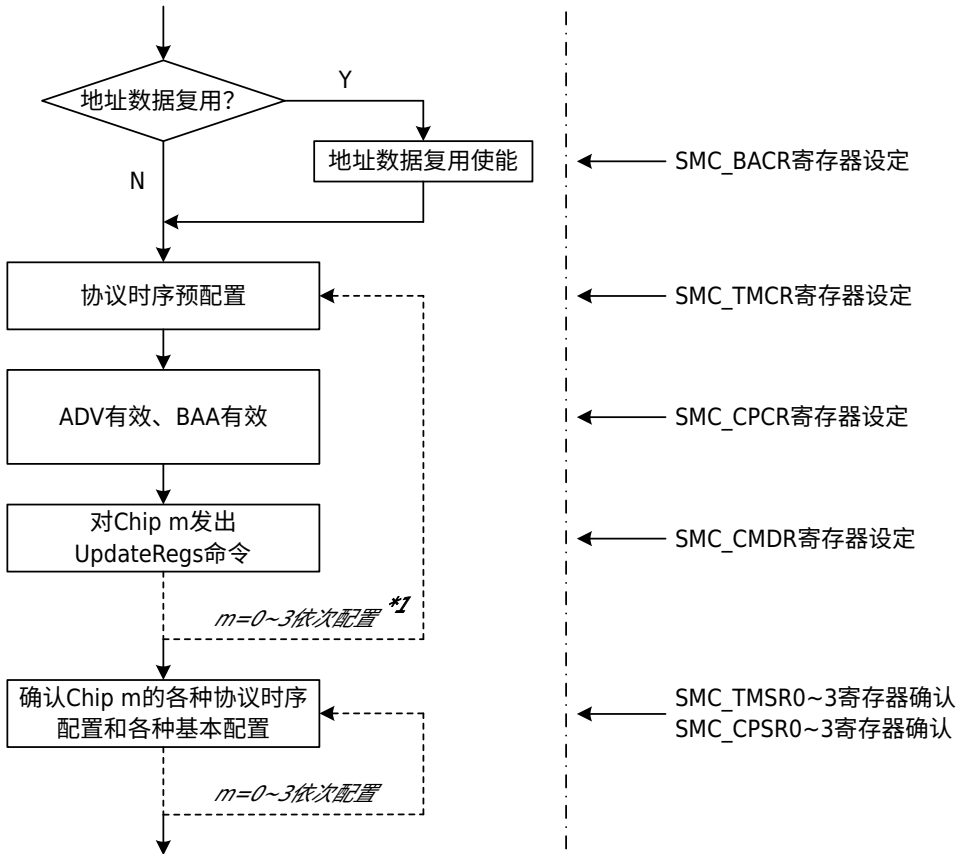


图 40-6 SMC 状态图

SMC 在和外部 SRAM、PSRAM、NOR Flash 等进行通信之前，必须进行初始设定，配置好相关的参数，以保证能进行正确的数据传输。具体的设定顺序参考下图。



注1：地址数据复用使能，配置T_ADV时，必须对Chip 0发出UpdateRegs命令

图 40-7 SMC 初始设定流程

40.3.5.4 SMC 访问动作

40.3.5.4.1 访问方式

SMC 对外部存储器的读写，可以选择同步访问方式或者异步访问方式。

同步访问方式是对应同步访问类型的外部存储器，SMC 在读写控制信号等输出时，时钟信号也同时输出，外部存储器根据 SMC 输出的时钟与 SMC 进行数据传输，即 SMC_CLK 有效；异步访问方式是对应异步访问类型的外部存储器，SMC 在读写控制信号等输出时，时钟信号不输出（输出始终为低），外部存储器依据其它控制信号实现数据访问。

在同步访问方式时，SMC 和外部存储器间通过 SMC_WAIT 信号来控制读写动作。在 SMC_WAIT 信号输入低电平时，SMC 处于等待状态，不执行任何操作，等到 SMC_WAIT 变为高电平后 SMC 才正常的进行读写访问。

通过设定 CHIP 配置寄存器（SMC_CPCR）的 RSYN、WSYN 位，可选择同步读写方式或异步读写方式。

 **说明**

若没有引脚配置为 EXMC_RB0 功能（详见《数据手册》“引脚配置及功能-引脚功能说明”），SMC 的读写动作将不受 SMC_WAIT 信号的控制。

40.3.5.4.2 访问时序

- SMC 支持多种类型的存储器的访问，可以把 AHB 的单次或突发读写操作转换成存储器的读写操作。下面列出几种控制器与外部存储器进行数据读写的时序例：
- 1. 单次读动作
 - 2. 地址数据线复用的单次读动作
 - 3. 单次写动作
 - 4. 地址数据线复用异步方式的单次写动作
 - 5. 地址数据线复用同步方式的单次写动作

单次读动作

单次读动作的基本时序图和设定例如下：

表 40-7 单次读动作基本设定例

基本设定	MW	RSYN	WSYN	-	ADVS	BLSS	-表示不关注 <set>表示 用户设定值
	<set>	<set>	-	-	0b0	0b0	
时序设定	T_RC	T_WC	T_CEOE	T_WP	T_TR	T_ADV	
	0b0011	-	0b001	-	0b001	-	

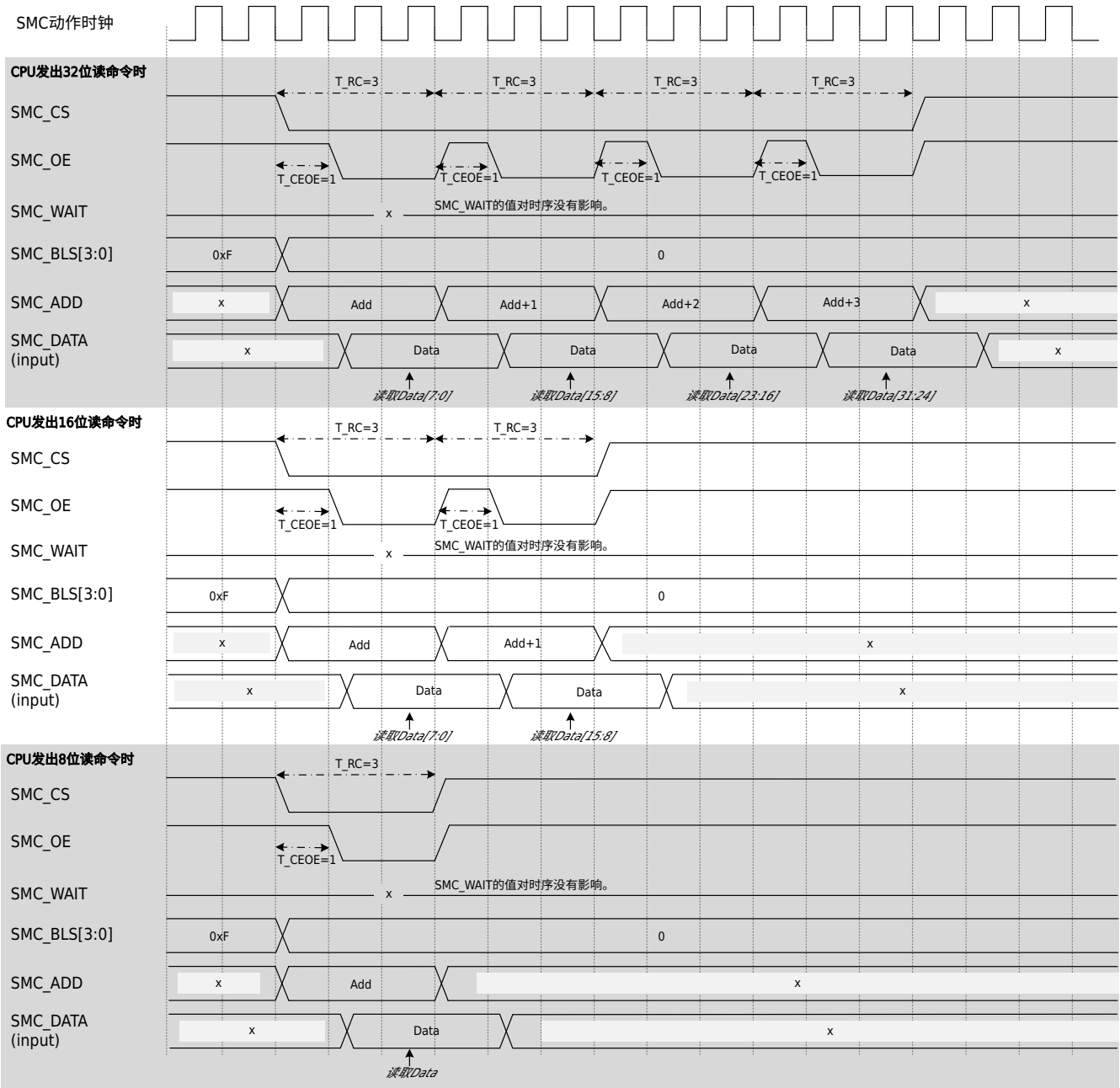


图 40-8 单次读动作基本时序（异步方式（RSYN=0b0）&8 位位宽（MW=0b00））

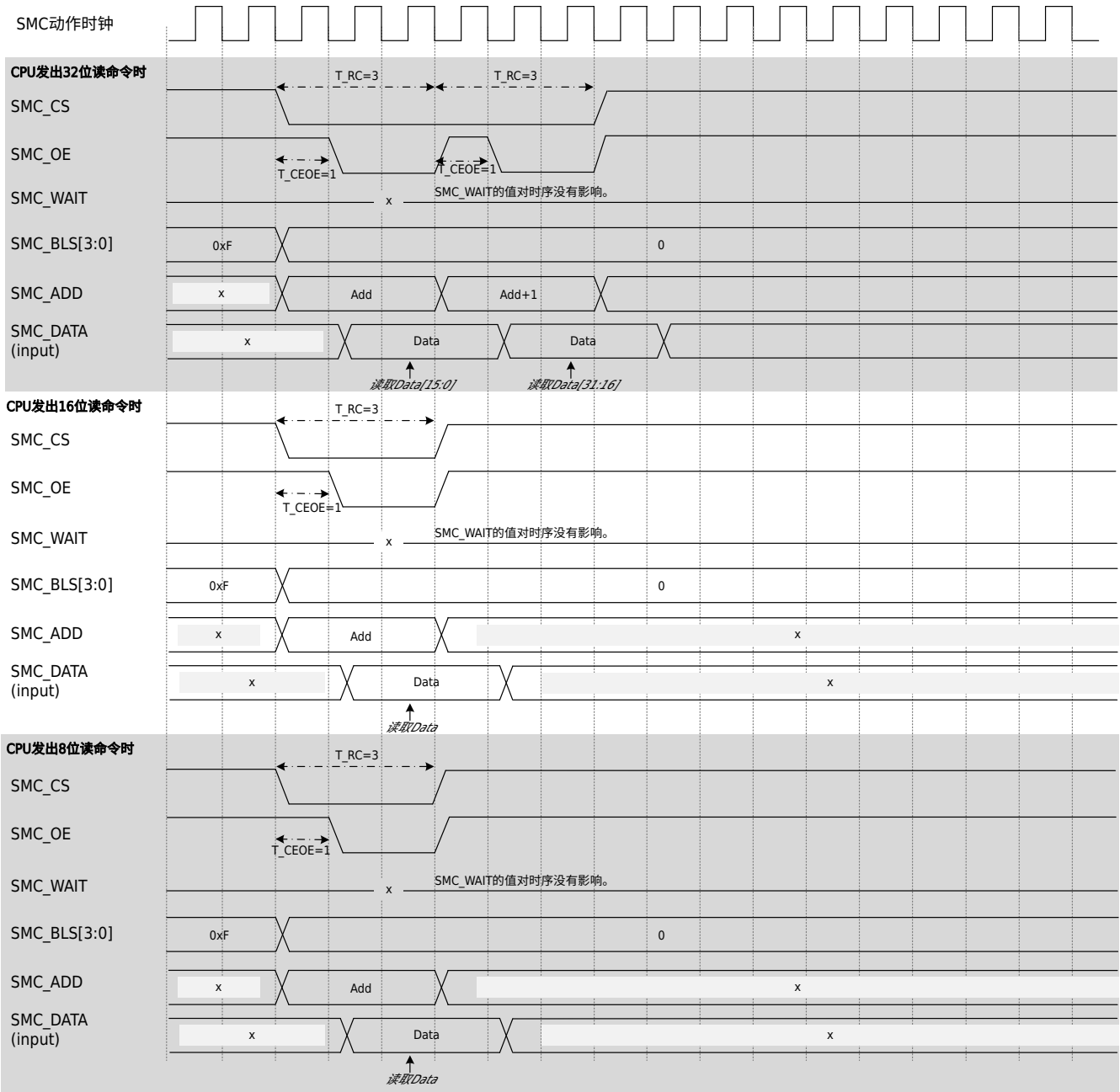


图 40-9 单次读动作基本时序（异步方式（RSYN=0b0）&16 位位宽（MW=0b01））

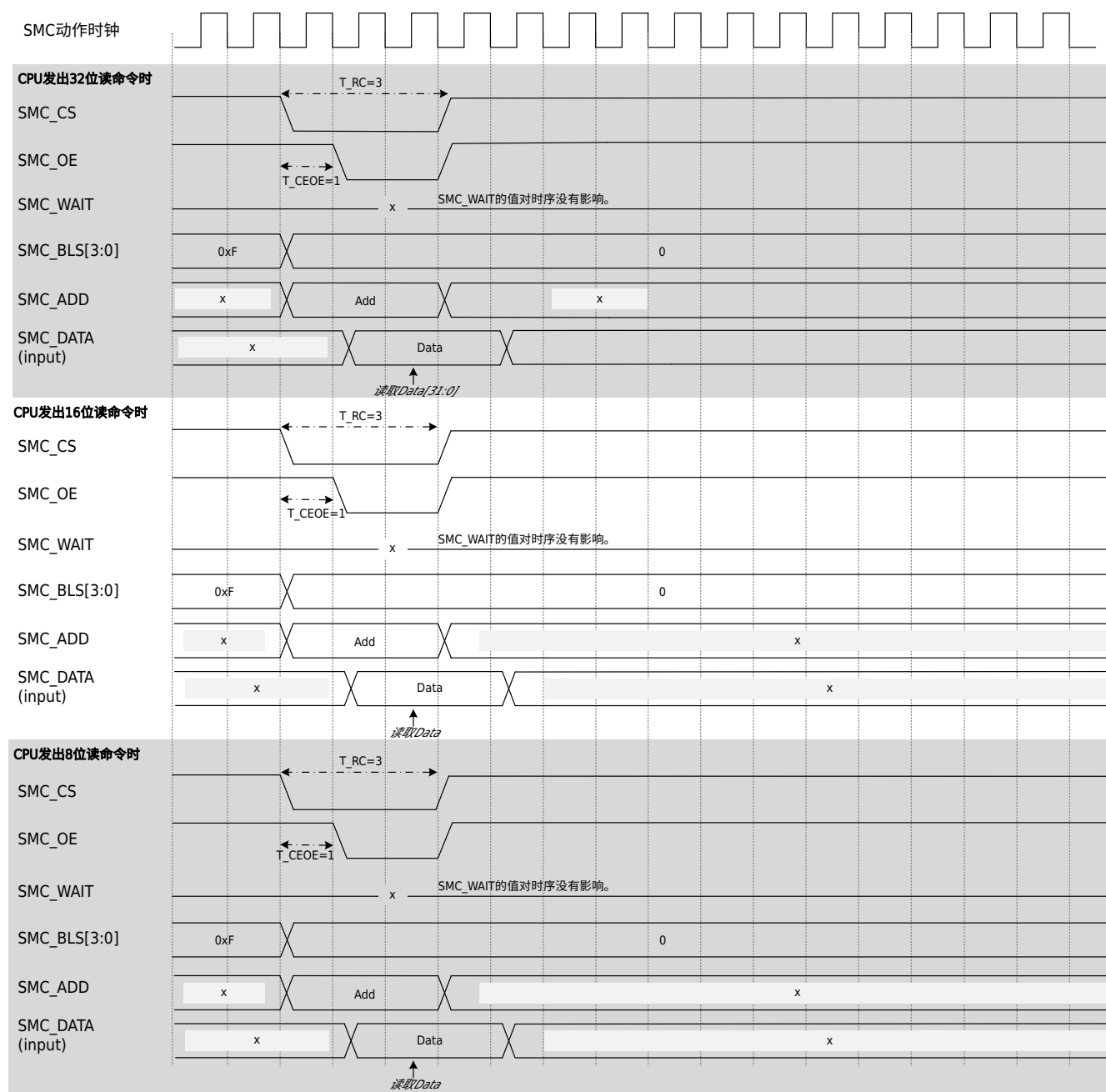


图 40-10 单次读动作基本时序（异步方式（RSYN=0b0）&32 位位宽（MW=0b10））

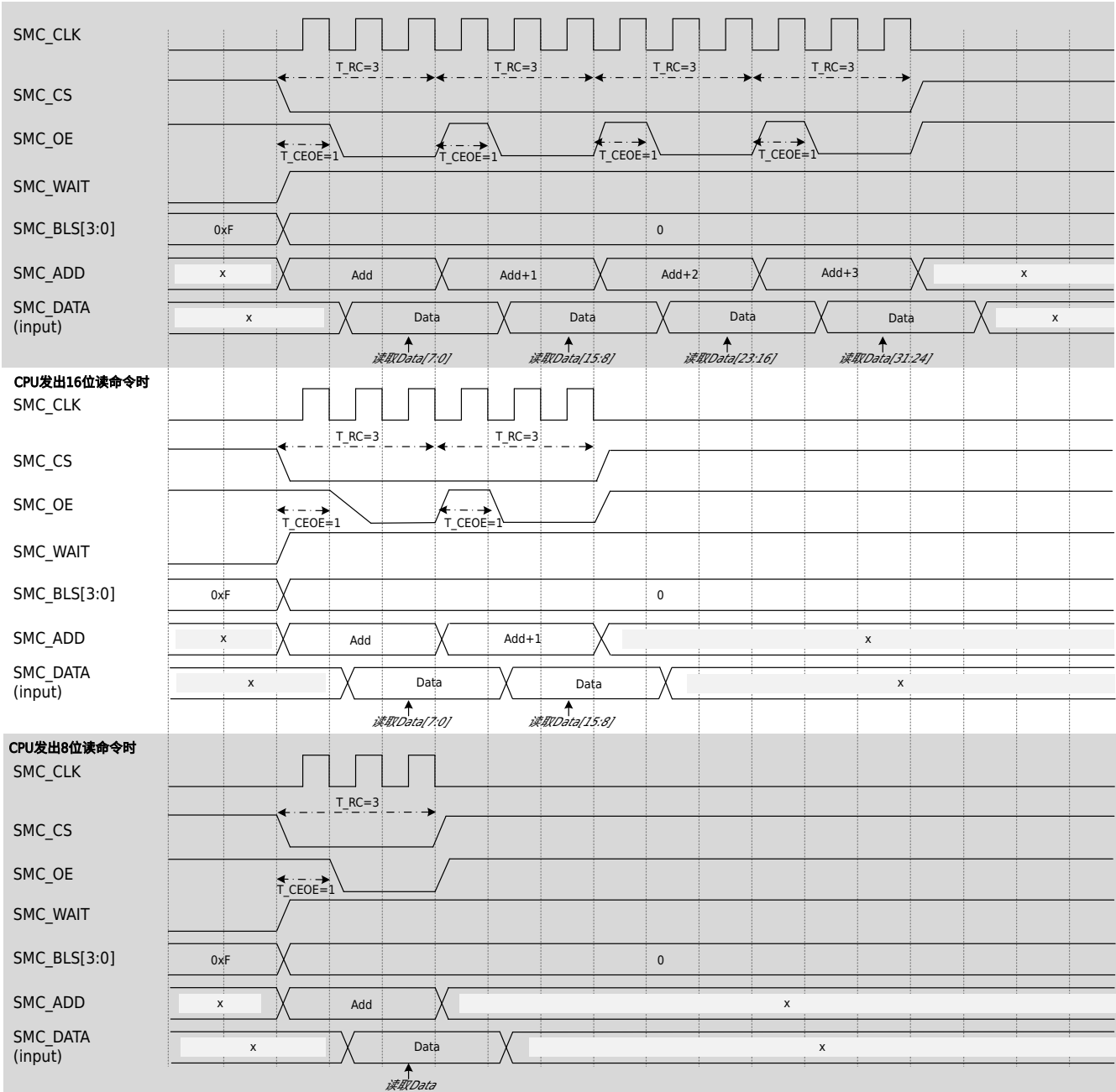


图 40-11 单次读动作基本时序（同步方式（RSYN=0b1）&8 位位宽（MW=0b00））

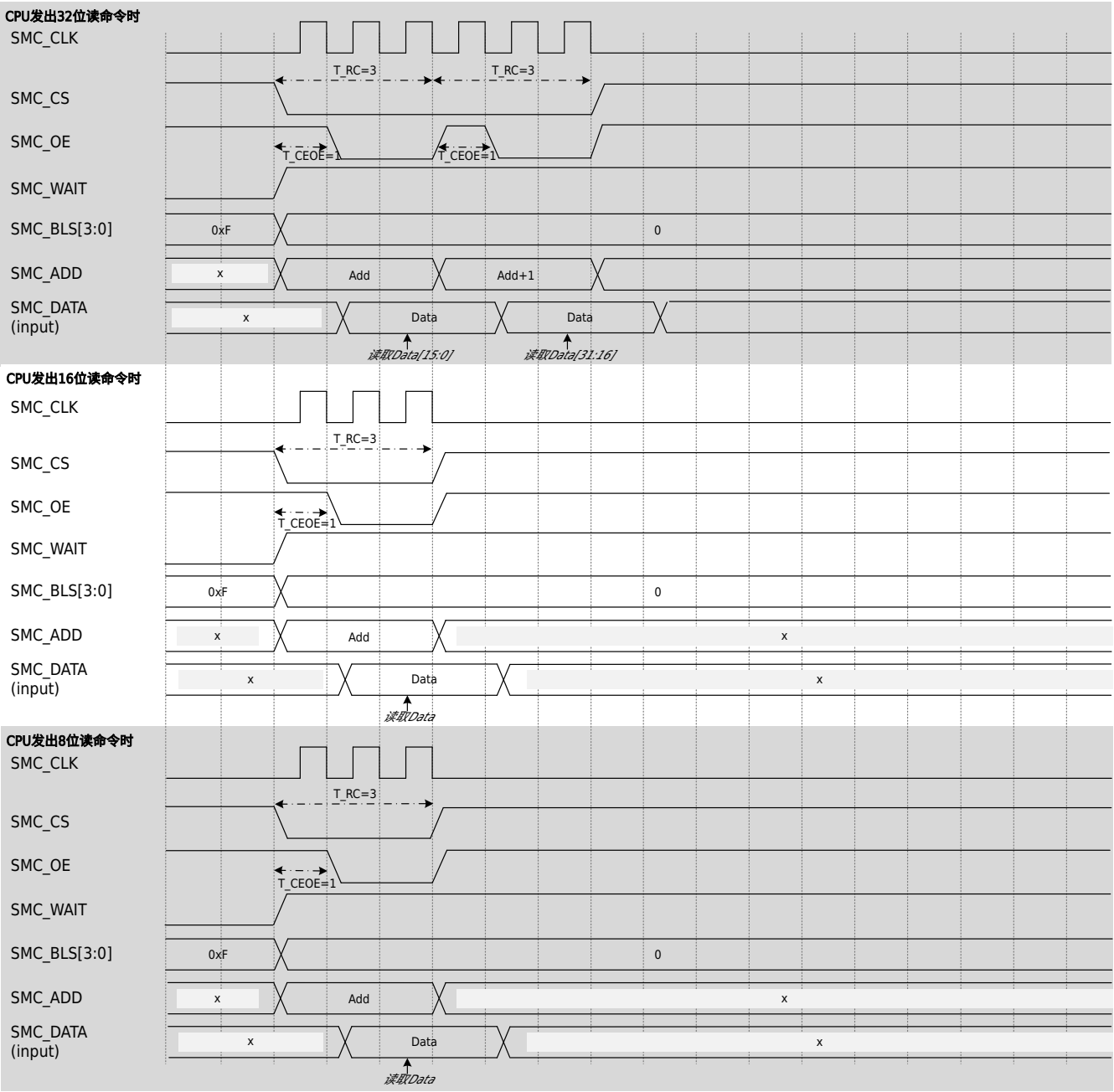


图 40-12 单次读动作基本时序（同步方式（RSYN=0b1）&16 位位宽（MW=0b01））

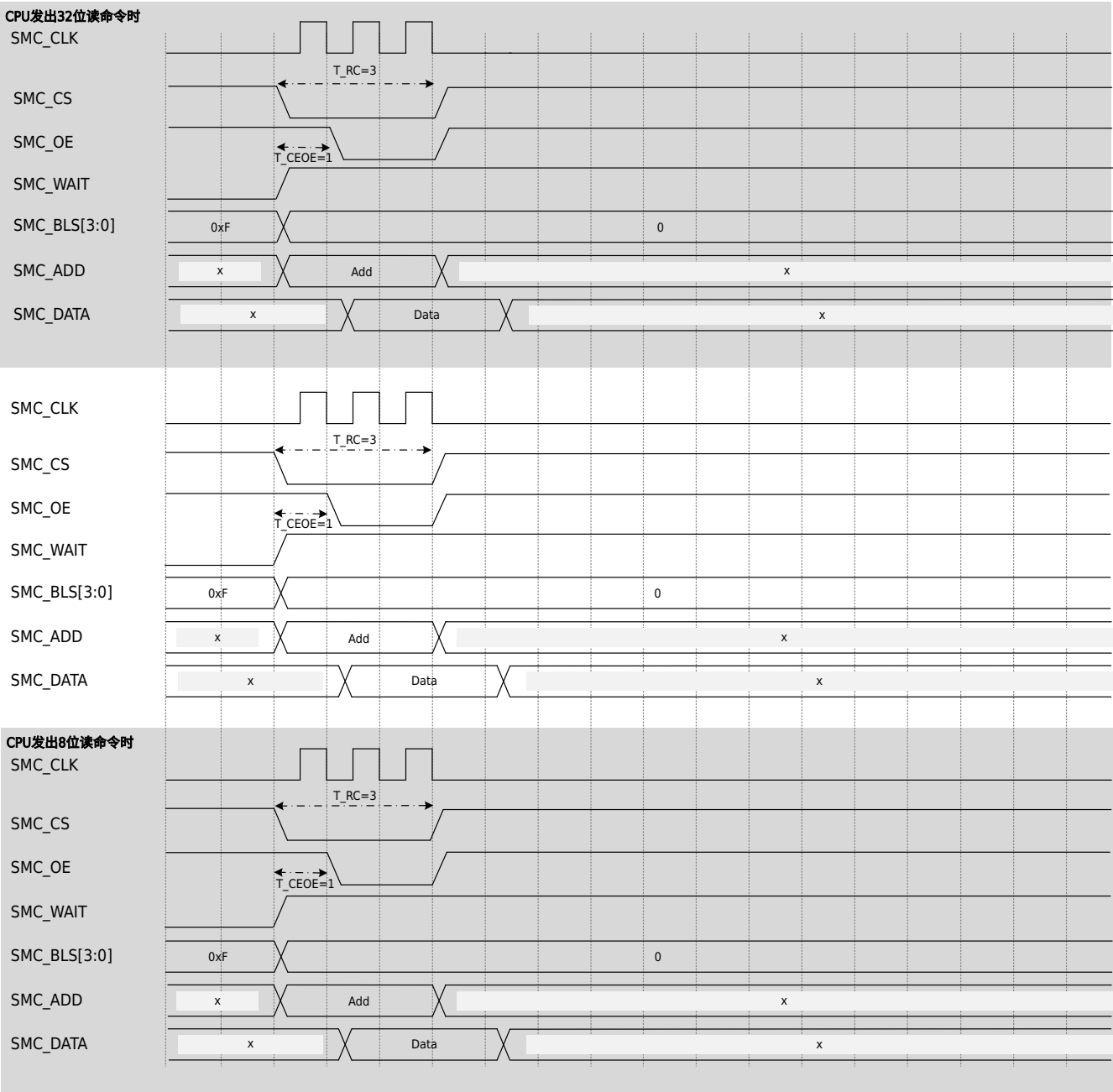


图 40-13 单次读动作基本时序（同步方式（RSYN=0b1）&32 位位宽（MW=0b10））

地址数据线复用的单次读动作

地址数据线复用的单次读动作的基本时序图和设定例如下：

表 40-8 地址数据线复用单次读动作基本设定例

基本设定	MW	RSYN	WSYN	-	ADVS	BLSS	-表示不关注 <set>表示 用户设定值
	<set>	<set>	-	-	0b1	0b0	
时序设定	T_RC	T_WC	T_CEOE	T_WP	T_TR	T_ADV	
	0b0110	-	0b011	-	0b010	0b001	

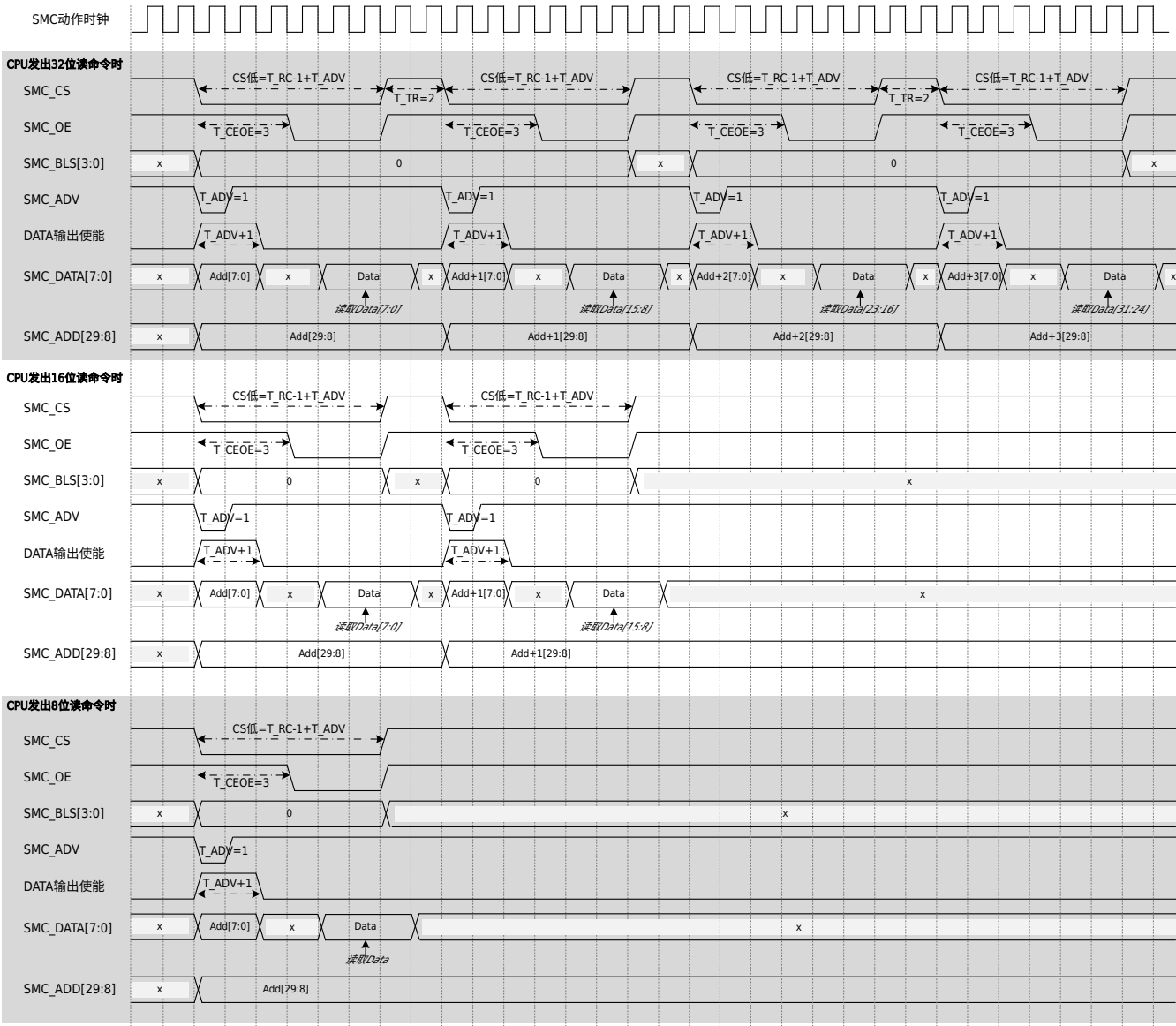


图 40-14 地址数据线复用单次读动作基本时序（8 位位宽（MW=0b00））

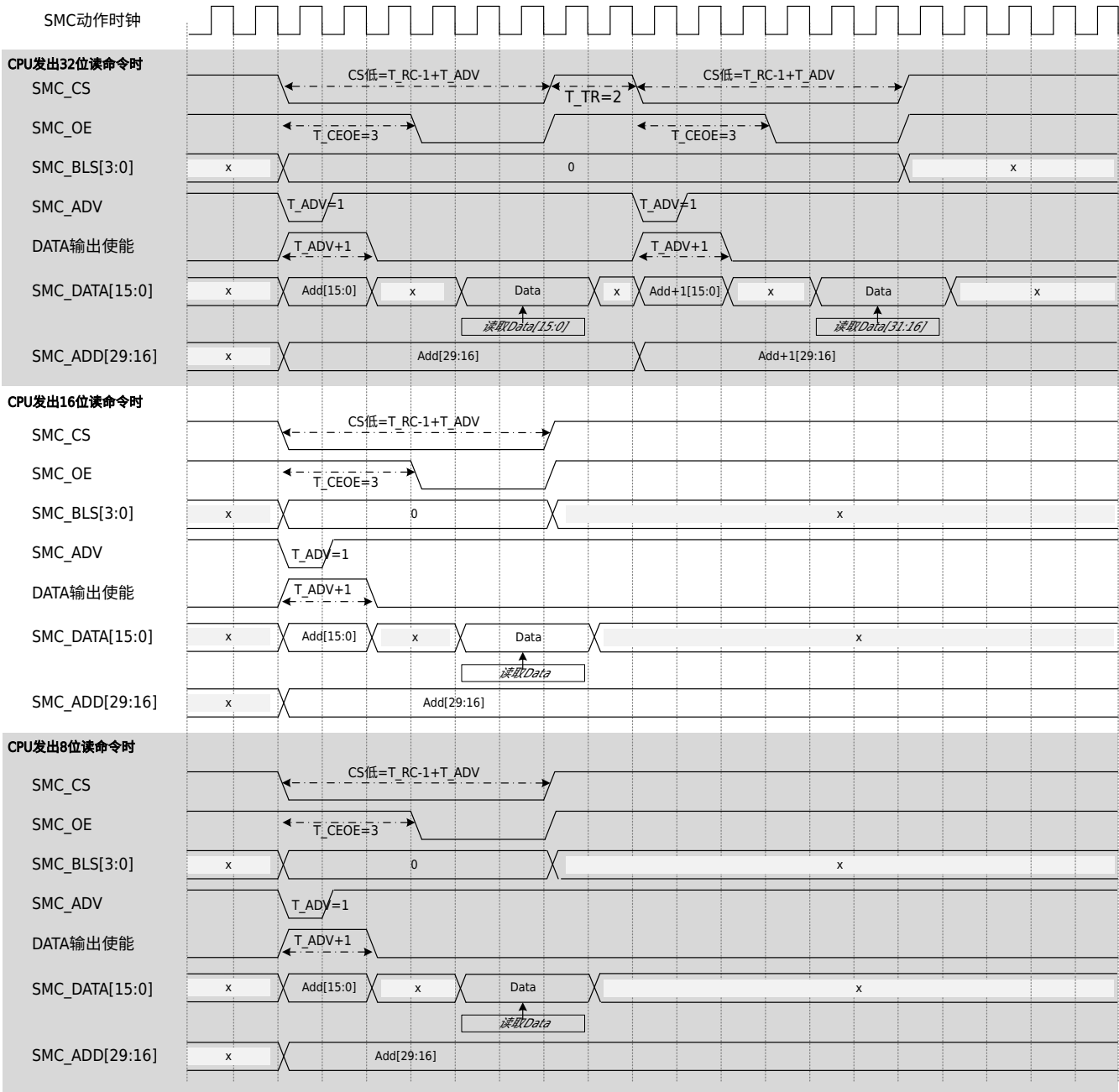


图 40-15 地址数据线复用单次读动作基本时序（16 位位宽（MW=0b01））

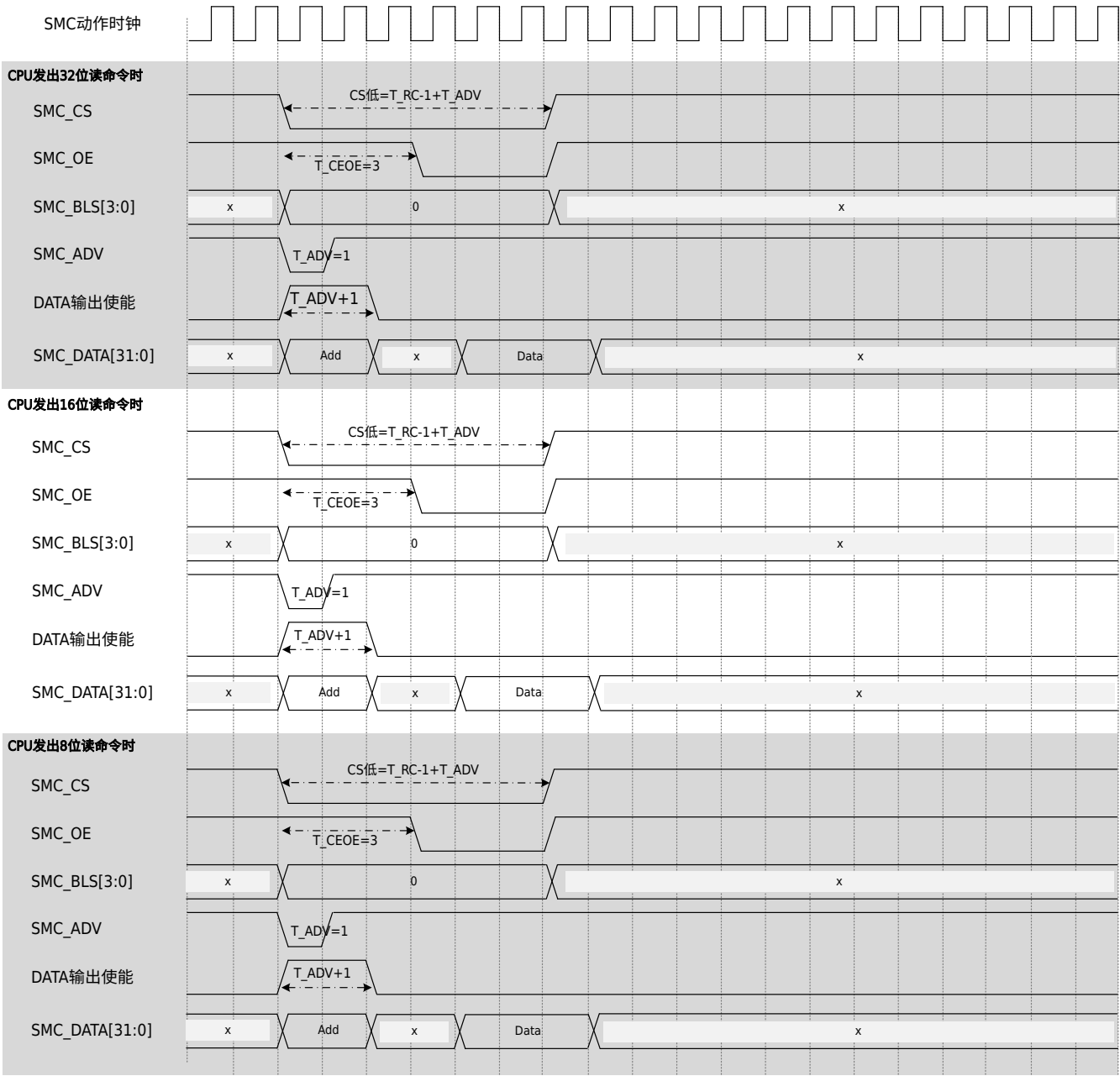


图 40-16 地址数据线复用单次读动作基本时序 (32 位位宽 (MW=0b10))

单次写动作

单次写动作的基本时序图和设定例如下：

表 40-9 单次写动作基本设定例

基本设定	MW	RSYN	WSYN	-	ADVS	BLSS	-表示不关注 <set>表示 用户设定值
	<set>	-	<set>	-	0b0	<set>	
时序设定	T_RC	T_WC	T_CEOE	T_WP	T_TR	T_ADV	
	-	0b0100	-	0b010	0b001	-	

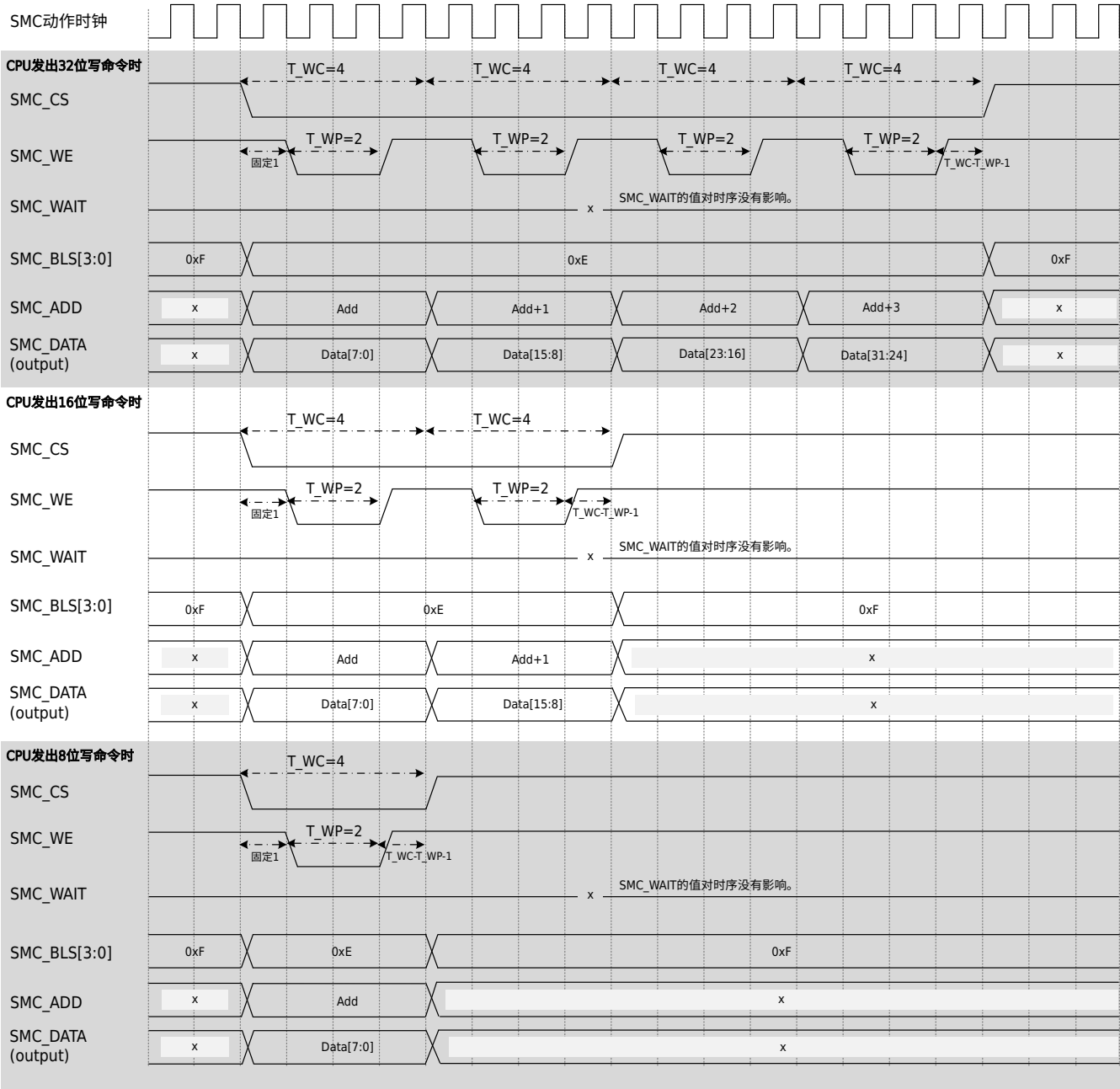


图 40-17 单次写动作基本时序（异步方式（WSYN=0b0）&8 位位宽（MW=0b00）&BLSS=0b0）

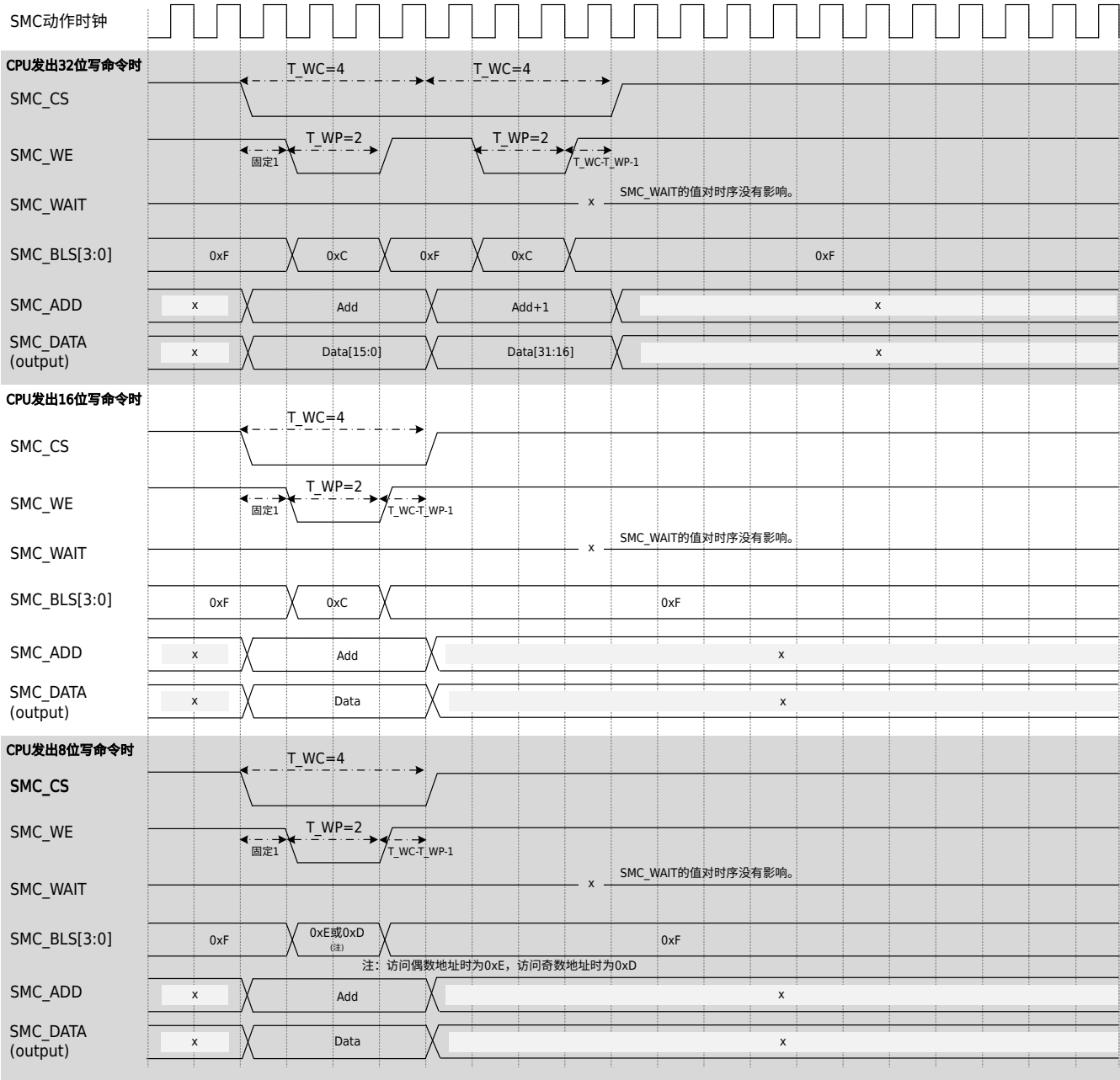


图 40-18 单次写动作基本时序（异步方式（WSYN=0b0）&16 位位宽（MW=0b01）&BLSS=0b1）

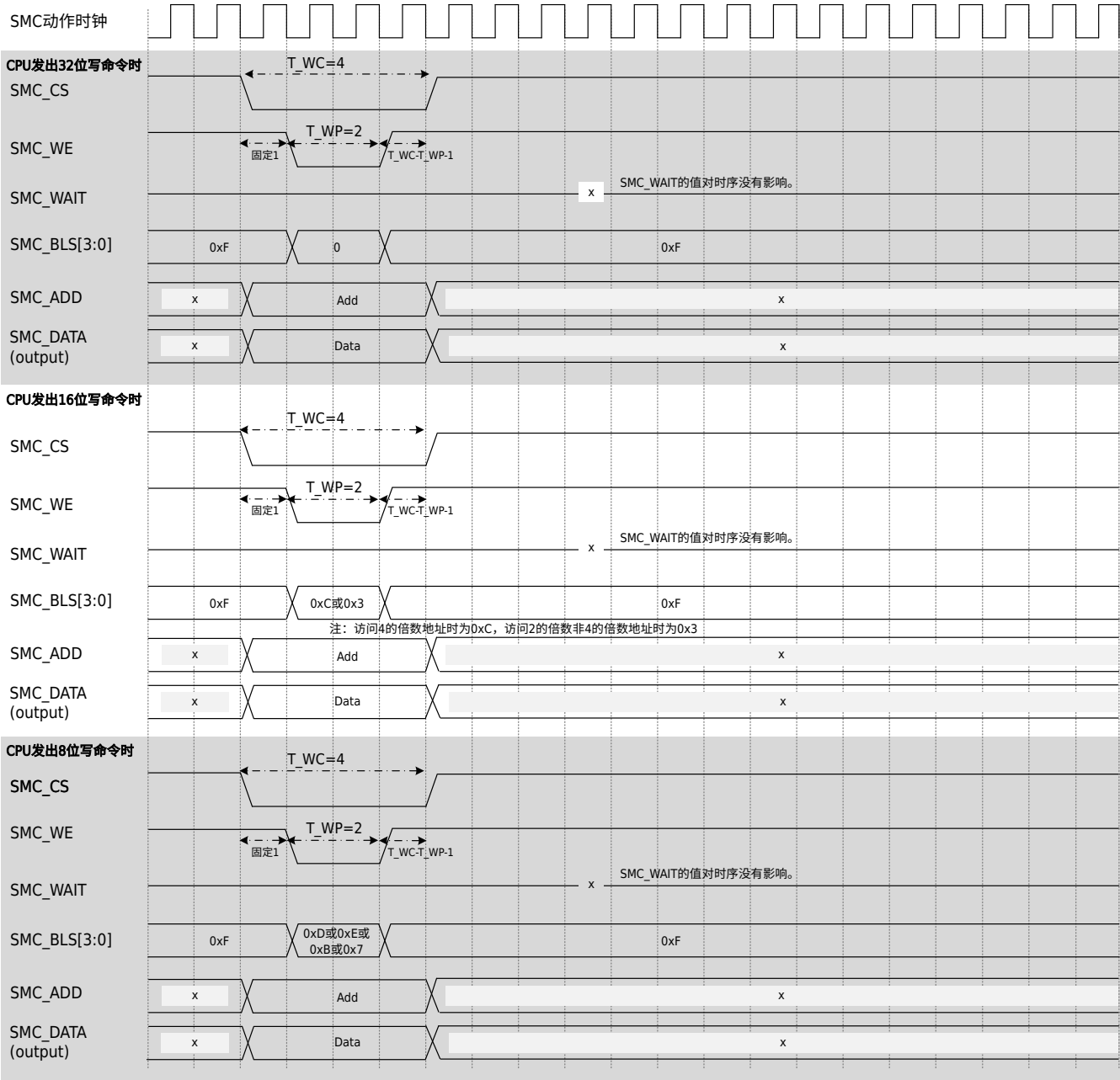


图 40-19 单次写动作基本时序（异步方式（WSYN=0b0）&32 位位宽（MW=0b10）&BLSS=0b1）

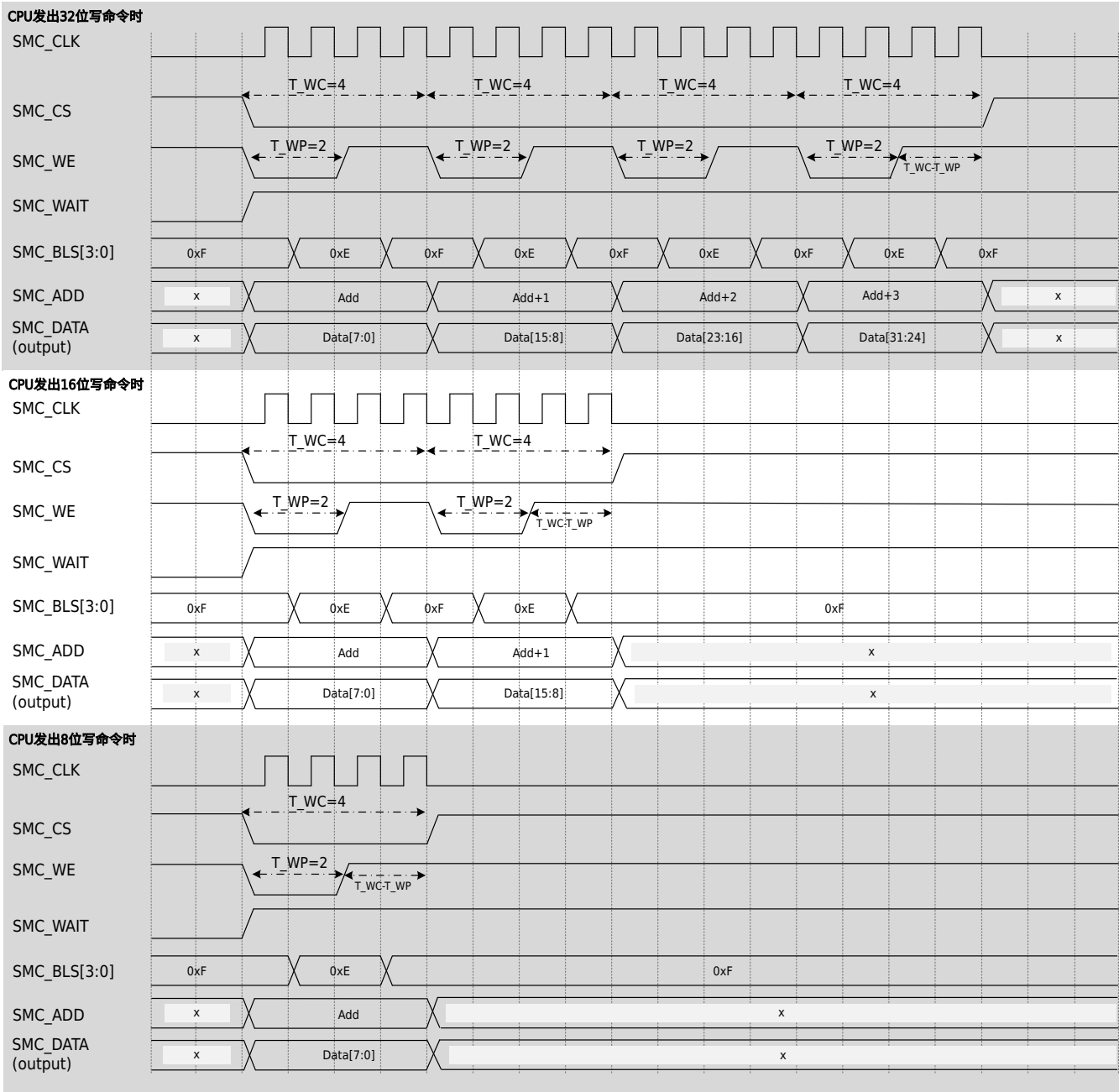


图 40-20 单次写动作基本时序（同步方式（WSYN=0b1）&8 位位宽（MW=0b00）&BLSS=0b1）

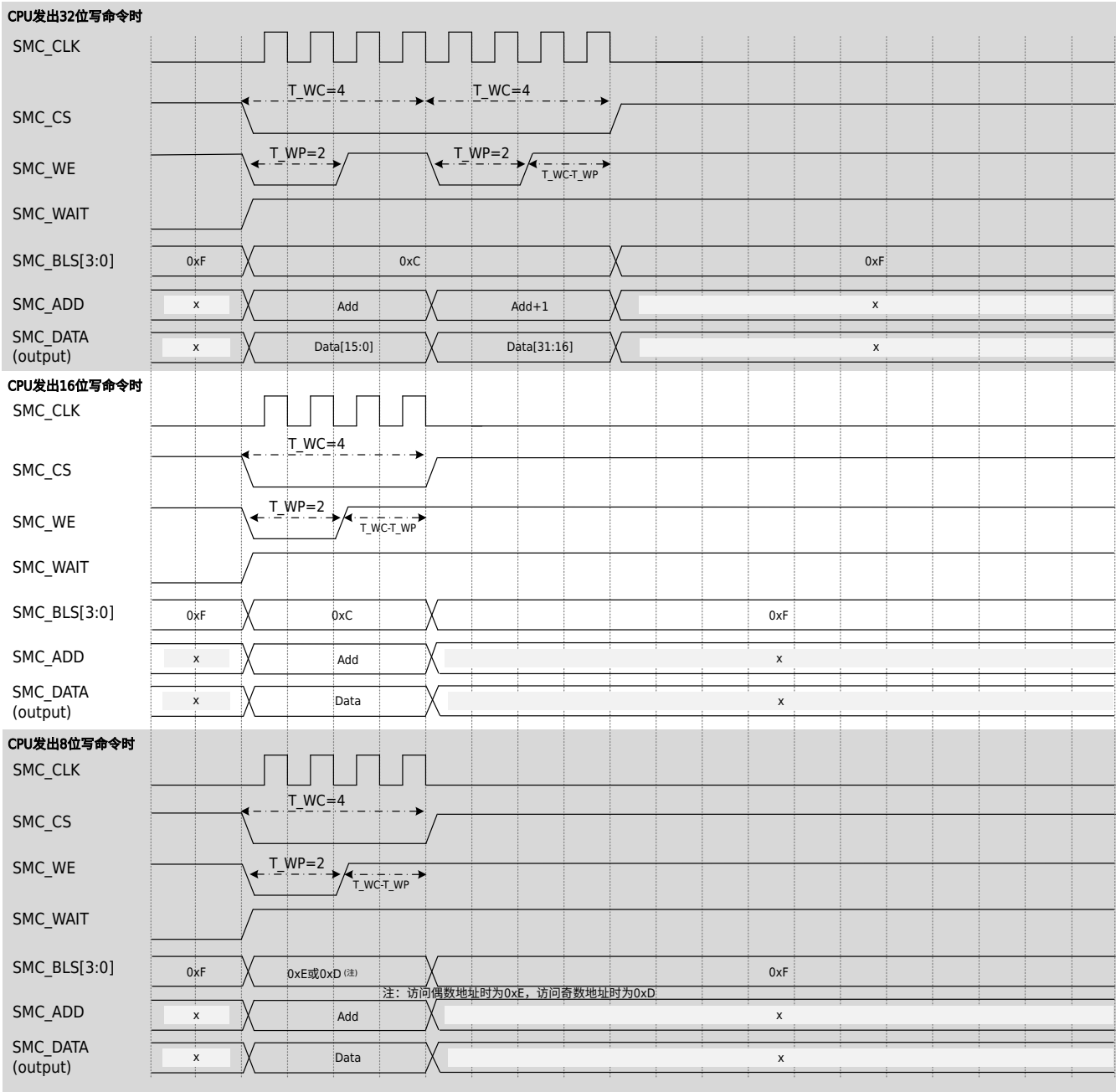


图 40-21 单次写动作基本时序（同步方式（WSYN=0b1）&16 位位宽（MW=0b01）&BLSS=0b0）

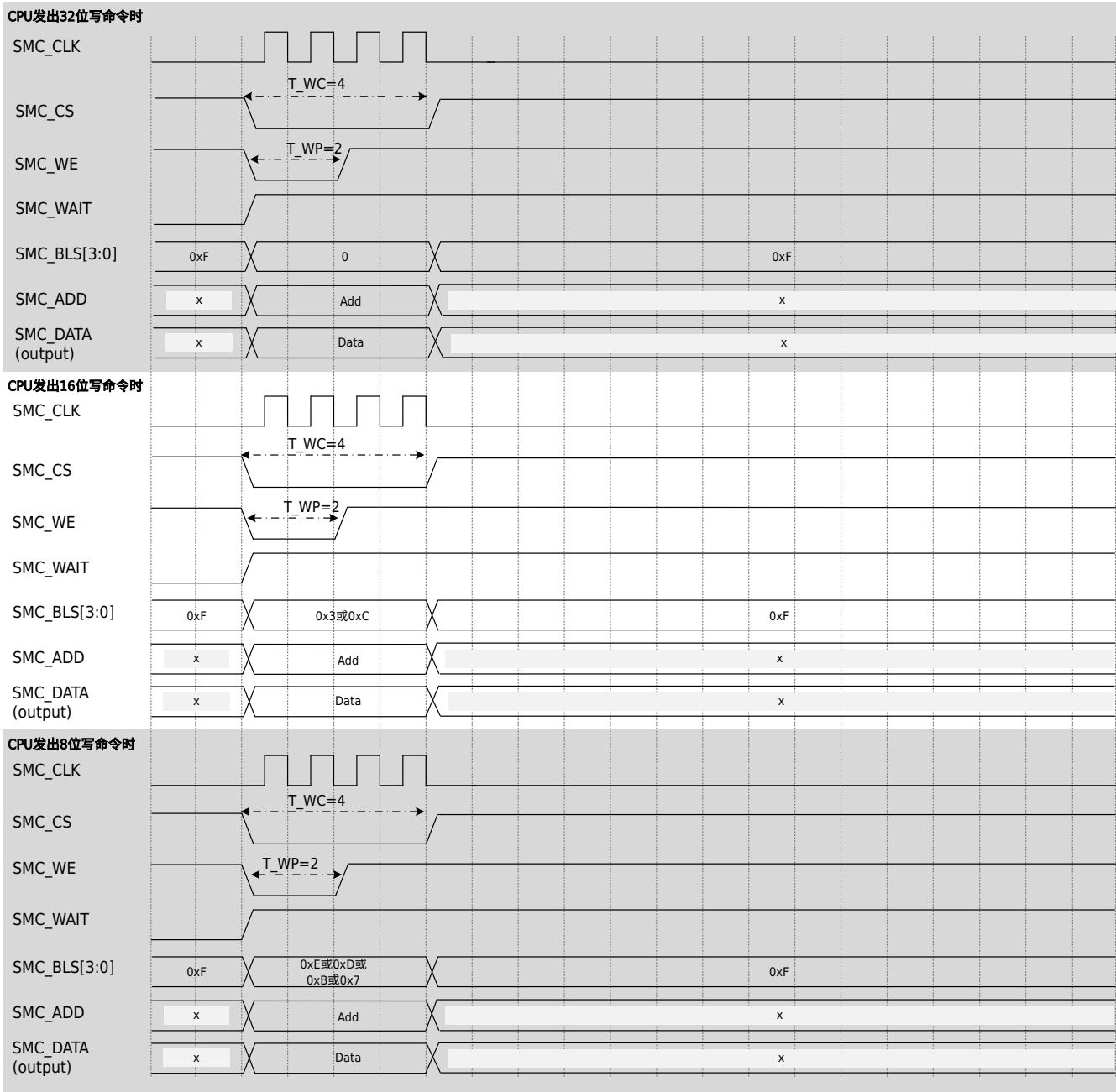


图 40-22 单次写动作基本时序（同步方式（WSYN=0b1）&32 位位宽（MW=0b10）&BLSS=0b0）

地址数据线复用异步方式的单次写动作

地址数据线复用异步方式的单次写动作的基本时序图和设定例如下：

表 40-10 地址数据线复用异步方式单次写动作基本设定例

基本设定	MW	RSYN	WSYN	-	ADVS	BLSS	-表示不关注 <set>表示 用户设定值
	<set>	-	0	-	0b1	0b1	
时序设定	T_RC	T_WC	T_CEOE	T_WP	T_TR	T_ADV	
	-	0b0101	-	0b010	0b010	0b001	

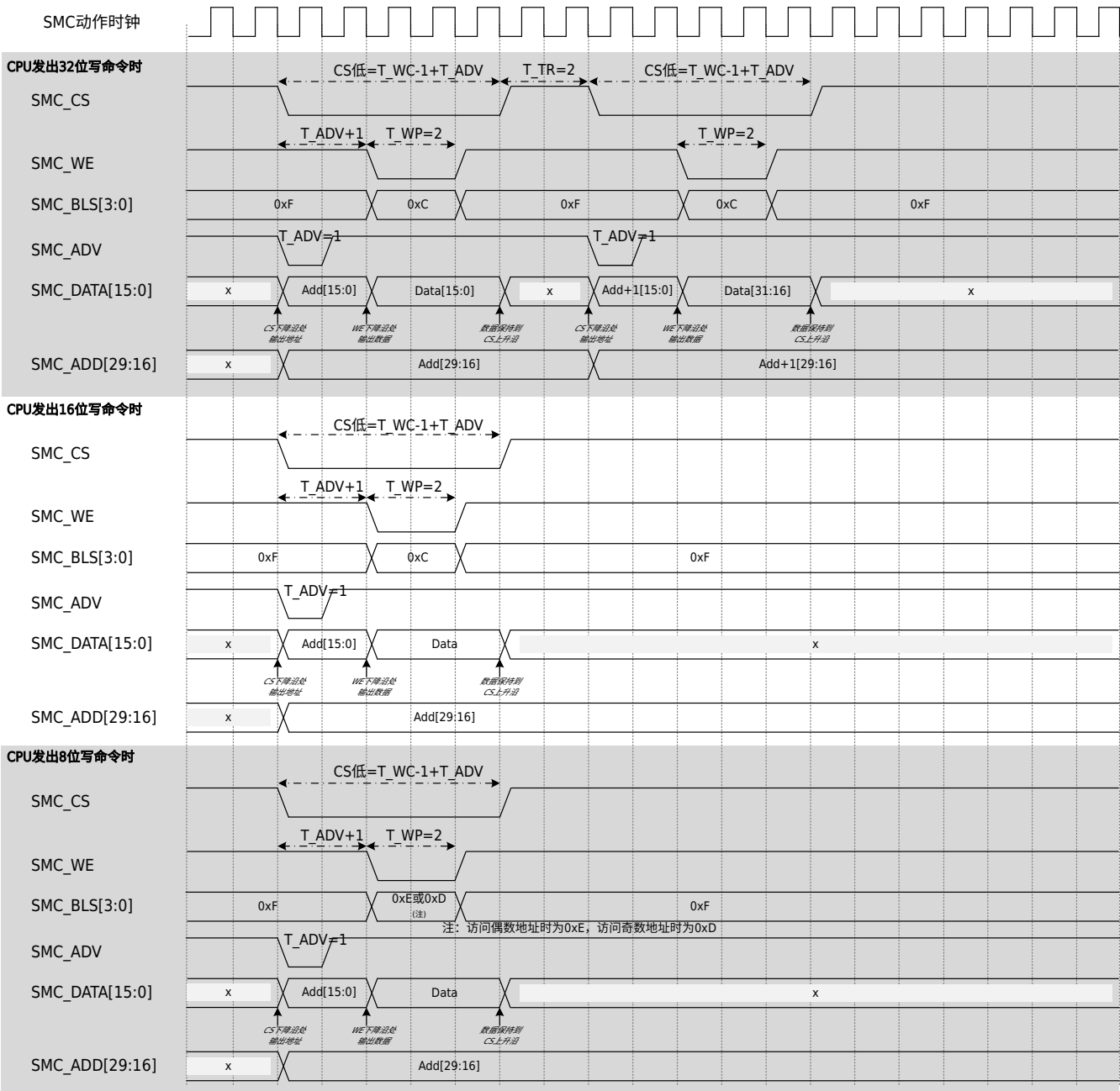


图 40-23 地址数据线复用单次写动作基本时序（异步方式（WSYN=0b0）&16 位位宽（MW=0b01））

地址数据线复用同步方式的单次写动作

地址数据线复用的同步方式的单次写动作的基本时序图和设定例如下：

表 40-11 地址数据线复用同步方式单次写动作基本设定例

基本设定	MW	RSYN	WSYN	-	ADVS	BLSS	-表示不关 注 <set>表示 用户设定值
	<set>	-	1	-	0b1	0b1	
时序设定	T_RC	T_WC	T_CEOE	T_WP	T_TR	T_ADV	
	-	0b0101	-	0b011	0b010	0b001	

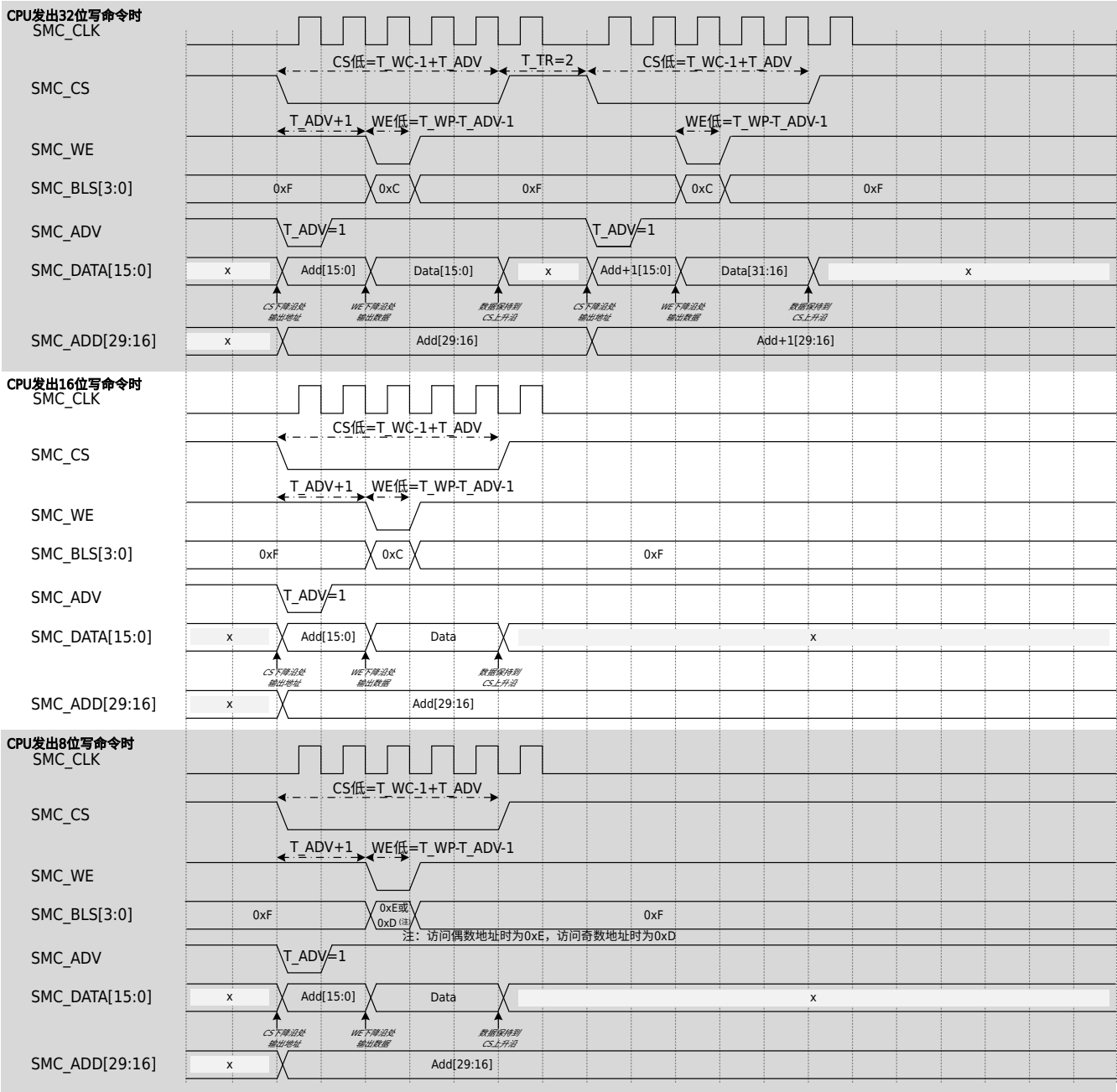


图 40-24 地址数据线复用单次写动作基本时序（同步方式（WSYN=0b1）&16 位位宽（MW=0b01））

40.3.5.4.3 FIFO 管理

SMC 中含有 3 类 FIFO。分别是命令 FIFO、写数据 FIFO、读数据 FIFO。

- 命令 FIFO：2 个命令 FIFO，用于缓存 2 次 SMC 命令及对应的命令地址、CS 等信息。

- 写数据 FIFO：4 个写数据 FIFO，用于缓存 AHB 的写数据。
- 读数据 FIFO：4 个读数据 FIFO，用于缓存从 SRAM/PSRAM/NOR Flash 读入的数据。

40.3.5.5 SMC 低功耗管理

SMC 处于 Low Power 状态时，如果将状态控制寄存器 (SMC_STCR1) 的 LPWOR 位设定为 1，则 SMC 从 Low Power 状态转为 Ready 状态，SMC 开始正常工作；如果对 SMC 长时间无操作，可将状态控制寄存器 (SMC_STCR0) 的 LPWIR 位设定为 1，使 SMC 进入 Low Power 状态，进入 Low Power 状态后，SMC 的内部动作停止，功耗会降低。

40.3.6 DMC-SDRAM 控制器

40.3.6.1 SDRAM 介绍

同步动态随机存储器 (SDRAM) 是基于外部同步时钟刷新的动态随机存储器 (DRAM)，它的同步时钟由 DMC 的 DMC_CLK 引脚提供。

Bank 特性

SDRAM 内部可以分为多个 Bank (EXMC 支持最大 4 个 Bank 的存储器)，允许设备以交错的方式进行访问，以获取更大的并发性和数据传输量。每个 Bank 是一个存储器矩阵，矩阵由行和列构成，行列的选择定位一个存储单元，存储单元上对应存储器存储宽度的空间，因此 SDRAM 存储器的每个 Bank 的大小可以认为是存储器数据宽度 x 行数 x 列数。用户可以通过片选控制寄存器 (DMC_CSCR) 的相关设定 (行地址长度、列地址长度等) 使 DMC 和外部 SDRAM 的参数匹配，实现与不同 SDRAM 的通信。

行激活

行激活是将行地址所在的 Bank 使能，完整的行地址由 2 Bit 的 Bank 地址 (DMC_BA[1:0]) 和 16 Bit 的行地址 (DMC_ADD[15:0]) 组成。行激活会将所选行的所有比特信息读入读写放大器，一旦行激活完成，读写操作就可以顺利的进行。

行激活需要一定的时间，这个时间间隔叫做行列延时，它是行寻址到列寻址的最小时间间隔。DMC 的行列延时 (T_{RCD})，是包含 SDRAM 行列延时的最小时钟周期数，它代表了行使能到 SDRAM 读写间的最小等待时间，可通过时序配置寄存器 (DMC_TMCR_T_RCD) 来设定该时间。在这段时间中，由于 SDRAM 控制器对 Bank 的操作是独立进行的，用户可向其它的 Bank 地址发出控制命令。下图是行激活到读写操作的时序图。

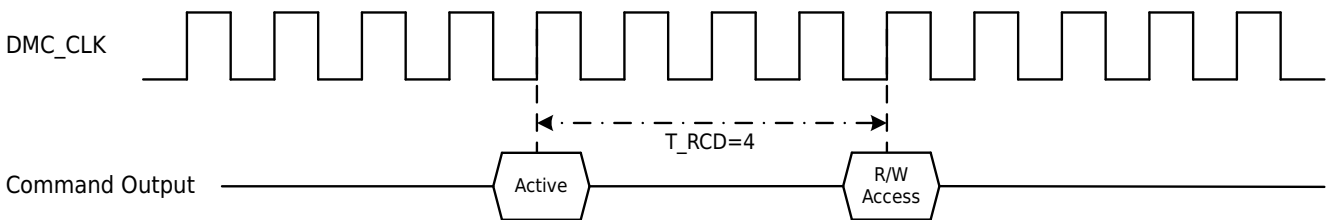


图 40-25 行激活到读写操作时序



说明

行激活在 EXMC 章节的时序图中都表示为“Active”。

预充电

如果 DMC 在访问外部数据时需要进行行切换，就需要将该 Bank 原来有效 (工作) 的行关闭，重新发送行列地址。Bank 关闭现有工作行，准备打开新行的操作就是预充电。预充电动作可以通过命令寄存器

(DMC_CMDR) 的预充电命令 (PrechargeAll) 触发，或者在读写换行时触发。

在突发读或突发写命令中，A10/A8 (由 DMC_CPCR.APBS 设定) 位变为“H”，在读写操作完成后自动附加一个预充电动作，实现自动预充电。

行预充电延时 (T_{RP}) 代表 SDRAM 行切换的最小时间，它是预充电完成到下一次行使能命令的最小时间间隔，可通过时序配置寄存器 (DMC_TMCR_T_RP) 来设定该时间。以下是两种不同方式的预充电的时序图。

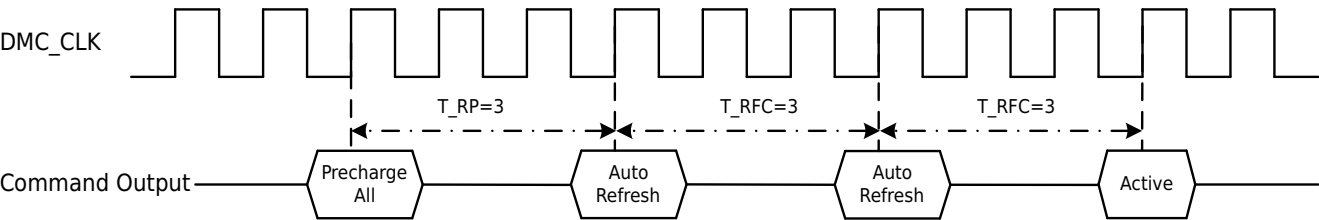


图 40-26 命令方式预充电

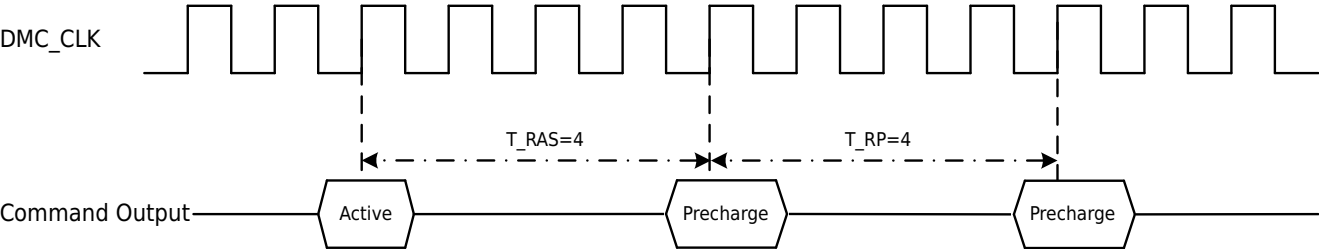


图 40-27 读写后自动预充电

刷新动作

由于动态存储器的存储单元存在漏电现象，为了保持每个存储单元数据的正确性，必须保证对所有的存储单元能定期实现刷新。EXMC 支持两种刷新方式，自动刷新方式（Auto Refresh，简称 AR）和自刷新方式（Self Refresh，简称 SR）。

自动刷新（AR）是 EXMC 周期性的提供刷新命令，刷新间隔由刷新时间寄存器（DMC_RFTR）决定。刷新对象为所有 Bank（所有 Bank 的对应行同时刷新），且只有在所有 Bank 都空闲并且未处于低功耗状态时才启动自动刷新操作。自动刷新执行期间只能输入空操作，其它操作指令只能等待而无法执行。自动刷新执行完毕后所有 Bank 都进入空闲状态。

自刷新（SR）主要用于休眠模式低功耗状态下的数据保存。在 AR 命令时 DMC_CKE 处于无效状态就进入了自刷新方式。SR 时 SDRAM 不再依靠 DMC_CLK 时钟工作，而是根据 SDRAM 内部的时钟进行刷新操作，在 SR 期间除了 DMC_CKE 之外的所有外部信号都是无效的（无需外部提供刷新指令）。DMC 在从 Pause 状态变为 Low Power 状态（通过发出 Sleep 命令）时，会自动发出自刷新指令。下图是自刷新动作的进入和退出时序图。

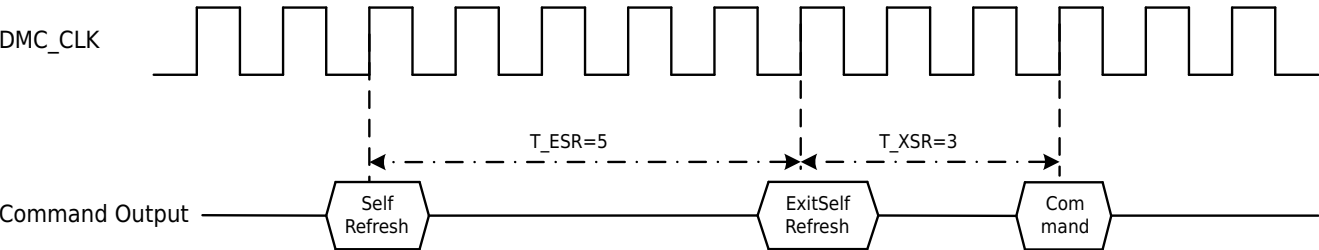


图 40-28 自刷新动作的进入和退出时序

命令真值表

对 SDRAM 进行控制和访问的各种命令都是由 DMC 的 SDRAM 协议接口发出，其具体的命令列表如下表所示。

表 40-12 DMC 的命令真值表

DMC_CS	DMC_RAS	DMC_CAS	DMC_WE	DMC_BA	DMC_ADD		命令
					ADD[xx]	ADD[10]/[8]	
0	0	0	0	0	Config		MdRegConfig 命令
0	0	0	1	-	-	-	SelfRefresh 命令（DMC_CKE=0）
							AutoRefresh 命令（DMC_CKE=1）

DMC_CS	DMC_RAS	DMC_CAS	DMC_WE	DMC_BA	DMC_ADD		命令
					ADD[xx]	ADD[10]/[8]	
0	0	1	0	-	-	1	PrechargeAll 命令
				Bank	-	0	指定 Bank 的 Prechagre 命令
0	0	1	1	Bank	Row	Row	行激活命令
0	1	0	0	Bank	Col	1	写动作时 Precharge 命令
				Bank	Col	0	写动作命令
0	1	0	1	Bank	Col	1	读动作时 Precharge 命令
				Bank	Col	0	读动作命令
0	1	1	0	-	-	-	中止突发传送
0	1	1	1	-	-	-	无操作
1	-	-	-	-	-	-	命令禁止

40.3.6.2 DMC 基本功能

DMC 基本功能特性如下：

- 支持 16 位、32 位外部存储器数据带宽
- 支持多达 16 位行地址、12 位列地址、2 位内部 Bank 地址
- AHB 字、半字、字节访问
- 为每个存储器 Chip 提供独立的片选控制
- 每个存储器 Chip 大小可独立配置
- 字节选择信号输出
- 自动进行行和 Bank 边界管理
- 可编程的协议时序参数
- 可编程速率的自动刷新动作
- 具有 2 个 41 位的命令 FIFO
- 具有 10 个 36 位的写数据 FIFO
- 具有 10 个 32 位的读数据 FIFO
- 低功耗管理

40.3.6.3 DMC 初始设定

DMC 在上电复位之后的状态图及状态之间切换图如下图所示。

初始复位时，DMC 进入 Config 状态，此时若通过状态控制寄存器（DMC_STCR）发出 Go 命令，之后无操作时，DMC 自动进入 Low Power 状态；正常动作期间，DMC 的状态切换可通过状态控制寄存器（DMC_STCR）的设定来控制，具体操作请参见“[DMC 低功耗管理](#)”。

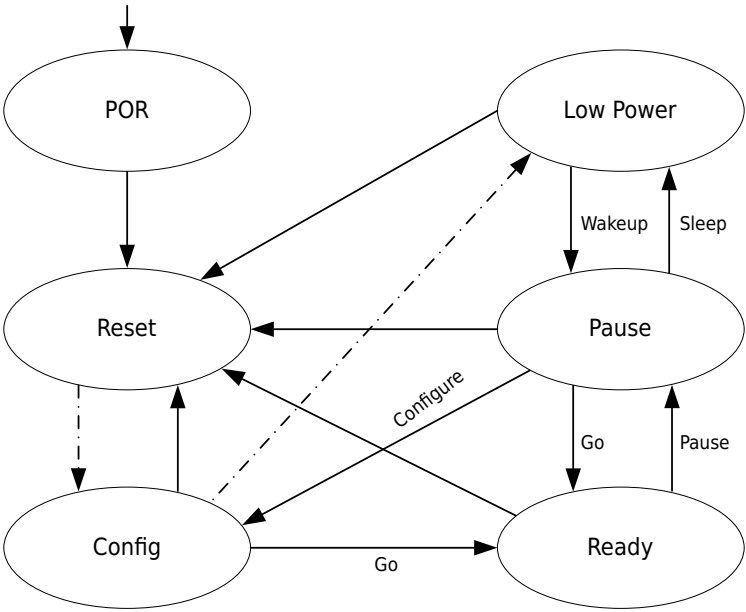


图 40-29 DMC 状态图

DMC 在和外部 SDRAM 进行通信之前，必须进行初始设定，配置好相关的参数，以保证能进行正确的数据交换。具体的设定顺序参考下图。

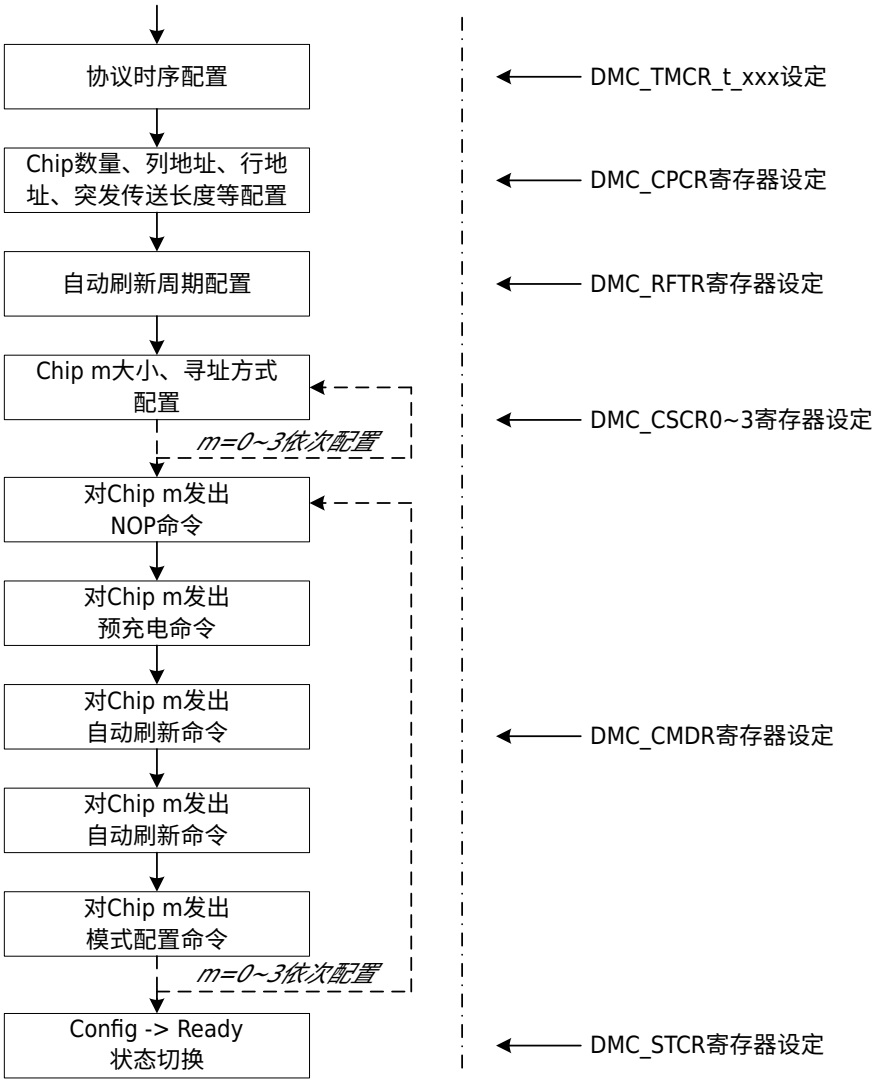


图 40-30 DMC 初始设定流程

40.3.6.4 DMC 访问动作

40.3.6.4.1 访问时序

DMC 可以把 AHB 的单次或突发读写操作转换成存储器的读写操作。与外部 SDRAM 存储器进行数据读写的方式有下面几种：

- 1. 单次读动作
- 2. 单次写动作
- 3. 突发读动作
- 4. 突发写动作

单次读动作

下图是单次读动作的基本时序图。

图中 cas_latency 为读命令到数据读出的延迟（DMC 模块在 DMC_CLK 的上升沿读入数据），可通过时序配置寄存器（DMC_TMCR_T_CASL）设置。

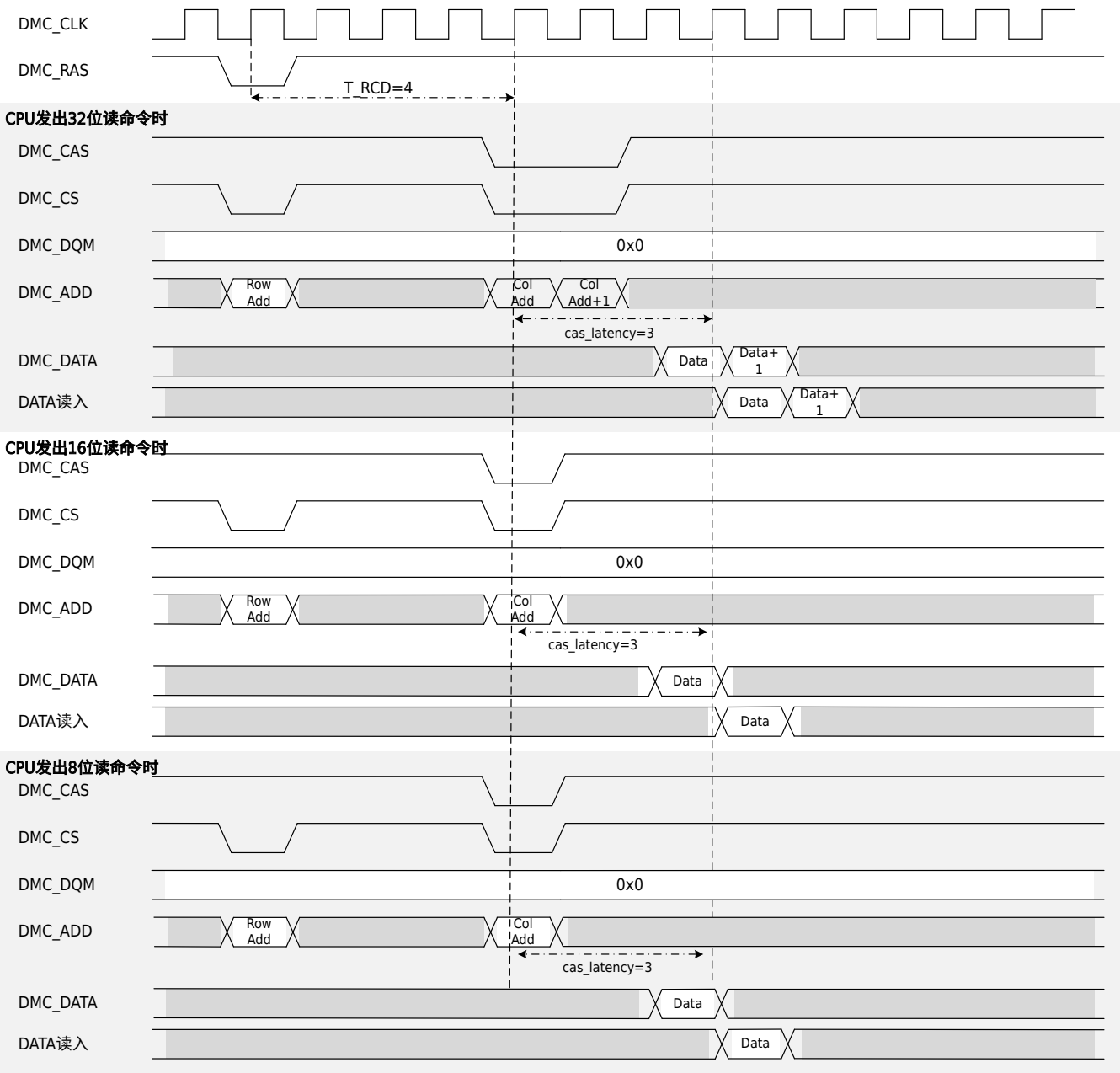


图 40-31 单次读动作基本时序（16 位位宽（DMCMW=0b00））

单次写动作

下图是单次写动作的基本时序图。
图中 t_{dqss} 为写使能和写数据之间的延迟，可通过时序配置寄存器 (DMC_TMCR_T_DQSS) 配置。

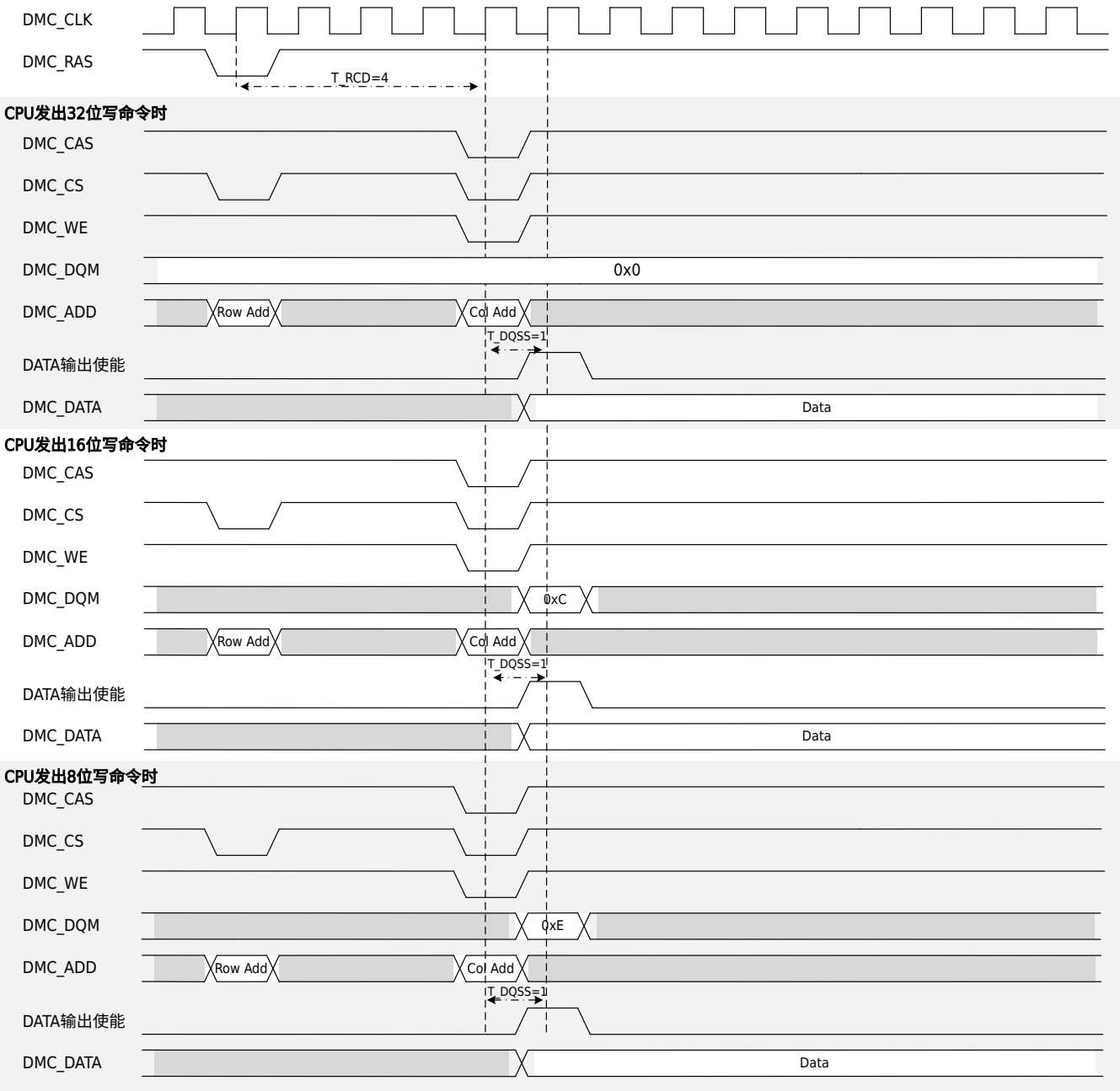


图 40-32 单次写动作基本时序 (32 位位宽 (DMCMW=0b01))

突发读动作

下图是突发读动作的基本时序图。

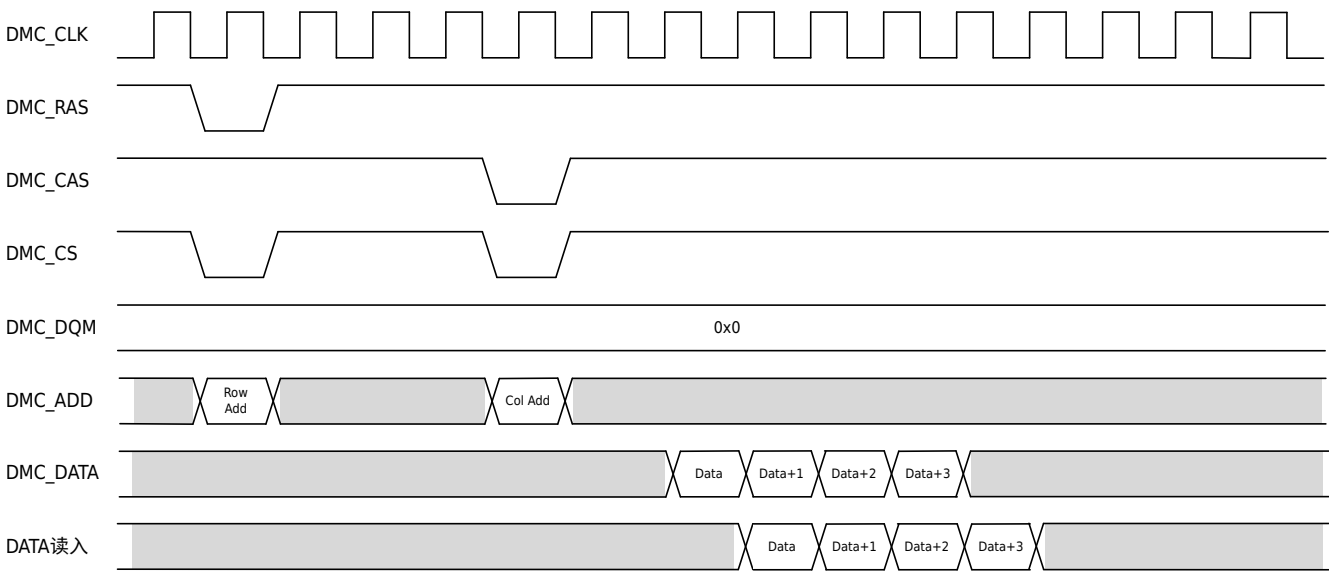


图 40-33 突发读动作基本时序 (32 位位宽 (DMCMW=0b10) &BURST=0b010)

突发写动作

下图是突发写动作的基本时序图。

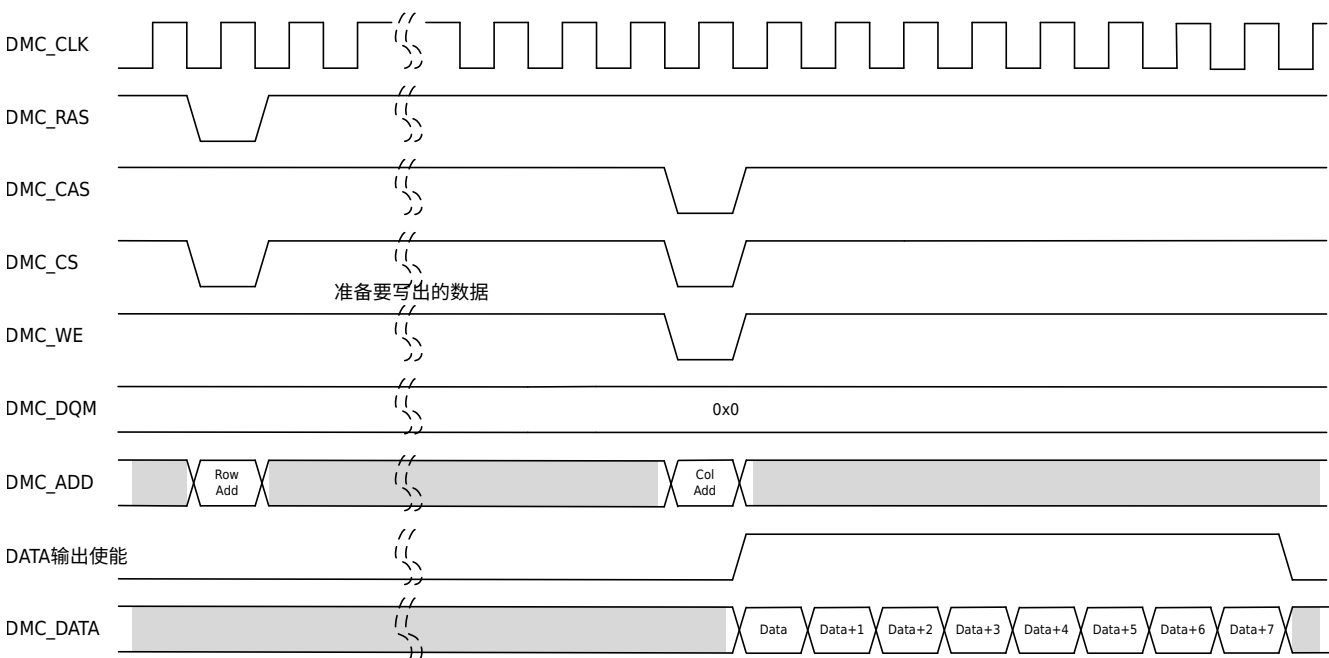


图 40-34 突发写动作基本时序 (32 位位宽 (DMCMW=0b10) &BURST=0b011)

40.3.6.4.2 FIFO 管理

DMC 中含有 3 类 FIFO。分别是命令 FIFO、写数据 FIFO、读数据 FIFO。

命令 FIFO：2 个命令 FIFO，用于缓存 2 次 DMC 命令及对应的命令地址、CS 等信息。

写数据 FIFO：10 个写数据 FIFO，用于缓存 AHB 的写数据。

读数据 FIFO：10 个读数据 FIFO，用于缓存从 SDRAM 读入的数据。

40.3.6.5 DMC 低功耗管理

DMC 内部可以在 Ready 状态和 Low Power 状态之间切换，实现 DMC 内部的功耗管理；同时可以控制空闲状态时 DMC 的输出时钟或时钟使能，以实现对外部 SDRAM 存储器的功耗管理。

Low Power 状态控制

DMC 处于 Config 状态时，通过状态控制寄存器 (DMC_STCR) 依次发出 Go 命令、Pause 命令和 Sleep 命令后，DMC 就进入 Low Power 状态。进入 Low Power 状态后，DMC 的内部动作停止，功耗会降低。在 Low Power 状态时，通过状态控制寄存器 (DMC_STCR) 依次发出 Wakeup 命令和 Go 命令后，DMC 就从 Low Power 状态退出，进入 Ready 状态开始正常动作。

DMC_CLK 输出控制

CHIP 配置寄存器 (DMC_CPCR) 的 CKSTOP 位设定为 1 时，DMC 在对外部 SDRAM 无操作时，DMC_CLK 端口的时钟输出会自动停止；当有新的操作时，DMC_CLK 的时钟会重新开启，动作完成之后，DMC_CLK 又自动停止，从而实现了在空闲状态时对时钟 DMC_CLK 的输出控制，进而达到控制 DMC 及 SDRAM 功耗的目的。

DMC_CKE 输出控制

CHIP 配置寄存器 (DMC_CPCR) 的 CKEDIS 位设定为 1 时，当 DMC 对外部 SDRAM 执行一次操作后，经过 DMC_CPCR.CKEDISPRD[5:0] 的设定周期后，DMC 变为空闲状态，DMC_CKE 的端口输出就变为无效；当有新的操作时，DMC_CKE 会重新变为有效，动作完成之后，DMC_CKE 又变为无效，从而实现了在空闲状态时对时钟输出使能 DMC_CKE 的控制，进而达到控制 SDRAM 功耗的目的。下图是 DMC_CKE 端口输出控制的时序图。

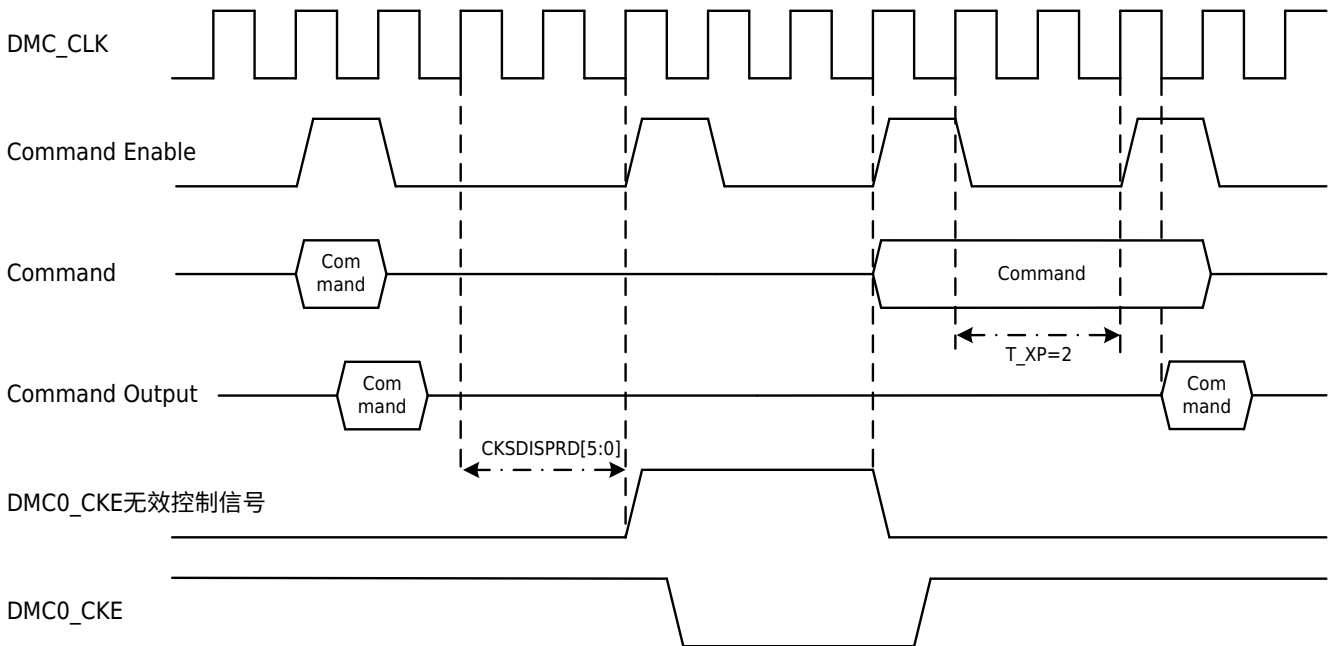


图 40-35 DMC_CKE 端口输出控制时序

40.3.7 NFC-NAND Flash 控制器

40.3.7.1 NAND Flash 介绍

NAND Flash 内存是 Flash 内存的一种，其内部采用非线性宏单元模式，为固态大容量内存的实现提供了廉价有效的解决方案。NAND Flash 存储器具有容量较大，改写速度快等优点，适用于大量数据的存储，因而在业界得到了越来越广泛的应用，如嵌入式产品中包括数码相机、MP3 随身听记忆卡、体积小巧的 U 盘等。

40.3.7.2 NFC 基本功能

NFC 基本功能特性如下：

- 支持 ONFI 协议，能够简单的实现对 SLC 和 MLC NAND Flash 设备的访问
- 支持页大小 2KB/4KB/8KB 的设备
- 支持 8-bit 和 16-bit 位宽的设备
- 支持每 512Byte 为单位的 1-bit 汉明码 ECC 纠错
- 能够计算每 512Byte 为单位的 4-bit BCH 码，并在 ECC 错误时给出伴随式
- 访问时序可通过寄存器配置

40.3.7.3 NFC 访问命令

下表列出了 NFC 支持的 ONFI 访问命令。其中“必须”表示所有 ONFI 设备必须支持的命令。“可选”表示某些 ONFI 设备能够支持的命令。

表 40-13 ONFI 访问命令

命令	必须(M)/可选(O)	第一周期 (command)	第二周期(confirm)
Read	M	0x00	0x30
Copyback read	O	0x00	0x35
Change read column	M	0x05	0xE0
Change read column enhanced	O	0x06	0xE0
Read cache random	O	0x00	0x31
Read cache sequential	O	0x31	-
Read cache end	O	0x3F	-
Block erase	M	0x60	0xD0
Block erase interleaved	O	0x60	0xD1
Read status	M	0x70	-
Read status enhanced	O	0x78	-
Page program	M	0x80	0x10
Page program interleaved	O	0x80	0x11
Page cache program	O	0x80	0x15
Copyback program	O	0x85	0x10
Copyback program interleaved	O	0x85	0x11
Change write column	M	0x85	-
Change row address	O	0x85	-
Read ID	M	0x90	-
Read parameter page	M	0xEC	-
Read Unique ID	O	0xED	-

命令	必须(M)/可选(O)	第一周期 (command)	第二周期(confirm)
Get Features	O	0xEE	-
Set Features	O	0xEF	-
Reset LUN	O	0xFA	-

40.3.7.4 NFC 访问动作

Page Read

- 步骤 1. 向命令寄存器写入 0x00000000;
- 步骤 2. 向索引寄存器写入 NAND Flash 地址;
- 步骤 3. 向命令寄存器写入 0x000000E0, 或者对数据寄存器执行一次读操作, 此时 NAND Flash 器件将拉低 RB 信号, 表示此时器件正在准备数据。当 RB 信号恢复为高时, 表示数据准备完毕。因此, 如果使用命令寄存器写 0x000000E0 的方式, 需等待至 RB 信号为高后读取数据寄存器, 使用对数据寄存器执行一次读操作时, 控制器将在总线上插入等待直到 RB 为高, 并返回第一个数据;
- 步骤 4. 通过读取数据寄存器得到整个 page 的数据;
- 步骤 5. 用户不需要读取该 page 全部数据。当读完所需数据后, 向命令寄存器写入 0x00000023, 控制器将自动读取剩余数据并开始计算 ECC。当 ECC 计算完成时将产生中断并将标志位置位, 并将 ECC 结果写入 ECC 结果寄存器中。如果用户选择 ECC 功能无效, 将省略此步骤;
- 步骤 6. 向命令寄存器写入 0x000000FE 结束当次读操作, 并将 CE 置为无效。

Page Write

- 步骤 1. 向命令寄存器写入 0x00000080;
- 步骤 2. 向索引寄存器写入 NAND Flash 地址;
- 步骤 3. 依次向数据寄存器写入数据, 当 ECC 有效时, 写入数据的同时控制器将计算 ECC;
- 步骤 4. 写完全部数据后, 向命令寄存器写入 0x00000010, NAND Flash 器件将开始 programming, 同时拉低 RB。当 ECC 有效时, 如果用户写入的数据不足 1page, 控制器将自动用 FF 补足剩余的地址并计算 ECC 值。(当 ECCM 为 1bitECC 模式时: 写完数据后将 ECC 值自动写入 NAND Flash 器件的 spare area 中; 当 ECCM 为 4bitECC 模式时: 每 512 字节后写入 8 字节 ECC 值);
- 步骤 5. programming 结束后, NAND Flash 器件将 RB 恢复为高。向命令寄存器写入 0x000000FE 结束此次写操作并将 CE 置为无效。

Block Erase

- 步骤 1. 向命令寄存器写入 0x81000M60, 其中 M 为 BANK 号;
- 步骤 2. 向命令寄存器写入 0x40000MAB, 其中 M 为 BANK 号, AB 为 NAND Flash row address 的最低位的字节;
- 步骤 3. 向命令寄存器写入 0x40000MWX, 其中 M 为 BANK 号, WX 为 NAND Flash row address 的次低位的字节;
- 步骤 4. 向命令寄存器写入 0x40000MYZ, 其中 M 为 BANK 号, YZ 为 NAND Flash row address 的最高位的字节;
- 步骤 5. 向命令寄存器写入 0x00000MD0, 其中 M 为 BANK 号;
- 步骤 6. 等待 NAND Flash 器件将 RB 恢复为高。

Reset

- 步骤 1. 向命令寄存器写入 0x00000MFF, 其中 M 为 BANK 号;
- 步骤 2. 等待 NAND Flash 器件将 RB 恢复为高。

Read ID

- 步骤 1. 读取一次数据寄存器；
- 步骤 2. 向命令寄存器写入 0x81000M90，其中 M 为 BANK 号；
- 步骤 3. 向命令寄存器写入 0x40000MAB，其中 M 为 BANK 号，AB 为 ID 的地址；
- 步骤 4. 读取数据寄存器得到 ID；
- 步骤 5. 向命令寄存器写入 0x000000FE 将 CE 置为无效。

Read Status

- 步骤 1. 向命令寄存器写入 0x81000M70，其中 M 为 BANK 号；
- 步骤 2. 读取数据寄存器，寄存器值的最低位字节为状态值；
- 步骤 3. 向命令寄存器写入 0x000000FE 将 CE 置为无效。

40.3.8 中断说明

NFC 有 3 类中断事件，由中断使能寄存器 (NFC_IENR) 控制是否发生相应中断，用户能够通过查询中断状态寄存器 (NFC_ISTR) 判断是否发生中断事件，并通过写寄存器清除中断事件。NFC 的 3 类中断事件如下：

1. ECC 计算结束时发生中断
2. ECC 校验错误时发生中断
3. 各个 RB 信号产生上升沿，即对应设备就绪时发生中断

40.4 寄存器

40.4.1 寄存器总表

EXMC 公共寄存器基地址：0x40055400

表 40-14 EXMC 公共寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x08	NFC_STCR	NFC 控制寄存器	32
0x28	NFC_STSR	NFC 状态寄存器	32
0x0C	EXMC_ENAR	EXMC 使能寄存器	32

SMC 基地址：0x88000000

表 40-15 外部存储器控制器 (EXMC) SMC 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	SMC_STSR	状态寄存器	32
0x0010	SMC_CMDR	命令寄存器	32
0x0008	SMC_STCR0	状态控制寄存器 0	32
0x000C	SMC_STCR1	状态控制寄存器 1	32
0x0200	SMC_BACR	基本控制寄存器	32
0x0208	SMC_CSCR0	片选控制寄存器	32
0x020C	SMC_CSCR1	片选控制寄存器	32
0x0018	SMC_CPCR	CHIP 配置寄存器	32
0x0104+0x20*x	SMC_CPSRx	CHIP 状态寄存器 x (x=0~3)	32
0x0014	SMC_TMCR	时序配置寄存器	32
0x0100+0x20*x	SMC_TMSRx	时序状态寄存器 x (x=0~3)	32

DMC 基地址：0x88000400

表 40-16 外部存储器控制器 (EXMC) DMC 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	DMC_STSR	状态寄存器	32
0x0008	DMC_CMDR	命令寄存器	32
0x0004	DMC_STCR	状态控制寄存器	32
0x0010	DMC_RFTR	刷新时间寄存器	32
0x0300	DMC_BACR	基本控制寄存器	32
0x0200+0x04*x	DMC_CSCRx	片选控制寄存器 x (x=0~3)	32

偏移地址	寄存器	描述	位宽
0x000C	DMC_CPCR	CHIP 配置寄存器	32
0x0014	DMC_TMCR_T_CASL	时序配置寄存器	32
0x0018	DMC_TMCR_T_DQSS	时序配置寄存器	32
0x001C	DMC_TMCR_T_MRD	时序配置寄存器	32
0x0020	DMC_TMCR_T_RAS	时序配置寄存器	32
0x0024	DMC_TMCR_T_RC	时序配置寄存器	32
0x0028	DMC_TMCR_T_RCD	时序配置寄存器	32
0x002C	DMC_TMCR_T_RFC	时序配置寄存器	32
0x0030	DMC_TMCR_T_RP	时序配置寄存器	32
0x0034	DMC_TMCR_T_RRD	时序配置寄存器	32
0x0038	DMC_TMCR_T_WR	时序配置寄存器	32
0x003C	DMC_TMCR_T_WTR	时序配置寄存器	32
0x0040	DMC_TMCR_T_XP	时序配置寄存器	32
0x0044	DMC_TMCR_T_XSR	时序配置寄存器	32
0x0048	DMC_TMCR_T_ESR	时序配置寄存器	32

NFC 基地址：0x88100000

表 40-17 外部存储器控制器 (EXMC) NFC 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000~0x7FFC	NFC_DATR	数据寄存器	32
0x8000	NFC_CMDR	命令寄存器	32
0x8004	NFC_IDXR0	索引寄存器 0	32
0x8048	NFC_IDXR1	索引寄存器 1	32
0x8054	NFC_BACR	基本配置寄存器	32
0x8030	NFC_IENR	中断使能寄存器	32
0x8034	NFC_ISTR	中断状态寄存器	32
0x8038	NFC_IRSR	中断结果寄存器	32
0x804C	NFC_TMCR0	时序配置寄存器 0	32
0x8050	NFC_TMCR1	时序配置寄存器 1	32
0x805C	NFC_TMCR2	时序配置寄存器 2	32
0x8060~0x817C	NFC_ECCR	ECC 校验寄存器	32

40.4.2 EXMC 公共寄存器

40.4.2.1 EXMC 使能寄存器 (EXMC_ENAR)

Offset	Bit Position																																		
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	Reserved																																NFCEN	SMCEN	DMCEN
Access																																	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	NFCEN	NFC 使能	● 0b0: NFC 模块的功能端口无效 ● 0b1: NFC 模块的功能端口有效	RW
1	SMCEN	SMC 使能	● 0b0: SMC 模块的功能端口无效 ● 0b1: SMC 模块的功能端口有效	RW
0	DMCEN	DMC 使能	● 0b0: DMC 模块的功能端口无效 ● 0b1: DMC 模块的功能端口有效	RW

40.4.3 SMC-SRAM/PSRAM/NOR Flash 寄存器

40.4.3.1 状态寄存器 (SMC_STSR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved																															STATUS
Access																																RO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	STATUS	SMC 当前状态	0b0: Ready 0b1: Low Power	RO

40.4.3.2 命令寄存器 (SMC_CMDR)

Offset	Bit Position																																
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	Reserved						CMDCHIP				CMD				CRES	CMDADD																	
Access							WO				WO				WO	WO																	

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:23	CMDCHIP	CS 对象设定	0b000: Chip 0 0b001: Chip 1 0b010: Chip 2 0b011: Chip 3 - 其他: 禁止设置	WO
22:21	CMD	命令输入	0b00: 禁止设置 0b01: MdRegConfig 0b10: UpdateRegs 0b11: MdRegConfig & UpdateRegs	WO
20	CRES	CRE 端口设定	0b0: SMC_CRE 输出低电平 0b1: 在发出 MdRegConfig 命令时, SMC_CRE 端口输出高电平	WO
19:0	CMDADD	地址对象设定	当 CMD=0b01 或 0b11 时, CMDADD[19:0]表示要访问的外部 MEM 地址, SMC_ADD[19:0]引脚输出 CMDADD[19:0]	WO

40.4.3.3 状态控制寄存器 0 (SMC_STCR0)

Offset	Bit Position																																
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	Reserved																														LPWIR		Reserved
Access																															WO		

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	LPWIR	Low Power 状态 进入请求	写 1 进入 Low Power 状态；写 0 无效	WO
1:0	Reserved	保留	-	-

40.4.3.4 状态控制寄存器 1 (SMC_STCR1)

Offset	Bit Position																																
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0xFFFFFFFF																																
Name	Reserved																														LPWOR		Reserved
Access																															WO		

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	LPWOR	Low Power 状态 退出请求	写 1 退出 Low Power 状态；写 0 无效	WO
1:0	Reserved	保留	-	-

40.4.3.5 基本控制寄存器 (SMC_BACR)

Offset	Bit Position																																			
0x0200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000300																																			
Name	Reserved																CKSEL	Reserved								MUXMD	Reserved									
Access																	RW									RW										

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:14	CKSEL	采样时钟选择	0b00: 读取锁存 SMC_DATA、SMC_WAIT 端口数据时, 使用内部 EXCLK (仅用于异步模式) 0b01: 读取锁存 SMC_DATA、SMC_WAIT 端口数据时, 使用内部 EXCLK 取反做时钟 (仅用于异步模式) 0b10: 读取锁存 SMC_DATA、SMC_WAIT 端口数据时, 使用 EXMC_CLK 端口状态反馈做时钟 (仅用于同步模式) 0b11: 禁止设置	RW
13:10	Reserved	保留	-	-
9:8	Reserved	保留	所有位读出为 1, 写入时必须写 1	-
7:5	Reserved	保留	-	-
4	MUXMD	SMC 地址数据复用	0b0: SMC 地址和数据不复用 0b1: SMC 地址和数据复用	RW
3:0	Reserved	保留	-	-

40.4.3.6 片选控制寄存器 (SMC_CSCR0)

Offset	Bit Position																															
0x0208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	ADDMSK3								ADDMSK2								ADDMSK1								ADDMSK0							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	ADDMSK3	CS3 空间屏蔽地址	CS3 空间屏蔽地址设定	RW
23:16	ADDMSK2	CS2 空间屏蔽地址	CS2 空间屏蔽地址设定	RW
15:8	ADDMSK1	CS1 空间屏蔽地址	CS1 空间屏蔽地址设定	RW

位/位域	标记	位名	功能描述	读写
7:0	ADDMSK0	CS0 空间屏蔽地址	CS0 空间屏蔽地址设定	RW

40.4.3.7 片选控制寄存器 (SMC_CSCR1)

Offset	Bit Position																															
0x020C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ADDMAT3								ADDMAT2								ADDMAT1								ADDMAT0							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	ADDMAT3	CS3 空间匹配地址	CS3 空间匹配地址设定	RW
23:16	ADDMAT2	CS2 空间匹配地址	CS2 空间匹配地址设定	RW
15:8	ADDMAT1	CS1 空间匹配地址	CS1 空间匹配地址设定	RW
7:0	ADDMAT0	CS0 空间匹配地址	CS0 空间匹配地址设定	RW

举例说明，不同 ADDMAT0 和 ADDMSK0 设定值时，SMC 访问空间的对应方式如下表所示。

表 40-34 CS0 设定和访问地址对应关系表

ADDMAT0[7:0]	ADDMSK0[7:0]	对应外部地址	CS0 空间大小
0x70	0xFF	0x70000000~0x70FFFFFF	16MB
	0xFE	0x70000000~0x71FFFFFF	32MB
	0xFC	0x70000000~0x73FFFFFF	64MB
	0xF8	0x70000000~0x77FFFFFF	128MB
	0xF0	0x70000000~0x7FFFFFFF	256MB
	其它设定禁止	-	-
0x68	0xFF	0x68000000~0x68FFFFFF	16MB
	0xFE	0x68000000~0x69FFFFFF	32MB
	0xFC	0x68000000~0x6BFFFFFF	64MB
	0xF8	0x68000000~0x6FFFFFFF	128MB
	其它设定禁止	-	-
0x66	0xFF	0x66000000~0x66FFFFFF	16MB
	0xFE	0x66000000~0x67FFFFFF	32MB
	其它设定禁止	-	-
0x64	0xFF	0x64000000~0x64FFFFFF	16MB
	0xFE	0x64000000~0x65FFFFFF	32MB
	0xFC	0x64000000~0x67FFFFFF	64MB
	其它设定禁止	-	-

ADDMAT0[7:0]	ADDMSK0[7:0]	对应外部地址	CS0 空间大小
0x62	0xFF	0x62000000~0x62FFFFFF	16MB
	0xFE	0x62000000~0x63FFFFFF	32MB
	其它设定禁止	-	-
0x60	0xFF	0x60000000~0x60FFFFFF	16MB
	0xFE	0x60000000~0x61FFFFFF	32MB
	0xFC	0x60000000~0x63FFFFFF	64MB
	0xF8	0x60000000~0x67FFFFFF	128MB
	0xF0	0x60000000~0x6FFFFFFF	256MB
	0xE0	0x60000000~0x7FFFFFFF	512MB
	其它设定禁止	-	-

40.4.3.8 CHIP 配置寄存器 (SMC_CPCR)

Offset	Bit Position																																			
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0xFFFFFFFF																																			
Name	Reserved																				BLSS		ADVS		Reserved		MW		Reserved		WSYN		Reserved		RSYN	
Access																					WO		WO				WO				WO				WO	

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12	BLSS	BLS 端口设定	0b0: BLS 的端口状态和 CS 端口的状态变化同步 0b1: BLS 的端口状态和 WE 端口的状态变化同步	WO
11	ADVS	ADV 端口设定	0b0: ADV 端口功能无效 0b1: 地址数据线复用功能有效时, ADV 端口功能有效	WO
10	Reserved	保留	-	-
9:8	MW	存储器位宽选择	0b00: 8 位位宽 0b01: 16 位位宽 0b10: 32 位位宽 - 其他: 禁止设置	WO
7:5	Reserved	保留	-	-
4	WSYN	写同步	0b0: 异步写使能 0b1: 同步写使能	WO
3:1	Reserved	保留	-	-
0	RSYN	读同步	0b0: 异步读使能 0b1: 同步读使能	WO

40.4.3.9 CHIP 状态寄存器 x (SMC_CPSRx) (x=0~3)

偏移地址 (Offset) : 0x0104+0x20*x

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00FF0A00																																	
Name	ADDMAT								ADDMSK								Reserved		BLSS		ADVS		Reserved		MW		Reserved		WSYN		Reserved			
Access	RO								RO										RO		RO				RO				RO				RO	

位/位域	标记	位名	功能描述	读写
31:24	ADDMAT	CS 空间匹配地址	CS 空间匹配地址信息	RO
23:16	ADDMSK	CS 空间屏蔽地址	CS 空间屏蔽地址信息	RO
15:13	Reserved	保留	-	-
12	BLSS	SMC_BLS 状态	● 0b0: BLS 的端口状态和 CS 端口的状态变化同步 ● 0b1: BLS 的端口状态和 WE 端口的状态变化同步	RO
11	ADVS	SMC_ADV 状态	● 0b0: ADV 端口功能无效 ● 0b1: 地址数据线复用功能有效时, ADV 端口功能有效	RO
10	Reserved	保留	-	-
9:8	MW	存储器位宽状态	● 0b00: 8 位位宽 ● 0b01: 16 位位宽 ● 0b10: 32 位位宽 ● 其他: Reserved	RO
7:5	Reserved	保留	-	-
4	WSYN	写动作方式状态	● 0b0: 异步写使能 ● 0b1: 同步写使能	RO
3:1	Reserved	保留	-	-
0	RSYN	读动作方式状态	● 0b0: 异步读使能 ● 0b1: 同步读使能	RO

40.4.3.10 时序配置寄存器 (SMC_TMCR)

Offset	Bit Position																																	
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0xFFFFFFFF																																	
Name	Reserved				T_ADV				Reserved	T_TR				Reserved				T_WP				Reserved	T_CEOE				T_WC				T_RC			
Access					WO					WO								WO					WO				WO				WO			

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26:24	T_ADV	T_ADV 周期设置	T_ADV 设定，设置值需要大于等于 1，每个 Chip 设置相同值  说明 仅在地址数据复用时有有效。 地址数据复用使用能时，配置 T_ADV 时注意： 1. 每个 chip SMC_TMCR 的 T_ADV 设置值需大于等于 1，且相同。 2. 配置 chip1~3 SMC_TMCR 的 T_ADV，必须对 chip 0 发出 UpdateRegs 命令。	WO
23	Reserved	保留	-	-
22:20	T_TR	T_TR 周期设定	T_TR 设定  说明 仅在地址数据复用时有有效，有效值为 1~7；地址数据不复用时写 1。	WO
19:15	Reserved	保留	-	-
14:12	T_WP	T_WP 周期设定	T_WP 设定	WO
11	Reserved	保留	-	-
10:8	T_CEOE	T_CEOE 周期设定	T_CEOE 设定，设定值需要大于 T_ADV	WO
7:4	T_WC	T_WC 周期设定	T_WC 设定	WO
3:0	T_RC	T_RC 周期设定	T_RC 设定	WO

40.4.3.11 时序状态寄存器 x (SMC_TMSRx) (x=0~3)

偏移地址 (Offset) : 0x0100+0x20*x

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x011263CC																																	
Name	Reserved				T_ADV				Reserved	T_TR				Reserved				T_WP				Reserved	T_CEOE				T_WC				T_RC			
Access					RO					RO								RO					RO				RO				RO			

位/位域	标记	位名	功能描述	读写
31:27	Reserved	保留	-	-
26:24	T_ADV	T_ADV 周期状态	T_ADV 周期	RO
23	Reserved	保留	-	-
22:20	T_TR	T_TR 周期状态	T_TR 周期	RO
19:15	Reserved	保留	-	-
14:12	T_WP	T_WP 周期状态	T_WP 周期	RO
11	Reserved	保留	-	-
10:8	T_CEOE	T_CEOE 周期状态	T_CEOE 周期	RO
7:4	T_WC	T_WC 周期状态	T_WC 周期	RO
3:0	T_RC	T_RC 周期状态	T_RC 周期	RO

40.4.4 DMC-SDRAM 寄存器

40.4.4.1 状态寄存器 (DMC_STSR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000700																															
Name	Reserved																												MEMW		STATUS	
Access																													RO		RO	

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:2	MEMW	DMC MEM 位宽	0b00: 16 位 0b01: 32 位 - 其他: Reserved  说明 该位读出值与 DMC_BACR.DMCMW[1:0]的设定值相同。	RO
1:0	STATUS	DMC 当前状态	0b00: Config 0b01: Ready 0b10: Paused 0b11: Low Power	RO

40.4.4.2 命令寄存器 (DMC_CMDR)

Offset	Bit Position																																			
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0xFFFFFFFF																																			
Name	Reserved										CMDCHIP		CMD		CMDBA		Reserved		CMDADD																	
Access											WO		WO		WO				WO																	

位/位域	标记	位名	功能描述	读写
31:22	Reserved	保留	-	-
21:20	CMDCHIP	CS 对象设定	0b00: Chip 0 0b01: Chip 1 0b10: Chip 2 0b11: Chip 3	WO
19:18	CMD	命令输入	0b00: PrechargeAll 0b01: AutoRefresh 0b10: MdRegConfig 0b11: NOP	WO
17:16	CMDBA	BANK 对象设定	CMD (MdRegConfig) 命令时指定的 Bank 对象 0b00: Bank 0 0b01: Bank 1 0b10: Bank 2 0b11: Bank 3	WO
15:14	Reserved	保留	-	-
13:0	CMDADD	地址对象设定	CMD (MdRegConfig) 命令时指定的 Address 对象, DMC_ADD[13:0]引脚输出 CMDADD[13:0]	WO

40.4.4.3 状态控制寄存器 (DMC_STCR)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																															STCTL
Access																																WO

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	STCTL	状态切换控制	0b000: Go 0b001: Sleep 0b010: Wakeup 0b011: Pause 0b100: Configure - 其他: 禁止设置	WO

40.4.4.4 刷新时间寄存器 (DMC_RFTR)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000A60																															
Name	Reserved																REFPRD															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14:0	REFPRD	刷新周期	刷新周期值设定	RW

40.4.4.5 基本控制寄存器 (DMC_BACR)

Offset	Bit Position																															
0x0300	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000300																															
Name	Reserved																CKSEL	Reserved														DMCMW
Access																	RW															RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:14	CKSEL	采样时钟选择	0b00: 禁止设置 0b01: 禁止设置 0b10: 读取锁存 DMC_DATA 端口数据时, 使用 EXMC_CLK 端口状态反馈做时钟 0b11: 禁止设置	RW
13:10	Reserved	保留	-	-
9:8	Reserved	保留	所有位读出的 1, 写入时必须写 1	-
7:2	Reserved	保留	-	-
1:0	DMCMW	存储器位宽选择	0b00: 16 位位宽 0b01: 32 位位宽 - 其他: 禁止设置	RW

40.4.4.6 片选控制寄存器 x (DMC_CSCRx) (x=0~3)

偏移地址 (Offset) : 0x0200+0x04*x

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x0000FF00																																
Name	Reserved																BRCx	ADDMATx								ADDMSKx							
Access																		RW								RW							

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	BRCx	地址译码方式	0b0: Row、Bank、Column 方式 0b1: Bank、Row、Column 方式	RW
15:8	ADDMATx	地址匹配	要匹配的地址	RW
7:0	ADDMSKx	地址屏蔽	要屏蔽的对应地址	RW

以 ADDMAT0 和 ADDMSK0 为例, 说明 DMC 的地址空间的对应方式。其它 CS 空间的设定和 CS0 设定类似。

表 40-55 CS0 设定和访问地址对应关系表

ADDMAT0[7:0]	ADDMSK0[7:0]	对应外部地址	CS0 空间大小
0x86	0xFF	0x86000000~0x86FFFFFF	16MB
	0xFE	0x86000000~0x87FFFFFF	32MB
	其它设定禁止	-	-
0x84	0xFF	0x84000000~0x84FFFFFF	16MB
	0xFE	0x84000000~0x85FFFFFF	32MB
	0xFC	0x84000000~0x87FFFFFF	64MB
	其它设定禁止	-	-
0x82	0xFF	0x82000000~0x82FFFFFF	16MB
	0xFE	0x82000000~0x83FFFFFF	32MB
	其它设定禁止	-	-
0x80	0xFF	0x80000000~0x80FFFFFF	16MB
	0xFE	0x80000000~0x81FFFFFF	32MB
	0xFC	0x80000000~0x83FFFFFF	64MB
	0xF8	0x80000000~0x87FFFFFF	128MB
	其它设定禁止	-	-

40.4.4.7 CHIP 配置寄存器 (DMC_CPCR)

Offset	Bit Position																																	
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00020040																																	
Name	Reserved				ACTCP				Reserved				BURST				CKEDISPRD				CKSTOP		CKEDIS		APBS		ROWBS				Reserved		COLBS	
Access					RW								RW				RW				RW		RW		RW		RW						RW	

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25:24	ACTCP	自动刷新 Chip 数目	0b00: Chip 0 自动刷新有效 0b01: Chip 0~1 自动刷新有效 0b10: Chip 0~2 自动刷新有效 0b11: Chip 0~3 自动刷新有效	RW
23:19	Reserved	保留	-	-
18:16	BURST	突发传输长度	0b000: 1 次传送 0b001: 2 次连续传送 0b010: 4 次连续传送 0b011: 8 次连续传送 0b100: 16 次连续传送 - 其他: 禁止设置  说明 - 该位的设定须和外部 Memory 的 Mode Register 的对应位设定一致。 - 同时使用 SMC 和 DMC 时, 如果 BURST 配置为非 0 值, DMC 的 AHB 总线读写数据位宽需要小于等于外部存储设备的位宽。	RW
15:10	CKEDISPRD	时钟输出关闭周期	时钟输出关闭的等待周期设定	RW
9	CKSTOP	时钟停止设定	0b0: MEM 时钟正常输出 0b1: 无操作时, MEM 时钟停止	RW
8	CKEDIS	CKE 输出设定	0b0: CKE 输出有效 (H) 0b1: 命令 FIFO 空时, 经过 CKEDISPRD 设定的周期后, CKE 输出变为无效 (L)	RW
7	APBS	自动预充电位	0b0: 地址线的第 10 位 0b1: 地址线的第 8 位	RW
6:4	ROWBS	行地址宽度	0b000: 行地址宽度为 11 位 0b001: 行地址宽度为 12 位 0b010: 行地址宽度为 13 位 - 其他: 禁止设置	RW
3	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
2:0	COLBS	列地址宽度	0b000: 列地址宽度为 8 位 0b001: 列地址宽度为 9 位 0b010: 列地址宽度为 10 位 0b011: 列地址宽度为 11 位 0b100: 列地址宽度为 12 位 - 其他: 禁止设置	RW

40.4.4.8 时序配置寄存器 (DMC_TMCR_T_CASL)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000003																															
Name	Reserved																														T_CASL	
Access																															RW	

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	T_CASL	T_CASL 周期设定	T_CASL 设定 (CAS latency time)	RW

40.4.4.9 时序配置寄存器 (DMC_TMCR_T_DQSS)

Offset	Bit Position																															
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved																															T_DQSS
Access																																RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1:0	T_DQSS	T_DQSS 周期设定	T_DQSS 设定	RW

40.4.4.10 时序配置寄存器 (DMC_TMCR_T_MRD)

Offset	Bit Position																															
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																										T_MRD					
Access																											RW					

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	T_MRD	T_MRD 周期设定	T_MRD 设定 (mode register command time)	RW

40.4.4.11 时序配置寄存器 (DMC_TMCR_T_RAS)

Offset	Bit Position																															
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000007																															
Name	Reserved																										T_RAS					
Access																											RW					

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:0	T_RAS	T_RAS 周期设定	T_RAS 设定 (RAS to precharge delay time)	RW

40.4.4.12 时序配置寄存器 (DMC_TMCR_T_RC)

Offset	Bit Position																															
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000000B																															
Name	Reserved																T_RC															
Access																																

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:0	T_RC	T_RC 周期设定	T_RC 设定 (Active bank x to Active bank x delay time)	RW

40.4.4.13 时序配置寄存器 (DMC_TMCR_T_RCD)

Offset	Bit Position																															
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000035																															
Name	Reserved																										T_RCD_P		Reserved	T_RCD_B		
Access																											RW			RW		

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:4	T_RCD_P	附加 T_RCD 周期设定	附加 T_RCD 设定	RW
3	Reserved	保留	-	-
2:0	T_RCD_B	基础 T_RCD 周期设定	基础 T_RCD 设定，需设置为非 0 值	RW

T_RCD (RAS to CAS minimum delay time) 计算方式如下：

- CKSTOP=0 时：
 - ▶ 当 T_RCD_B>=T_RCD_P+3 时，实际 T_RCD 宽度为 T_RCD_B 个 DMC 模块时钟周期；
 - ▶ 当 T_RCD_B<T_RCD_P+3 时，实际 T_RCD 宽度为(T_RCD_P+3) 个 DMC 模块时钟周期。
- CKSTOP=1 时：
 - ▶ 当 T_RCD_B>=T_RCD_P+2 时，实际 T_RCD 宽度为 T_RCD_B 个 DMC 模块时钟周期；
 - ▶ 当 T_RCD_B<T_RCD_P+2 时，实际 T_RCD 宽度为(T_RCD_P+3) 个 DMC 模块时钟周期。

40.4.4.14 时序配置寄存器 (DMC_TMCR_T_RFC)

Offset	Bit Position																															
0x002C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00001012																															
Name	Reserved																T_RFC_P				Reserved				T_RFC_B							
Access																	RW								RW							

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:8	T_RFC_P	附加 T_RFC 周期 设定	附加 T_RFC 设定	RW
7:5	Reserved	保留	-	-
4:0	T_RFC_B	基础 T_RFC 周期 设定	基础 T_RFC 设定	RW

T_RFC (autorefresh command time) 计算方式如下：

- 当 T_RFC_B>=T_RFC_P+3 时，实际 T_RFC 宽度为 T_RFC_B 个 DMC 模块时钟周期；
- 当 T_RFC_B<T_RFC_P+3 时，实际 T_RFC 宽度为(T_RFC_P+3)个 DMC 模块时钟周期。

40.4.4.15 时序配置寄存器 (DMC_TMCR_T_RP)

Offset	Bit Position																															
0x0030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000035																															
Name	Reserved																								T_RP_P		Reserved	T_RP_B				
Access																									RW			RW				

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:4	T_RP_P	附加 T_RP 周期设定	附加 T_RP 设定	RW
3	Reserved	保留	-	-
2:0	T_RP_B	基础 T_RP 周期设定	基础 T_RP 设定, 需设置为非 0 值	RW

T_RP 计算方式如下:

- CKSTOP=0 时:
 - ▶ 当 T_RP_B ≥ T_RP_P+3 时, 实际 T_RP 宽度为 T_RP_B 个 DMC 模块时钟周期;
 - ▶ 当 T_RP_B < T_RP_P+3 时, 实际 T_RP 宽度为 (T_RP_P+3) 个 DMC 模块时钟周期。
- CKSTOP=1 时:
 - ▶ 当 T_RP_B ≥ T_RP_P+2 时, 实际 T_RP 宽度为 T_RP_B 个 DMC 模块时钟周期;
 - ▶ 当 T_RP_B < T_RP_P+2 时, 实际 T_RP 宽度为 (T_RP_P+3) 个 DMC 模块时钟周期。

40.4.4.16 时序配置寄存器 (DMC_TMCR_T_RRD)

Offset	Bit Position																															
0x0034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																														T_RRD	
Access																																

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:0	T_RRD	T_RRD 周期设定	T_RRD 设定 (Active bank x to Active bank y delay time)	RW

40.4.4.17 时序配置寄存器 (DMC_TMCR_T_WR)

Offset	Bit Position																															
0x0038	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000003																															
Name	Reserved																														RW T_WR	
Access																																

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	T_WR	T_WR 周期设定	T_WR 设定 (write to precharge delay time)	RW

40.4.4.18 时序配置寄存器 (DMC_TMCR_T_WTR)

Offset	Bit Position																															
0x003C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000002																															
Name	Reserved																															T_WTR
Access																																

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2:0	T_WTR	T_WTR 周期设定	T_WTR 设定 (write to read delay time)	RW

40.4.4.19 时序配置寄存器 (DMC_TMCR_T_XP)

Offset	Bit Position																															
0x0040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved																								T_XP							
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	T_XP	T_XP 周期设定	T_XP 设定 (exit power-down command time)	RW

40.4.4.20 时序配置寄存器 (DMC_TMCR_T_XSR)

Offset	Bit Position																															
0x0044	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000000A																															
Name	Reserved																								T_XSR							
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	T_XSR	T_XSR 周期设定	T_XSR 设定 (exit self-refresh command time)	RW

40.4.4.21 时序配置寄存器 (DMC_TMCR_T_ESR)

Offset	Bit Position																															
0x0048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000014																															
Name	Reserved																								T_ESR							
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	T_ESR	T_ESR 周期设定	T_ESR 设定 (self-refresh command time)	RW

40.4.5 NFC-NAND Flash 寄存器

40.4.5.1 状态寄存器 (NFC_STSR)

Offset	Bit Position																															
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								PECC		CHIP_BUSY					
Access																									RO		RO					

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8	PECC	ECC 状态	0b0: NFC 没有进行 ECC 计算 0b1: NFC 正在进行 ECC 计算	RO
7:0	CHIP_BUSY	设备状态	对应 BANK7~0 设备： 0b0: 设备正忙 0b1: 设备就绪	RO

40.4.5.2 控制寄存器 (NFC_STCR)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															OPENP
Access																																

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	OPENP	open page 使能	0b0: 不允许 NFC 在复位后自动发送一个读命令 0b1: 允许 NFC 在复位后自动发送一个读命令	RW

40.4.5.3 数据寄存器 (NFC_DATR)

Offset	Bit Position																															
0x0000~0x7FFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DATA																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	数据	数据寄存器用于读写 Flash 设备上的数据。读写数据的地址由索引寄存器写入的地址决定	RW

40.4.5.4 命令寄存器 (NFC_CMDR)

Offset	Bit Position																															
0x8000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ARG												CMD																			
Access	RW												RW																			

位/位域	标记	位名	功能描述	读写
31:8	ARG	参数	用于对命令进行调整, 详见下表	RW
7:0	CMD	命令	发送至设备的命令, 详见下表	RW

所有命令和参数的设定方法如下:

表 40-94 命令参数列表

命令 (CMD)	参数 (ARG)	功能
0x00	0x000000	Read 命令
0x10	0x000000	Page program confirm
0x11	0x000000	用于 two-plane page program 第一个 page 结束后插入的一个 confirm
0x15	0x000000	Page cache program 中的第二个命令
0x23	0x000000	读取 ECC 状态。写入此命令后 NFC 将从 NAND Flash 的额外区域读取 ECC 数据并开始 ECC 计算
0x60	0x810000~0x810007	Block erase 命令, 参数用于指示操作对象 BANK0~BANK7
0x70	0x810000~0x810007	Read status 命令, 参数用于指示操作对象 BANK0~BANK7
0x80	0x000000	Page program 命令
0x81	0x000000	用于 two-plane page program 第二个 page 开始写入
0x90	0x000000~0x000007	Read ID 命令, 参数用于指示操作对象 BANK0~BANK7
0xD0	0x810000~0x810007	Block erase confirm, 参数用于指示操作对象 BANK0~BANK7
0xD1	0x810000~0x810007	Block erase interleaved confirm, 参数用于指示操作对象 BANK0~BANK7
0xE0	0x000000	开始读数据。用于指示 NFC 开始读取数据, 写入该命令后, NFC 将开放 1 个 page 用于数据读取, 此时 CHIP_BUSY 将置位, 等待 CHIP_BUSY 复位后可以读取数据寄存器获得数据
0xFE	0x000000	读操作完成后写入该命令指示 NFC 将片选信号 CE 置为无效
0xFF	0x000000~0x000007	Reset 命令, 参数用于指示操作对象 BANK0~BANK7
0x05	0x810000~0x810007	Change read column 命令的第一周期
0x06	0x810000~0x810007	Change read column enhanced 命令的第一周期
0x31	0x830000~0x830007	Read cache sequential 命令的第一周期和 Read cache random 命令的第二周期
0x32	0x810000~0x810007	Copyback read interleaved 命令
0x35	0x810000~0x810007	Copyback read 命令
0x3F	0x830000~0x830007	Read cache end
0x78	0x800000~0x800007	Read status enhanced 命令

命令 (CMD)	参数 (ARG)	功能
0x85	0x810000~0x810007	Copyback program interleaved 和 Copyback program 命令
0x85	0x840000~0x840007	Change write column 和 Change row address
0xE0	0x850000~0x850007	Change read column 和 Change read column enhanced 命令的第二周期
0xEC	0x830000~0x830007	Read parameter page 命令
0xED	0x830000~0x830007	Read Unique ID 命令
0xEE	0x830000~0x830007	Get Features 命令
0xEF	0x830000~0x830007	Set Features 命令
0xFA	0x820000~0x820007	Reset LUN 命令
0xFC	0x820000~0x820007	Asynchronous reset 命令

40.4.5.5 索引寄存器 0 (NFC_IDXR0)

Offset	Bit Position																															
0x8004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IDX0																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	IDX0	索引	组成访问地址的低 32 位，高 8 位由 IDX1 寄存器组成	RW

40.4.5.6 索引寄存器 1 (NFC_IDXR1)

Offset	Bit Position																															
0x8048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																IDX1															
Access																																

位/位域	标记	位名	功能描述	读写
31:8	Reserved	保留	-	-
7:0	IDX1	索引	组成访问地址的高 8 位，低 32 位由 IDX0 寄存器组成	RW

IDXR0 和 IDX1 共同组成了访问 NAND Flash 设备时的 40 位地址。寄存器写入的值和实际设备的地址关系取决于设备的容量。下表列出了访问各个容量设备时的地址构成：

表 40-99 索引寄存器值与 MEM 地址对应表

容量	2K page			4K page			8K page		
	page	column	bank	page	column	bank	page	column	bank
512Mbits	26:12	11:0	29:27	26:13	12:0	29:27	26:14	13:0	29:27
1Gbits	27:12	11:0	30:28	27:13	12:0	30:28	27:14	13:0	30:28
2Gbits	28:12	11:0	31:29	28:13	12:0	31:29	28:14	13:0	31:29
4Gbits	29:12	11:0	32:30	29:13	12:0	32:30	29:14	13:0	32:30
8Gbits	30:12	11:0	33:31	30:13	12:0	33:31	30:14	13:0	33:31
16Gbits	31:12	11:0	34:32	31:13	12:0	34:32	31:14	13:0	34:32
32Gbits	32:12	11:0	35:33	32:13	12:0	35:33	32:14	13:0	35:33
64Gbits	33:12	11:0	36:34	33:13	12:0	36:34	33:14	13:0	36:34

40.4.5.7 基本配置寄存器 (NFC_BACR)

Offset	Bit Position																																
0x8054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00002187																																
Name	Reserved								SCS								Reserved	RAC	ECCM	WP	PAGE	BANK	Reserved	16BIT	SIZE								
Access									RW									RW	RW	RW	RW	RW	RW		RW	RW							

位/位域	标记	位名	功能描述	读写
31:24	Reserved	保留	-	-
23:16	SCS	spare column size	设置 spare column size 的大小，以 word 为单位	RW
15:14	Reserved	保留	-	-
13	RAC	row address cycle	0b0: 2 row address cycles 0b1: 3 row address cycles	RW
12:11	ECCM	ECC 模式	0b00: 1bitECC 模式 0b01: 4bitECC 模式 - 其他: 禁止设置	RW
10	WP	写保护	0b0: 将设备置为写保护状态 0b1: 将设备置为未写保护状态	RW
9:8	PAGE	page size 大小	0b00: 禁止设置 0b01: 2K 0b10: 4K 0b11: 8K	RW
7:6	BANK	bank(CE)数	0b00: 1bank 0b01: 2bank 0b10: 4bank 0b11: 8bank	RW
5:4	Reserved	保留	-	-
3	16BIT	数据位宽	0b0: 8-bit 0b1: 16-bit	RW
2:0	SIZE	设备容量	0b000: 16Gbits 0b001: 32Gbits 0b010: 64Gbits 0b011: 512Mbits 0b100: 1Gbits 0b101: 2Gbits 0b110: 4Gbits 0b111: 8Gbits	RW

40.4.5.8 中断使能寄存器 (NFC_IENR)

Offset	Bit Position																																					
0x8030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																RBEN				ECCDIS	ECCEEN	Reserved	ECCEN	Reserved	ECCEEN	Reserved											
Access																	RW				RW	RW		RW														

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	RBEN	就绪中断使能	对应 BANK7~0 设备： 0b0：当访问设备就绪（RB 信号置位）时不允许产生中断 0b1：当访问设备就绪（RB 信号置位）时允许产生中断	RW
7	ECCDIS	ECC 禁用	0b0：启用 ECC 功能 0b1：禁用 ECC 功能	RW
6	ECCEEN	ECC 错误中断使能	0b0：ECC 错误发生时不产生中断 0b1：ECC 错误发生时产生中断	RW
5	Reserved	保留	-	-
4	ECCEN	ECC 计算完成中断使能	0b0：ECC 计算完成时不产生中断 0b1：ECC 计算完成时产生中断	RW
3:2	Reserved	保留	-	-
1	ECCEEN	可纠正 ECC 错误中断使能	0b0：1bitECC 模式下发生可纠正 ECC 错误时不产生中断 0b1：1bitECC 模式下发生可纠正 ECC 错误时产生中断	RW
0	ECCEUEN	不可纠正 ECC 错误中断使能	0b0：1bitECC 模式下发生不可纠正 ECC 错误时不产生中断 0b1：1bitECC 模式下发生不可纠正 ECC 错误时产生中断	RW

40.4.5.9 中断状态寄存器 (NFC_ISTR)

Offset	Bit Position																																									
0x8034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
Reset	0x00000000																																									
Name	Reserved																RBST								Reserved	ECCEST	Reserved	ECCEST	Reserved	ECCEST	Reserved	ECCEST										
Access																	RW									RW		RW		RW		RW		RW		RW		RW		RW		RW

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	RBST	就绪状态	对应 BANK7~0 设备： 写 0 将该位清零；写 1 无效 读取： ● 0b0：访问设备未就绪 ● 0b1：访问设备已就绪（RB 信号置位）	RW
7	Reserved	保留	-	-
6	ECCEST	ECC 错误状态	写 0 将该位清零；写 1 无效 读取： ● 0b0：4bitECC 错误未发生 ● 0b1：4bitECC 错误已发生	RW
5	Reserved	保留	-	-
4	ECCEST	ECC 计算完成状态	写 0 将该位清零；写 1 无效 读取： ● 0b0：ECC 计算未完成 ● 0b1：ECC 计算已完成	RW
3:2	Reserved	保留	-	-
1	ECCEST	可纠正 ECC 错误状态	写 0 将该位清零；写 1 无效 读取： ● 0b0：1bitECC 模式下未发生可纠正 ECC 错误 ● 0b1：1bitECC 模式下已发生可纠正 ECC 错误	RW
0	ECCEUST	不可纠正 ECC 错误状态	写 0 将该位清零；写 1 无效 读取： ● 0b0：1bitECC 模式下未发生不可纠正 ECC 错误 ● 0b1：1bitECC 模式下已发生不可纠正 ECC 错误	RW

40.4.5.10 中断结果寄存器 (NFC_IRSR)

Offset	Bit Position																															
0x8038	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RBRS								Reserved	ECCERS	Reserved	ECCCRS	Reserved	ECCECRS	ECCEURS	
Access																									RO		RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:8	RBRS	就绪中断	对应 BANK7~0 设备： 该位为 NFC_ISTR.RBST 和 NFC_IENR.RBEN 的与逻辑结果 0b0：访问设备就绪中断未产生 0b1：访问设备就绪中断产生	RO
7	Reserved	保留	-	-
6	ECCERS	ECC 错误中断	该位为 NFC_ISTR.ECCEST 和 NFC_IENR.ECCEEN 的与逻辑结果 0b0：4bitECC 错误中断未产生 0b1：4bitECC 错误中断产生	RO
5	Reserved	保留	-	-
4	ECCCRS	ECC 计算完成中断	该位为 NFC_ISTR.ECCCST 和 NFC_IENR.ECCCEEN 的与逻辑结果 0b0：ECC 计算完成中断未产生 0b1：ECC 计算完成中断产生	RO
3:2	Reserved	保留	-	-
1	ECCECRS	可纠正 ECC 错误中断	该位为 NFC_ISTR.ECCECST 和 NFC_IENR.ECCECEN 的与逻辑结果 0b0：1bitECC 模式下可纠正 ECC 错误中断未产生 0b1：1bitECC 模式下可纠正 ECC 错误中断产生	RO
0	ECCEURS	不可纠正 ECC 错误中断	该位为 NFC_ISTR.ECCEUST 和 NFC_IENR.ECCEUEN 的与逻辑结果 0b0：1bitECC 模式下不可纠正 ECC 错误中断未产生 0b1：1bitECC 模式下不可纠正 ECC 错误中断产生	RO

40.4.5.11 时序配置寄存器 0 (NFC_TMCRO)

Offset	Bit Position																															
0x804C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x03030202																															
Name	TH								TRP								TWP								TS							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	TH	-	设置 CLE/ALE/CE 在 WE/RE 变化后的保持时间	RW
23:16	TRP	-	设置 RE 的有效（低电平）脉冲宽度	RW
15:8	TWP	-	设置 WE 的有效（低电平）脉冲宽度	RW
7:0	TS	-	设置 CLE/ALE/CE 在 WE/RE 变化前的建立时间	RW

40.4.5.12 时序配置寄存器 1 (NFC_TMCRI1)

Offset	Bit Position																															
0x8050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x28080303																															
Name	TWB								TRR								TRH								TWH							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	TWB	-	设置 WE 上升沿到 RB 下降沿的时间	RW
23:16	TRR	-	设置 RB 上升沿到 RE 下降沿的时间	RW
15:8	TRH	-	设置 RE 的无效（高电平）脉冲宽度	RW
7:0	TWH	-	设置 WE 的无效（高电平）脉冲宽度	RW

时序配置寄存器 0 和时序配置寄存器 1 设置的参数如下图所示：

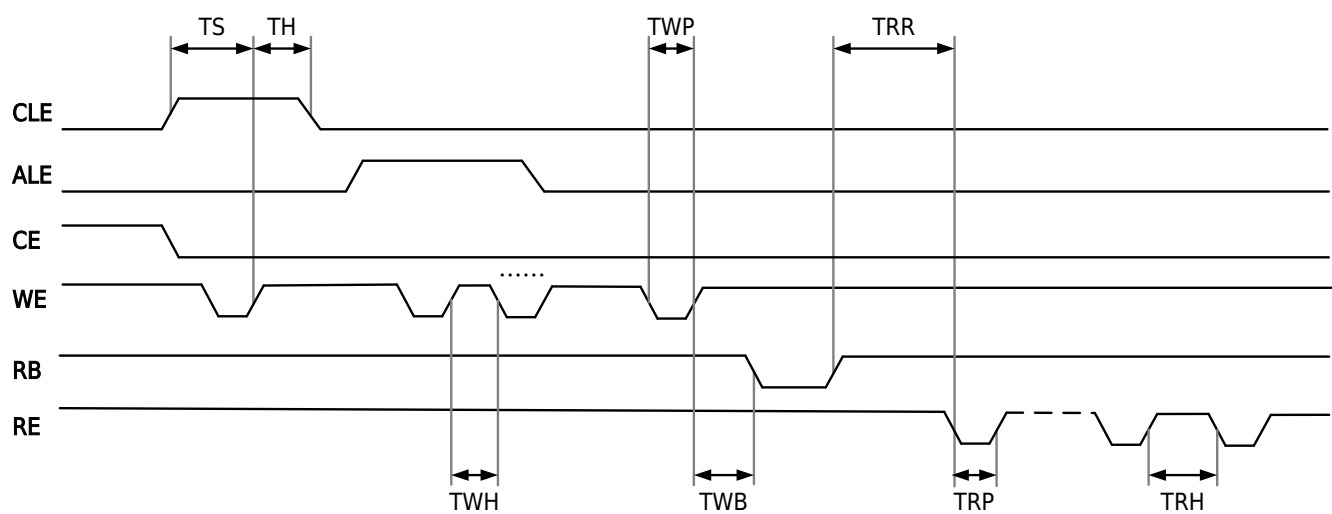


图 40-36 NFC 时序控制图 1

40.4.5.13 时序配置寄存器 2 (NFC_TMCR2)

Offset	Bit Position																															
0x805C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x03050D03																															
Name	TADL								TRTW								TWTR								TCCS							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	TADL	-	设置 ALE 到读写数据的时间	RW
23:16	TRTW	-	设置 RE 上升沿到 WE 下降沿的时间	RW
15:8	TWTR	-	设置 WE 上升沿到 RE 下降沿的时间	RW
7:0	TCCS	-	设置 Change read column 和 Change write column 命令后的延迟时间	RW

时序配置寄存器 2 设置的参数如下图所示：

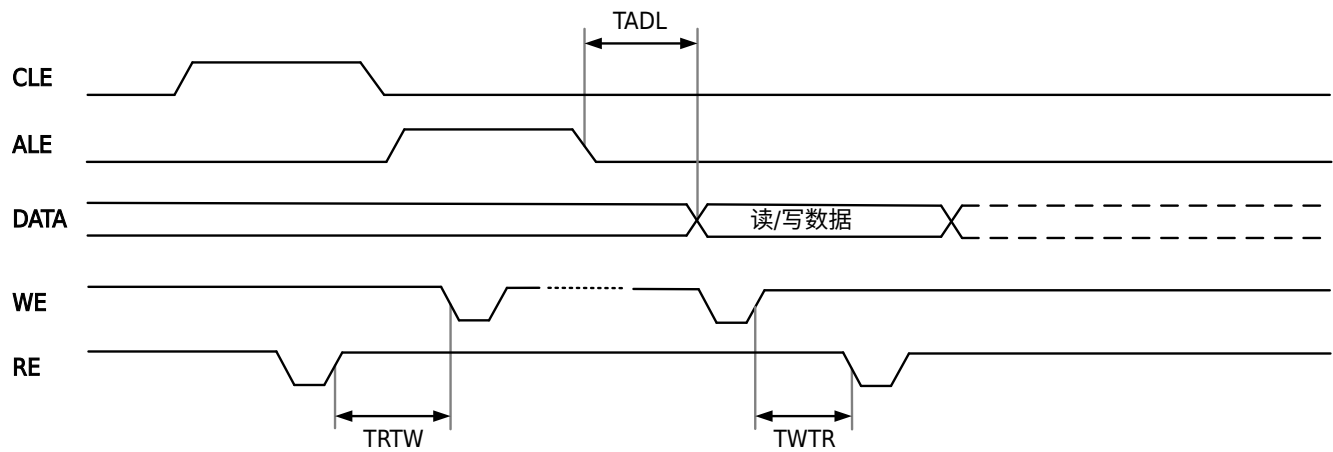


图 40-37 NFC 时序控制图 2

40.4.5.14 ECC 校验寄存器 (NFC_ECCR)

1bitECC 模式

在 1bitECC 模式下，ECCR0~15（偏移地址 Offset：0x8060~0x809C）作为 ECC 结果寄存器：

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																			RO ME		RO SE		RO ERRLOC										
Access																				RO		RO												

位/位域	标记	位名	功能描述	读写
31:14	Reserved	保留	-	-
13	ME	多位错误	0b0：未检测到多位 ECC 错误 0b1：检测到多位 ECC 错误	RO
12	SE	一位错误	0b0：未检测到一位 ECC 错误 0b1：检测到一位 ECC 错误	RO
11:0	ERRLOC	错误位置	表示错误的位置，只有当 SE 置位时有意义 [2:0]：错误 Byte 中的 Bit 位置 0b000：Bit0 错误 0b111：Bit7 错误 [11:3]：错误 Byte 在 512Byte 中的位置 0x000：第 1 个 Byte 0x1FF：第 512 个 Byte	RO

4bitECC 模式

在 4bitECC 模式下，ECCR7~71（偏移地址 Offset：0x807C~0x817C）作为 ECC 状态寄存器和伴随式寄存器。如下表所示，其中以 512 字节为一个段，每个段对应 8 个 ECC 伴随式系数。例如：段 0 伴随式寄存器对应第一个 512 字节的 ECC 值，总共包含 4 个寄存器（段 0 伴随式系数 1 和 2 寄存器

（ECC_SECTION0_SYND12）、段 0 伴随式系数 3 和 4 寄存器（ECC_SECTION0_SYND34）、段 0 伴随式系数 5 和 6 寄存器（ECC_SECTION0_SYND56）、段 0 伴随式系数 7 和 8 寄存器（ECC_SECTION0_SYND78）），每个寄存器包含 2 个 ECC 伴随式系数。

表 40-116 4bitECC 模式时的 ECCR 寄存器列表

偏移地址	寄存器	描述	位宽	复位值
0x807C	ECC_STAT	ECC 状态寄存器	32	0x00000000
0x8080~0x808C	ECC_SECTION0_SYND	段 0 伴随式寄存器	32	0x00000000
0x8090~0x809C	ECC_SECTION1_SYND	段 1 伴随式寄存器	32	0x00000000
0x80A0~0x80AC	ECC_SECTION2_SYND	段 2 伴随式寄存器	32	0x00000000
0x80B0~0x80BC	ECC_SECTION3_SYND	段 3 伴随式寄存器	32	0x00000000
0x80C0~0x80CC	ECC_SECTION4_SYND	段 4 伴随式寄存器	32	0x00000000
0x80D0~0x80DC	ECC_SECTION5_SYND	段 5 伴随式寄存器	32	0x00000000

偏移地址	寄存器	描述	位宽	复位值
0x80E0~0x80EC	ECC_SECTION6_SYND	段 6 伴随式寄存器	32	0x00000000
0x80F0~0x80FC	ECC_SECTION7_SYND	段 7 伴随式寄存器	32	0x00000000
0x8100~0x810C	ECC_SECTION8_SYND	段 8 伴随式寄存器	32	0x00000000
0x8110~0x811C	ECC_SECTION9_SYND	段 9 伴随式寄存器	32	0x00000000
0x8120~0x812C	ECC_SECTION10_SYND	段 10 伴随式寄存器	32	0x00000000
0x8130~0x813C	ECC_SECTION11_SYND	段 11 伴随式寄存器	32	0x00000000
0x8140~0x814C	ECC_SECTION12_SYND	段 12 伴随式寄存器	32	0x00000000
0x8150~0x815C	ECC_SECTION13_SYND	段 13 伴随式寄存器	32	0x00000000
0x8160~0x816C	ECC_SECTION14_SYND	段 14 伴随式寄存器	32	0x00000000
0x8170~0x817C	ECC_SECTION15_SYND	段 15 伴随式寄存器	32	0x00000000

下表为每段（512 字节）对应的 8bitECC 值寄存器，以伴随式寄存器 0（0x8080~0x808C）为例。

表 40-117 4bitECC 模式时每段 ECCR 寄存器

偏移地址	寄存器	段伴随式系数寄存器名	位宽	复位值
0x8080	ECC_SECTION0_SYND12	段 0 伴随式系数 1 和 2 寄存器	32	0x00000000
0x8084	ECC_SECTION0_SYND34	段 0 伴随式系数 3 和 4 寄存器	32	0x00000000
0x8088	ECC_SECTION0_SYND56	段 0 伴随式系数 5 和 6 寄存器	32	0x00000000
0x808C	ECC_SECTION0_SYND78	段 0 伴随式系数 7 和 8 寄存器	32	0x00000000

ECC 状态寄存器 (ECC_STAT)

Offset	Bit Position																															
0x807C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																																
Name	Reserved																ECCSEC															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	ECCSEC	错误位置	对应段 15~0： ● 0b0：对应 section 未出现 ECC 错误 ● 0b1：对应 section 出现 ECC 错误	RO

段 x 伴随式系数 1 和 2 寄存器 (ECC_SECTIONx_SYND12) (x=0~15)

偏移地址 (Offset) : 0x8080+0x10*x

Offset	Bit Position																																			
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved		RO S2																Reserved		RO S1															
Access			RO																		RO															

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:16	S2	-	α 2 伴随式系数	RO
15:13	Reserved	保留	-	-
12:0	S1	-	α 伴随式系数	RO

段 x 伴随式系数 3 和 4 寄存器 (ECC_SECTIONx_SYND34) (x=0~15)

偏移地址 (Offset) : 0x8084+0x10*x

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved		RO S4														Reserved		RO S3													
Access																																

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:16	S4	-	α 4 伴随式系数	RO
15:13	Reserved	保留	-	-
12:0	S3	-	α 3 伴随式系数	RO

段 x 伴随式系数 5 和 6 寄存器 (ECC_SECTIONx_SYND56) (x=0~15)

偏移地址 (Offset) : 0x8088+0x10*x

Offset	Bit Position																																			
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved		RO S6														Reserved		RO S5																	
Access			RO S6																RO S5																	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:16	S6	-	α6 伴随式系数	RO
15:13	Reserved	保留	-	-
12:0	S5	-	α5 伴随式系数	RO

段 x 伴随式系数 7 和 8 寄存器 (ECC_SECTIONx_SYND78) (x=0~15)

偏移地址 (Offset) : 0x808C+0x10*x

Offset	Bit Position																																			
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved		RO S8														Reserved		RO S7																	
Access			RO S8																RO S7																	

位/位域	标记	位名	功能描述	读写
31:29	Reserved	保留	-	-
28:16	S8	-	α8 伴随式系数	RO
15:13	Reserved	保留	-	-
12:0	S7	-	α7 伴随式系数	RO

40.5 注意事项

- 1. SMC 的相关寄存器只能在 Ready 状态下设定；DMC 的相关寄存器只能在 Config 或 Low Power 状态下设定。
- 2. 在使用 SMC、DMC 模块对外部 Memory 进行读访问时，Memory Device 在接收到控制器发出的读命令后将数据传送给控制器，控制器对传送回的数据进行采样的时钟可以通过 SMC_BACR.CKSEL 或 DMC_BACR.CKSEL 进行选择，具体设定请参考寄存器章节。选择不同的采样时钟，可以改善通信路径的时间裕量。

下图是同步模式下的 SMC 访问例。

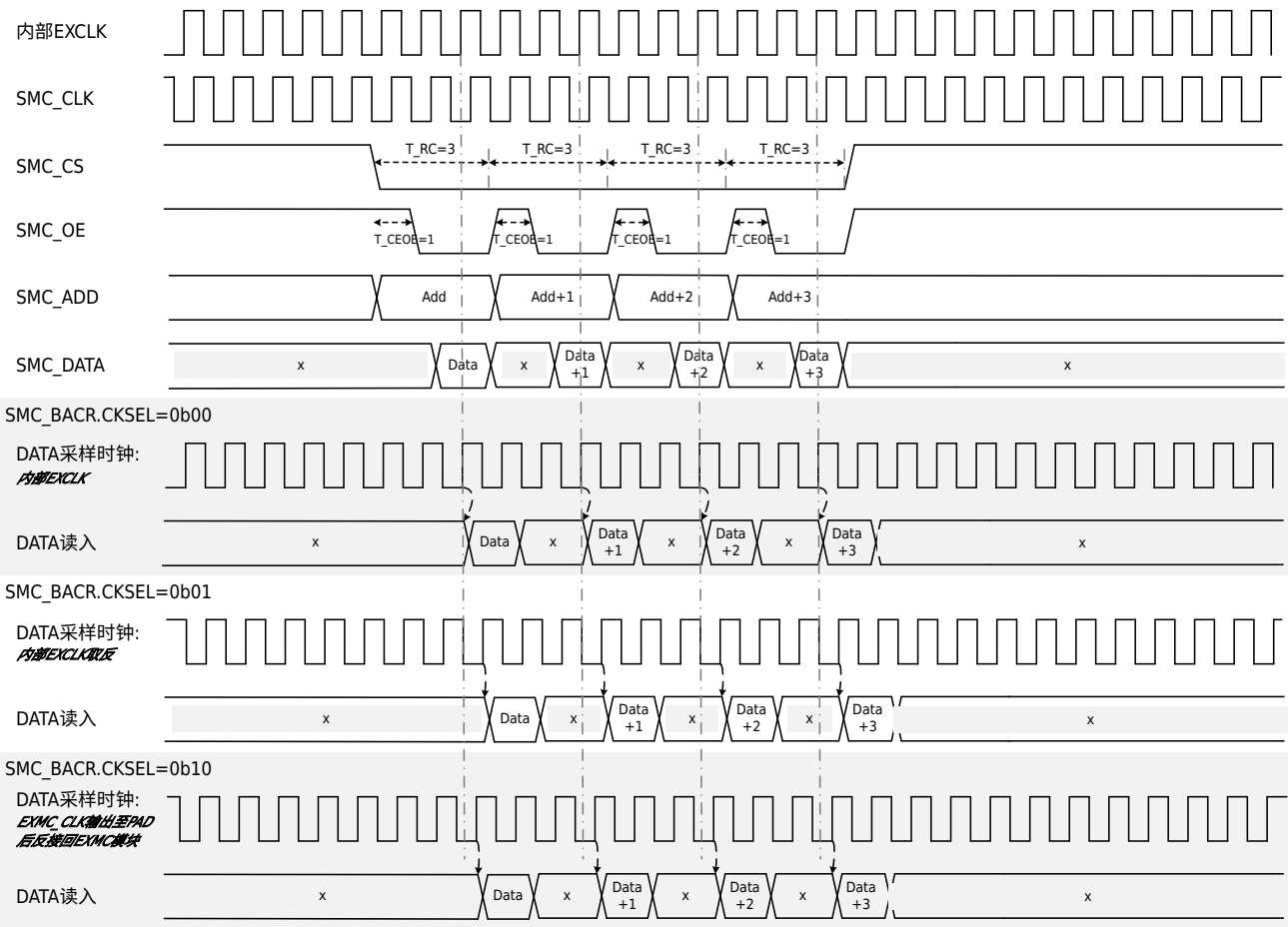


图 40-38 外部反馈时钟访问

- 3. 使用 SMC、DMC 功能时，如果访问 NFC 寄存器，则访问结束后需要插入 NFC 的 DUMMY 读访问来清除 NFC 的访问状态。

41 数字视频接口 (DVP)

41.1 简介

数字摄像头接口 (DVP) 是一个同步并行接口，可采集从外部 CMOS 摄像头模块传入的 8 位、10 位、12 位或 14 位高速数据流。支持软件同步和硬件同步。支持对数据流的采集频率控制和窗口裁剪控制。支持单色或原始拜尔格式/YCbCr4:2:2/RGB565 逐行视频和压缩数据(JPEG)等不同格式的数据流采集。

41.2 主要特性

DVP 基本的功能及特性如下表所示。

表 41-1 DVP 基本功能及特性

基本功能	8 位、10 位、12 位、14 位并行接口
	单帧模式和连续模式
	软件、硬件行同步和帧同步
	帧采集频率控制
	窗口裁剪
	FIFO 控制
数据格式	单色或原始拜尔格式
	YCbCr4:2:2 逐行视频
	RGB565 逐行视频
	JPEG 压缩数据
中断事件	帧开始中断和事件
	帧结束中断和事件
	行开始中断和事件
	行结束中断和事件
	软件同步错误中断和事件
	FIFO 溢出错误中断和事件

41.3 功能说明

41.3.1 系统框图

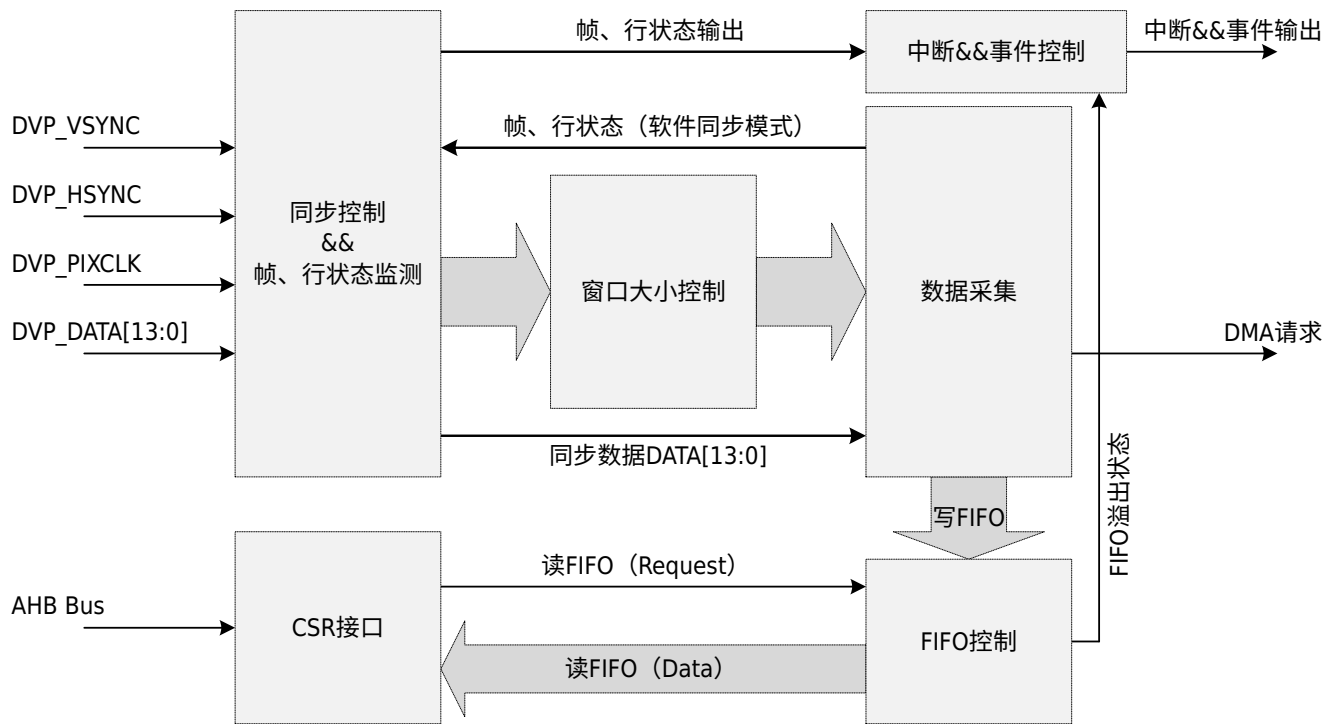


图 41-1 DVP 基本框图

下表所示为 DVP 的输入端口列表。

表 41-2 DVP 端口列表

端口名	方向	功能
DVP_VSYNC	输入	帧同步输入端口
DVP_HSYNC	输入	行同步输入端口
DVP_PIXCLK	输入	时钟输入端口
DVP_DATA[13:0]	输入	数据输入端口

41.3.2 视频数据格式

单色格式

- 每个像素 8 位。

下表显示了数据的存储方式。

表 41-3 单色格式视频数据存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	n+3	n+2	n+1	n
0x04	n+7	n+6	n+5	n+4

YCbCr 格式

- 一个缓冲区交替存储 Y、Cb 和 Cr：CbYCrYCbYCr.....
- DVP 模块支持 YCbCr4:2:2 格式数据传送，像素分量包括 Y（亮度）、Cb 和 Cr（蓝色色度和红色色度）。每个分量都采用 8 位进行编码，亮度和色度（交替）存储在一起。

下表显示了数据的存储方式。

表 41-4 YCbCr 格式视频数据存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	Y n+1	Cr n	Y n	Cb n
0x04	Y n+3	Cr n+2	Y n+2	Cb n+2

RGB565 格式

- 一个缓冲区交替存储 RGB 信号，BGRBGRBGR.....
- DVP 模块支持 16BPP（每个像素 16 位）：RGB565（每 32 位表示 2 个像素），不支持 24BPP（托盘化格式）和灰度格式。

下表显示了数据的存储方式。

表 41-5 RGB565 格式视频数据存储方式

字节地址	31:27	26:21	20:16	15:11	10:5	4:0
0x00	R n+1	G n+1	B n+1	R n	G n	B n
0x04	R n+3	G n+3	B n+3	R n+3	G n+3	B n+3

JPEG 格式

- JPEG 图像不按行和帧存储，DVP_VSYNC 信号用于启动采集过程，DVP_HSYNC 用作数据使能信号。
- DVP_HSYNC 有效区间包含的字节数可能不是 4 的倍数。
- DVP 模块支持 JPEG 格式数据传输（DVP_CTR.JPEGEN=1）。

41.3.3 并口存储格式

DVP 接口由 11、13、15、17 个输入信号组成。仅支持从模式。根据 DVP_CTR 寄存器中 BITSEL[1:0]位的设置，可以采集 8 位、10 位、12 位或 14 位数据。

摄像头模块输出的数据 DVP_DATA 与像素时钟 DVP_PIXCLK 保持同步，并根据像素时钟的极性在像素时钟上升沿/下降沿发生变化；DVP_VSYNC 信号指示帧的开始/结束；DVP_HSYNC 信号指示行的开始/结束。

DVP_VSYNC 或 DVP_HSYNC 有效时，表示同步区间，数据传输无效；DVP_VSYNC 和 DVP_HSYNC 无效时，表示数据区间，数据传输有效。下图所示信号波形图，DVP_DATA 在 DVP_PIXCLK 的上升沿发生变化（DVP 模块通过设定，在 DVP_PIXCLK 的下降沿采集数据 DVP_DATA），DVP_VSYNC 或 DVP_HSYNC 在高电平时为同步区间，数据传输无效。

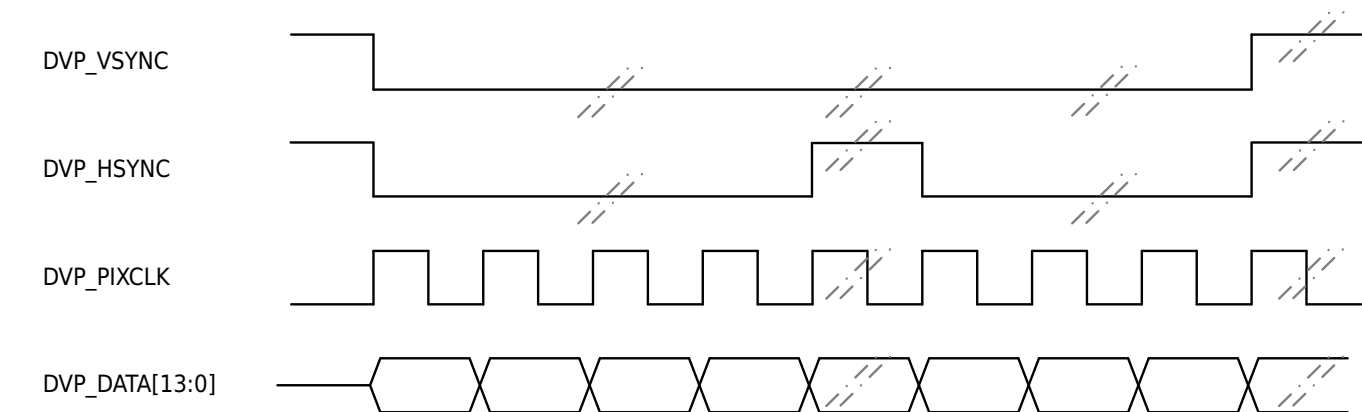


图 41-2 DVP 信号波形示意图

8-bit 数据

当 DVP_CTR.BITSEL[1:0]=0b00 时，DVP 接口将采集其输入 DVP_DATA[7:0]的 8 位数据，并将其存储为 8 位数据，忽略 DVP_DATA[13:8]的输入。在此情况下，每四个像素时钟周期会生成一个 32 位数据字。下表列举了采集到的数据字节在两个 32 位字中的位置排布。

表 41-6 8-bit 数据 DVP 存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	Dn+3[7:0]	Dn+2[7:0]	Dn+1[7:0]	Dn[7:0]
0x04	Dn+7[7:0]	Dn+6[7:0]	Dn+5[7:0]	Dn+4[7:0]

10-bit 数据

当 DVP_CTR.BITSEL[1:0]=0b01 时，DVP 接口将采集其输入 DVP_DATA[9:0]的 10 位数据，并将其存储到 16 位字的 10 个最低有效位，忽略 DVP_DATA[13:10]的输入。此时 DVP_DTR 寄存器中的其余最高有效位（Bit15~10）将清零。在此情况下，每两个像素时钟周期会生成一个 32 位数据字，如下表所示。

表 41-7 10-bit 数据 DVP 存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	0	Dn+1[9:0]	0	Dn[9:0]
0x04	0	Dn+3[9:0]	0	Dn+2[9:0]

12-bit 数据

当 DVP_CTR.BITSEL[1:0]=0b10 时，DVP 接口将采集其输入 DVP_DATA[11:0]的 12 位数据，并将其存储到 16 位字的 12 个最低有效位，忽略 DVP_DATA[13:12]的输入。此时 DVP_DTR 寄存器中的其余最高有效位（Bit15~12）将清零。在此情况下，每两个像素时钟周期会生成一个 32 位数据字，如下表所示。

表 41-8 12-bit 数据 DVP 存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	0	Dn+1[11:0]	0	Dn[11:0]

字节地址	31:24	23:16	15:8	7:0
0x04	0	Dn+3[11:0]	0	Dn+2[11:0]

14-bit 数据

当 DVP_CTR.BITSEL[1:0]=0b11 时，DVP 接口将采集其输入 DVP_DATA[13:0]的 14 位数据，并将其存储到 16 位字的 14 个最低有效位。此时 DVP_DTR 寄存器中的其余最高有效位（Bit15~14）将清零。在此情况下，每两个像素时钟周期会生成一个 32 位数据字，如下表所示。

表 41-9 14-bit 数据 DVP 存储方式

字节地址	31:24	23:16	15:8	7:0
0x00	0	Dn+1[13:0]	0	Dn[13:0]
0x04	0	Dn+3[13:0]	0	Dn+2[13:0]

41.3.4 模式选择

41.3.4.1 单帧模式

单帧模式下（DVP_CTR.CAPMD=1）只采集单个帧。在 DVP_CTR.CAPEN 位置 1 后，DVP 接口将等待系统检测到帧开始，然后对数据进行采集。收到完整的一帧数据后，将自动清零 DVP_CTR 中的 CAPEN 位，摄像头接口不再接收数据。如果使能帧结束中断，将生成对应的中断或事件(DVP_FRAMEND)。

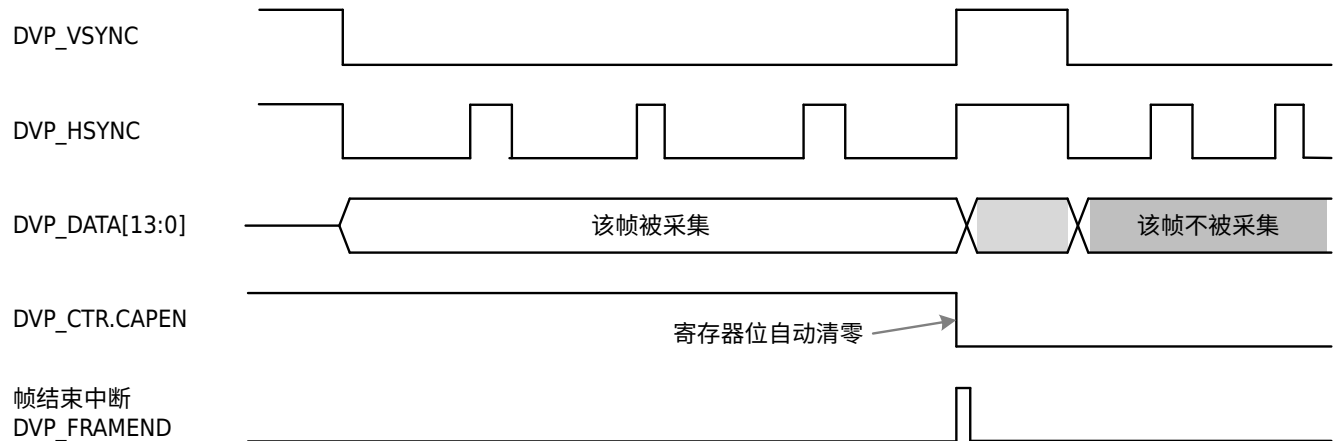


图 41-3 单帧模式数据采集动作

41.3.4.2 连续模式

连续模式下（DVP_CTR.CAPMD=0）会连续采集图像帧。在 DVP_CTR.CAPEN 位置 1 后，DVP 接口将等待系统检测到帧开始，然后对数据进行采集。该过程一直持续到 DVP_CTR 中的 CAPEN 位清零。CPU 对 DVP_CTR.CAPEN 位发出软件清零动作后，数据采集过程持续到当前帧结束，然后 DVP_CTR.CAPEN 位才被清零。

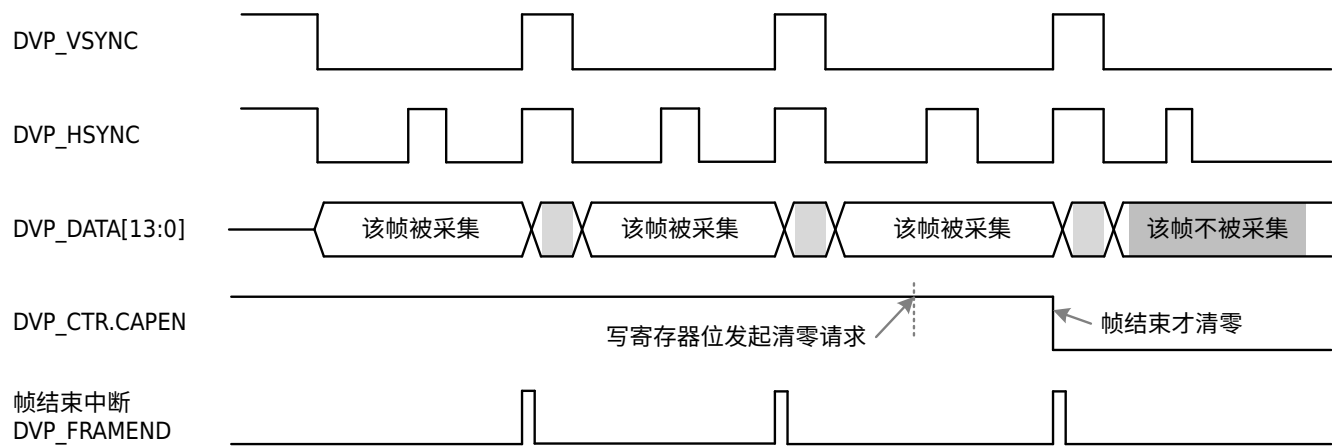


图 41-4 连续模式数据采集动作

在连续模式下，可以通过设定 DVP_CTR.CAPFRC[1:0]位来控制图像帧采集频率，以实现降低带宽的目的。例如当 DVP_CTR.CAPFRC[1:0]=0b01 时，摄像头接口对输入数据隔帧进行采样，即每隔一帧采集一次图像数据。

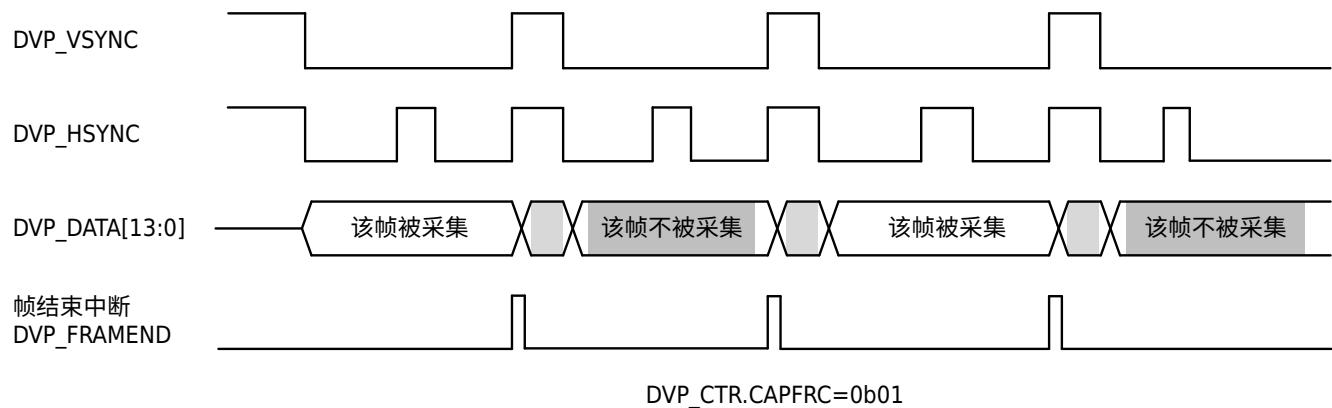


图 41-5 帧采集频率控制

41.3.5 同步控制

41.3.5.1 软件同步

软件同步模式下，将使用数据流中嵌入的 32 位同步码来同步数据流。这些同步码使用数据流中不再使用的 0x00/0xFF。同步码共有 4 种类型，均采用 0xFF0000XY 的格式，DVP 接口通过采集到的一个 32 位字数据判断帧的同步状态。

四个软件同步码分别表示以下事件：

- 帧开始 FS (DVP_SSYNDR.FSDAT[7:0])
- 行开始 LS (DVP_SSYNDR.LSDAT[7:0])
- 行结束 LE (DVP_SSYNDR.LEDAT[7:0])
- 帧结束 FE (DVP_SSYNDR.FEDAT[7:0])

同步码格式 0xFF0000XY 中的 XY 由上述 4 组寄存器位决定。例如，当采集到的数据中 XY 和 DVP_SSYNDR.FSDAT[7:0]相同时，代表一帧数据传送开始。

4 个同步码还具有位屏蔽功能，可以仅使用同步码中未被屏蔽的位进行比较，各同步码的屏蔽位由 DVP_SSYNMR 中的对应位设定。屏蔽功能有效时，可以选择一个位用于同步码的比较，来检测帧/行起始和结束。这意味着可以多个同步码表示帧/行的起始和结束，只要未被屏蔽的位相同即可。例如当 DVP_SSYNDR.FSDAT[7:0]设定为 0xA5 时、DVP_SSYNMR.FSMASK[7:0]设定为 0xF0 时，只需要比较数据码的高 4 位来检测是否是 FS 信号。

根据软件同步机理，DVP 只支持 8 位数据格式接口 (DVP_CTR.BITSEL[1:0]=0b00) 和全帧采集 (DVP_CTR.CAPFRC[1:0]=0b00)，其它数据宽度格式或隔帧采集时不适用，会产生无法预知的结果。JPEG 格式数据传输也不适用 (DVP_CTR.JPEGEN=0)。

41.3.5.2 硬件同步

硬件同步模式下将使用两个端口同步信号(DVP_VSYNC/DVP_HSYNC)。摄像头接口模块根据 DVP_VSYNC/ DVP_HSYNC 的电平变化判断帧/行起始和帧/行结束。
在硬件同步模式时启用数据流采集功能 (DVP_CTR.CAPEN=1)，在 DVP_VSYNC 端口上采集到无效边沿后，数据传输便可以开始执行，由 DMA 将连续帧传输到多个连续的缓冲区或一个具有循环特性的缓冲区。每一帧结束时都将激活帧传送结束中断（当帧传送结束中断 DVP_FRAMEND 使能时）。

41.3.6 窗口裁剪

摄像头接口可以使用裁剪功能从收到的图像中选择一个矩形窗口。窗口的起始坐标和尺寸大小（像素时钟数表示水平尺寸、行数表示垂直尺寸）由窗口裁剪偏移寄存器 (DVP_CPSFTR) 和窗口裁剪尺寸寄存器 (DVP_CPSZER) 指定。

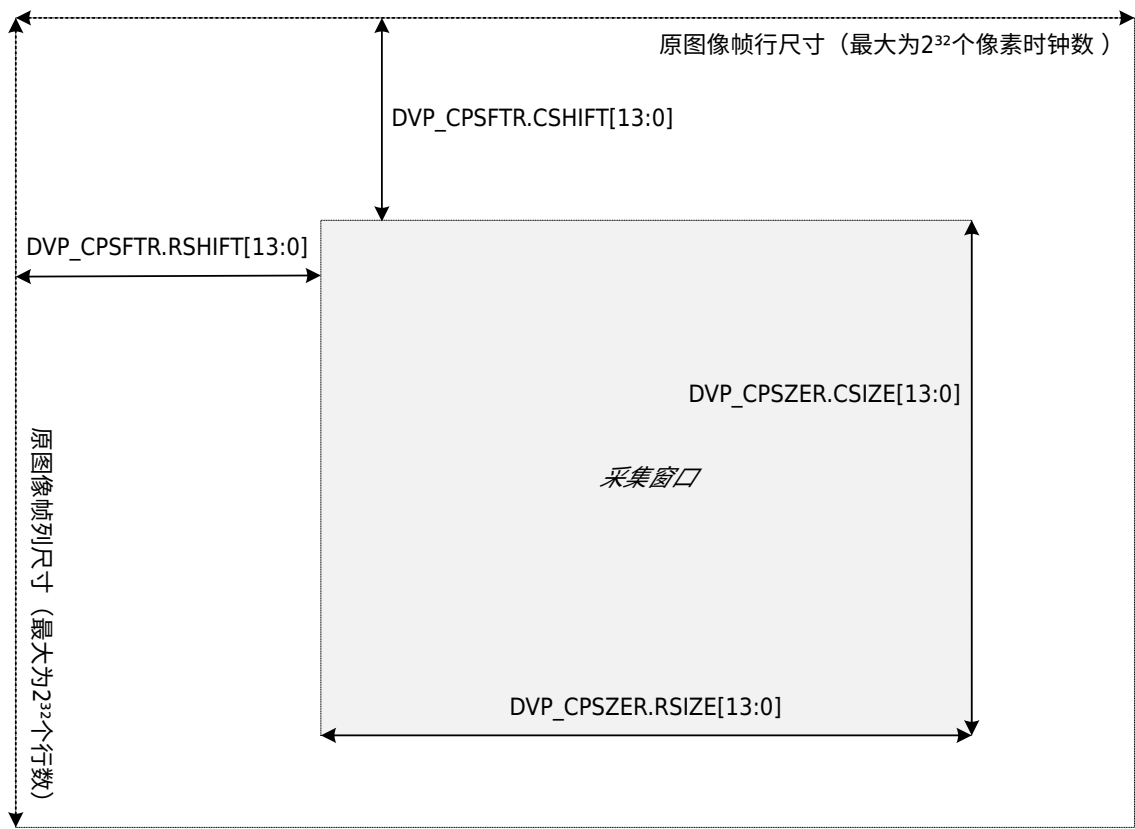


图 41-6 DVP 窗口设定图

裁剪窗口的起点坐标指定为某一行号（从 0 开始）和某个像素时钟数（从 0 开始），窗口大小则指定为行数和像素时钟数。如果在窗口裁剪尺寸寄存器 (DVP_CPSZER) 中指定的行数完成之前，DVP_VSYNC 信号已有效，那么图像采集将停止，并且在中断使能时生成帧传送结束中断 (DVP_FRAMEND)。下图是窗口裁剪数据采集的示意图。

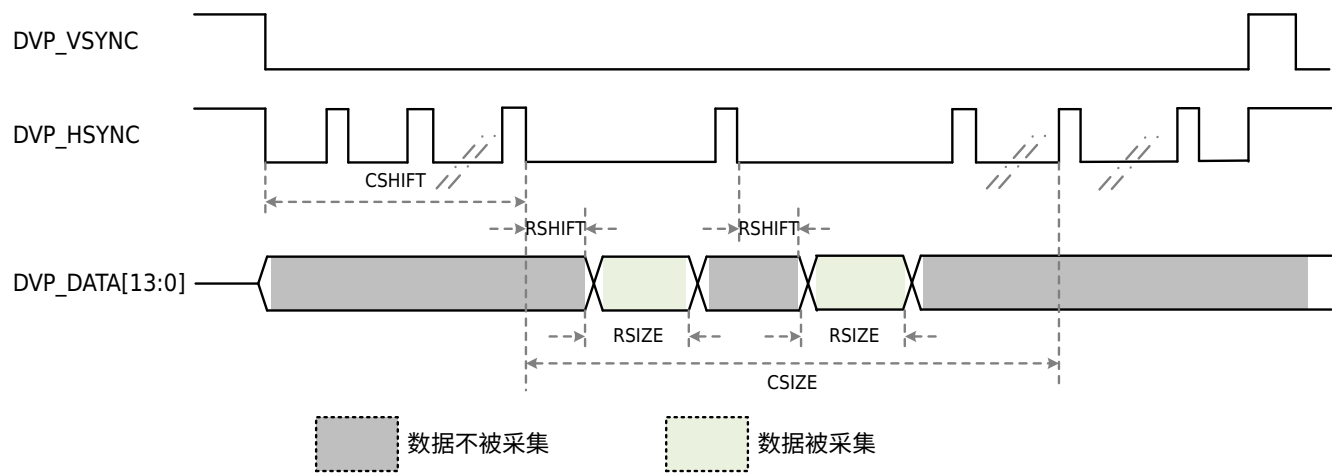


图 41-7 窗口裁剪数据采集图

JPEG 格式数据传输（DVP_CTL.JPEGEN=1）时，窗口裁剪功能不适用，需将窗口裁剪功能无效。

41.3.7 FIFO 控制

为了对 AHB 上的数据传输速度加以管理，在 DVP 模块中搭载了 8 个字深度的 FIFO 并配有一个 FIFO 控制器。每次从摄像头接口 FIFO 读取数据时读指针递增、每次向摄像头接口 FIFO 写入数据时写指针递增。如果数据写入速度超过了 AHB 接口能够承受的速率，FIFO 会产生溢出，数据会被覆盖。此时若使能 FIFO 溢出错误中断，将产生相应的中断。

如果发生软件同步错误事件，FIFO 将复位，DVP 接口将等待新的数据帧开始。

41.3.8 DMA 控制

当 DVP_CTL.CAPEN=1 时，DMA 接口就被激活。摄像头接口每次在数据寄存器（DVP_DTR）中收到一个完整的 32 位数据块时，就可以触发一个 DMA 读请求。DMA 通过 AHB 接口对 DMA 数据传输寄存器（DVP_DMR）访问时，即从 FIFO 中读出对应的缓存数据。该 DMA 读请求是事件列表的第 90 号事件，相关信息请参见《中断控制器（INTC）》的“中断事件表”；DMA 的相关设定请参见《DMA 控制器（DMA）》。在 JPEG 数据传输时，行中包含的字节数可能不是 4 的倍数，DVP 接口在检测到行结束并且尚未凑成 32 位字时，将使用 0 进行填充，并触发一个 DMA 读请求。

41.3.9 中断和事件

41.3.9.1 帧传送状态中断和事件

在图像帧正常传送期间的各个状态时，分别可产生相应的中断。

表 41-10 帧传送中断

中断/事件名	中断/事件符号	中断使能位
帧传送开始	DVP_FRAMSTA	IER.FSIEN
行传送开始	DVP_LINESTA	IER.LSIEN
行传送结束	DVP_LINEEND	IER.LEIEN
帧传送结束	DVP_FRAMEND	IER.FEIEN

41.3.9.2 软件同步错误中断及事件

在软件同步模式时，若接收到的同步码出现下表所示的错误序列时，对应的状态位（DVP_STR.SQUERF）会被置 1，此时若设定相应的中断使能（DVP_IER.SQUERIEN）有效，则产生软件同步错误（DVP_SQUERR）中断。

表 41-11 同步码错误序列

FS	LS	LE	FE
0	0	0	1
0	0	1	x
0	1	x	x
1	0	1	x
1	1	0	1

表中 1 表示接收到该同步序列，0 表示未接收到该同步序列，x 表示该同步序列是否被接收到不影响序列错误判断。

例如，当 FS:LS:LE:FE=0b1101 时，表明在接收到帧开始、行开始序列之后，又接收到了帧结束序列，行结束序列未接收到，此时产生软件同步序列错误事件。

产生同步码错误时，内部的帧采集控制逻辑会复位，不再接收当前帧的数据，直至 DVP_STR.SQUERF 被清零之后，开始接收并监测下一帧的帧开始序列。

41.3.9.3 FIFO 溢出错误中断及事件

在 FIFO 发生溢出时，对应的状态位 (DVP_STR.FIFOERF) 会被置 1，此时若设定相应的中断使能 (DVP_IER.FIFOERIEN) 有效，则产生 FIFO 溢出错误 (DVP_FIFOERR) 中断。

41.4 寄存器

41.4.1 寄存器总表

基地址：0x40055800

表 41-12 数字视频接口 (DVP) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	DVP_CTR	控制寄存器	32
0x0004	DVP_DTR	数据寄存器	32
0x0008	DVP_STR	状态寄存器	32
0x000C	DVP_IER	中断寄存器	32
0x0010	DVP_DMR	DMA 数据传输寄存器	32
0x0020	DVP_SSYNDR	软件同步数据寄存器	32
0x0024	DVP_SSYNMR	软件同步屏蔽寄存器	32
0x0028	DVP_CPSFTR	窗口裁剪偏移寄存器	32
0x002C	DVP_CPSZER	窗口裁剪尺寸寄存器	32

41.4.2 控制寄存器 (DVP_CTR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																		DVPEN	Reserved	BITSEL		CAPFRC		VSYNCSSEL	HSYNCSSEL	PIXCKSEL	SWSYNC	JPEGEN	CROPEN	CAPMD	CAPEN
Access																			RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
14	DVPEN	DVP 模块使能	0b0: DVP 模块无效 0b1: DVP 模块使能	RW
13:12	Reserved	保留	-	-
11:10	BITSEL	数据宽度选择	0b00: 8 位数据宽度 0b01: 10 位数据宽度 0b10: 12 位数据宽度 0b11: 14 位数据宽度	RW
9:8	CAPFRC	帧采集频率选择	0b00: 全帧采集 (采集所有帧) 0b01: 每隔 1 帧采集一次 (带宽降低 50%) 0b10: 每隔 3 帧采集一次 (带宽降低 75%) 0b11: 全帧采集 (采集所有帧)	RW
7	VSYNCSSEL	帧同步电平选择	0b0: DVP_VSYNC 为低电平时, 同步功能有效 0b1: DVP_VSYNC 为高电平时, 同步功能有效	RW
6	HSYNCSSEL	行同步电平选择	0b0: DVP_HSYNC 为低电平时, 同步功能有效 0b1: DVP_HSYNC 为高电平时, 同步功能有效	RW
5	PIXCKSEL	像素时钟选择	0b0: DVP_PIXCLK 下降沿时, 采集数据 0b1: DVP_PIXCLK 上升沿时, 采集数据	RW
4	SWSYNC	软件同步使能	0b0: 软件同步无效 0b1: 软件同步有效	RW
3	JPEGEN	JPEG 传送使能	0b0: 非 JPEG 格式数据传送 0b1: JPEG 格式数据传送	RW
2	CROPEN	窗口裁剪使能	0b0: 窗口裁剪功能无效 0b1: 窗口裁剪功能有效	RW
1	CAPMD	采集模式选择	0b0: 连续模式 0b1: 单帧模式	RW
0	CAPEN	采集功能使能	0b0: 禁止采集 0b1: 使能采集  说明 单帧模式下一帧数据结束后该位自动清零; 连续模式下, 该位写零后, 到当前帧结束后, 清零才生效。	RW

41.4.3 数据寄存器 (DVP_DTR)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DATA																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	采集数据	当前采集到的一组数据	RO

41.4.4 状态寄存器 (DVP_STR)

Offset	Bit Position																															
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										FIFOERF	SQUERF	FEF	LEF	LSF	FSF
Access																											RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	FIFOERF	FIFO 溢出错误标志	● 0b0: 未发生 FIFO 溢出错误 ● 0b1: 发生 FIFO 溢出错误	RW
4	SQUERF	软件同步错误标志	● 0b0: 未发生软件同步错误 ● 0b1: 发生软件同步错误	RW
3	FEF	帧传送结束标志	● 0b0: 未监测到帧传送结束 ● 0b1: 检测到帧传送结束	RW
2	LEF	行传送结束标志	● 0b0: 未监测到行传送结束 ● 0b1: 检测到行传送结束	RW
1	LSF	行传送开始标志	● 0b0: 未监测到行传送开始 ● 0b1: 检测到行传送开始	RW
0	FSF	帧传送开始标志	● 0b0: 未监测到帧传送开始 ● 0b1: 检测到帧传送开始	RW

41.4.5 中断寄存器 (DVP_IER)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										FIFOERIEN	SQUERIEN	FEIEN	LEIEN	LSIEN	FSIEN
Access																											RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	FIFOERIEN	FIFO 溢出错误中断使能	● 0b0: 禁止 ● 0b1: 使能	RW
4	SQUERIEN	软件同步错误中断使能	● 0b0: 禁止 ● 0b1: 使能	RW
3	FEIEN	帧传送结束中断使能	● 0b0: 禁止 ● 0b1: 使能	RW
2	LEIEN	行传送结束中断使能	● 0b0: 禁止 ● 0b1: 使能	RW
1	LSIEN	行传送开始中断使能	● 0b0: 禁止 ● 0b1: 使能	RW
0	FSIEN	帧传送开始中断使能	● 0b0: 禁止 ● 0b1: 使能	RW

41.4.6 DMA 数据传输寄存器 (DVP_DMR)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DMR																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DMR	DMA 传输值	从该寄存器读取当前读 FIFO 中的数据	RO

41.4.7 软件同步数据寄存器 (DVP_SSYNDR)

Offset	Bit Position																															
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	FEDAT								LEDAT								LSDAT								FSDAT							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	FEDAT	帧结束同步码	软件同步模式时的帧结束同步码	RW
23:16	LEDAT	行结束同步码	软件同步模式时的行结束同步码	RW
15:8	LSDAT	行开始同步码	软件同步模式时的行开始同步码	RW
7:0	FSDAT	帧开始同步码	软件同步模式时的帧开始同步码	RW

41.4.8 软件同步屏蔽寄存器 (DVP_SSYNMR)

Offset	Bit Position																															
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	FEMSK								LEMSK								LSMSK								FSMSK							
Access	RW								RW								RW								RW							

位/位域	标记	位名	功能描述	读写
31:24	FEMSK	帧结束屏蔽码	软件同步模式时的帧结束屏蔽码	RW
23:16	LEMSK	行结束屏蔽码	软件同步模式时的行结束屏蔽码	RW
15:8	LSMSK	行开始屏蔽码	软件同步模式时的行开始屏蔽码	RW
7:0	FSMSK	帧开始屏蔽码	软件同步模式时的帧开始屏蔽码	RW

41.4.9 窗口裁剪偏移寄存器 (DVP_CPSFTR)

Offset	Bit Position																															
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	CSHIFT														Reserved	RSHIFT															
Access		RW															RW															

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29:16	CSHIFT	列偏移数	表示窗口开始的行数	RW
15:14	Reserved	保留	-	-
13:0	RSHIFT	行偏移数	表示窗口开始的列数 (时钟像素数)  说明 在软件同步模式时，行偏移数以 4 的倍数进行偏移，Bit1~Bit0 可读写，但功能设定无效。	RW

41.4.10 窗口裁剪尺寸寄存器 (DVP_CPSZER)

Offset	Bit Position																															
0x002C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	CSIZE														Reserved	RSIZE															
Access		RW															RW															

位/位域	标记	位名	功能描述	读写
31:30	Reserved	保留	-	-
29:16	CSIZE	列尺寸数	表示窗口每一列上的尺寸大小，即窗口的行数	RW
15:14	Reserved	保留	-	-
13:0	RSIZE	行尺寸数	表示窗口每一行上的尺寸大小 (时钟像素数)，即窗口的列数  说明 窗口大小列数需设定为 4 的倍数，Bit1~Bit0 设定无效，固定为 0。	RW

42 加密协处理模块 (CPM)

42.1 简介

加密协处理模块 (CPM) 包括 HASH 安全散列算法, TRNG 真随机数发生器, PKE 公钥引擎, SKE 对称加密算法模块四个子模块。

- HASH 安全散列算法是 SHA-2 版本的 SHA-256 (Secure Hash Algorithm), 符合美国国家标准和技术局发布的国家标准“FIPS PUB 180-3”, 可以对长度不超过 2^{64} 位的消息产生 256 位的消息摘要输出。支持 HMAC (密钥散列消息认证码) 应用, 使用 SHA-256 散列函数对消息进行认证。
- TRNG 真随机数发生器是以连续模拟噪声为基础的随机数发生器, 提供 64-bit 随机数。
- PKE 公钥引擎实现了非对称签名/验签或加解密算法的硬件加速。
- SKE 对称加密算法包含 AES、DES、SM4 三种算法和 ECB、CBC、CTR、CFB、OFB、GCM、CCM、CMAC 共 8 种工作模式。

42.2 主要特性

42.2.1 公钥引擎 (PKE) 主要特性

- 支持模运算: 模加、模减、模乘、模幂、模逆
- 支持椭圆曲线点运算: 点加、点倍、点乘、验证点是否在曲线上
- 支持大数运算: 大数乘法
- 支持中断、轮询两种方式查询状态

42.2.2 对称加密算法模块 (SKE) 主要特性

对称加密算法模块 (SKE) 支持 AES-128、AES-192、AES-256、DES 和 SM4 共 5 种算法, 其中: AES 和 SM4 支持 ECB、CBC、CTR、CFB、OFB、GCM、CCM 和 CMAC 共 8 种工作模式; DES 支持除 GCM 和 CCM 模式以外的其他 6 种工作模式。

42.3 功能说明

42.3.1 安全散列算法 (HASH)

42.3.1.1 算法简介

安全散列算法的步骤如下：

首先填充消息使其长度恰好为一个比 512 的倍数仅小 64 位的数。填充方法是附一个 1 在消息后面，后接所要求的多个 0，然后在其后附上 64 位的消息长度（填充前），使消息长度恰好是 512 位的整数倍。

其次将 A、B、C、D、E、F、G、H 8 个 32 位变量，用十六进制初始化。然后开始算法的主循环，一次处理 512 位消息，循环次数是消息中 512 位分组的数目。

主循环一共进行 64 次操作，此操作称为压缩函数。每次操作都包含移位、循环移位、逻辑运算、模 232 加等，运算的过程如下图所示，最后的输出由 A、B、C、D、E、F、G、H 级联而成。其中 W_t 为由 512 位消息得到的第 t 步所用的临时值， K_t 为第 t 步所用的常数值， t ($0 \leq t \leq 63$) 是 64 步循环中的一步。

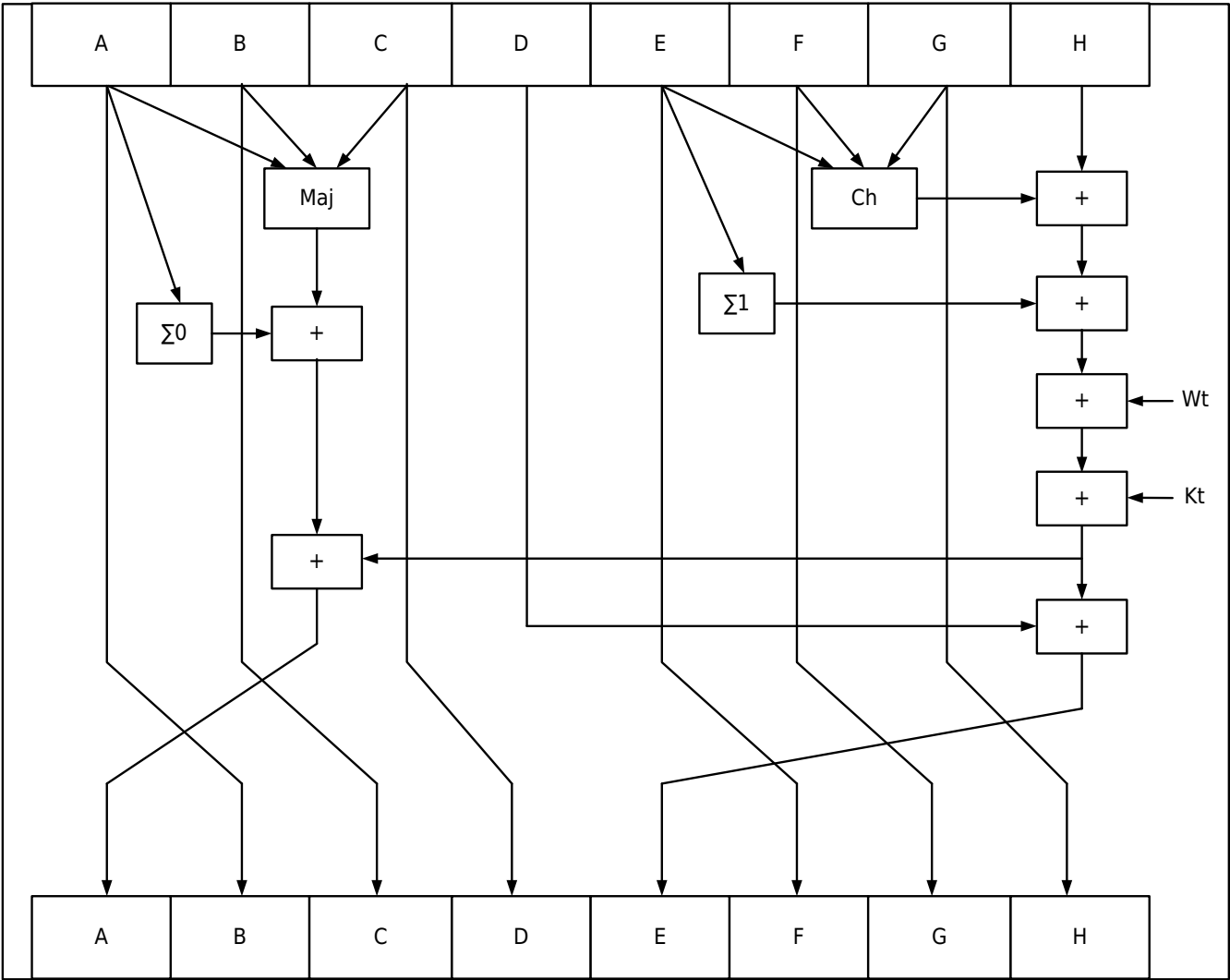


图 42-1 HASH 算法流程图

HASH 模块集成了硬件 HMAC 功能；也可以通过软件方式根据 HMAC 算法直接调用 HASH 运算实现 HMAC 功能。

42.3.1.2 操作流程

HASH 模块的操作流程如下：

- 软件操作流程：

- 步骤 1.** 软件将原始数据按照算法规则进行填充，并且将填充后的消息按 512 位进行分组。
- 步骤 2.** 将数据写入数据寄存器 HASH_DR 中。
- 步骤 3.** 如果本次运算为消息分组的第一组数据，设置 HASH_CR.FST_GRP=1。
- 步骤 4.** 设置 HASH_CR.START=1，启动本模块进行运算。



说明

步骤 3 和 4 可以同时进行。

- 步骤 5.** 判断本模块本次运算是否完成，通过以下方法：
不断读取 HASH_CR.START 直到读到该位为 0 即表示运算完成；或者不断读取 HASH_CR.BUSY 直到读到该位为 0 即表示运算完成；又或者设置 HASH 中断，通过中断方式判断运算结束。
- 步骤 6.** 如果本次运算不是消息分组的最后一组数据，返回到步骤 2。
- 步骤 7.** 如果本次运算为消息分组的最后一组数据，读取摘要寄存器 HASH_HR 获得本次运算的结果。如需再次进行运算则返回到步骤 1。

● 使用 DMA 操作流程：

- 步骤 1.** 将需要处理的消息按照 512 比特进行分组，最后一组消息需要按照规则预先填充为 512 比特的块。
- 步骤 2.** 设置内部触发事件寄存器的 HASH 运算启动方式为 DMA 传送完成和数据块传送完成；设置内部触发事件寄存器的 DMA 传送开始方式为 HASH 运算完成（或者其他内部触发事件）。
- 步骤 3.** 设置 HASH_CR.MODE=0b00，HASH 运算方式；设置 HASH_CR.FST_GRP=1，并设置 HASH_CR.START=1，启动 HASH 运算。如果选择其他内部触发事件启动 DMA，则需要等待触发事件发生后再开始 HASH 运算。
- 步骤 4.** 等待 HASH_CR.CYC_END=1，HASH 运算完成，并清除该位。
- 步骤 5.** 读取摘要寄存器 HASH_HR 的结果。



说明

当只有一组运算数据时，请同时设置 HASH_CR.FST_GRP=1 和 HASH_CR.KMSG_END=1。

42.3.1.3 消息填充

SHA-256 的填充分组处理步骤如下：

步骤 1. 原始消息分组

将原始消息以 512-bit 的大小分为 L 组。令原始消息总的 bit 个数为 I。如果 $I \% 512 < 448$ ，那么分组数 L 为 $I / 512$ ；如果 $I \% 512 \geq 448$ ，那么分组数 L 为 $I / 512 + 1$ 。

步骤 2. 添加长度

a. 添加填充位：

在消息分组的第 $I / 512$ 组的最后添加填充位：一个 1 和若干个 0，0 的个数可以为零。如果 $I \% 512 < 448$ ，填充使得数据位的长度满足长度为 $448 \bmod 512$ （后 64-bit 留做原始消息长度的表示）；如果 $I \% 512 \geq 448$ ，用一个 1 和若干个 0 把第 $I / 512$ 组的 512-bit 的数据块填满，并将第 L ($L = I / 512 + 1$) 组的前 448 位填 0。

b. 添加原始消息长度：

一个 64-bit 块，表示原始消息长度，为 64-bit 无符号整数。将原始消息长度添加在第 L 个分组的最后 64-bit 处。

举例说明填充分组的过程如下：

● **示例 1:**

原始消息为字符串“abcde”，用二进制位串表示其 ASCII 码为：“01100001 01100010 01100011 01100100 01100101”，添加长度的步骤如下：

1. 添加"1"。填充后的消息为"01100001 01100010 01100011 01100100 01100101 1"。
2. 添加"0"。因为原始消息长度为 40-bit，所以要添加的 0 的个数为 $512 - 64 - 40 - 1 = 407$ 。

填充后的消息变为 (十六进制)：

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000
```

3. 添加原始消息长度。原始消息长度 40-bit，用两个 32-bit 字表达为 (十六进制)：00000000 00000028。

填充后的消息变为 (十六进制)：

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028
```

● 示例 2:

原始消息为字符串“abcbcdcedefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq”。每个字符都可通过其 ASCII 码转换为 8-bit，所以消息的长度为 $l = 56 * 8 = 448$ 。

1. 添加"1"和"0"。填充后的消息 (十六进制) 为第一个消息块：

```
61626364 62636465 63646566 64656667
65666768 66676869 6768696A 68696A6B
696A6B6C 6A6B6C6D 6B6C6D6E 6C6D6E6F
6D6E6F70 6E6F7071 80000000 00000000
```

2. 添加原始消息长度。原始消息长度 448-bit，用两个 32-bit 字表达为 (十六进制)：00000000 000001C0。

填充后的消息 (十六进制) 为第二个消息块：

```
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 000001C0
```

42.3.1.4 HMAC 运算

HMAC (Keyed-Hashing for Message Authentication) 以不可逆方式将正在处理的消息与用户所选的密钥进行绑定，从而用于消息验证。基本而言，HMAC 是通过加入共享密钥的方式由两个嵌套的底层 HASH 运算组成：

$$\text{HMAC}(\text{message}) = \text{Hash}(((\text{key XOR opad}) \mid \text{Hash}((\text{key XOR ipad}) \mid \text{message})))$$

其中：

- ipad: 64 字节(SHA-256)0x36。
- opad: 64 字节(SHA-256)0x5C。
- |: 表示连接运算符。

42.3.1.4.1 HMAC 运算过程

步骤 1. 设置 HASH_CR.MODE=0b01，选择 HMAC 工作模式；

- 步骤 2. 如果使用的密钥长度超过 64 字节, 则需要设置 HASH_CR.LKEY=1 (在此情况下, HMAC 将使用规范规定的密钥替代给定密钥) ;
- 步骤 3. 将密钥写入 HASH_DR 寄存器, 直到所有的密钥分组运算完成;
- 步骤 4. 将消息写入 HASH_DR 寄存器, 直到所有的消息分组运算完成;
- 步骤 5. 读取 HASH_HR 寄存器的最终散列运算结果。

HMAC 运算过程中的数据加载, 可以选择软件操作流程和 DMA 操作流程的方式, 具体举例如下:

● HMAC 软件操作流程

- 步骤 1. 设置模式 HASH_CR.MODE=0b01, HMAC 工作模式; 设置 HASH_CR.LKEY=1, 选择长密钥方式;
- 步骤 2. HASH_DR 中写入密钥分组数据;
- 步骤 3. 如果为第一组数据, 则设置 HASH_CR.FST_GRP=1。设置 HASH_CR.START=1 启动密钥运算;
- 步骤 4. 等待 HASH_CR.BUSY=0 运算完成;
- 步骤 5. 如果不是最后一组数据, 则重复步骤 2~4;
- 步骤 6. 如果是最后一组数据, 则设置 HASH_CR.KMSG_END=1, 并设置 HASH_CR.START=1 启动最后一组密钥运算。等待 HASH_CR.CYC_END=1 后, 密钥运算完成, 并将该位清除。
- 步骤 7. HASH_DR 中写入消息分组数据;
- 步骤 8. 如果为第一组消息数据, 则设置 HASH_CR.FST_GRP=1。设置 HASH_CR.START=1 启动消息运算;
- 步骤 9. 等待 HASH_CR.BUSY=0 运算完成;
- 步骤 10. 如果不是最后一组消息数据, 则重复步骤 7~9;
- 步骤 11. 如果是最后一组消息, 则设置 HASH_CR.KMSG_END=1, 并设置 HASH_CR.START=1 启动最后一组消息运算。等待 HASH_CR.CYC_END=1 后, 所有 HMAC 运算完成, 并将该位清除;
- 步骤 12. 读取 HASH_HR 中的最终消息认证结果。

● HMAC 使用 DMA 操作流程

- 步骤 1. 设置 DMA 为数据块传送方式, 数据块的大小为 16 个字, 不管是密钥还是消息都需要按照 16 个字的大小分组, 未满 16 个字时, 请按照 HASH 消息填充规则进行填充, 配置 DMA 数据块源地址为密钥地址;
- 步骤 2. 设置内部触发事件寄存器的 HASH 运算启动方式为 DMA 传送完成和数据块传送完成; 设置内部触发事件寄存器的 DMA 传送开始方式为 HASH 运算完成 (或者其他内部触发事件) ;
- 步骤 3. 设置 HASH_CR.MODE=0b01, HMAC 运算模式; 根据密钥长度配置 HASH_CR.LKEY 位; 设置 HASH_CR.FST_GRP=1, HASH_CR.START=1 启动密钥运算; 如果选择其他内部触发事件, 则需要等待触发事件后开始启动运算;
- 步骤 4. 等待 HASH_CR.CYC_END=1 密钥运算完成后, 将该位清除;
- 步骤 5. 停止 DMA, 将 DMA 数据块源地址指向消息地址;
- 步骤 6. 设置 HASH_CR.FST_GRP=1, HASH_CR.START=1 启动消息运算; 或等待其他触发事件后启动消息运算。
- 步骤 7. 等待 HASH_CR.CYC_END=1 消息运算完成后, 将该位清除;
- 步骤 8. 读取 HASH_HR 寄存器中最终的消息认证结果。

42.3.1.4.2 HMAC 运算举例

密钥和消息三组 HMAC 运算结果如下:

[illegible]

42.3.1.4.3 注意事项

1. 当填充后的密钥和消息只有一组数据，即 512 位时，启动运算前请同时设置 `HASH_CR.FST_GRT=1` 和 `HASH_CR.KMSG_END=1`；
2. 关于密钥填充，当密钥小于 64 字节时，未满 64 字节部分请填写 0x00；当密钥大于 64 字节时，未满 64 字节部分请按照 HASH 填充规则进行填充。

42.3.1.5 中断和事件

42.3.1.5.1 中断

HASH 中断标记 HASH_INT。不管是在 HASH 运算模式还是在 HMAC 运算模式下，在设置 HASH_CR.HEIE=1 中断许可条件下，每组数据运算完成后将产生一次中断请求。HASH_CR.HCIE=1 中断许可条件下，所有的密钥或消息运算完成后将产生一次中断请求。HEIE 和 HCIE 同时许可时，任何一个条件满足都将产生中断请

求。另外，HEIE 使能后，在模块复位后或者运算完成后，直接设置 HASH_CR.FST_GRP 且 HASH_CR.START=1 时，也将产生一次中断用于数据加载。

42.3.1.5.2 硬件触发事件选择

通过配置内部硬件触发事件选择寄存器 HASH_TRGSELA，HASH_TRGSELB 在数据传送完成后启动 HASH 运算，通常与 DMA 配合使用。例如，当选择 DMA_2 传送数据和启动 HASH 运算时，首先，配置 DMA2_TRGSELx (x=0~7) 选择 HASH_INT 启动 DMA 传送；其次，配置 HASH_TRGSELA 选择 DMA_BTCx (x=0~7)，表示完成一个数据块 (512-bit) 传输并启动 HASH 运算，配置 HASH_TRGSELB 选择 DMA_TCx (x=0~7)，表示完成所有数据块传输。当启动 HASH 后，DMA 和 HASH 相互触发，直到所有数据块完成传送和运算。

42.3.2 真随机数发生器 (TRNG)

42.3.2.1 模块框图

TRNG 模块提供 1 个真随机数发生器，产生 1 个 64 位随机数。

TRNG 的系统框图如下图所示。其中随机数发生器是一个模拟随机数发振器电路，用于得到随机性的噪声；算法模块捕捉随机噪声并把结果保存至数据模块、并通过总线输出；控制模块控制 TRNG 的模式和启动。

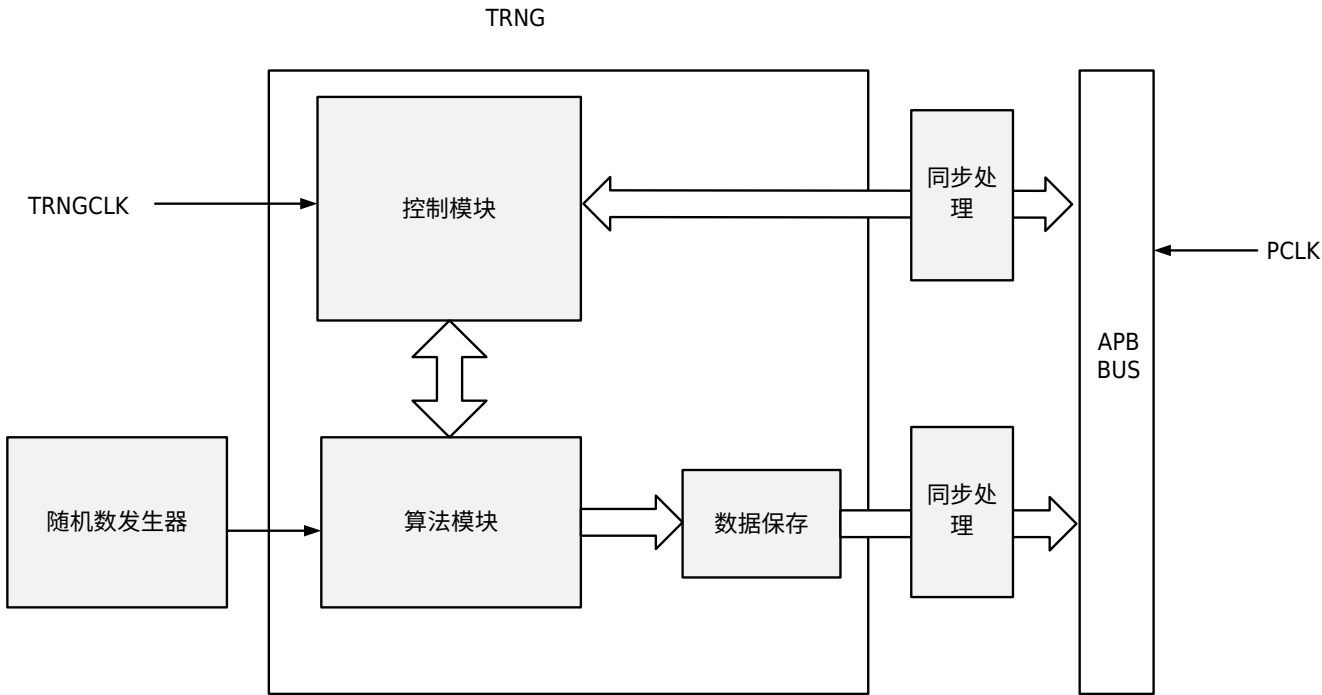


图 42-2 TRNG 系统框图

42.3.2.2 操作流程

真随机数生成流程如下：

- 步骤 1. 打开随机数发生器电路（设定 TRNG_CR 的 EN 位为 1）。
- 步骤 2. 配置随机数生成模式（设定 TRNG_MR）。
- 步骤 3. 启动随机数生成（设定 TRNG_CR 的 RUN 位为 1）。
- 步骤 4. 读取随机数（读取 TRNG_DR）。
- 步骤 5. 关闭随机数发生器电路（设定 TRNG_CR 的 EN 位为 0）。

42.3.2.3 中断和事件

当随机数生成结束后，寄存器位 TRNG_CR.RUN 硬件清零并产生随机数生成完中断请求（TRNG_END）。随机数生成结束同时也产生事件输出，可以触发其他模块联动。

42.3.3 公钥引擎 (PKE)

42.3.3.1 模块简介

本产品的 PKE (Public Key Engine) 系统由 1 个运算模块、2 个专用 RAM 存储器构成。配合软件驱动可以实现 RSA (RSA1024/RSA2048/RSA3072/RSA4096)、ECC (brainpoolp 160/brainpoolp 256/brainpoolp 512/SECP 192/SECP 224/SECP 256/SECP 384/SECP 521)、SM2。

表 42-2 运算速度

运算模式			速度 (次/秒)	模块工作频率
硬件运算最快速度	256 比特点乘		75	120MHz
	1024 比特模幂		32.4	
软件运算最快速度	ECC SECP256R1	ECDSA/ECDH 密钥生成	72	
		ECDH 密钥交换	75	
		ECDSA 签名	72	
		ECDSA 验签	36	
	SM2	SM2 密钥生成	72	
		SM2 密钥交换	45	
		SM2 签名	72	
		SM2 验签	36	
		SM2 加密	36	
		SM2 解密	69	
	ECC SECP384R1	ECDSA/ECDH 密钥生成	30	
		ECDH 密钥交换	30	
		ECDSA 签名	30	
		ECDSA 验签	14.4	
	ECC SECP521R1	ECDSA/ECDH 密钥生成	13.2	
		ECDH 密钥交换	13.8	
		ECDSA 签名	13.2	
		ECDSA 验签	6.6	
	RSA 1024	密钥生成 (e=65537)	3	
		CRT 密钥生成 (e=65537)	3	
		签名/解密	30	
		验签/加密	828	
		CRT-签名/解密	96	

运算模式			速度 (次/秒)	模块工作频率
	RSA 2048	密钥生成 (e=65537)	1.32	120MHz
		CRT 密钥生成 (e=65537)	1.26	
		签名/解密	4.2	
		验签/加密	240	
		CRT-签名/解密	15	
	RSA 4096	密钥生成 (e=65537)	16.2	
		CRT 密钥生成 (e=65537)	15.6	
		签名/解密	0.6	
		验签/加密	66	
		CRT-签名/解密	1.8	

42.3.3.2 模块框图

PKE 模块将 CPU 从复杂的公钥密码运算中解放了出来。CPU 只需要将输入参数配置好，PKE 就会根据目前的配置完成指定的操作。目前，PKE 可以支持直接完成 RSA 中的模幂运算和 ECC 中的点乘运算。CPU 可以通过轮询或者是中断的方式来查询 PKE 的工作情况。

PKE 由一块程序存储单元 (ROM)、命令运算单元 (IEU)、32 位的算法运算单元 (ALU)、寄存器组和接口模块构成。

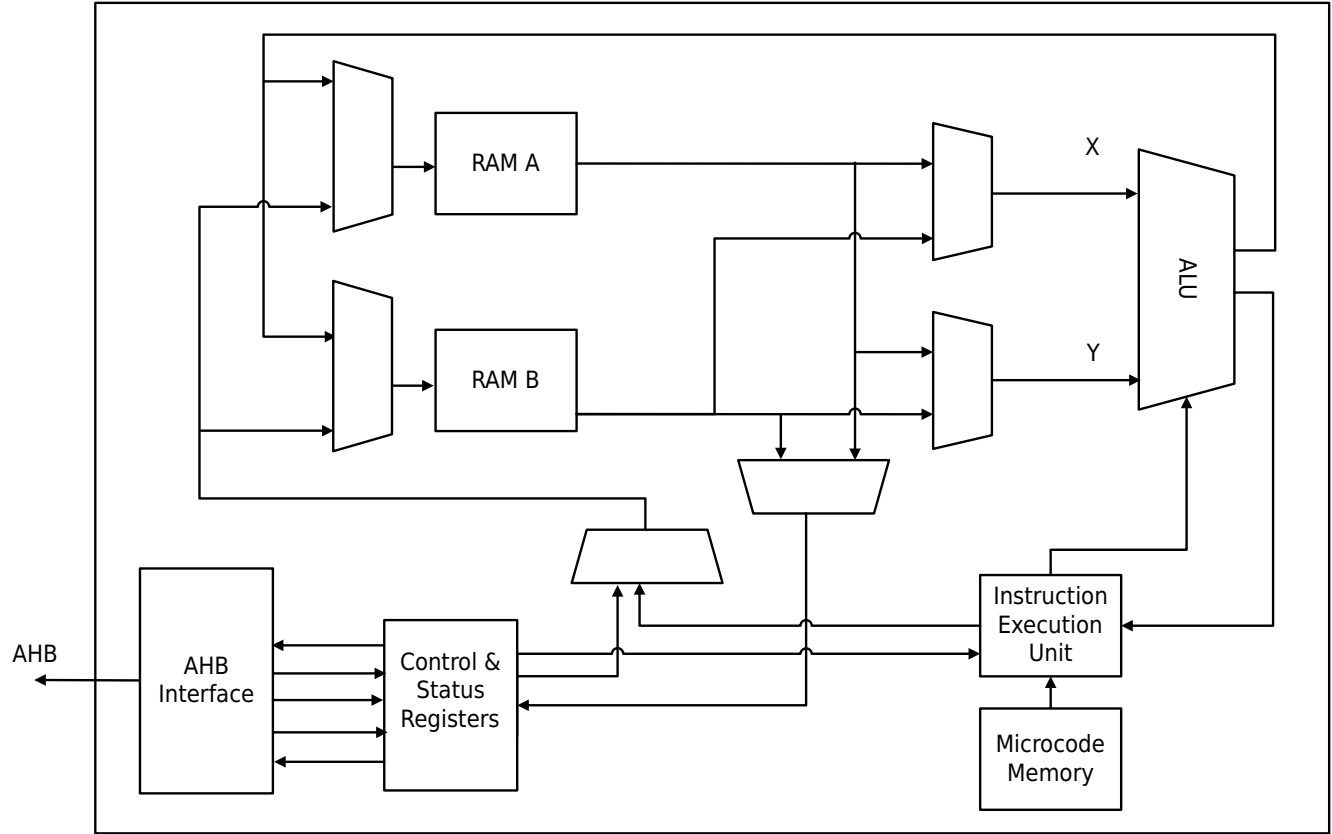


图 42-3 PKE 系统框图

42.3.3.3 编程模型

PKE 使用的外部存储器都被映射到了总线地址空间内。这一地址映射主要包含 CPU 可寻址的所有操作数以及运算结果。这些操作数包含了模数、幂指数、部分中间变量等。

除此之外，该地址映射也包含了 PKE 的控制和状态寄存器。CPU 可以通过这些控制、状态寄存器来配置、监控 PKE 模块。

在常见的公钥密码运算中，运算数最短为 192 比特，在将数据写入数据 RAM 中会遇到字间大小端的问题。在 PKE 模块中，字与字之间都是按照小端进行排列。在 PKE 中，最小的操作数位宽为 32 比特（1 个字），若操作数位宽并不是 32 比特的整数倍，那么外部输入操作数时需要提前将高位补零。

在触发 PKE 启动运算后，可以通过查询状态寄存器了解 PKE 的运行状态。可以通过轮询或是中断的方式来获取 PKE 是否完成目标运算的结果。

说明

数据 RAM 都是字对齐，不支持字节对齐。

表 42-3 RAM 操作数分布

RAM	操作数空间	ECC			RSA			
		256-bit	512-bit	1024-bit	512-bit	1024-bit	2048-bit	4096-bit
RAM0	A0	0x1400	0x1400	0x1400	0x1400	0x1400	0x1400	0x1400
	A1	0x1424	0x1444	0x1484	0x1444	0x1484	0x1504	0x1604
	A2	0x1448	0x1488	0x1508	0x1488	0x1508	0x1608	0x1808
	A3	0x146C	0x14CC	0x158C	0x14CC	0x158C	0x170C	0x1A0C
	A4	0x1490	0x1510	0x1610	0x1510	0x1610	0x1810	0x1C10
	A5	0x14B4	0x1554	0x1694	-	-	-	-
	A6	0x14D8	0x1598	0x1718	-	-	-	-
	A7	0x14FC	0x15DC	0x179C	-	-	-	-
	A8	0x1520	0x1620	0x1820	-	-	-	-
	A9	0x1544	0x1664	0x18A4	-	-	-	-
RAM1	B0	0x2000	0x2000	0x2000	0x2000	0x2000	0x2000	0x2000
	B1	0x2024	0x2044	0x2084	0x2044	0x2084	0x2104	0x2204
	B2	0x2048	0x2088	0x2108	0x2088	0x2108	0x2208	0x2408
	B3	0x206C	0x20CC	0x218C	0x20CC	0x218C	0x230C	0x260C
	B4	0x2090	0x2110	0x2210	0x2110	0x2210	0x2410	0x2810
	B5	0x20B4	0x2154	0x2294	-	-	-	-
	B6	0x20D8	0x2198	0x2318	-	-	-	-
	B7	0x20FC	0x21DC	0x239C	-	-	-	-
	B8	0x2120	0x2220	0x2420	-	-	-	-
	B9	0x2144	0x2264	0x24A4	-	-	-	-

上表给出了 ECC 和 RSA 模式下不同运算长度时两块 RAM 的地址分配情况。操作数寄存器分布在两块数据 RAM 中，使用前缀 A 和 B 来区别两块 RAM。表中列出的地址都是 CPU 可寻址的地址，RAMA 带有 0x1400 的地址偏移，RAMB 带有 0x2000 的地址偏移。RAM 实际使用的空间会比表中列出的空间更大，因为还有部分空间用于存储其他中间变量，而这些中间变量并不对 CPU 开放。

- 在 ECC 模式下，PKE 会用到两块 RAM 中 0~9 的寄存器空间，每个寄存器空间有（8+1）字、（16+1）字和（32+1）字三种选项，其中的 1 个字用来存储可能的进位。
- 在 RSA 模式下，PKE 会用到两块 RAM 中 0~4 的寄存器空间，每个寄存器空间有（16+1）字、（32+1）字、（64+1）字和（128+1）字三种选项，其中的 1 个字用来存储可能的进位。

42.3.3.4 编程入口

所有程序都基于下表所示的过程入口地址。

表 42-4 编程入口地址

过程名	过程入口地址	说明
PDBL	0x04	椭圆曲线点倍
PADD	0x08	椭圆曲线点加
PVER	0x0C	椭圆曲线验证是否在曲线上
PMUL	0x10	椭圆曲线点乘
MODEXP	0x14	模幂
MODMUL	0x18	模乘
MODINV	0x1C	模逆
MODADD	0x20	模加
MODSUB	0x24	模减
CAL_PRE_MON	0x28	蒙哥马利预计算
IMUL	0x2C	大数乘法

42.3.4 对称加密算法模块（SKE）

42.3.4.1 算法简介

对称加密算法模块（SKE）内含对称加密算法引擎，支持如下功能：

- AES 加解密算法
- DES 加解密算法
- SM4 加解密算法
- 支持 ECB、CBC、CTR、CFB、OFB、GCM、CCM 和 CMAC 共 8 种工作模式，并支持 GCM、CCM 和 CMAC 模式下的数据交织
- 支持加解密完成中断

AES 加解密算法遵循《FIPS PUB 197》对算法原理的描述，分组长度固定为 128 位，密钥长度支持 128/192/256 位。

DES 加解密算法遵循《FIPS PUB 46-3》对算法原理的描述，分组长度固定为 64 位，密钥长度 64 位。

SM4 加解密算法遵循《GMT0002-2012》对算法原理的描述，分组长度固定为 128 位，密钥长度 128 位。

42.3.4.2 模块框图

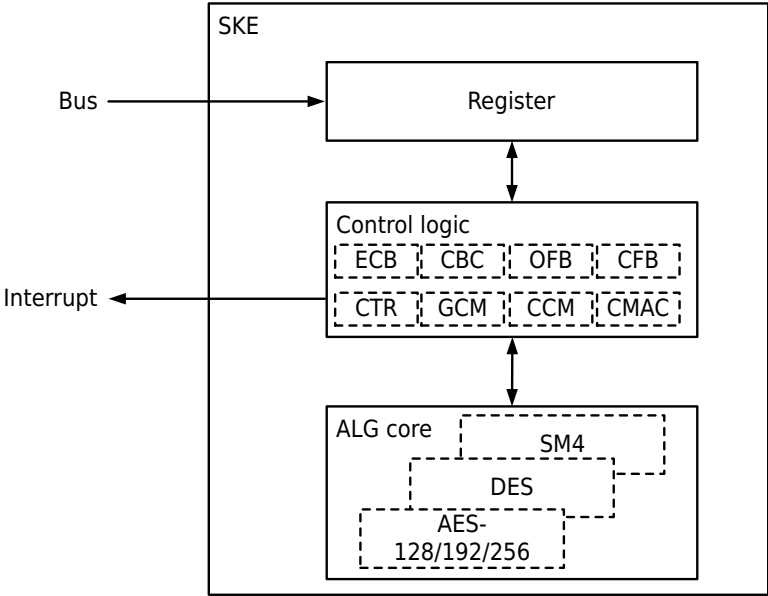


图 42-4 SKE 系统框图

42.3.4.3 工作模式说明

- ECB：由 NIST SP800-38A 所定义的电码本模式（Electronic Code Book）
- CBC：由 NIST SP800-38A 所定义的密码分组链接模式（Cipher Block Chaining）
- CTR：由 NIST SP800-38A 所定义的计数器模式（Counter），特指 CTR-32，参考 RFC 3686 标准
- CFB：由 NIST SP800-38A 所定义的密码反馈模式（Cipher Feedback），特指 CFB-128
- OFB：由 NIST SP800-38A 所定义的输出反馈模式（Output Feedback）
- GCM：由 NIST SP800-38D 所定义的伽罗瓦计数器模式（Galois/Counter Mode）
- CCM：由 NIST SP800-38C 所定义的认证加密模式（Counter with CBC-MAC）
- CMAC：由 NIST SP800-38B 所定义的用于认证的 CMAC 模式（Cipher-based Message Authentication Code）

42.3.4.4 运算周期

下面是每种算法下常用模式的运算周期：

表 42-5 AES 硬件运算周期个数（分组数量:128-bit）

密钥长度	ECB	CBC	CTR	CFB	OFB	CMAC	GCM				CCM			
							配置	AAD	加解密	TAG	配置	AAD	加解密	TAG
128 位	13	14	14	14	14	18	1	10	18	19	1	18	18	2
192 位	15	16	16	16	16	20	1	10	20	21	1	20	20	2
256 位	17	18	18	18	18	22	1	10	22	23	1	22	22	2

表 42-6 SM4 硬件运算周期个数（分组数量:128-bit）

密钥长度	ECB	CBC	CTR	CFB	OFB	GCM				CCM			
						配置	AAD	加解密	TAG	配置	AAD	加解密	TAG
128 位	36	37	37	37	37	1	10	41	42	1	41	41	2

表 42-7 DES 硬件运算周期个数（分组数量:64-bit）

密钥长度	ECB	CBC	CTR	CFB	OFB
64 位	35	36	36	36	36

42.3.4.5 操作流程

SKE 的加解密操作流程例如下：

1. 上电复位后，SKE 空闲，等待调用。
2. 判断是否需要更新 SKE 中的配置，若是则进行配置更新流程，否则进行数据加/解密流程；在不更新配置的情况下，SEK 会采用上一次保存的配置进行运算。即使没有切换密钥（KEY）、初始向量（IV，Initial Value）等配置，在切换数据包时也需要更新配置。
 - 配置更新时：
 - a. 对 SKE_AADx/SKE_CLENx 写入 AAD 长度和数据长度（仅 GCM/CCM 模式适用）。
 - b. 先对 SKE_CFG 写入所需要的配置，选择加密或者解密，SKE_CFG.UP_CFG 需保证为 1。
 - c. 通过 SKE_KEYx 写入密钥。并不是每次都需要输入密钥，若之前已经输入过密钥了，则无需重复输入相同密钥，SKE 会采用之前输入的密钥进行运算。
 - d. 通过 SKE_IVx 写入初始向量，对于 CCM、GCM、CMAC 模式下的分段交织，如果处于中间段或者末尾段，则需要输入 SKE_MID_MACx、SKE_MID_IVx 的中间值。
 - e. 对 SKE_CTRL 写 1，触发 SKE 进行运算。
 - f. 轮询 SKE_SR2 对应的标志位，直到该标志位为 1 为止。
 - g. 当检测到 SKE_SR2 中标志位为 1 后，对 SKE_SR2 进行写操作，清除。
 - h. 对 SKE_CFG.UP_CFG 写 0；至此，配置更新流程结束。
 - 数据加解密时：
 - a. （可选）当 AAD 输入时（仅 GCM/CCM 模式适用），需要执行以下步骤：
 - 1) 确保 SKE_DIN_CR.LAST=0b0，通过写寄存器 SKE_DINx，输入第一组 AAD。
 - 2) 对 SKE_CTRL 写 1，触发 SKE 进行运算。
 - 3) 轮询 SKE_SR2 对应的标志位，直到该标志位为 1 为止。
 - 4) 当检测到 SKE_SR2 中标志位为 1 后，对 SKE_SR2 进行写操作，清除。
 - 5) 重复步骤 1)~4)，输入最后一组 AAD 时，写入 SKE_DIN_CR.LAST=0b1 后再重复步骤 1)~4)。
 - b. 确保 SKE_DIN_CR.LAST=0b0。
 - c. 通过写寄存器 SKE_DINx，输入下一组需要加密/解密的数据。
 - d. 对 SKE_CTRL.START 写 1，触发 SKE 进行运算。
 - e. 轮询 SKE_SR2.DONE 标志位，直到该标志位为 1 为止。
 - f. 当检测到 SKE_SR2.DONE 为 1 后，清零该标志位。
 - g. 重复步骤 c~f。注意，若是输入最后一组数据且是 GCM/CCM/CMAC 模式时，需在输入数据之前设置 SKE_DIN_CR.LAST=0b1。
 - h. 最后一组数据加密/解密完成后，确保 SKE_DIN_CR.LAST=0b0。
 - i. 读取 SKE_DOUT，获得运算结果；如果是分段交织的初始段或者中间段，还需要读取 SKE_MID_MACx、SKE_MID_IVx；如果需要生成 TAG，则再重复一次步骤 b~e 并读取 SKE_DOUTx。

至此，SKE 完成了一次完整调用，可循环以上步骤，进行多次调用。

42.3.4.6 寄存器地址映射

SKE 的 SKE_KEYx、SKE_IVx、SKE_DINx、SKE_DOUTx、SKE_MID_IVx 和 SKE_MID_MACx 寄存器均采用小端存储，以 SKE_KEYx 寄存器举例如下，其他寄存器类似：

- DES 64 比特长度密钥：0x00，0x01，0x02，0x03，0x04，0x05，0x06，0x07

寄存器	写值
SKE_KEY1	0x03020100

寄存器	写值
SKE_KEY2	0x07060504

- AES 128 比特长度密钥: 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F

寄存器	写值
SKE_KEY1	0x03020100
SKE_KEY2	0x07060504
SKE_KEY3	0x0B0A0908
SKE_KEY4	0x0F0E0D0C

- AES 192 比特长度密钥: 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17

寄存器	写值
SKE_KEY1	0x03020100
SKE_KEY2	0x07060504
SKE_KEY3	0x0B0A0908
SKE_KEY4	0x0F0E0D0C
SKE_KEY5	0x13121110
SKE_KEY6	0x17161614

- AES 256 比特长度密钥: 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, 0x1B, 0x1C, 0x1D, 0x1E, 0x1F

寄存器	写值
SKE_KEY1	0x03020100
SKE_KEY2	0x07060504
SKE_KEY3	0x0B0A0908
SKE_KEY4	0x0F0E0D0C
SKE_KEY5	0x13121110
SKE_KEY6	0x17161614
SKE_KEY7	0x1B1A1918
SKE_KEY8	0x1F1E1D1C

42.3.4.7 数据示例

2 个分组数据 256 位明文:

0x6BC1BEE2 2E409F96 E93D7E11 7393172A AE2D8A57 1E03AC9C 9EB76FAC 45AF8E51

128 位密钥:

0x2B7E1516 28AED2A6 ABF71588 09CF4F3C

256 位密文:

0x3AD77BB4 0D7A3660 A89ECAF3 2466EF97 F5D3D585 03B9699D E785895A 96FDBAAF

表 42-12 128 位 AES ECB 模式操作寄存器示例

加密前			
寄存器	写值 (密钥)	寄存器	写值 (明文)
SKE_KEY1	0x16157E2B	SKE_DIN1	0xE2BEC16B
SKE_KEY2	0xA6D2AE28	SKE_DIN2	0x969F402E
SKE_KEY3	0x8815F7AB	SKE_DIN3	0x117E3DE9
SKE_KEY4	0x3C4FCF09	SKE_DIN4	0x2A179373
第一轮加密后			
寄存器	读值 (密文)	寄存器	写值 (明文)
SKE_DOUT1	0xB47BD73A	SKE_DIN1	0x578A2DAE
SKE_DOUT2	0x60367A0D	SKE_DIN2	0x9CAC031E
SKE_DOUT3	0xF3CA9EA8	SKE_DIN3	0xAC6FB79E
SKE_DOUT4	0x97EF6624	SKE_DIN4	0x518EAF45
第二轮加密后			
寄存器	读值 (密文)	寄存器	写值 (明文)
SKE_DOUT1	0x85D5D3F5	-	-
SKE_DOUT2	0x9D69B903	-	-
SKE_DOUT3	0x5A8985E7	-	-
SKE_DOUT4	0xAFBAFD96	-	-

42.3.4.8 中断

SKE 含有 1 个中断, 当 SKE 的加解密操作完成, SKE_SR2.SKE_DONE 标志位被置为 1, 如果此时 SKE_CFG.IRQEN=1, 则产生中断信号。SKE_SR2.SKE_DONE 标志位可以被软件清零。

42.4 寄存器

42.4.1 寄存器总表

HASH 基地址：0x40008400

表 42-13 安全散列算法（HASH）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	HASH_CR	HASH 控制寄存器	32
0x0010+0x4*(7-n)	HASH_HRn	HASH 摘要寄存器 n（n=0~7）	32
0x0040+0x4*(15-n)	HASH_DRn	HASH 数据寄存器 n（n=0~15）	32

TRNG 基地址：0x40042000

表 42-14 真随机数发生器（TRNG）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	TRNG_CR	TRNG 控制寄存器	32
0x0004	TRNG_MR	TRNG 模式寄存器	32
0x000C+0x4*n	TRNG_DRn	TRNG 数据寄存器 n（n=0~1）	32

PKE 基地址：0x4000B000

表 42-15 公钥引擎（PKE）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x1000	PKE_CTRL	PKE 控制寄存器	32
0x1004	PKE_CFG	PKE 配置寄存器	32
0x1008	PKE_MC_PTR	PKE 过程入口寄存器	32
0x100C	PKE_RISR	PKE 中断源寄存器	32
0x1010	PKE_IMCR	PKE 中断使能寄存器	32
0x1014	PKE_MISR	PKE 中断输出寄存器	32
0x1024	PKE_RT_CODE	PKE 返回日志寄存器	32
0x1050	PKE_EXE_CFG	PKE 运算配置寄存器	32
0x10FC	VERSION	PKE 版本寄存器	32
0x1400~0x1E10	PKE_MEM_RAM0	PKE 运算操作数 RAM0	32
0x2000~0x2A10	PKE_MEM_RAM1	PKE 运算操作数 RAM1	32
0x0000	PKE_RAM0_CKCR	PKE RAM0 校验控制寄存器	32
0x0004	PKE_RAM0_CKSR	PKE RAM0 校验状态寄存器	32

偏移地址	寄存器	描述	位宽
0x0008	PKE_RAM0_EIEN	PKE RAM0 错误注入使能寄存器	32
0x000C	PKE_RAM0_EIBIT0	PKE RAM0 错误注入位使能寄存器 0	32
0x0010	PKE_RAM0_EIBIT1	PKE RAM0 错误注入位使能寄存器 1	32
0x0014	PKE_RAM0_ECCERRADDR	PKE RAM0 错误地址寄存器	32
0x0020	PKE_RAM1_CKCR	PKE RAM1 校验控制寄存器	32
0x0024	PKE_RAM1_CKSR	PKE RAM1 校验状态寄存器	32
0x0028	PKE_RAM1_EIEN	PKE RAM1 错误注入使能寄存器	32
0x002C	PKE_RAM1_EIBIT0	PKE RAM1 错误注入位使能寄存器 0	32
0x0030	PKE_RAM1_EIBIT1	PKE RAM1 错误注入位使能寄存器 1	32
0x0034	PKE_RAM1_ECCERRADDR	PKE RAM1 错误地址寄存器	32

SKE 基地址：0x4000E000

表 42-16 对称加密算法模块 (SKE) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	SKE_CTRL	SKE 控制寄存器	32
0x0004	SKE_CFG	SKE 配置寄存器	32
0x0008	SKE_SR1	SKE 状态寄存器 1	32
0x000C	SKE_SR2	SKE 状态寄存器 2	32
0x0010+0x4*(x-1)	SKE_KEYx	SKE 密钥寄存器 x (x=1~8)	32
0x0060+0x4*(x-1)	SKE_AADx	SKE AAD 长度寄存器 x (x=1~2)	32
0x0068+0x4*(x-1)	SKE_CLENx	SKE 数据长度寄存器 x (x=1~2)	32
0x0070+0x4*(x-1)	SKE_IVx	SKE 初始向量寄存器 x (x=1~4)	32
0x0080	SKE_DIN_CR	SKE 数据标志寄存器	32
0x0090+0x4*(x-1)	SKE_DINx	SKE 数据输入寄存器 x (x=1~4)	32
0x00B0+0x4*(x-1)	SKE_DOUTx	SKE 数据输出寄存器 x (x=1~4)	32
0x00FC	SKE_VERSION	SKE 版本寄存器	32
0x01A0+0x4*(x-1)	SKE_MID_IVx	SKE 中间向量寄存器 x (x=1~4)	32
0x01B0+0x4*(x-1)	SKE_MID_MACx	SKE 中间 MAC 寄存器 x (x=1~4)	32
0x0060+0x4*(x-1)	SKE_TOTAL_AADx	SKE AAD 总长度寄存器 x (x=1~2)	32
0x0068+0x4*(x-1)	SKE_TOTAL_CLENx	SKE 数据总长度寄存器 x (x=1~2)	32

42.4.2 HASH 寄存器

42.4.2.1 HASH 控制寄存器 (HASH_CR)

Offset	Bit Position																																		
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	Reserved																DATATYPE	HEIE	HCIE	Reserved				HMAC_END	CYC_END	BUSY	Reserved	LKEY	MODE		Reserved	KMSG_END	FST_GRP	START	
Access																	RW	RW	RW					RW	RW	RO		RW	RW			RW	KMSG_END	FST_GRP	START

位/位域	标记	位名	功能描述	读写
31:18	Reserved	保留	-	-
17:16	DATATYPE	输入数据顺序	0b00: 字内按字节反序 0b01: 字内按半字反序 0b10: 字内按位反序 0b11: 保持原数据顺序	RW
15	HEIE	每组数据运算完成中断许可	0b0: 每组运算完成中断禁止 0b1: 每组运算完成中断许可	RW
14	HCIE	全部密钥或消息运算完成中断许可	0b0: 全部密钥或消息运算完成中断禁止 0b1: 全部密钥或消息运算完成中断许可	RW
13:11	Reserved	保留	-	-
10	HMAC_END	HMAC 运算完成标志	0b0: HMAC 未运算完成 0b1: HMAC 运算完成  说明 当标志位为“1”后，写“0”清除。注意，软件在确认标志位后请及时清“0”，否则标志位将保持为“1”。	RW
9	CYC_END	密钥或消息运算完成标志	0b0: 密钥或消息运算未完成 0b1: 密钥或消息运算完成  说明 当标志位为“1”后，写“0”清除。注意，软件在确认标志位后请及时清“0”，否则标志位将保持为“1”。	RW
8	BUSY	转换标志	0b0: HASH 运算处于空闲状态 0b1: HASH 运算处理中	RO
7	Reserved	保留	-	-
6	LKEY	长密钥选择	0b0: 短密钥 (≤64 字节) 0b1: 长密钥 (>64 字节) 在 HMAC 模式中，该位在短密钥 (≤64 字节) 或长密钥 (>64 字节) 之间进行选择。在 MODE=0b01，HMAC 模式时有效。	RW

位/位域	标记	位名	功能描述	读写
5:4	MODE	工作模式	● 0b00: SHA-256 工作模式 ● 0b01: HMAC 工作模式 ● 0b1X: 设定禁止	RW
3	Reserved	保留	-	-
2	KMSG_END	密钥或消息分组的最后一组	● 0b0: 密钥或消息非最后一组 ● 0b1: 密钥或消息最后一组 当最后一组密钥或消息转换完成后, 该位自动清“0”	RW
1	FST_GRP	密钥或消息分组的最后一组	● 0b0: 密钥或消息非第一组 ● 0b1: 密钥或消息第一组 当第一组密钥或消息转换完成后, 该位自动清“0”	RW
0	START	启动	0b0: 本模块运算结束或未被启动 0b1: 启动本模块进行运算	RW

说明

- START 位的操作方法是：软件对该位写入 1 后，本模块将启动运行；本次运行结束后硬件会自动将该位清 0；软件查询到该位为 0 即表示本次运行完成。
- 对该寄存器的写入操作只能在本模块非处于运算状态时（即 START 位为 0 时）才能进行，否则硬件将自动忽略写操作。读操作则不受此限制。
- 若要改变输入数据顺序，需在输入数据前配置 DATATYPE。

42.4.2.2 HASH 摘要寄存器 n（HASH_HRn）（n=0~7）

8 个 HASH 摘要寄存器 HASH_HR0~HASH_HR7，对应 256 位 HASH 摘要。HASH_HR0 对应摘要 Bit31~Bit0，HASH_HR1 对应摘要 Bit63~Bit32，依次类推。

偏移地址（Offset）：0x0010+0x4*(7-n)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	HASH																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	HASH	摘要值	在模块运算完成后通过读取本寄存器得到消息摘要	RO

说明

- 本寄存器由 8 个 32 位寄存器拼接而成。访问时依次对 8 个 32 位寄存器进行操作，低地址对应的 32 位寄存器存放消息摘要的高字。
- 对于本寄存器的读取只能在本模块非处于运算状态时（HASH_CR.START=0）才能进行，否则对本寄存器的读取将得到全 0。

42.4.2.3 HASH 数据寄存器 n (HASH_DRn) (n=0~15)

16 个 HASH 数据寄存器 HASH_DR0~HASH_DR15, 对应 512 位 HASH 数据。HASH_DR0 对应数据 Bit31~Bit0, HASH_DR1 对应数据 Bit63~Bit32, 依次类推。

偏移地址 (Offset) : 0x0040+0x4*(15-n)

Offset	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DATA																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	数据寄存器	用于在模块运算前写入消息	RW

 **说明**

- 本寄存器由 16 个 32 位寄存器拼接而成。访问时依次对 16 个 32 位寄存器进行操作, 低地址对应的 32 位寄存器存放数据的高字。
- 对于本寄存器的写入只能在本模块非处于运算状态时 (HASH_CR.START 为 0) 才能进行, 否则硬件将自动忽略对本寄存器的写操作。
- 在本模块处于运算状态时 (HASH_CR.START 为 1), 对本寄存器的读取将总是得到全 0。在非运算状态时 (HASH_CR.START 为 0), 对本寄存器读取的数据 (该数据为计算过程的中间结果) 按 HASH_CR.DATATYPE 为 0 的格式排列。

42.4.3 TRNG 寄存器

42.4.3.1 TRNG 控制寄存器 (TRNG_CR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RUN	随机数运算开始	● 0b0: 随机数运算停止 ● 0b1: 随机数运算开始  说明 软件写 1, 产生新的 64 位随机数; 运行完毕后, 硬件清零。	RW
0	EN	模拟发振器使能	● 0b0: 关闭模拟随机数发生器电路 ● 0b1: 打开模拟随机数发生器电路	RW

42.4.3.2 TRNG 模式寄存器（TRNG_MR）

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000012																															
Name	Reserved																												CNT		Reserved	LOAD
Access																															RW	

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4:2	CNT	移位次数控制位	捕捉随机噪声时，移位次数控制位 0b000~0b010：禁止设置 0b011：移位 32 次 0b100：移位 64 次 0b101：移位 128 次 0b110：移位 256 次 0b111：禁止设置	RW
1	Reserved	保留	读出为 1，写入时必须写 1	-
0	LOAD	装载控制位	随机数生成前，数据寄存器是否从随机数发生器装载新的初始值 0b0：不装载新的初始值 0b1：装载新的初始值	RW

42.4.3.3 TRNG 数据寄存器 n（TRNG_DRn）（n=0~1）

2 个 TRNG 数据寄存器 TRNG_DR0~TRNG_DR1，对应 64 位 TRNG 数据。TRNG_DR0 对应数据 Bit31~Bit0，TRNG_DR1 对应数据 Bit63~Bit32。

偏移地址（Offset）：0x000C+0x4*n

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DATA																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	随机数	随机数	RO

说明

TRNG_DR 寄存器由 2 个 32 位寄存器拼接而成。

42.4.4 PKE 寄存器

42.4.4.1 PKE 控制寄存器（PKE_CTRL）

Offset	Bit Position																															
0x1000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															START
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	START	开始计算	启动 PKE 运算 ● 0b0：PKE 未开始计算 ● 0b1：PKE 会在下一个时钟周期开始运算 说明 写 0 无效，不会改变该位当前值；PKE 计算完成后硬件自动清零。	RW

42.4.4.2 PKE 配置寄存器 (PKE_CFG)

Offset	Bit Position																															
0x1004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00020100																															
Name	Reserved														BASE_RADIX		Reserved		PARTIAL_RADIX													
Access															RW				RW													

位/位域	标记	位名	功能描述	读写
31:19	Reserved	保留	-	-
18:16	BASE_RADIX	基本配置	该域表示进行运算的位宽基数。同时，该基数也表示了操作数在数据 RAM 中存储需要的空间： ● 2：256-bit ● 3：512-bit ● 4：1024-bit ● 5：2048-bit ● 6：4096-bit ● 其他：禁止设置  说明 ● 对于 RSA 模运算，该域的取值为 4，5 或 6； ● 对于 ECC 点运算，该域的取值应为 2，3 或 4。	RW
15:13	Reserved	保留	-	-
12:0	PARTIAL_RADIX	其他配置	PARTIAL_RADIX 决定该操作在运算过程中真正需要使用的比特位宽。该域的数值表示比特的数目，操作数的比特位宽也就是 PARTIAL_RADIX 比特。 如果要进行 SECP521R1 运算，那么 BASE_RADIX 需要配置为 4，PARTIAL_RADIX 需要配置为 512。	RW

42.4.4.3 PKE 过程入口寄存器 (PKE_MC_PTR)

Offset	Bit Position																															
0x1008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ADDR											
Access																					RW											

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	ADDR	过程入口地址	PKE 微码执行入口  说明 该寄存器只有在 PKE 不工作时才能被改写，任何在 PKE 工作时的写操作都将会被忽略。 在 PKE 运行过程中，该域也会实时更新；总是指向下一条将被执行的指令地址。 需要注意的是，指令都是字对齐。因此该域的最低 2 位都是 0。	RW

42.4.4.4 PKE 中断源寄存器 (PKE_RISR)

Offset	Bit Position																															
0x100C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															CORE_RIS
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	CORE_RIS	操作完成标志位	写 0 清除标志，写 1 无效 读取： 0b0：PKE CPU 模式未完成操作 0b1：PKE CPU 模式已完成操作。若 PKE_IMCR.CORE_IRQEN 使能，则中断产生	RW

42.4.4.5 PKE 中断使能寄存器 (PKE_IMCR)

Offset	Bit Position																																
0x1010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																																CORE_IRQEN
Access																																	RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	CORE_IRQEN	CPU 模式中断使能	0b0: 禁止 PKE 在 CPU 模式下产生中断 0b1: 允许 PKE 在 CPU 模式下产生中断	RW

42.4.4.6 PKE 中断输出寄存器 (PKE_MISR)

Offset	Bit Position																																
0x1014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																																CORE_MI
Access																																	RO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	CORE_MI	CPU 模式中断标志	0b0: PKE 在 CPU 模式下未产生中断 0b1: PKE 在 CPU 模式下产生中断	RO

42.4.4.7 PKE 返回日志寄存器 (PKE_RT_CODE)

Offset	Bit Position																															
0x1024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																												STOP_LOG			
Access																													RO			

位/位域	标记	位名	功能描述	读写
31:4	Reserved	保留	-	-
3:0	STOP_LOG	停止日志	用来表示 PKE 停止的原因 如果 PKE 是因为运算完成而停止的，那么该域的值为 0。如果该域的值为非零，则 PKE 的运算未完成，遇到了一些异常，需要外部进行处理，结果不可用 0：正常停止 2：无有效的模逆结果 3：点不在曲线上 (CTRL.CMD：PVER) - 其他：Reserved	RO

42.4.4.8 PKE 运算配置寄存器 (PKE_EXE_CFG)

Offset	Bit Position																																					
0x1050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000015																																					
Name	Reserved																																					
Access																																						

位/位域	标记	位名	功能描述	读写
31:6	Reserved	保留	-	-
5	OMON	使能蒙哥马利形式输出	0b0: 输出结果为普通形式 0b1: 输出结果为蒙哥马利形式	RO
4	OAFF	使能仿射坐标系形式输出	0b0: 输出的点为雅可比坐标系上的点 0b1: 输出的点为仿射坐标系上的点  说明 该位设置仅对 ECC 运算有效。	RW
3	IMON_R1	使能蒙哥马利形式输入 R1	0b0: 输入数据为普通形式；当涉及到模乘运算时，若该比特位为低，将会先将其作用域上的数转换为蒙哥马利形式之后再进行运算 0b1: 输入数据为蒙哥马利形式	RW
2	IAFF_R1	使能仿射坐标系形式输入 R1	0b0: 输入的点为雅可比坐标系上的点；当涉及到模乘运算时，若该比特位为低，将会先将其作用域上的点转换为雅可比坐标系上的点后再进行运算 0b1: 输入的点为仿射坐标系上的点  说明 该位设置仅对 ECC 运算有效。	RW
1	IMON_R0	使能蒙哥马利形式输入 R0	0b0: 输入数据为普通形式；当涉及到模乘运算时，若该比特位为低，将会先将其作用域上的数转换为蒙哥马利形式之后再进行运算 0b1: 输入数据为蒙哥马利形式	RW
0	IAFF_R0	使能仿射坐标系形式输入 R0	0b0: 输入的点为雅可比坐标系上的点；当涉及到模乘运算时，若该比特位为低，将会先将其作用域上的点转换为雅可比坐标系上的点后再进行运算 0b1: 输入的点为仿射坐标系上的点  说明 该位设置仅对 ECC 运算有效。	RW

42.4.4.9 PKE 版本寄存器 (VERSION)

Offset	Bit Position																															
0x10FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xC4060010																															
Name	PROJECT																Reserved								MAR				MIR			
Access	RO																								RO				RO			

位/位域	标记	位名	功能描述	读写
31:16	PROJECT	项目编号	-	RO
15:8	Reserved	保留	-	-
7:4	MAR	主版本编号	-	RO
3:0	MIR	副版本编号	-	RO

42.4.4.10 PKE 运算操作数 RAM0 (PKE_MEM_RAM0)

该寄存器用于存储运算数据，根据运算模式和预算长度的不同，选择不同偏移地址的寄存器进行操作，详细信息请参见“编程模型”。

偏移地址 (Offset) : 0x1400~0x1E10

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	DATA																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	运算数据	该域用来存储运算数据	RW

42.4.4.11 PKE 运算操作数 RAM1 (PKE_MEM_RAM1)

该寄存器用于存储运算数据，根据运算模式和预算长度的不同，选择不同偏移地址的寄存器进行操作，详细信息请参见“编程模型”。

偏移地址 (Offset) : 0x2000~0x2A10

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	DATA																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DATA	运算数据	该域用来存储运算数据	RW

42.4.4.12 PKE RAM0 校验控制寄存器 (PKE_RAM0_CKCR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															ECCMOD
Access																																RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1:0	ECCMOD	PKE RAM0 的 ECC 校验允许位	● 0b00: 禁止 ECC 校验功能 ● 0b01: ▶ 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b10: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 ● 0b11: ▶ 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断 ▶ 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断	RW

42.4.4.13 PKE RAM0 校验状态寄存器 (PKE_RAM0_CKSR)

Offset	Bit Position																																	
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																														RW	RAM_2ERR	RW	RAM_1ERR
Access																															RW		RW	

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RAM_2ERR	PKE RAM0 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
0	RAM_1ERR	PKE RAM0 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW

42.4.4.14 PKE RAM0 错误注入使能寄存器 (PKE_RAM0_EIEN)

Offset	Bit Position																															
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	● 0b0: 错误注入禁止 ● 0b1: 错误注入使能	RW

42.4.4.15 PKE RAM0 错误注入位使能寄存器 0 (PKE_RAM0_EIBIT0)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 PKE RAM0 输出数据的 31~0 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

42.4.4.16 PKE RAM0 错误注入位使能寄存器 1 (PKE_RAM0_EIBIT1)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 PKE RAM0 输出数据的 38~32 位是否取反： ● 0b0：不取反，该位不注入 ECC 错误 ● 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

 说明 PKE RAM0 输出数据的 38~32 位为 ECC 位。

42.4.4.17 PKE RAM0 错误地址寄存器 (PKE_RAM0_ECCERRADDR)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RO ECCERRADDR															
Access																																

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:0	ECCERRADDR	校验错误地址	记录 PKE RAM0 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

42.4.4.18 PKE RAM1 校验控制寄存器 (PKE_RAM1_CKCR)

Offset	Bit Position																															
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															ECCMOD
Access																																RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1:0	ECCMOD	PKE RAM1 的 ECC 校验允许位	0b00: 禁止 ECC 校验功能 0b01: - 若 1 位错误, ECC 纠错, 不产生 1 位错误标志, 不产生中断 - 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 0b10: - 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 不产生中断 - 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断 0b11: - 若 1 位错误, ECC 纠错, 产生 1 位错误标志, 产生中断 - 若 2 位错误, ECC 检错, 产生 2 位错误标志, 产生中断	RW

42.4.4.19 PKE RAM1 校验状态寄存器 (PKE_RAM1_CKSR)

Offset	Bit Position																															
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														RW	RAM_2ERR
Access																															RW	RAM_1ERR

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	RAM_2ERR	PKE RAM1 发生 ECC 2-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 2-bit ECC 错误 ● 0b1: 产生 2-bit ECC 错误	RW
0	RAM_1ERR	PKE RAM1 发生 ECC 1-bit 错误标志	写 1 清除标志, 写 0 无效 读取: ● 0b0: 未产生 1-bit ECC 错误 ● 0b1: 产生 1-bit ECC 错误	RW

42.4.4.20 PKE RAM1 错误注入使能寄存器 (PKE_RAM1_EIEN)

Offset	Bit Position																																
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																															RW	EIEN
Access																																RW	EIEN

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	EIEN	错误注入使能	● 0b0: 错误注入禁止 ● 0b1: 错误注入使能	RW

42.4.4.21 PKE RAM1 错误注入位使能寄存器 0 (PKE_RAM1_EIBIT0)

Offset	Bit Position																															
0x002C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	EIBIT[31:0]																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	EIBIT[31:0]	错误注入位使能	控制 PKE RAM1 输出数据的 31~0 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW

42.4.4.22 PKE RAM1 错误注入位使能寄存器 1 (PKE_RAM1_EIBIT1)

Offset	Bit Position																															
0x0030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								EIBIT[38:32]							
Access																									RW							

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6:0	EIBIT[38:32]	错误注入位使能	控制 PKE RAM1 输出数据的 38~32 位是否取反： 0b0：不取反，该位不注入 ECC 错误 0b1：取反，该位注入 ECC 错误  说明 每个 Bit 对应一个输出数据位。	RW



说明

PKE RAM1 输出数据的 38~32 位为 ECC 位。

42.4.4.23 PKE RAM1 错误地址寄存器 (PKE_RAM1_ECCERRADDR)

Offset	Bit Position																															
0x0034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				RO ECCERRADDR											
Access																																

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:0	ECCERRADDR	校验错误地址	记录 PKE RAM1 产生 1-bit 或者 2-bit ECC 错误时的偏移地址	RO

42.4.5 SKE 寄存器

42.4.5.1 SKE 控制寄存器 (SKE_CTRL)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															WO START
Access																																WO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	START	启动	写 1 启动 SKE，写 0 无效  说明 - 当 SKE_CFG.UP_CFG 为 1 时，启动 SKE 更新配置，不进行数据运算； - 当 SKE_CFG.UP_CFG 为 0 时，启动 SKE 进行数据运算，数据输入采用 SKE_DINx 的数值。	WO

42.4.5.2 SKE 配置寄存器 (SKE_CFG)

Offset	Bit Position																																	
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	MODE				Reserved		DATA_TYPE		SEC_STAGE		Reserved				IRQEN		Reserved				UP_CFG		DEC		Reserved						ALG			
Access	RW						RW		RW						RW						RW		RW								RW			

位/位域	标记	位名	功能描述	读写
31:28	MODE	模式选择	1: ECB 模式 3: CBC 模式 4: CFB 模式 5: OFB 模式 6: CTR 模式 7: CMAC 模式 9: GCM 模式 10: CCM 模式 - 其他: 禁止设置	RW
27:26	Reserved	保留	-	-
25:24	DATA_TYPE	数据格式	对 SKE_DIN, SKE_DOUT 进行数据交换 ¹ 0b00: 无交换 0b01: 半字交换 0b10: 字节交换 0b11: 比特交换	RW
23:22	SEC_STAGE	数据交织中的数据段标识	0b00: 不分段 0b01: 初始段 0b10: 中间段 0b11: 末尾段	RW
21:19	Reserved	保留	-	-
17	IRQEN	中断使能	1. 0b0: 禁止 2. 0b1: 使能	RW
16:13	Reserved	保留	-	-
12	UP_CFG	更新配置	0b0: 不更新配置, SKE 采用上次输入的配置进行运算 0b1: 更新配置, SKE 采用新输入的配置进行运算 配置包含: SKE_CFG 中 ALG, DEC 以及 SKE_KEYx 的值。	RW
11	DEC	加解密方向选择	0b0: 加密 0b1: 解密	RW
10:4	Reserved	保留	-	-

位/位域	标记	位名	功能描述	读写
3:0	ALG	算法选择	● 1: AES 128 ● 2: SM4 ● 3: DES ● 4: AES 192 ● 5: AES 256 ● 其他: 禁止设置	RW

 **说明**

1. DATA_TYPE 用于在模块运算前存放在 SKE_DIN 寄存器的需要被加密的明文或者需要被解密的密文，和运算完成后存放加密后的密文或者解密后的明文的 SKE_DOUT 寄存器。

数据举例：0x11223344

DATA_TYPE	数据处理	使用数据
0x0	无交换	0x11223344
0x1	半字交换	0x33441122
0x2	字节交换	0x44332211
0x3	比特交换	0x22CC4488

42.4.5.3 SKE 状态寄存器 1 (SKE_SR1)

 **说明**

在触发 SKE 计算前，需先读该寄存器 BUSY 的值，确认为 0 后再进行后续操作。

Offset	Bit Position																																	
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																														MID_O_VALID		BUSY	
Access																															RW	RO		

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	MID_O_VALID	输出中间值有效标志位	计算结束后该位自动置 1；写 0 清零 读取： ● 0b0: 输出中间值无效 ● 0b1: 输出中间值有效	RW
0	BUSY	忙标志位	● 0b0: SKE 处于空闲状态，可接收指令或数据输入 ● 0b1: SKE 处于运行状态，不能接收指令或数据输入	RO

42.4.5.4 SKE 状态寄存器 2（SKE_SR2）

说明

在触发 SKE 计算前，需先清除该寄存器的值。

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															SKE_DONE
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	SKE_DONE	SKE 操作完成指示位	写 0 清零，写 1 无效 读取： 0b0：SKE 操作未完成 0b1：SKE 操作完成	RW

42.4.5.5 SKE 密钥寄存器（SKE_KEYx）（x=1~8）

偏移地址（Offset）：0x0010+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	KEY																															
Access	WO																															

位/位域	标记	位名	功能描述	读写
31:0	KEY	密钥输入	该寄存器作为加解密密钥，需要在 SKE_CFG.UP_CFG=1 时输入密钥才有效，且只支持 32 位操作	WO

42.4.5.6 SKE AAD 长度寄存器 (SKE_AADx) (x=1~2)

偏移地址 (Offset) : 0x0060+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	A_LEN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	A_LEN	AAD 比特长度	该寄存器在 GCM、CCM 模式下有效	RW

42.4.5.7 SKE 数据长度寄存器 (SKE_CLENx) (x=1~2)

偏移地址 (Offset) : 0x0068+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	C_LEN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	C_LEN	密文比特长度	该寄存器在 GCM、CCM 模式下有效	RW

42.4.5.8 SKE 初始向量寄存器 (SKE_IVx) (x=1~4)

偏移地址 (Offset) : 0x0070+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IV																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	IV	初始向量	该寄存器在非 ECB 模式下有效, 且只支持 32 位操作	RW

42.4.5.9 SKE 数据标志寄存器（SKE_DIN_CR）

Offset	Bit Position																															
0x0080	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																LAST	Reserved								LAST_LEN						
Access																	RW									RW						

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	LAST	数据末尾标志位	写 0 清零，计算完成需要清零 读取： ● 0b0：当前数据不包含数据末尾 ● 0b1：当前数据包含数据末尾  说明 该位仅在 GCM、CCM、CMAC 模式有效。	RW
15:8	Reserved	保留	-	-
7:0	LAST_LEN	CMAC 模式下的最后一个 Block 长度	以比特为单位，计算完成需要清零（数据实际长度，不包含填充位）	RW

42.4.5.10 SKE 数据输入寄存器（SKE_DINx）（x=1~4）

偏移地址（Offset）：0x0090+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DATA_IN																															
Access	WO																															

位/位域	标记	位名	功能描述	读写
31:0	DATA_IN	数据输入	该寄存器只支持 32 位操作，CMAC 模式填入该寄存器的是未进行填充的数据	WO

42.4.5.11 SKE 数据输出寄存器（SKE_DOUTx）（x=1~4）

偏移地址（Offset）：0x00B0+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DATA_OUT																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DATA_OUT	数据输出	当 SKE_SR2.CORE_DONE=1 时，可读取本寄存器的值作为运算结果	RO

42.4.5.12 SKE 版本寄存器（SKE_VERSION）

Offset	Bit Position																															
0x00FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x001D0011																															
Name	PROJECT																Reserved								MAR				MIR			
Access	RO																								RO				RO			

位/位域	标记	位名	功能描述	读写
31:16	PROJECT	项目编号	-	RO
15:8	Reserved	保留	-	-
7:4	MAR	主版本编号	-	RO
3:0	MIR	副版本编号	-	RO

42.4.5.13 SKE 中间向量寄存器（SKE_MID_IVx）（x=1~4）

偏移地址（Offset）：0x01A0+0x4*(x-1)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	MID_IV																																					
Access	RW																																					

位/位域	标记	位名	功能描述	读写
31:0	MID_IV	数据交织中的初始向量中间值	该寄存器在 SKE_CFG.SEC_STAGE 不为 0 时有效且只支持 32 位操作。 用于 GCM、CCM、CMAC 的初始段、中间段和末尾段。 - 在计算开始前（中间段和末尾段），写入对应的初始值； - 在计算结束时（初始段和中间段），读取输出的中间值。	RW

 **说明**

中间值包括输入值和输出值，使用同一个地址进行访问，且输入值写入后无法读取。当 SKE_SR1.MID_O_VALID=1 时，表示该寄存器中的输出值有效。

42.4.5.14 SKE 中间 MAC 寄存器（SKE_MID_MACx）（x=1~4）

偏移地址（Offset）：0x01B0+0x4*(x-1)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	MID_MAC																																					
Access	RW																																					

位/位域	标记	位名	功能描述	读写
31:0	MID_MAC	数据交织中的 MAC/TAG 中间值	该寄存器在 SKE_CFG.SEC_STAGE 不为 0 时有效且只支持 32 位操作。 用于 GCM、CCM 的初始段、中间段和末尾段。 - 在计算开始前（中间段和末尾段），写入对应的初始值； - 在计算结束时（初始段和中间段），读取输出的中间值。	RW

 **说明**

中间值包括输入值和输出值，使用同一个地址进行访问，且输入值写入后无法读取。当 SKE_SR1.MID_O_VALID=1 时，表示该寄存器中的输出值有效。

42.4.5.15 SKE AAD 总长度寄存器 (SKE_TOTAL_AADx) (x=1~2)

偏移地址 (Offset) : 0x0060+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TOTAL_A_LEN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TOTAL_A_LEN	数据交织中的 AAD 总比特长度	该寄存器在 SKE_CFG.SEC_STAGE=3 时有效。用于 GCM 模式的末尾段，标识数据 AAD 的总长度。	RW



说明

该寄存器的值表示整个 GCM 事务的 AAD 的长度，不是当前分段的 AAD 长度。

42.4.5.16 SKE 数据总长度寄存器 (SKE_TOTAL_CLENx) (x=1~2)

偏移地址 (Offset) : 0x0068+0x4*(x-1)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	TOTAL_C_LEN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	TOTAL_C_LEN	数据交织中的 Payload 总比特长度	该寄存器在 SKE_CFG.SEC_STAGE=3 时有效。用于 GCM 模式的末尾段，标识数据 Payload 的总长度。	RW

43 CRC 运算（CRC）

43.1 简介

在许多应用中，都需要 CRC 算法来校验数据的完整性和正确性。尤其是在数据传输中，CRC 校验更是被广泛应用。本模块可采用 CRC16/CCITT，CRC16/CCITT-FALSE，CRC16/XMODE，CRC16/IBM，CRC16/MAXIM，CRC16/USB，CRC16/MODBUS，CRC16/X25，CRC32，CRC32/MPEG-2 和 CRC64 共计 11 种算法对数据进行运算和校验。

43.2 功能说明

43.2.1 功能框图

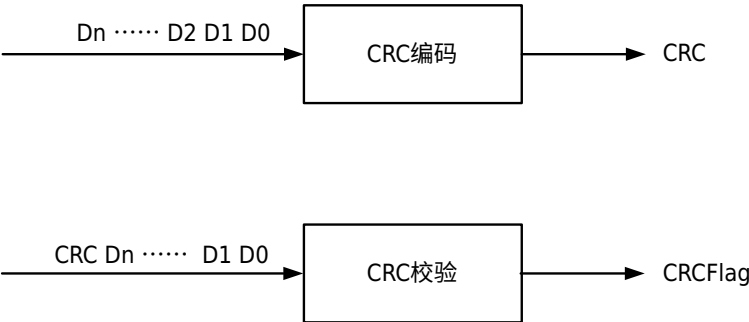


图 43-1 CRC 应用示意图

43.2.2 CRC 参数模型

本模块 CRC 算法遵从 ISO/IEC13239 的定义，分别采用 16 位、32 位和 64 位的 CRC。CRC 各算法对应参数模型如下表。

表 43-1 CRC 参数模型

CRC 算法名称	多项式公式	初始值	结果异或值	输入反转	输出反转
CRC16/IBM	$X^{16}+X^{15}+X^2+1$	0x0000	0x0000	是	是
CRC16/MAXIM	$X^{16}+X^{15}+X^2+1$	0x0000	0xFFFF	是	是
CRC16/USB	$X^{16}+X^{15}+X^2+1$	0xFFFF	0xFFFF	是	是
CRC16/MODBUS	$X^{16}+X^{15}+X^2+1$	0xFFFF	0x0000	是	是
CRC16/CCITT	$X^{16}+X^{12}+X^5+1$	0x0000	0x0000	是	是
CRC16/CCITT-FALSE	$X^{16}+X^{12}+X^5+1$	0xFFFF	0x0000	否	否
CRC16/X25	$X^{16}+X^{12}+X^5+1$	0xFFFF	0xFFFF	是	是
CRC16/XMODEM	$X^{16}+X^{12}+X^5+1$	0x0000	0x0000	否	否
CRC32	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$	0xFFFFFFFF	0xFFFFFFFF	是	是

CRC 算法名称	多项式公式	初始值	结果异或值	输入反转	输出反转
CRC32/MPEG-2	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$	0xFFFFFFFF	0x00000000	否	否
CRC64	$X^{64}+X^4+X^3+X+1$	0xFFFFFFFF FFFFFFFF	0xFFFFFFFF FFFFFFFF	是	是

本模块功能包括：

- CRC 编码和 CRC 校验；
- 3 种位宽访问方式 8 位，16 位，32 位：
 - ▶ 8 位位宽下输入数据示例为 0x00、0x11、0x22、0x33、0x44、0x55、0x66、0x77；
 - ▶ 16 位位宽下输入数据示例为 0x1100、0x3322、0x5544、0x7766；
 - ▶ 32 位位宽下输入数据示例为 0x33221100、0x77665544。

43.2.3 CRC16 编码模式

CRC 编码就是对原始数据编码以计算其 CRC 值，操作流程如下：

1. 向 CRC_CR.CR 写入 0b0 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC16 规格算法。
2. 根据 [CRC 参数模型](#)向 CRC_RESULT1[15:0]写入 0x0000 或者 0xFFFF，初始化 CRC 计算。
3. 将待编码的原始数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。



说明

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 读取 CRC_RESULT1[15:0]，即可获得 16 位 CRC 编码值。

43.2.4 CRC16 校验模式

校验模式可以校验已编码的数据是否被篡改，操作流程如下所示：

1. 向 CRC_CR.CR 写入 0b0 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC16 规格算法。
2. 根据 [CRC 参数模型](#)向 CRC_RESULT1[15:0]写入 0x0000 或者 0xFFFF，初始化 CRC 计算。
3. 将已编码的数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。



说明

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 校验码以 16 位位宽将数据写入 CRC_DAT 寄存器。



说明

CRC16/CCITT-FALSE 和 CRC16/XMODEM 在校验时，需要对校验码进行高 8 位和低 8 位对调操作。
例如：0x1234->0x3412

5. 读取 CRC_CR.FLAG 的值，为 1 则表示校验成功，为 0 则表示校验失败。

43.2.5 CRC32 编码模式

CRC 编码就是对原始数据编码以计算其 CRC 值，操作流程如下：

1. 向 CRC_CR.CR 写入 0b1 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC32 规格算法。
2. 向 CRC_RESULT1[31:0]写入 0xFFFFFFFF，初始化 CRC 计算。
3. 将待编码的原始数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。

**说明**

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 读取 CRC_RESULT1[31:0]，即可获得 32 位 CRC 编码值。

43.2.6 CRC32 校验模式

校验模式可以校验已编码的数据是否被篡改，操作流程如下所示：

1. 向 CRC_CR.CR 写入 0b1 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC32 规格算法。
2. 向 CRC_RESULT1[31:0]写入 0xFFFFFFFF，初始化 CRC 计算。
3. 将已编码的数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。

**说明**

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 校验码按 8 位/16 位/32 位组织方式，写入 CRC_DAT 寄存器。

**说明**

CRC32/MPEG-2 在校验时，需要对其高 16 位和低 16 位进行对调操作，且分别对其高、低 16 位中的 16 位数据进行高 8 位和低 8 位进行对调操作。

例如：0x12345678->0x78563412

5. 读取 CRC_CR.FLAG 的值，为 1 则表示校验成功，为 0 则表示校验失败。

43.2.7 CRC64 编码模式

CRC 编码就是对原始数据编码以计算其 CRC 值，操作流程如下：

1. 向 CRC_CR.CR 写入 0b1 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC64 规格算法。
2. 向 CRC_RESULT1[31:0]和 CRC_RESULT2[31:0]均写入 0xFFFFFFFF，初始化 CRC 计算。
3. 将待编码的原始数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。

**说明**

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 读取 CRC_RESULT1[31:0]和 CRC_RESULT2[31:0]，即可获得 64 位 CRC 编码值。

**说明**

CRC64_DATA= {CRC_RESULT2[31:0], CRC_RESULT1[31:0]}。

43.2.8 CRC64 校验模式

校验模式可以校验已编码的数据是否被篡改，操作流程如下所示：

1. 向 CRC_CR.CR 写入 0b1 和 CRC_CR.SELECT 写入对应数据，选择对应 CRC64 规格算法。
2. 向 CRC_RESULT1[31:0]和 CRC_RESULT2[31:0]均写入 0xFFFFFFFF，初始化 CRC 计算。
3. 将已编码的数据按 8 位/16 位/32 位组织方式，依次写入 CRC_DAT 寄存器。

**说明**

按 8 位组织方式写 CRC 值，应先写低位，后写高位。

4. 校验码按 8 位/16 位/32 位组织方式，写入 CRC_DAT 寄存器。
5. 读取 CRC_CR.FLAG 寄存器，为 1 则表示校验成功，为 0 则表示校验失败。

43.3 寄存器

43.3.1 寄存器总表

基地址：0x40008C00

表 43-2 CRC 运算（CRC）寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	CRC_CR	CRC 控制寄存器	32
0x04	CRC_RESULT1	CRC 结果寄存器 1	32
0x08	CRC_RESULT2	CRC 结果寄存器 2	32
0x80~0xFC	CRC_DAT	CRC 数据寄存器	32

43.3.2 控制寄存器 (CRC_CR)

Offset	Bit Position																																	
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000001																																	
Name	Reserved																										RW		SELECT		RO		FLAG	
Access																																		

位/位域	标记	位名	功能描述	读写
31:5	Reserved	保留	-	-
4:2	SELECT	CRC 选择位	CR=0b0 时, Bit4~2 对应 CRC 算法选择: 0b000: CRC16/X25 0b001: CRC16/CCITT 0b010: CRC16/CCITT-FALSE 0b011: CRC16/XMODEM 0b100: CRC16/IBM 0b101: CRC16/MAXIM 0b110: CRC16/USB 0b111: CRC16/MODBUS CR=0b1 时, Bit4~2 对应 CRC 算法选择: 0b000: CRC32 0b001: CRC32/MPEG-2 0b010: CRC64 - 其他: 禁止设置	RW
1	FLAG	校验结果位	0b0: 当前校验错误 0b1: 当前校验正确	RO
0	CR	运算控制位	0b0: CRC16 0b1: CRC32/CRC64	RW

43.3.3 结果寄存器 1 (CRC_RESULT1)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RESULT1																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	RESULT1	结果位	- 选择 CRC16 时，取 RESULT1[15:0] - 选择 CRC32 时，取 RESULT1[31:0] - 选择 CRC64 时，CRC_RESULT1 和 CRC_RESULT2 联合使用，CRC_RESULT1.RESULT1[31:0]对应 CRC 结果的 Bit31~Bit0; CRC_RESULT2.RESULT2[31:0] 对应 CRC 结果的 Bit63~Bit32	RW

43.3.4 结果寄存器 2 (CRC_RESULT2)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RESULT2																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	RESULT2	结果位	选择 CRC64 时，CRC_RESULT1 和 CRC_RESULT2 联合使用： - CRC_RESULT1.RESULT1[31:0]对应 CRC 结果的 Bit31~Bit0 - CRC_RESULT2.RESULT2[31:0]对应 CRC 结果的 Bit63~Bit32	RW

43.3.5 数据寄存器 (CRC_DAT)

Offset	Bit Position																															
0x80~0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CRC_DAT																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	CRC_DAT	需要运算的数据	写入需要运算的数据；读出恒为 0	RW

说明

CRC_DAT 寄存器位于 CRC 基地址的 0x80 偏移地址处，基地址的 0x84~0xFC 偏移地址处为 CRC_DAT 寄存器的映射地址，其访问和操作权限与 CRC_DAT 寄存器相同。

44 数据计算单元 (DCU)

44.1 简介

数据计算单元 (Data Computing Unit) 是一个不借助于 CPU 的简单处理数据的模块。每个 DCU 单元具有 3 个数据寄存器，能够进行 2 个数据的加减运算和比较大小以及窗口比较功能。还可以通过定时器触发为数模转换模块 (DAC) 提供连续变化的数字量以产生三角波和锯齿波输出。本产品搭载 8 个 DCU 单元。

44.2 主要特性

- 可进行 4 种数据处理：2 个数据的加法，减法，比较以及 3 个数据窗口比较
- 加法和减法对 DATA0 和 DATA1 寄存器中的数据进行运算，结果存放在 DATA0 中；同时自动将结果进行一次减半操作，减半的结果存放到 DATA1
- 加法和减法可以选择写寄存器后计算或由其他外围电路事件触发进行运算
- 比较模式可以对 DATA0 和 DATA1 寄存器之间，以及 DATA0 和 DATA2 寄存器之间的 2 个数据进行比较，可以分别选择当满足大于、小于、等于时产生中断和标志位
- 比较模式可以用来进行窗口比较，即分别设置 DATA1 和 DATA2 为窗口的上下限，根据 DATA0 和 DATA1 及 DATA0 和 DATA2 的比较结果判断 DATA0 在窗口内或窗口外
- 能够由其他外围电路事件触发进行运算，并根据运算结果产生多种中断和事件信号。当其他具有硬件触发启动功能的外围电路选择 DCU 作为触发源时，由 DCU 产生事件信号来启动外围电路开始动作
- 在三角波输出模式下，通过定时器定时触发产生连续递增和递减的数字量给数模转换模块 (DAC)，后者能够将这些数字量通过数模转换转换为三角波输出
- 在递增锯齿波输出模式下，通过定时器定时触发产生连续递增的数字量给数模转换模块 (DAC)，后者能够将这些数字量通过数模转换转换为递增的锯齿波输出
- 在递减锯齿波输出模式下，通过定时器定时触发产生连续递减的数字量给数模转换模块 (DAC)，后者能够将这些数字量通过数模转换转换为递减的锯齿波输出

44.3 功能说明

44.3.1 加法模式

加法模式计算 DATA0 和 DATA1 的和，其中 DATA0 作为被加数，DATA1 作为加数。每次写 DATA1 寄存器进行一次 DATA0+DATA1 的加法运算，加法运算结果存入 DATA0，加法运算结果除以 2 存入 DATA2。当 DATA0+DATA1 结果大于 0xFF (8-bit 模式) 或 0xFFFF (16-bit 模式) 或 0xFFFFFFFF (32-bit 模式) 时产生中断和标志位。

加法模式应用举例：

- 步骤 1.** 控制寄存器 DCU_CTL 选择加法模式，数据宽度为 16-bit；
- 步骤 2.** DCU_INTEVTSEL 选择中断产生条件；
- 步骤 3.** 分别在 DATA0 和 DATA1 中写入 0xFF00 和 0x55，此时计算结果为 0xFF55，结果保存在 DATA0 中，0xFF55/2 保存在 DATA2；
- 步骤 4.** DATA1 中继续写入 0xFF，此时计算结果 (0xFF55+0xFF>0xFFFF) 溢出，产生结果标志。读取标志寄存器 DCU_FLAG 得到结果标志；
- 步骤 5.** 写标志复位寄存器 DCU_FLAGCLR 清除标志位。

44.3.2 减法模式

减法模式计算 DATA0 和 DATA1 的差，其中 DATA0 作为被减数，DATA1 作为减数。每次写 DATA1 寄存器进行一次 DATA0-DATA1 的减法运算，减法运算结果存入 DATA0，减法运算结果除以 2 存入 DATA2。当 DATA0-DATA1 结果小于 0x0 (8-bit、16-bit、32-bit 模式) 时产生中断和标志位。

44.3.3 硬件触发启动模式

DCU 能够根据外围电路产生的事件触发启动运算。使用硬件触发启动模式时，需要先将功能时钟控制 0 寄存器 (FCG0) 的外围电路触发功能使能位置为有效。每个 DCU 单元可独立选择由其他外围电路送来的触发启动信号，选择启动信号时，在触发源选择寄存器 (DCU_TRGSEL) 中写入要选择的外围电路启动源的编号，寄存器详情请参见《自动运行系统 (AOS) 》。当该外围电路事件发生时，会将事件信号输入 DCU 并触发 DCU 启动运算。

硬件触发启动模式包括触发加模式和触发减模式：

- 在触发加模式时，每发生一次事件触发，DCU 将启动并进行一次 DATA0+DATA1 的加法运算，加法运算结果存入 DATA0，加法运算结果除以 2 存入 DATA2。当 DATA0+DATA1 的结果大于 0xFF (8-bit 模式) 或 0xFFFF (16-bit 模式) 或 0xFFFFFFFF (32-bit 模式) 时置位标志位并产生中断。
- 在触发减模式时，每发生一次事件触发，DCU 将启动并进行一次 DATA0-DATA1 的减法运算，减法运算结果存入 DATA0，减法运算结果除以 2 存入 DATA2。当 DATA0-DATA1 的结果小于 0x0 (8-bit、16-bit、32-bit 模式) 时置位标志位并产生中断。

硬件触发启动模式应用举例：

- 步骤 1. 控制寄存器 DCU_CTL 选择触发加模式，数据宽度为 8-bit；
- 步骤 2. 中断条件选择寄存器 DCU_INTEVTSEL 选择运算条件；
- 步骤 3. 分别在 DATA0 和 DATA1 中写入 0x00 和 0x56；
- 步骤 4. 在触发源选择寄存器 DCU_TRGSEL 中写入事件编号；
- 步骤 5. 使选中的外围电路动作并产生事件，DCU 由该事件触发并进行一次加法运算，结果为 0x56 并保存在 DATA0 中，0x56 除以 2 保存在 DATA2；
- 步骤 6. 在连续 3 次触发后，计算结果溢出，产生结果标志。读取标志寄存器 DCU_FLAG 得到结果标志；
- 步骤 7. 写标志复位寄存器 DCU_FLAGCLR 清除标志位。

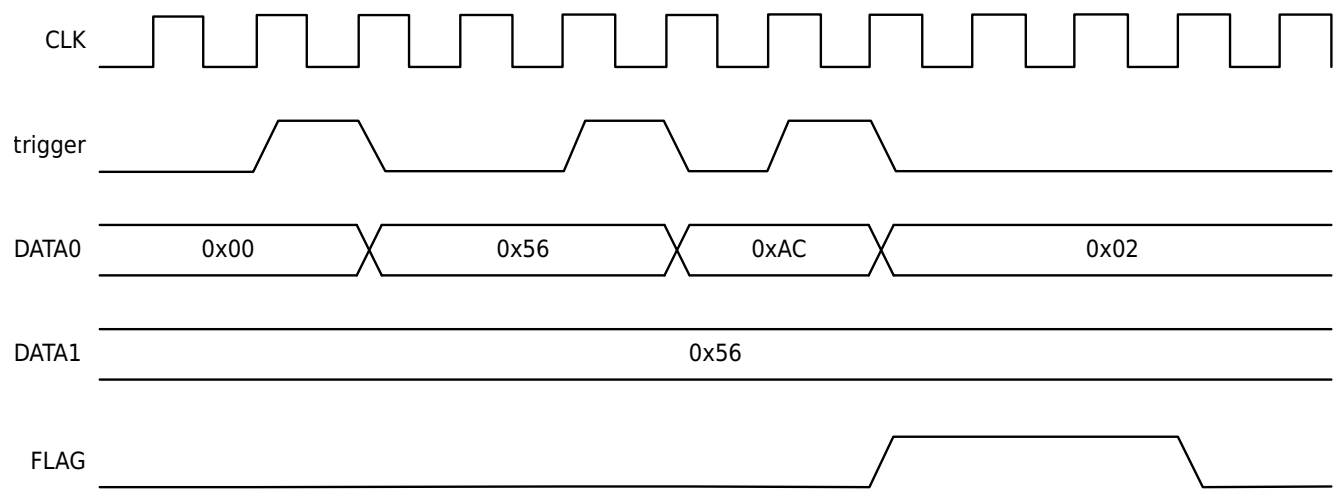


图 44-1 硬件触发启动示意图

44.3.4 比较模式

比较 DATA0 和 DATA1 以及 DATA0 和 DATA2 的大小，可选择当 DATA0 大于 DATA1，DATA0 小于 DATA1，DATA0 等于 DATA1 时以及当 DATA0 大于 DATA2，DATA0 小于 DATA2，DATA0 等于 DATA2 时产生标志位并产生中断。比较模式下可以选择数据开始比较的条件，写 DATA0 后比较或写任何数据寄存器后比较。

比较模式应用举例：

- 步骤 1. 控制寄存器 DCU_CTL 选择比较模式，数据宽度为 16-bit。在写入 DATA0 后开始比较；
- 步骤 2. 中断条件选择寄存器 DCU_INTEVTSEL 选择 DATA0>DATA1 时产生标志；

- 步骤 3. 在 DATA1 和 DATA0 中分别写入 0xB BBBB 和 0xA AAAA，此时由于不满足 DATA0>DATA1 因此不产生标志；
- 步骤 4. 在 DATA1 写入 0x8888，此时虽然满足 DATA0>DATA1，但由于设置为写入 DATA0 后开始比较，因此也不产生标志；
- 步骤 5. DATA0 写入 0x9999 后满足标志产生条件，产生标志；
- 步骤 6. 写标志复位寄存器 DCU_FLAGCLR 清除标志位。

44.3.5 三角波输出模式

三角波输出模式是 DCU 通过定时器作为触发源，为数模转换模块 (DAC) 提供一个连续变化的数字量，DAC 将这些数字量转化为三角波输出。三角波输出模式通过 DCU 控制寄存器 (DCU_CTL) 设定，设定成三角波输出模式后，数据寄存器 DATA0~2 的功能将发生变化，数据寄存器 DATA0 将作为输出给 DAC 的数字量的数据寄存器并复位。DATA0 变为输出数据寄存器，DATA.DO (Bit0~Bit11) 为有效位。DATA1 变为振幅设定寄存器，DATA1.UPL (Bit16~Bit27) 和 DATA1.LWL (Bit0~Bit11) 为有效位。DATA2 变为步长设定寄存器，DATA2.STEP (Bit0~Bit11) 为有效位。

 说明

仅 DCU1~4 支持三角波输出模式。

使用三角波输出模式步骤如下：

- 步骤 1. 将 DCU 控制寄存器 (DCU_CTL) 的 MODE 设为 0b1000；
- 步骤 2. 设定三角波振幅的上限和下限。上限通过 DATA1.UPL[11:0]位设定，下限通过 DATA1.LWL[11:0]设定；
- 步骤 3. 设定三角波递增和递减的步长值，通过 DATA2.STEP[11:0]设定；
- 步骤 4. 通过触发源选择寄存器 (DCU_TRGSEL) 选择合适的定时器事件或软件触发作为触发源；
- 步骤 5. 配置选定的定时器，使其在设定的时间间隔上产生触发事件；
- 步骤 6. 启动定时器，第一次产生触发事件时，DATA1.LWL 的值加载到 DATA0.DO 并输出给 DAC，以后每次产生触发事件时，DATA0.DO 将根据 DATA2.STEP 设定的步长值递增；
- 步骤 7. 当 DATA0.DO 递增至 DATA1.UPL 设定的上限时，将自动根据 DATA2.STEP 设定的步长值开始递减。当递减至 DATA1.LWL 设定的下限后重新开始递增；
- 步骤 8. 如果需要在三角波到达峰点或谷点时发生中断，请先将 DCU 控制寄存器 (DCU_CTL) 的中断使能位 (INTEN) 置为无效，再设置 DATA0~DATA2 的值，在启动定时器开始产生触发事件前清除一次 DCU 标志位后，将 DCU 控制寄存器中的中断使能位 (INTEN) 置为有效后，再启动定时器；
- 步骤 9. 将 DCU 控制寄存器 (DCU_CTL) 的 MODE 设为 0b0000 以退出三角波输出模式。

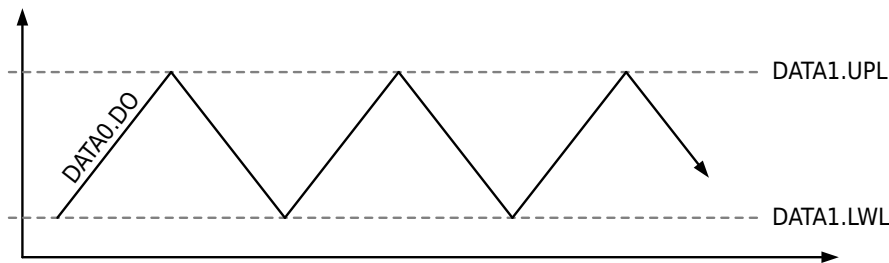


图 44-2 三角波输出模式波形

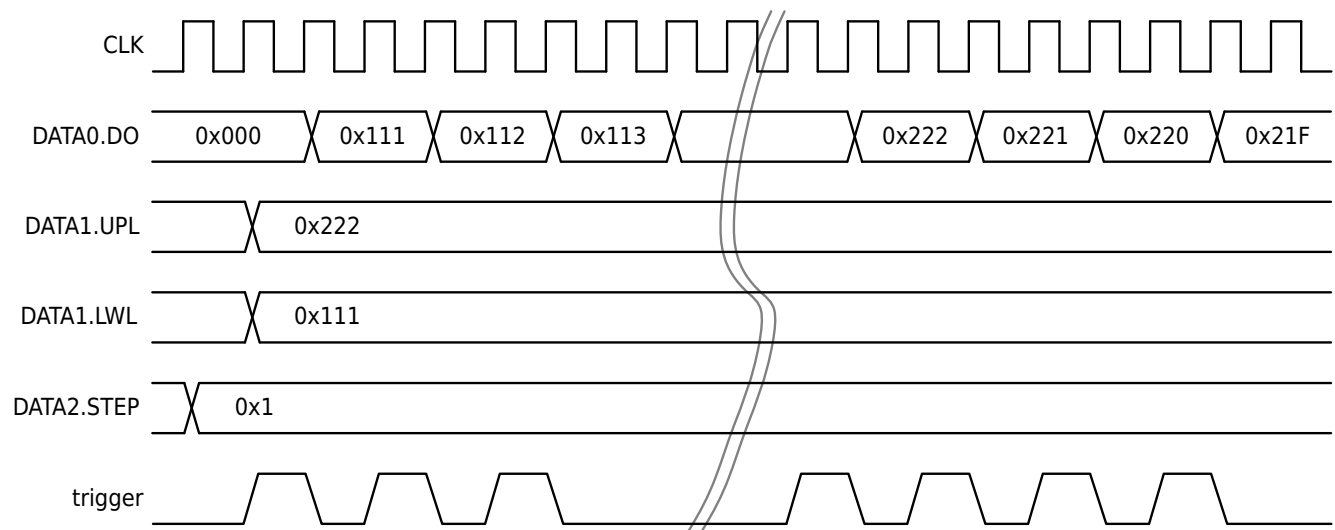


图 44-3 三角波输出模式

44.3.6 递增锯齿波输出模式

递增锯齿波输出模式是 DCU 通过定时器作为触发源，为数模转换模块（DAC）提供一个连续变化的数字量，DAC 将这些数字量转化为锯齿波输出。递增锯齿波输出模式通过 DCU 控制寄存器（DCU_CTL）设定，设定成递增锯齿波输出模式后，数据寄存器 DATA0~2 的功能将发生变化，数据寄存器 DATA0 将作为输出给 DAC 的数字量的数据寄存器并复位。DATA0 变为输出数据寄存器，DATA.DO（Bit0~Bit11）为有效位。DATA1 变为振幅设定寄存器，DATA1.UPL（Bit16~Bit27）和 DATA1.LWL（Bit0~Bit11）为有效位。DATA2 变为步长设定寄存器，DATA2.STEP（Bit0~Bit11）为有效位。

 **说明**
仅 DCU1~4 支持递增锯齿波输出模式。

使用递增锯齿波输出模式步骤如下：

- 步骤 1.** 将 DCU 控制寄存器（DCU_CTL）的 MODE 设为 0b1001；
- 步骤 2.** 设定锯齿波振幅的上限和下限。上限通过 DATA1.UPL[11:0]位设定，下限通过 DATA1.LWL[11:0]设定；
- 步骤 3.** 设定锯齿波递增的步长值，通过 DATA2.STEP[11:0]设定；
- 步骤 4.** 通过触发源选择寄存器（DCU_TRGSEL）选择合适的定时器事件或软件触发作为触发源；
- 步骤 5.** 配置选定的定时器，使其在设定的时间间隔上产生触发事件；
- 步骤 6.** 启动定时器，第一次产生触发事件时，DATA1.LWL 的值加载到 DATA0.DO 并输出给 DAC，以后每次产生触发事件时，DATA0.DO 将根据 DATA2.STEP 设定的步长值递增；
- 步骤 7.** 当 DATA0.DO 递增至 DATA1.UPL 设定的上限时，将自动重新加载 DATA1.LWL 到 DATA0.DO，使其从 DATA1.LWL 设定的下限重新开始递增；
- 步骤 8.** 如果需要在锯齿波重载时发生中断，请先将 DCU 控制寄存器（DCU_CTL）的中断使能位（INTEN）置为无效，再设置 DATA0~DATA2 的值，在启动定时器开始产生触发事件前清除一次 DCU 标志位后，将 DCU 控制寄存器中的中断使能位（INTEN）置为有效后，再启动定时器；
- 步骤 9.** 将 DCU 控制寄存器（DCU_CTL）的 MODE 设为 0b0000 以退出锯齿波输出模式。

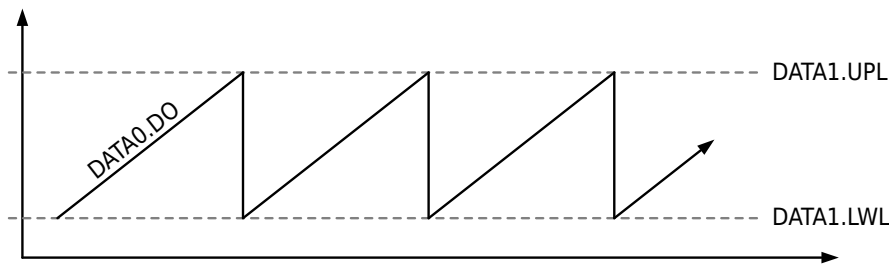


图 44-4 递增锯齿波输出模式波形

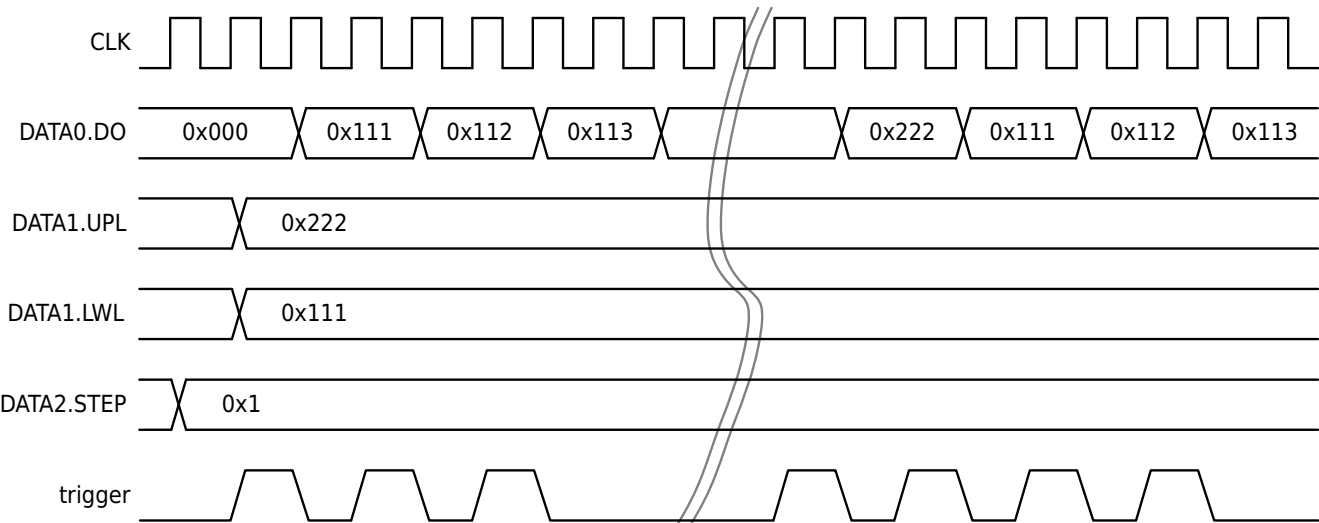


图 44-5 递增锯齿波输出模式

44.3.7 递减锯齿波输出模式

递减锯齿波输出模式是 DCU 通过定时器作为触发源，为数模转换模块（DAC）提供一个连续变化的数字量，DAC 将这些数字量转化为锯齿波输出。递减锯齿波输出模式通过 DCU 控制寄存器（DCU_CTL）设定，设定成递减锯齿波输出模式后，数据寄存器 DATA0~2 的功能将发生变化，数据寄存器 DATA0 将作为输出给 DAC 的数字量的数据寄存器并复位。DATA0 变为输出数据寄存器，DATA.DO（Bit0~Bit11）为有效位。DATA1 变为振幅设定寄存器，DATA1.UPL（Bit16~Bit27）和 DATA1.LWL（Bit0~Bit11）为有效位。DATA2 变为步长设定寄存器，DATA2.STEP（Bit0~Bit11）为有效位。

 **说明**
仅 DCU1~4 支持递减锯齿波输出模式。

使用递减锯齿波输出模式步骤如下：

- 步骤 1.** 将 DCU 控制寄存器（DCU_CTL）的 MODE 设为 0b1010；
- 步骤 2.** 设定锯齿波振幅的上限和下限。上限通过 DATA1.UPL[11:0]位设定，下限通过 DATA1.LWL[11:0]设定；
- 步骤 3.** 设定锯齿波递减的步长值，通过 DATA2.STEP[11:0]设定；
- 步骤 4.** 通过触发源选择寄存器（DCU_TRGSEL）选择合适的定时器事件或软件触发作为触发源；
- 步骤 5.** 配置选定的定时器，使其在设定的时间间隔上产生触发事件；
- 步骤 6.** 启动定时器，第一次产生触发事件时，DATA1.UPL 的值加载到 DATA0.DO 并输出给 DAC，以后每次产生触发事件时，DATA0.DO 将根据 DATA2.STEP 设定的步长值递减；
- 步骤 7.** 当 DATA0.DO 递减至 DATA1.LWL 设定的下限时，将自动重新加载 DATA1.UPL 到 DATA0.DO，使其从 DATA1.UPL 设定的上限重新开始递减；

- 步骤 8.** 如果需要在锯齿波重载时发生中断, 请先将 DCU 控制寄存器 (DCU_CTL) 的中断使能位 (INTEN) 置为无效, 再设置 DATA0~DATA2 的值, 在启动定时器开始产生触发事件前清除一次 DCU 标志位后, 将 DCU 控制寄存器中的中断使能位 (INTEN) 置为有效后, 再启动定时器;
- 步骤 9.** 将 DCU 控制寄存器 (DCU_CTL) 的 MODE 设为 0b0000 以退出锯齿波输出模式。

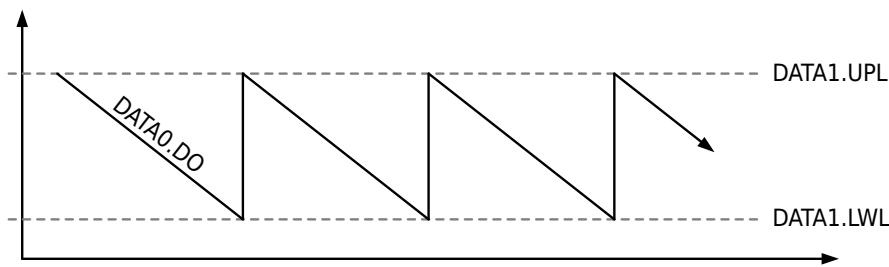


图 44-6 递减锯齿波输出模式波形

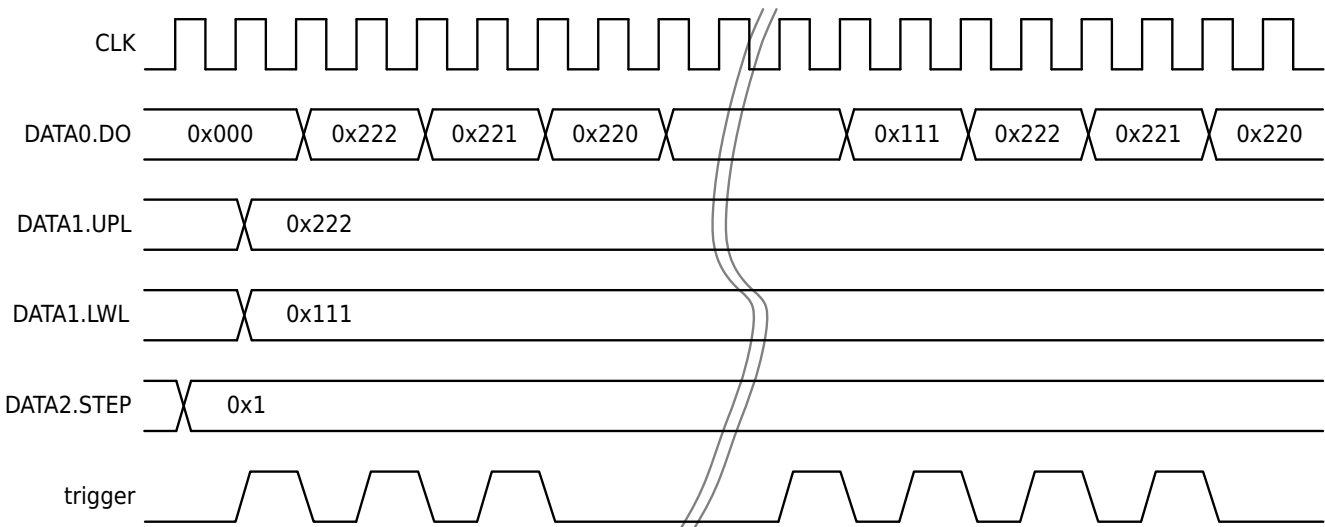


图 44-7 递减锯齿波输出模式

44.3.8 中断和事件

DCU 具有多种中断和用于触发启动其他外围电路的事件输出供用户选择。中断和事件输出的控制由中断和事件寄存器 (DCU_INTEVTSEL) 控制, 当需要输出事件信号时, 用户需要将中断和事件寄存器 (DCU_INTEVTSEL) 的相应控制位置为有效。每个 DCU 单元各输出一个 DCU 事件信号, 在事件列表中分别为 DCU1~DCU8。当需要对应事件在发生时产生中断, 用户需要将中断和事件寄存器 (DCU_INTEVTSEL) 的相应控制位置为有效的同时, 将控制寄存器 (DCU_CTL) 的 INTEN 位置 1。每个 DCU 单元各输出一个 DCU 中断信号, 在中断列表中分别为 DCU1~DCU8。

44.4 寄存器

44.4.1 寄存器总表

- DCU1 基地址：0x40056000
- DCU2 基地址：0x40056400
- DCU3 基地址：0x40056800
- DCU4 基地址：0x40056C00
- DCU5 基地址：0x40057000
- DCU6 基地址：0x40057400
- DCU7 基地址：0x40057800
- DCU8 基地址：0x40057C00

表 44-1 数据计算单元 (DCU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x00	DCU_CTL	DCU 控制寄存器	32
0x04	DCU_FLAG	DCU 标志寄存器 (DCU1~DCU4)	32
0x04	DCU_FLAG	DCU 标志寄存器 (DCU5~DCU8)	32
0x08+0x04*x	DCU_DATAx	DCU 数据寄存器 x (x=0~2)	32
0x14	DCU_FLAGCLR	DCU 标志复位寄存器 (DCU1~DCU4)	32
0x14	DCU_FLAGCLR	DCU 标志复位寄存器 (DCU5~DCU8)	32
0x18	DCU_INTEVTSEL	DCU 中断和事件寄存器 (DCU1~DCU4)	32
0x18	DCU_INTEVTSEL	DCU 中断和事件寄存器 (DCU5~DCU8)	32

44.4.2 DCU 控制寄存器 (DCU_CTL)

该寄存器用来配置 DCU 的动作模式。

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80000000																															
Name	INTEN	Reserved																					COMPTRG	Reserved		DATASIZE	MODE					
Access	RW																						RW			RW	RW					

位/位域	标记	位名	功能描述	读写
31	INTEN	中断使能	0b0: 禁止 0b1: 允许	RW
30:9	Reserved	保留	-	-
8	COMPTRG	比较模式触发比较的时机	0b0: 写入 DATA0 后比较 0b1: 写入 DATA0 或 DATA1 或 DATA2 后比较	RW
7:6	Reserved	保留	-	-
5:4	DATASIZE	加/减法模式，比较模式数据大小	0b00: 8-bit 0b01: 16-bit 0b10: 32-bit 0b11: 禁止设置	RW
3:0	MODE	动作模式	0b0000: DCU 无效 0b0001: 加法模式，运算在 DATA1 寄存器写入数据后进行 0b0010: 减法模式，运算在 DATA1 寄存器写入数据后进行 0b0011: 硬件触发加法模式，由其他外围电路触发启动进行加法运算 0b0100: 硬件触发减法模式，由其他外围电路触发启动进行减法运算 0b0101: 比较模式 0b1000*: 三角波输出模式 0b1001*: 递增锯齿波输出模式 0b1010*: 递减锯齿波输出模式 - 其他: 禁止设置  说明 *: 仅 DCU1/DCU2/DCU3/DCU4 支持该功能。	RW

44.4.3 DCU 标志寄存器 (DCU_FLAG) (DCU1~DCU4)

该寄存器产生 DCU 的结果标识。

Offset	Bit Position																																									
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
Reset	0x00000000																																									
Name	Reserved																				RO	FLAG_TOP	RO	FLAG_BTM	RO	FLAG_RLD	Reserved		RO	FLAG_GT1	RO	FLAG_EQ1	RO	FLAG_LS1	RO	FLAG_GT2	RO	FLAG_EQ2	RO	FLAG_LS2	RO	FLAG_OP
Access																					RO		RO		RO				RO		RO		RO		RO		RO		RO		RO	

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	FLAG_TOP	三角波峰点	三角波输出模式下，当 DATA0[11:0]的值达到最大时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_TOP 位写 1 时清除	RO
10	FLAG_BTM	三角波谷点	三角波输出模式下，当 DATA0[11:0]的值达到最小时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_TOP 位写 1 时清除	RO
9	FLAG_RLD	锯齿波重载	递增锯齿波输出模式下，当 DATA0[11:0]的值达到最大时置位；递减锯齿波输出模式下，当 DATA0[11:0]的值达到最小时置位。 DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_RLD 位写 1 时清除	RO
8:7	Reserved	保留	-	-
6	FLAG_GT1	大于标志位 1	比较模式下，当 DATA0>DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_GT1 位写 1 时清除	RO
5	FLAG_EQ1	等于标志位 1	比较模式下，当 DATA0=DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_EQ1 位写 1 时清除	RO
4	FLAG_LS1	小于标志位 1	比较模式下，当 DATA0<DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_LS1 位写 1 时清除	RO
3	FLAG_GT2	大于标志位 2	比较模式下，当 DATA0>DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_GT2 位写 1 时清除	RO
2	FLAG_EQ2	等于标志位 2	比较模式下，当 DATA0=DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_EQ2 位写 1 时清除	RO
1	FLAG_LS2	小于标志位 2	比较模式下，当 DATA0<DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_LS2 位写 1 时清除	RO
0	FLAG_OP	运算标志位	加法/减法和触发加/触发减模式时，当加法产生上溢或减法产生下溢时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_OP 位写 1 时清除	RO

44.4.4 DCU 标志寄存器 (DCU_FLAG) (DCU5~DCU8)

该寄存器产生 DCU 的结果标识。

Offset	Bit Position																																
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																										FLAG_GT1	FLAG_EQ1	FLAG_LS1	FLAG_GT2	FLAG_EQ2	FLAG_LS2	FLAG_OP
Access																											RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6	FLAG_GT1	大于标志位 1	比较模式下，当 DATA0>DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_GT1 位写 1 时清除	RO
5	FLAG_EQ1	等于标志位 1	比较模式下，当 DATA0=DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_EQ1 位写 1 时清除	RO
4	FLAG_LS1	小于标志位 1	比较模式下，当 DATA0<DATA1 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_LS1 位写 1 时清除	RO
3	FLAG_GT2	大于标志位 2	比较模式下，当 DATA0>DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_GT2 位写 1 时清除	RO
2	FLAG_EQ2	等于标志位 2	比较模式下，当 DATA0=DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_EQ2 位写 1 时清除	RO
1	FLAG_LS2	小于标志位 2	比较模式下，当 DATA0<DATA2 时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_LS2 位写 1 时清除	RO
0	FLAG_OP	运算标志位	加法/减法和触发加/触发减模式时，当加法产生上溢或减法产生下溢时置位，DCU 标志复位寄存器 DCU_FLAGCLR 的 CLR_OP 位写 1 时清除	RO

44.4.5 DCU 数据寄存器（DCU_DATAx）（x=0~2）

各个数据寄存器在各模式下的功能如下：

模式功能	DATA0	DATA1	DATA2
加法模式	被加数/存放结果	加数	存放减半结果
触发加模式	被加数/存放结果	加数	存放减半结果
减法模式	被减数/存放结果	减数	存放减半结果
触发减模式	被减数/存放结果	减数	存放减半结果
比较模式	被比较对象	比较对象 1	比较对象 2
比较模式(窗口比较)	被比较对象	窗口上限	窗口下限
三角波输出模式*	输出数据寄存器	振幅设置寄存器	步长寄存器
递增锯齿波输出模式*	输出数据寄存器	振幅设置寄存器	步长寄存器
递减锯齿波输出模式*	输出数据寄存器	振幅设置寄存器	步长寄存器

说明

*: 仅 DCU1/DCU2/DCU3/DCU4 支持该功能。

偏移地址（Offset）：0x08+0x04*x

加/减法模式、触发加/减法模式、比较模式

该寄存器用来存放运算数据。

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DAT																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	DAT	运算数据	存放运算数据，实际使用的位数根据 DCU_CTL.DATASIZE 设置： ● 当 DCU_CTL.DATASIZE=0b00 时 DATA[7:0]为有效数据； ● 当 DCU_CTL.DATASIZE=0b01 时 DATA[15:0]为有效数据； ● 当 DCU_CTL.DATASIZE=0b10 时 DATA[31:0]为有效数据。	RW

三角波输出模式

寄存器 DCU_DATA0 作为输出数据寄存器，Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																DO															
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	DO	输出数据	输出到 DAC 的数据	RW

寄存器 DCU_DATA1 作为振幅设定寄存器，Bit16~Bit27 及 Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				UPL												Reserved				LWL											
Access					RW																RW											

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27:16	UPL	振幅上限	设定三角波振幅的上限	RW
15:12	Reserved	保留	-	-
11:0	LWL	振幅下限	设定三角波振幅的下限	RW

寄存器 DCU_DATA2 作为步长寄存器，Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																STEP															
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	STEP	步长设置	设定三角波递增和递减的步长值	RW

递增/递减锯齿波输出模式

寄存器 DCU_DATA0 作为输出数据寄存器，Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																DO															
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	DO	输出数据	输出到 DAC 的数据	RW

寄存器 DCU_DATA1 作为振幅设定寄存器，Bit16~Bit27 及 Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				UPL												Reserved				LWL											
Access					RW																RW											

位/位域	标记	位名	功能描述	读写
31:28	Reserved	保留	-	-
27:16	UPL	振幅上限	设定锯齿波振幅的上限	RW
15:12	Reserved	保留	-	-
11:0	LWL	振幅下限	设定锯齿波振幅的下限	RW

寄存器 DCU_DATA2 作为步长寄存器，Bit0~Bit11 为有效位。

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				STEP											
Access																																

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	STEP	步长设置	设定锯齿波递增或递减的步长值	RW

44.4.6 DCU 标志复位寄存器 (DCU_FLAGCLR) (DCU1~DCU4)

该寄存器用于清除 DCU 的结果标识。

Offset	Bit Position																																	
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																				CLR_TOP	CLR_BTM	CLR_RLD	Reserved				CLR_GT1	CLR_EQ1	CLR_LS1	CLR_GT2	CLR_EQ2	CLR_LS2	CLR_OP
Access																					WO	WO	WO					WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	CLR_TOP	清除三角波峰点标志位	写 1 清除 DCU_FLAG 的 FLAG_TOP 位，写 0 无效	WO
10	CLR_BTM	清除三角波谷点标志位	写 1 清除 DCU_FLAG 的 FLAG_BTM 位，写 0 无效	WO
9	CLR_RLD	清除锯齿波重载标志位	写 1 清除 DCU_FLAG 的 FLAG_RLD 位，写 0 无效	WO
8:7	Reserved	保留	-	-
6	CLR_GT1	清除大于标志位 1	写 1 清除 DCU_FLAG 的 FLAG_GT1 位，写 0 无效	WO
5	CLR_EQ1	清除等于标志位 1	写 1 清除 DCU_FLAG 的 FLAG_EQ1 位，写 0 无效	WO
4	CLR_LS1	清除小于标志位 1	写 1 清除 DCU_FLAG 的 FLAG_LS1 位，写 0 无效	WO
3	CLR_GT2	清除大于标志位 2	写 1 清除 DCU_FLAG 的 FLAG_GT2 位，写 0 无效	WO
2	CLR_EQ2	清除等于标志位 2	写 1 清除 DCU_FLAG 的 FLAG_EQ2 位，写 0 无效	WO
1	CLR_LS2	清除小于标志位 2	写 1 清除 DCU_FLAG 的 FLAG_LS2 位，写 0 无效	WO
0	CLR_OP	清除运算标志位	写 1 清除 DCU_FLAG 的 FLAG_OP 位，写 0 无效	WO

44.4.7 DCU 标志复位寄存器 (DCU_FLAGCLR) (DCU5~DCU8)

该寄存器用于清除 DCU 的结果标识。

Offset	Bit Position																																
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																										CLR_GT1	CLR_EQ1	CLR_LS1	CLR_GT2	CLR_EQ2	CLR_LS2	CLR_OP
Access																											WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
31:7	Reserved	保留	-	-
6	CLR_GT1	清除大于标志位1	写 1 清除 DCU_FLAG 的 FLAG_GT1 位；写 0 无效	WO
5	CLR_EQ1	清除等于标志位1	写 1 清除 DCU_FLAG 的 FLAG_EQ1 位；写 0 无效	WO
4	CLR_LS1	清除小于标志位1	写 1 清除 DCU_FLAG 的 FLAG_LS1 位；写 0 无效	WO
3	CLR_GT2	清除大于标志位2	写 1 清除 DCU_FLAG 的 FLAG_GT2 位；写 0 无效	WO
2	CLR_EQ2	清除等于标志位2	写 1 清除 DCU_FLAG 的 FLAG_EQ2 位；写 0 无效	WO
1	CLR_LS2	清除小于标志位2	写 1 清除 DCU_FLAG 的 FLAG_LS2 位；写 0 无效	WO
0	CLR_OP	清除运算标志位	写 1 清除 DCU_FLAG 的 FLAG_OP 位；写 0 无效	WO

44.4.8 DCU 中断和事件寄存器 (DCU_INTEVTSEL) (DCU1~DCU4)

该寄存器能够选择 DCU 在何种条件下产生中断和输出事件信号。

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Name	Reserved														RW		SEL_TOP		RW		SEL_BTM		RW		SEL_RLD		RW		SEL_WIN		RW		SEL_GT1		RW		SEL_EQ1		RW		SEL_LS1		RW		SEL_GT2		RW		SEL_EQ2		RW		SEL_LS2		RW		SEL_OP																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
Access																													RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW				RW			

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	SEL_TOP	三角波峰点条件选择	0b0: 在三角波和输出模式下当 DATA0[11:0]达到最大时不产生中断和输出事件信号 0b1: 在三角波输出模式下当 DATA0[11:0]达到最大时产生中断和输出事件信号	RW
10	SEL_BTM	三角波谷点条件选择	0b0: 在三角波输出模式下当 DATA0[11:0]达到最小值时不产生中断和输出事件信号 0b1: 在三角波输出模式下当 DATA0[11:0]达到最小值时产生中断和输出事件信号	RW
9	SEL_RLD	锯齿波重载条件选择	0b0: 在递增/递减锯齿波输出模式下不产生中断和输出事件信号 0b1: 在递增锯齿波输出模式下当 DATA0[11:0]达到最大值时产生中断和输出事件信号；在递减锯齿波输出模式下当 DATA0[11:0]达到最小值时产生中断和输出事件信号	RW
8:7	SEL_WIN	窗口比较条件选择	比较模式下满足 SEL_WIN 设定的窗口比较条件时产生中断和输出事件信号，在 SEL_WIN 设定有效时，满足其他比较条件时不会产生中断和输出事件信号： 0b00: 不产生窗口比较中断和输出事件信号，此设定下由该寄存器的 Bit1~Bit6 选择其他中断和事件信号产生条件 0b01: 当 DATA0 数据在窗口内时产生中断和输出事件信号，即 $DATA2 \leq DATA0 \leq DATA1$ 0b10: 当 DATA0 数据在窗口外时产生中断和输出事件信号，即 $DATA0 > DATA1$ 或 $DATA0 < DATA2$ 0b11: 比较模式下不产生任何中断和事件信号	RW
6	SEL_GT1	大于条件选择 1	0b0: 比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 > DATA1$ 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 > DATA1$ 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW

位/位域	标记	位名	功能描述	读写
5	SEL_EQ1	等于条件选择 1	0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA1 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA1 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
4	SEL_LS1	小于条件选择 1	0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA1 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA1 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
3	SEL_GT2	大于条件选择 2	0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0>DATA2 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0>DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
2	SEL_EQ2	等于条件选择 2	0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA2 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
1	SEL_LS2	小于条件选择 2	0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA2 时不产生中断和输出事件信号 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
0	SEL_OP	运算条件选择	0b0: 加法和减法模式下当运算结果上溢或下溢时不产生中断和输出事件信号 0b1: 加法和减法模式下当运算结果上溢或下溢时产生中断和输出事件信号	RW

44.4.9 DCU 中断和事件寄存器 (DCU_INTEVTSEL) (DCU5~DCU8)

该寄存器能够选择 DCU 在何种条件下产生中断和输出事件信号。

Offset	Bit Position																																							
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved																								RW	SEL_WIN	RW	SEL_GT1	RW	SEL_EQ1	RW	SEL_LS1	RW	SEL_GT2	RW	SEL_EQ2	RW	SEL_LS2	RW	SEL_OP
Access																									RW		RW		RW		RW		RW		RW		RW		RW	

位/位域	标记	位名	功能描述	读写
31:9	Reserved	保留	-	-
8:7	SEL_WIN	窗口比较条件选择	比较模式下满足 SEL_WIN 设定的窗口比较条件时产生中断和输出事件信号，在 SEL_WIN 设定有效时，满足其他比较条件时不会产生中断和输出事件信号： 0b00：不产生窗口比较中断和输出事件信号，此设定下由该寄存器的 Bit1~Bit6 选择其他中断和事件信号产生条件 0b01：当 DATA0 数据在窗口内时产生中断和输出事件信号，即 $DATA2 \leq DATA0 \leq DATA1$ 0b10：当 DATA0 数据在窗口外时产生中断和输出事件信号，即 $DATA0 > DATA1$ 或 $DATA0 < DATA2$ 0b11：比较模式下不产生任何中断和事件信号	RW
6	SEL_GT1	大于条件选择 1	0b0：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 > DATA1$ 时不产生中断和输出事件信号 0b1：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 > DATA1$ 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
5	SEL_EQ1	等于条件选择 1	0b0：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 = DATA1$ 时不产生中断和输出事件信号 0b1：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 = DATA1$ 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
4	SEL_LS1	小于条件选择 1	0b0：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 < DATA1$ 时不产生中断和输出事件信号 0b1：比较模式下并且 SEL_WIN=0b00 时，当 $DATA0 < DATA1$ 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW

位/位域	标记	位名	功能描述	读写
3	SEL_GT2	大于条件选择 2	● 0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0>DATA2 时不产生中断和输出事件信号 ● 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0>DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
2	SEL_EQ2	等于条件选择 2	● 0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA2 时不产生中断和输出事件信号 ● 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0=DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
1	SEL_LS2	小于条件选择 2	● 0b0: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA2 时不产生中断和输出事件信号 ● 0b1: 比较模式下并且 SEL_WIN=0b00 时, 当 DATA0<DATA2 时产生中断和输出事件信号  说明 SEL_WIN≠0b00 时该位设置无效。	RW
0	SEL_OP	运算条件选择	● 0b0: 加法和减法模式下当运算结果上溢或下溢时不产生中断和输出事件信号 ● 0b1: 加法和减法模式下当运算结果上溢或下溢时产生中断和输出事件信号	RW

45 数学运算单元 (MAU)

45.1 简介

数学运算单元 (MAU) 是一个内含开方运算和正弦运算两种运算类型的硬件加速运算模块，支持定点数的开方和正弦运算。正弦函数支持 $360^\circ/2^{12}$ 运算精度。

45.2 功能说明

45.2.1 开方运算

开方运算支持 32 位定点数输入，运算结果为 32 位定点数输出。根据输入数据的大小不同，运算执行周期会不同。

操作流程

中断无效 (MAU_CSR.INTEN=0) 时，开方运算的操作流程如下；中断有效时，用户根据该流程自定义加入中断处理过程：

- 步骤 1. 在数据输入寄存器 0 (MAU_DTR0) 中写入被开方数据；
- 步骤 2. 将控制状态寄存器的启动位置 1 (MAU_CSR.START=1)；
- 步骤 3. 运算开始后，控制状态寄存器的状态位 (MAU_CSR.BUSY) 会置位，直到运算完成，该位自动清零。期待 BUSY 位清零，表明运算完成；
- 步骤 4. 从结果输出寄存器 0 (MAU_RTR0) 中读取运算结果；
- 步骤 5. 重复以上步骤 1~4。

关于步骤 3，从 MAU_CSR.START 置位后，BUSY 位开始变高，在持续 8~16 个运算时钟周期后（根据输入数据的大小运算时间不等）自动变为 0。用户在使用开方运算模块时可根据此位的变化灵活判断当前的运算状态。

结果处理

开方运算的运算结果是 32 位数据输出，其最低位结果采用四舍五入方式得到。例如，当输入数据为 0x0000000C 时，其结果为 0x00000003；当输入数据为 0x0000000D 时，其结果为 0x00000004。

定点数处理

当输入数据为定点小数，且需要保持输入输出数据的格式统一时，用户可通过控制状态寄存器的移位控制位 (MAU_CSR.SHIFT) 来对输出数据进行相应的处理。

例如，当采用 Q14.2 格式进行定点数运算时，可设定 MAU_CSR.SHIFT=1，使输出结果也遵循 Q14.2 格式。定点数 6.25 (Q14.2 格式为 0x00000019) 在移位之前的输出结果位 5 (0x00000005)，移位 1 位之后的结果为 2.5 (Q14.2 格式为 0x0000000A)。



说明

当 MAU_DTR0 写入的被开方数据大于 0xFFFF0000 时，可设定 MAU_CSR.SHIFT 最大值为 15；当 MAU_DTR0 写入的被开方数据小于等于 0xFFFF0000 时，可设定 MAU_CSR.SHIFT 最大值为 16。

45.2.2 正弦运算

正弦运算支持 12 位数据输入，运算结果为 16 位有符号数输出。其中 12 位数据输入表示要运算的直角坐标系角度；16 位数据输出中，最高位表示结果的正负。

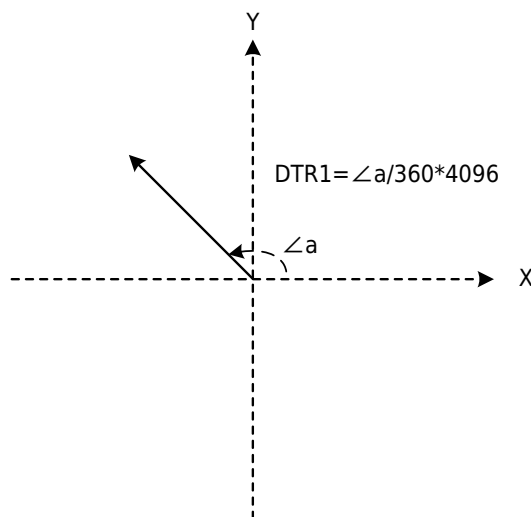


图 45-1 正弦运算角度值示意图

操作流程

正弦运算的操作流程如下：

- 步骤 1. 在数据输入寄存器 1 (MAU_DTR1) 中写入角度值；
- 步骤 2. 在 1 个周期之后从结果输出寄存器 1 (MAU_RTR1) 中读取运算结果；
- 步骤 3. 重复以上步骤 1~2。

45.2.3 中断及事件输出

中断输出

在每一次开方运算完成之后，都会有相应的运算完成事件产生。若使能中断控制位 (MAU_CSR.INTEN=1)，则对应的中断请求信号 (MAU_SQRT) 也会产生。

事件输出

在每一次开方运算完成之后，都会有相应的运算完成事件产生，相应的事件请求信号 (MAU_SQRT) 也会输出，可以用于选择触发其它模块。

45.3 寄存器

45.3.1 寄存器总表

基地址：0x40055000

表 45-1 数学运算单元 (MAU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	MAU_CSR	控制状态寄存器	32
0x0004	MAU_DTR0	数据输入寄存器 0	32
0x000C	MAU_RTR0	结果输出寄存器 0	32
0x0010	MAU_DTR1	数据输入寄存器 1	32
0x0014	MAU_RTR1	结果输出寄存器 1	32

45.3.2 控制状态寄存器 (MAU_CSR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SHIFT				Reserved				BUSY	Reserved	INTEN	START				
Access																	RW								RO		RW	RW				

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:8	SHIFT	开方结果移位	0b00000: 运算结果不移位输出 0b00001: 运算结果左移 1 位后输出 0b00010: 运算结果左移 2 位后输出 0b10000: 运算结果左移 16 位后输出 - 其他: 禁止设置	RW
7:4	Reserved	保留	-	-
3	BUSY	开方运算状态	0b0: 开方运算未开始或运算完成 0b1: 正在进行开方运算	RO
2	Reserved	保留	-	-
1	INTEN	开方运算中断使能	0b0: 开方运算完成后不产生中断 0b1: 开方运算完成后产生中断	RW
0	START	开方运算开始	写 1 开始开方运算 (内部产生一次 START 请求) , 写 0 无效; 读出恒为 0	RW

45.3.3 数据输入寄存器 0 (MAU_DTR0)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SQRT_DIN																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	SQRT_DIN	被开方数	被开方数输入	RW

45.3.4 结果输出寄存器 0 (MAU_RTR0)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SQRT_DOUT																															
Access	RW																															

位/位域	标记	位名	功能描述	读写
31:0	SQRT_DOUT	开方结果	开方结果输出	RW

45.3.5 数据输入寄存器 1 (MAU_DTR1)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SIN_DIN															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11:0	SIN_DIN	角度设定	直角坐标系角度输入	RW

45.3.6 结果输出寄存器 1 (MAU_RTR1)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SIN_DOUT															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	SIN_DOUT	正弦运算结果	正弦运算结果输出	RW

46 滤波数学加速器 (FMAC)

46.1 简介

滤波数学加速器 (FMAC) 是一个 FIR 滤波计算的硬件加速模块。该模块可进行最大 16 阶, 且阶数可配置的 FIR 数字滤波。内置 16x16bits 乘法器、32+5bits 加法器, 用户可自定义输出数据精度。本系列产品搭载 4 个 FMAC 模块。

46.2 主要特性

- 最大 16 阶 FIR 滤波, 阶数可配置
- 滤波系数可配置
- 16 位有符号数输入、输出数据, 16 位有符号数滤波系数
- 16x16bits 乘法器
- 32+5bits 加法器
- 提供中断信号和状态查询
- AHB slave 接口配置参数和输入输出数据

46.3 功能说明

46.3.1 基本框图

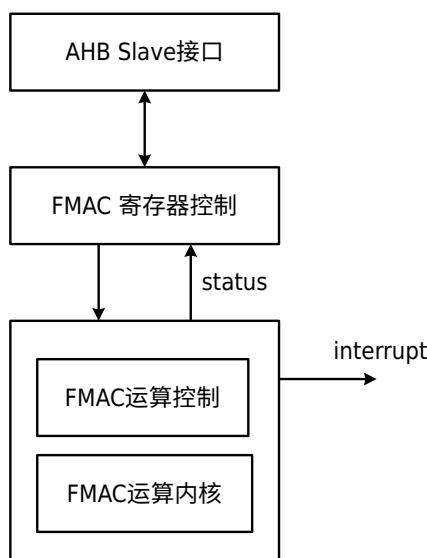


图 46-1 FMAC 基本框图

在选择 16 阶 FIR 滤波时, FMAC 运算控制和运算内核需要 17 个时钟周期处理一个数据, 在 240MHz 的工作时钟频率下, 可以处理最大 14.1MHz 的输入数据流。不同的阶数, 处理所需的时间不同。阶数越小, 处理速度越快。最大输入数据频率与阶数和工作时钟频率的关系如下:

最大输入数据频率 (MHz) = 工作时钟频率 (MHz) ÷ (阶数+1)

46.3.2 操作流程

本模块需要按照以下步骤来使用：

- 步骤 1. 将使能位 (FMAC_ENR.FMACEN) 置为 1；
- 步骤 2. 配置滤波器阶数 (FMAC_CTR.STAGE_NUM[4:0]) 及输出有效结果移位数 (FMAC_CTR.SHIFT[4:0])；
- 步骤 3. 如果需要使用中断，则配置中断使能位 (FMAC_IER.INTEN) 为 1；
- 步骤 4. 写入滤波器系数 (有 N 阶，就只需配置 FMAC_COR0~FMAC_CORN)；
- 步骤 5. 写入待处理数据 (FMAC_DTR)；
- 步骤 6. 等待完成中断或者查询完成状态 (FMAC_STR.READY) 为 1；
- 步骤 7. 读取结果 (FAMC_RTR0 或 FAMC_RTR1)；
- 步骤 8. 重复步骤 5~7，直到所有数据处理完。

46.3.3 模块使能

模块使能位 (FMAC_ENR.FMACEN) 的作用有两个，作用一是使能整个模块，只有其为 1 时，模块才允许工作；作用二是模块的软复位作用，当某种阶数和系数的配置使用完以后，需要换另一种阶数和系数的配置，那么需要将 FMACEN 置为 0，将前面计算残留的信息清零，再按照操作流程来操作。

FMACEN 清 0 时，不会将配置参数清零，只会将内部寄存器和中间结果清零。

46.3.4 系数归一化

如果输出结果只取 16 位，那么在计算过程中以及计算结果，总共截去的位数多达 20 位，所以如果输入的参与计算的数据太小，则输出结果有可能为 0，或者很小的数（损失了很大的精度）。所以滤波器系数有必要做归一化，即将其中的最大值（如果是正数）归一化成 32767（如果是负数，归一化成-32768），其他系数乘以相同的比例。

例如，有 5 个系数为 10, 13, 35, 96, 42，要将其归一化。首先找出其中的最大值，为 96，然后用 32767 除以该最大值， $32767/96=341.3$ ，取值 341，即为归一化的倍数，其他系数都要乘以该倍数，即为归一化以后的数。

所以归一化以后的系数为：3410, 4433, 11935, 32736, 14322。

46.3.5 中断和事件

46.3.5.1 中断输出

在每一次运算完成之后，都会有相应的运算完成事件产生。若使能中断控制位 (FMAC_IER.INTEN=1)，则对应的中断请求信号 (FMAC_m_FIR, m=1~4) 也会产生。

46.3.5.2 事件输出

在每一次运算完成之后，都会有相应的运算完成事件产生，相应的事件请求信号 (FMAC_m_FIR, m=1~4) 也会输出，可以用于选择触发其它模块。

46.4 寄存器

所有寄存器写完之后不能立即读取，写入寄存器时需要两个 HCLK 周期才能加载到寄存器。

46.4.1 寄存器总表

- FMAC1 基地址：0x40058000
- FMAC2 基地址：0x40058400
- FMAC3 基地址：0x40058800
- FMAC4 基地址：0x40058C00

表 46-1 滤波数学加速器 (FMAC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0000	FMAC_ENR	模块使能寄存器	32
0x0004	FMAC_CTR	基本控制寄存器	32
0x0008	FMAC_IER	中断控制寄存器	32
0x000C	FMAC_DTR	数据输入寄存器	32
0x0020+0x04*n	FMAC_CORn	滤波系数寄存器 n (n=0~16)	32
0x0010	FMAC_RTR0	结果输出寄存器 0	32
0x0014	FMAC_RTR1	结果输出寄存器 1	32
0x0018	FMAC_STR	运算状态寄存器	32

46.4.2 模块使能寄存器 (FMAC_ENR)

Offset	Bit Position																															
0x0000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															FMACEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	FMACEN	模块使能	0b0: FMAC 无效 0b1: FMAC 使能 说明 该信号作为本模块的数据输入复位信号使用，即使能后需要改变滤波阶数 FMAC_DTR 或滤波系数 FMAC_COR0~16 的时候，需要先将该位信号置为 0，再将该位置为 1。	RW

46.4.3 基本控制寄存器 (FMAC_CTRL)

Offset	Bit Position																															
0x0004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000010																															
Name	Reserved																SHIFT				Reserved				STAGE_NUM							
Access																	RW								RW							

位/位域	标记	位名	功能描述	读写
31:13	Reserved	保留	-	-
12:8	SHIFT	滤波结果移位	滤波结果右移位数，控制 MAU_RTR1 输出的结果： ● 0b00000：不移位，即选取运算结果的[31:0]给 MAU_RTR1 ● 0b00001：右移 1 位，即选取运算结果的[32:1]给 MAU_RTR1 ● 0b00010：右移 2 位，即选取运算结果的[33:2]给 MAU_RTR1 ● ● 0b00101：右移 5 位，即选取运算结果的[36:5]给 MAU_RTR1 ● 0b00110：右移 6 位，即选取运算结果的[36:6]给 MAU_RTR1，高位移入符号位 ● 0b00111：右移 7 位，即选取运算结果的[36:7]给 MAU_RTR1，高位移入符号位 ● ● 0b10100：右移 20 位，即选取运算结果的[36:20]给 MAU_RTR1，高位移入符号位 ● 0b10101：右移 21 位，即选取运算结果的[36:21]给 MAU_RTR1，高位移入符号位 ● 其他：不移位	RW
7:5	Reserved	保留	-	-
4:0	STAGE_NUM	滤波阶数设定	设定滤波器的滤波阶数	RW

46.4.4 中断控制寄存器 (FMAC_IER)

Offset	Bit Position																															
0x0008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															INTEN
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	INTEN	中断使能	0b0: 运算完成后不产生中断 0b1: 开方运算完成后产生中断	RW

46.4.5 数据输入寄存器 (FMAC_DTR)

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																DIN															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	DIN	数据输入	FMAC 数据输入	RW

46.4.6 滤波系数寄存器 n (FMAC_CORn) (n=0~16)

偏移地址 (Offset) : 0x0020+0x04*n

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CIN															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CIN	滤波系数	滤波系数输入	RW

46.4.7 结果输出寄存器 0 (FMAC_RTR0)

Offset	Bit Position																															
0x0010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DOUT0																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DOUT0	结果输出	累加器结果的高 32 位结果输出	RO

46.4.8 结果输出寄存器 1 (FMAC_RTR1)

Offset	Bit Position																															
0x0014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	DOUT1																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	DOUT1	结果输出	累加器结果的低 32 位结果输出	RO

46.4.9 运算状态寄存器 (FMAC_STR)

Offset	Bit Position																															
0x0018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RC READY	Reserved																														
Access	RC																															

位/位域	标记	位名	功能描述	读写
31	READY	运算完成位	● 0b0: 计算未完成, 结果不可用 ● 0b1: 计算完成, 结果可用	RC
30:0	Reserved	保留	-	-

47 错误管理单元 (ERMU)

47.1 简介

ERMU 收集来自本产品安全机制的错误通知。每个错误通知可以配置为三种响应行为：系统中断、系统复位和输出至错误输出端口，用以通知系统进行进一步错误处理操作。

47.2 主要特性

- 每个错误通知可配置为三种行为：系统中断、系统复位和输出至错误输出端口
- 错误输出可从以下三种类型进行配置
 - ▶ 单 pin 输出
 - ▶ 单 pin 翻转输出
 - ▶ 双 pin 互补输出
- 最多支持 4 组错误输出
- 支持伪错误源触发，可用于诊断和调试
- 支持错误中断处理的响应超时功能

47.3 功能说明

47.3.1 功能框图

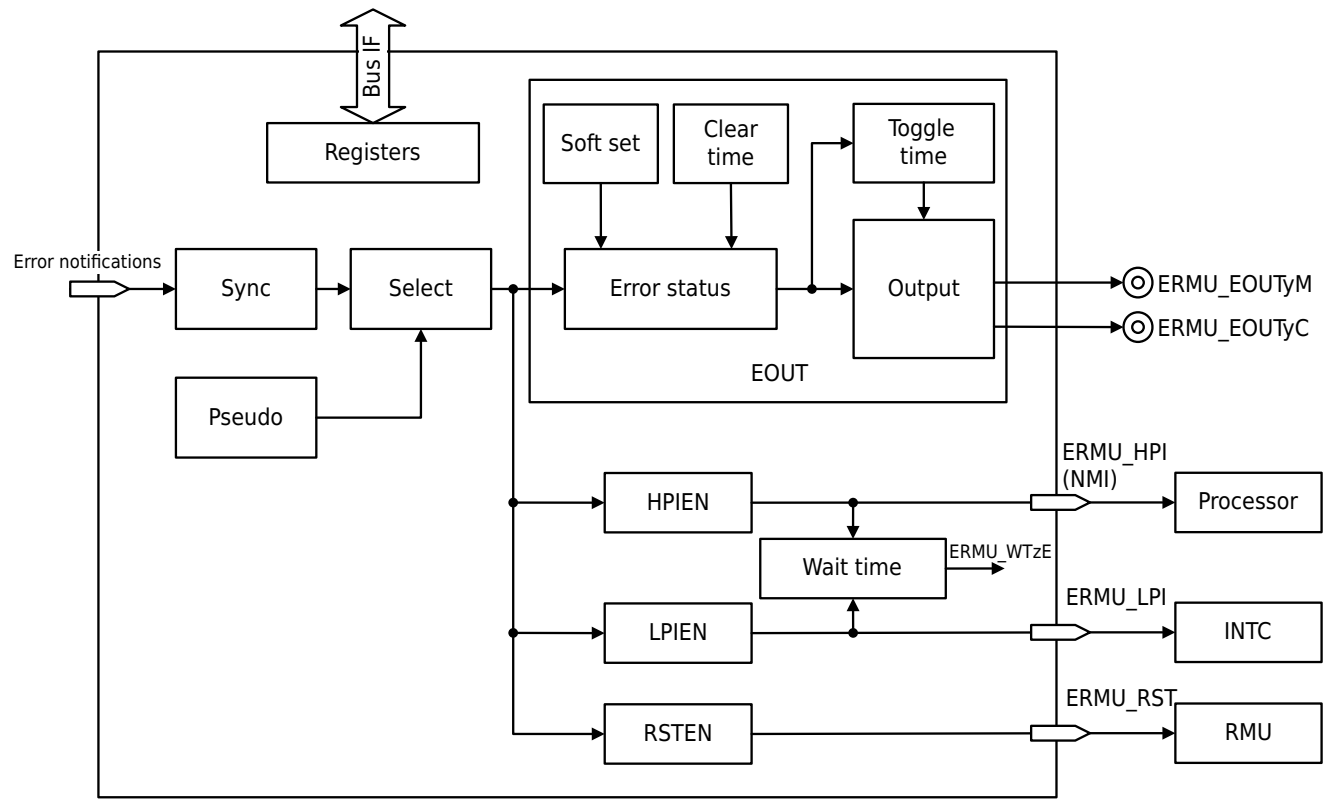


图 47-1 ERMU 框图

表 47-1 ERMU 参数定义表

参数名称	取值	功能说明
y	0~3	错误输出通道分组由编号 'y' 标识
z	0~3	单个等待定时器由编号 'z' 标识
j	0~1	错误源分组由编号 'j' 标识
k	0~31	单个错误源在每组的编号 'k' 标识
i	0~63	单个错误源 ID 由编号 'i' 标识

 **说明**
该表内容适用于 ERMU 所有章节内容。

表 47-2 ERMU 输入/输出端口

端口名称	端口类型	功能说明
ERMU_EOUTyM ¹⁾	输出	错误输出正向端口
ERMU_EOUTyC	输出	错误输出反向端口，与错误输出正向端口的输出反相



说明

- 1. 其中 ERMU_EOUT0M 可通过 ICG8 配置 ERMU_EOUT0M 默认输出引脚。
- 2. 除 1 外，其他错误输出端口可通过端口配置寄存器选择相应功能。

47.3.2 错误输出通道

47.3.2.1 错误输出操作

复位释放后，默认 ERMU_EOyS.EOS 为 1，使用 ERMU 前，需要写 ERMU_EOyC.CLR 清除 ERMU_EOyS.EOS，输出端口 ERMU_EOUTyM 和 ERMU_EOUTyC 在复位期间和复位后的输出电平行为请参见[错误输出说明](#)。

除了 ERMU 的输出 ERMU_EOUT0M 可通过 ICG8 配置为默认输出引脚之外，其他 ERMU 的输出 ERMU_EOUTyM 和 ERMU_EOUTyC，在使用 ERMU 之前，要把相应端口输出功能先设置为 ERMU_EOUTyM 和 ERMU_EOUTyC，具体配置请参见《通用 IO (GPIO) 》章节。

ERMU_EOyS.EOS 状态改变时，ERMU 的输出 ERMU_EOUTyM 和 ERMU_EOUTyC 电平也会做相应改变，具体见下表。

表 47-3 错误输出操作

错误源状态 ERMU_ESSj.ESSk	翻转定时器启用	ERMU_EOUTyM 输出电平 ⁽¹⁾	ERMU_EOUTyC 输出电平 ⁽¹⁾	错误输出状态 ERMU_EOyS.EOS
0	0(禁用)	高	低	0(无错误)
	1(启用)	翻转	翻转	0(无错误)
1	0(禁用)	低	高	1(有错误)
	1(启用)	低	高 ⁽²⁾	1(有错误)



说明

- 1. 从错误源到错误端口的输出可以通过 ERMU_EOyOMj 进行屏蔽，ERMU_EOUT0M 可通过 ICG8 配置为默认输出引脚，其他输出 ERMU_EOUTyM 和 ERMU_EOUTyC 输出端口，软件设置端口配置寄存器选择 ERMU 相应错误输出功能，才能作为错误输出端口。
- 2. 假设在启用翻转定时器时，配置了 ERMU_EOUTyC 为端口输出功能。

47.3.2.2 错误输出说明

ERMU_EOUTyM 和 ERMU_EOUTyC 在复位期间和复位后的行为参见下表。

表 47-4 ERMU_EOUT0M 输出情况说明

类别	复位类别			
	上电/掉电复位	系统复位	模块复位	JTAG 复位
复位时的输出电平	Hi-Z	Hi-Z	保持复位前输出	保持复位前输出
复位后的输出电平	低电平	低电平	保持复位前输出	保持复位前输出

表 47-5 ERMU_EOUTyM (y=1~3)和 ERMU_EOUTyC (y=0~3)输出端口说明

类别	复位类别			
	上电/掉电复位	系统复位	模块复位	JTAG 复位
复位时的输出电平	Hi-Z	Hi-Z	保持复位前输出	保持复位前输出
复位后的输出电平	通用 IO ⁽¹⁾	通用 IO ⁽¹⁾	保持复位前输出	保持复位前输出

 **说明**

1. ERMU_EOUTyM (y=1~3)和 ERMU_EOUTyC (y=0~3)输出端口复位解除时是普通通用 IO 功能，需要软件设置端口配置寄存器选择 ERMU 相应错误输出功能，才能作为错误输出端口。

47.3.2.3 错误输出状态由软件控制

设置错误输出状态

按照以下步骤通过软件方式设置错误输出状态：

- 1. 设置 ERMU_EOyC.SET 为 1
- 2. 等待 ERMU_EOyS.EOS 变为 1

清除错误输出状态

按照以下步骤清除错误输出状态：

- 1. 确认所有未被 ERMU_EOyOMj 屏蔽的 ERMU_ESSj.ESSk 均为 0
- 2. 等待 ERMU_EOyS.CTS 变为 0
- 3. 设置 ERMU_EOyC.CLR 为 1
- 4. 等待 ERMU_EOyS.EOS 变为 0

47.3.2.4 翻转定时器

翻转定时器计数时钟是 MRC(8MHz)经过预分频之后的时钟，如果 ERMU_EOyTTC.TTE=1，翻转定时器从 0 开始计数，当计数器值匹配到 ERMU_EOyTTCMP 时，从 0 开始重新计数。当计数器值匹配时，翻转输出信号 ERMU_EOUTyM 和 ERMU_EOUTyC。

47.3.2.5 启用翻转定时器

按照以下步骤启用翻转定时器：

- 1. 设置 ERMU_EOyTTCMP 来定义翻转周期
- 2. 设置 ERMU_EOyC.CLR 为 1 清除错误输出状态
- 3. 等待 ERMU_EOyS.EOS 变为 0
- 4. 设置 ERMU_EOyTTC.TTE 为 1

47.3.2.6 禁用翻转定时器

按照以下步骤禁用翻转定时器：

- 1. 设置 ERMU_EOyC.SET 为 1，设置错误输出状态
- 2. 等待 ERMU_EOyS.EOS 变为 1
- 3. 设置 ERMU_EOyTTC.TTE 为 0

47.3.2.7 清除定时器

使用清除定时器，可以将 ERMU 输出端口输出的错误状态保持有效一定时间周期，该时间周期可通过 ERMU_EOyCTCMP 进行配置。

清除定时器计数时钟是 MRC(8MHz)经过预分频之后的时钟，设置 ERMU_EOyCTCMP 非零值，同时 ERMU_EOyC.CTE=1，即可使能清除定时器，当有效错误源产生时，计数器开始计数；注意，即使在清除定

时器计数过程中，当出现新的有效错误源时，计数器将从 0 开始计数，当计数器值与 ERMU_EOyCTCMP 匹配时，清除定时器停止计数。当 ERMU_EOyS.CTS=1 时，对 ERMU_EOyC.CLR 写 1 将被忽略。

47.3.3 等待定时器

使用等待定时器，可以检测 CPU 处理 ERMU 中断（高优先级中断或低优先级中断）的时间，一旦检测超时，则可进一步请求端口输出错误状态或请求系统复位。

等待定时器计数时钟是 MRC（8MHz）经过预分频之后的时钟，设置 ERMU_WTzCMP 非零值，同时 ERMU_WTzC.WTE=1，即可使能等待定时器，当由 ERMU_WTzSE 设置的启动中断有效时，计数器从 0 开始计数。在等待定时器计数过程中，新中断产生不影响等待定时器计数。当计数器值与 ERMU_WTzCMP 匹配时，等待定时器停止计数，并产生对应等待定时器超时错误信号 ERMU_WTzE。

ERMU_WTzE 可用于配置为错误输出、中断或请求复位。



注意

要避免同一等待定时器的超时事件循环自触发，这种情况可描述为当前阶段等待定时器超时事件发生时（即 ERMU_WTzE 有效），且该信号又被配置为 ERMU 等待定时器的启动中断，触发等待定时器开始下一阶段的计数，然后就这样超时事件循环自触发。建议将 ERMU_WTzE 设置为不同的错误通知类型，例如，当 ERMU_WTzE 的触发源为 ERMU_LPI 时，设置 ERMU_WTzE 事件为高优先级中断 ERMU_HPI；再比如，当 ERMU_WTzE 的触发源为 ERMU_HPI 时，设置 ERMU_WTzE 事件为复位 ERMU_RST 或 EOUTyM。

47.3.4 复位请求

当 ERMU_REj 中使能的错误源发生错误时，ERMU 向复位控制模块 RMU 发出复位请求，最终产生系统复位。复位时序及复位标志，请参见《复位控制 (RMU)》章节。

47.3.5 中断请求

当 ERMU_HPIEj 或 ERMU_LPIEj 使能的错误源发生错误时，ERMU 发出 ERMU_HPI 请求或 ERMU_LPI 请求，作为系统不可屏蔽中断 (NMI) 的 ERMU_HPI 将直接发生系统中断，而 ERMU_LPI 需要同时设置 INTC 中对应的使能寄存器，才能发生有效的系统中断，具体请参见《中断控制器 (INTC)》章节。

47.3.6 伪错误触发器

伪错误源触发 ERMU_PETj.PETk 作为测试或调试手段，可通过软件方式注入错误源错误行为，伪错误源错误行为在 ERMU 模块中的处理与真实错误源发生错误时的处理是基本一致。

47.3.7 中断

ERMU 支持从不同的事件产生中断请求，具体信息如下：

中断事件	中断标志	中断使能	说明
ERMU 高优先级中断	ERMU_HPI	寄存器 ERMU_HPIEj	用作 NMI 中断，直接输出给处理器
ERMU 低优先级中断	ERMU_LPI	寄存器 ERMU_LPIEj	普通中断，经 INTC 模块输出给处理器

47.3.8 错误源

ERMU 支持从不同的错误源请求，具体信息如下：

表 47-7 ERMU 错误源列表

ID	模块	名称	描述
0	ERMU	ERMU_WT0E	ERMU 等待定时器 0 错误
1	ERMU	ERMU_WT1E	ERMU 等待定时器 1 错误
2	ERMU	ERMU_WT2E	ERMU 等待定时器 2 错误
3	ERMU	ERMU_WT3E	ERMU 等待定时器 3 错误
4~7	-	-	保留
8	SWDT	SWDT_ERR	SWDT 超时溢出错误
9	WDT	WDT_ERR	WDT 超时溢出错误
10	-	-	保留
11	EFM	FLASH_CERR	FLASH ECC 可纠正错误
12	EFM	FLASH_FERR	FLASH ECC 不可纠正错误
13	EFM	FLASH_OERR	FLASH ECC 溢出错误
14	SRAMH	SRAMH_CERR	SRAMH ECC 可纠正错误
15	SRAM1	SRAM1_CERR	SRAM1 ECC 可纠正错误
16	SRAM2	SRAM2_CERR	SRAM2 ECC 可纠正错误
17	SRAM3	SRAM3_CERR	SRAM3 ECC 可纠正错误
18	SRAM4	SRAM4_CERR	SRAM4 ECC 可纠正错误
19	SRAMB	SRAMB_CERR	SRAMB ECC 可纠正错误
20~22	-	-	保留
23	SRAMH	SRAMH_FERR	SRAMH ECC 不可纠正错误
24	SRAM1	SRAM1_FERR	SRAM1 ECC 不可纠正错误
25	SRAM2	SRAM2_FERR	SRAM2 ECC 不可纠正错误
26	SRAM3	SRAM3_FERR	SRAM3 ECC 不可纠正错误
27	SRAM4	SRAM4_FERR	SRAM4 ECC 不可纠正错误
28	SRAMB	SRAMB_FERR	SRAMB ECC 不可纠正错误
29-31	-	-	保留
32	CACHE	CACHE_CERR	Cache RAM ECC 可纠正错误
33	CACHE	CACHE_FERR	Cache RAM ECC 不可纠正错误
34~35	-	-	保留
36	CAN1	CREC1_CERR	CAN1 RAM ECC 可纠正错误

ID	模块	名称	描述
37	CAN2	CREC2_CERR	CAN2 RAM ECC 可纠正错误
38	MCAN	MCREC_CERR	MCAN RAM ECC 可纠正错误
39~41	-	-	保留
42	CAN1	CREC1_FERR	CAN1 RAM ECC 不可纠正错误
43	CAN2	CREC2_FERR	CAN2 RAM ECC 不可纠正错误
44	MCAN	MCREC_FERR	MCAN RAM ECC 不可纠正错误
45~47	-	-	保留
48	PKE_LP	PKERAM_CERR	PKE RAM ECC 可纠正错误
49	PKE_LP	PKERAM_FERR	PKE RAM ECC 不可纠正错误
50	-	-	保留
51	MPU	MPU_SMPU1ERR	MPU SMPU 1 总线错误
52	MPU	MPU_SMPU2ERR	MPU SMPU 2 总线错误
53	MPU	MPU_FMPUERR	MPU FMPU 总线错误
54	MPU	MPU_HMPUERR	MPU HMPU 总线错误
55	MPU	MPU_EMPUERR	MPU EMPU 总线错误
56	-	-	保留
57	DMA	DMA1_ERR	DMA1 传输错误
58	DMA	DMA2_ERR	DMA2 传输错误
59	-	-	保留
60	INTC	FPU_ERR ⁽¹⁾	FPU 计算错误
61	FCM	FCM_ERR	时钟检测错误
62~63	-	-	保留



说明

1. 需要先使能 FPU 相关错误中断，对应错误产生时才能输出至 ERMU。

47.4 操作示例

47.4.1 ERMU 错误管理示例

操作步骤

- 步骤 1. 读取并检测 ERMU_ESSj 中的每一位，确认是否有错误；
- 步骤 2. 若有错误，直接进入系统错误处理程序，若无错误则进入后边流程；
- 步骤 3. ERMU_EOyC.CLR 写入 1；
- 步骤 4. 读取并检测确认 ERMU_EOyS.EOS 直至为 0，若一直不为 0 则转至 Step1；
- 步骤 5. 根据需要设置端口功能寄存器，选择 ERMU_EOUTyM、ERMU_EOUTyC 输出功能；
- 步骤 6. 根据需要设置好 ERMU_EOyOMj, ERMU_HPIEj, ERMU_LPIEj, ERMU_REj；
- 步骤 7. 根据需要设置好 ERMU_EOyCTCMP, ERMU_EOyC.CTE；
- 步骤 8. 根据需要设置好 ERMU_EOyTTCMP, ERMU_EOyTTC.TTE；
- 步骤 9. 根据需要设置好 ERMU_WTzCMP, ERMU_WTzSE, ERMU_WTzC.WTE；
- 步骤 10. 设置对 ERMU 总线访问权限，取消写权限，仅保留读权限；
- 步骤 11. 根据需要配置好 ERMU_HPI、ERMU_LPI 中断发生时的中断处理函数。

47.5 寄存器

47.5.1 寄存器总表

基地址：0x40048800

表 47-8 错误管理单元 (ERMU) 寄存器偏移地址

偏移地址	寄存器	描述	位宽
0x0080*y	ERMU_EOyC	错误输出 y 控制寄存器 (y=0~3)	32
0x0004+0x0080*y	ERMU_EOyS	错误输出 y 状态寄存器 (y=0~3)	32
0x0008+0x0080*y	ERMU_EOyCTCNT	错误输出 y 清除定时器计数寄存器 (y=0~3)	32
0x000C+0x0080*y	ERMU_EOyCTCMP	错误输出 y 清除定时器比较寄存器 (y=0~3)	32
0x0010+0x0080*y	ERMU_EOyTTC	错误输出 y 翻转定时器控制寄存器 (y=0~3)	32
0x0018+0x0080*y	ERMU_EOyTTCMP	错误输出 y 翻转定时器比较寄存器 (y=0~3)	32
0x0040+0x0080*y	ERMU_EOyOM0	错误输出 y 输出屏蔽 0 寄存器 (y=0~3)	32
0x0044+0x0080*y	ERMU_EOyOM1	错误输出 y 输出屏蔽 1 寄存器 (y=0~3)	32
0x0200+0x0020*z	ERMU_WTzC	等待定时器 z 控制寄存器 (z=0~3)	32
0x0204+0x0020*z	ERMU_WTzS	等待定时器 z 状态寄存器 (z=0~3)	32
0x0208+0x0020*z	ERMU_WTzCNT	等待定时器 z 计数寄存器 (z=0~3)	32
0x020C+0x0020*z	ERMU_WTzCMP	等待定时器 z 比较寄存器 (z=0~3)	32
0x0210+0x0020*z	ERMU_WTzSE	等待定时器 z 启动使能寄存器 (z=0~3)	32
0x0300+0x0004*j	ERMU_ESSj	错误源状态 j 寄存器 (j=0~1)	32
0x0340+0x0004*j	ERMU_ESSCj	错误源状态清除 j 寄存器 (j=0~1)	32
0x0380+0x0004*j	ERMU_PETj	伪错误源触发 j 寄存器 (j=0~1)	32
0x03C0+0x0004*j	ERMU_REj	复位使能 j 寄存器 (j=0~1)	32
0x0400+0x0004*j	ERMU_LPIEj	低优先级中断使能 j 寄存器 (j=0~1)	32
0x0500+0x0004*j	ERMU_HPIEj	高优先级中断使能 j 寄存器 (j=0~1)	32
0x0600	ERMU_CCPS	计数时钟预分频寄存器	32



说明

对 ERMU 寄存器的写访问仅支持 word 写方式。

47.5.2 错误输出 y 控制寄存器 (ERMU_EOyC) (y=0~3)

Offset	Bit Position																																		
0x0080*y	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000000																																		
Name	Reserved																	Reserved																	
																	CTE																	CLR	SET
Access																	RW																	WO	WO

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	CTE	清除定时器使能	0b0: 关闭清除定时器 0b1: 开启清除定时器	RW
15:2	Reserved	保留	-	-
1	CLR	错误输出状态清除	写 1 清除错误输出状态 ERMU_EoyS.EOS, 写 0 无效; 读出恒为 0  说明 - 当未被 ERMU_EOyOMj 屏蔽的 ERMU_ESSj.ESSk 不为 0 时, 向该位写入 1 将被忽略。 - 当 ERMU_EOyS.CTS=1 时, 向该位写入 1 将被忽略。	WO
0	SET	设置错误输出状态	写 1 置位错误输出状态 ERMU_EoyS.EOS, 写 0 无效; 读出恒为 0	WO

47.5.3 错误输出 y 状态寄存器 (ERMU_EOyS) (y=0~3)

偏移地址 (Offset) : 0x0004+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved															CTS	Reserved															EOS
Access																																

位/位域	标记	位名	功能描述	读写
31:17	Reserved	保留	-	-
16	CTS	清除定时器状态	0b0: 清除定时器计数停止 0b1: 清除定时器正在计数	RO
15:1	Reserved	保留	-	-
0	EOS	设置错误输出状态	0b0: 无错误输出 0b1: 有错误输出	RO

47.5.4 错误输出 y 清除定时器计数寄存器 (ERMU_EOyCTCNT) (y=0~3)

偏移地址 (Offset) : 0x0008+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																RO CNT															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNT	清除定时器计数值	当内核停止时, 清除定时器不会停止计数。	RO

47.5.5 错误输出 y 清除定时器比较寄存器 (ERMU_EOyCTCMP) (y=0~3)

偏移地址 (Offset) : 0x000C+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CMP															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CMP	清除定时器比较 值	ERMU_EOyC.CTE 为 0 时才能配置该值，当清除定时器计数值达到该比较值时，计数停止并自动清零 ERMU_EOyS.CTS	RW

47.5.6 错误输出 y 翻转定时器控制寄存器 (ERMU_EOyTTC) (y=0~3)

偏移地址 (Offset) : 0x0010+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															TTE
Access																																RW

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	TTE	翻转定时器使能	● 0b0: 关闭翻转定时器 ● 0b1: 开启翻转定时器	RW

47.5.7 错误输出 y 翻转定时器比较寄存器 (ERMU_EOyTTCMP) (y=0~3)

偏移地址 (Offset) : 0x0018+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CMP															
Access																																

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CMP	翻转定时器比较 值	ERMU_EOyC.TTE 为 0 时才能配置该值，当翻转定时器计数值达到该比较值时，计数器自动从零开始计数	RW

47.5.8 错误输出 y 屏蔽 0 寄存器 (ERMU_EOyOM0) (y=0~3)

偏移地址 (Offset) : 0x0040+0x0080*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OM31	OM30	OM29	OM28	OM27	OM26	OM25	OM24	OM23	OM22	OM21	OM20	OM19	OM18	OM17	OM16	OM15	OM14	OM13	OM12	OM11	OM10	OM9	OM8	OM7	OM6	OM5	OM4	OM3	OM2	OM1	OM0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	OMk (k=31~0)	输出屏蔽控制位	错误输出 y 输出屏蔽控制位，用以屏蔽错误源编号 i (i = k) ● 0b0：错误输出未被屏蔽 ● 0b1：错误输出被屏蔽	RW

47.5.9 错误输出 y 屏蔽 1 寄存器 (ERMU_EOyOM1) (y=0~3)

偏移地址 (Offset) : 0x0044+0x0080*y

Offset	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	OM31	OM30	OM29	OM28	OM27	OM26	OM25	OM24	OM23	OM22	OM21	OM20	OM19	OM18	OM17	OM16	OM15	OM14	OM13	OM12	OM11	OM10	OM9	OM8	OM7	OM6	OM5	OM4	OM3	OM2	OM1	OM0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	OMk (k=31~0)	输出屏蔽控制位	错误输出 y 输出屏蔽控制位，用以屏蔽错误源编号 i (i = 32+k) 0b0: 错误输出未被屏蔽 0b1: 错误输出被屏蔽	RW

47.5.10 等待定时器 z 控制寄存器 (ERMU_WTzC) (z=0~3)

偏移地址 (Offset) : 0x0200+0x0020*z

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved															STP	Reserved															WTE
Access																																

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
16	STP	等待定时器停止计数控制	写 1 等待定时器停止计数，并等待下一次有效的启动使能，写 0 无效；读出恒为 0	WO
15:1	Reserved	保留	-	-
0	WTE	等待定时器使能	0b0: 关闭等待定时器 0b1: 使能等待定时器	RW

47.5.11 等待定时器 z 状态寄存器 (ERMU_WTzS) (z=0~3)

偏移地址 (Offset) : 0x0204+0x0020*z

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															WTS
Access																																RO

位/位域	标记	位名	功能描述	读写
31:1	Reserved	保留	-	-
0	WTS	等待定时器计数状态	● 0b0：等待定时器计数停止 ● 0b1：等待定时器正在计数过程中	RO

47.5.12 等待定时器 z 计数寄存器 (ERMU_WTzCNT) (z=0~3)

偏移地址 (Offset) : 0x0208+0x0020*z

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																	RO															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CNT	等待定时器计数值	当内核停止时，等待定时器不会停止计数。	RO

47.5.13 等待定时器 z 比较寄存器 (ERMU_WTzCMP) (z=0~3)

偏移地址 (Offset) : 0x020C+0x0020*z

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CMP															
Access																	RW															

位/位域	标记	位名	功能描述	读写
31:16	Reserved	保留	-	-
15:0	CMP	等待定时器比较 值	ERMU_WTzC.WTE 为 0 时才能配置该值，当等待 定时器计数值达到该比较值时，计数停止并自动清零 ERMU_WTzS.WTS	RW

47.5.14 等待定时器 z 启动使能寄存器 (ERMU_WTzSE) (z=0~3)

偏移地址 (Offset) : 0x0210+0x0020*z

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																HPISE	Reserved																LPISE
Access																	RW																	RW

位/位域	标记	位名	功能描述	读写
31:15	Reserved	保留	-	-
16	HPISE	HPI 启动定时器 使能	0b0: HPI 启动定时器使能关闭 0b1: HPI 启动定时器使能打开，当 ERMU 触发 HPI 中断时，启动等待定时器开始计数	RW
15:1	Reserved	保留	-	-
0	LPISE	LPI 启动定时器 使能	0b0: LPI 启动定时器使能关闭 0b1: LPI 启动定时器使能打开，当 ERMU 触发 LPI 中断时，启动等待定时器开始计数	RW

47.5.15 错误源状态 j 寄存器 (ERMU_ESSj) (j=0~1)

偏移地址 (Offset) : 0x0300+0x0004*j

Offset	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ESS31	ESS30	ESS29	ESS28	ESS27	ESS26	ESS25	ESS24	ESS23	ESS22	ESS21	ESS20	ESS19	ESS18	ESS17	ESS16	ESS15	ESS14	ESS13	ESS12	ESS11	ESS10	ESS9	ESS8	ESS7	ESS6	ESS5	ESS4	ESS3	ESS2	ESS1	ESS0
Access	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位/位域	标记	位名	功能描述	读写
31:0	ESSk (k=31~0)	错误源状态位	错误源状态位 i (i = j * 32 + k) 0b0: 错误源没有错误 0b1: 错误源发生错误 错误源状态位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	RO

 **注意**

当 SWDT 配置为溢出复位时, 系统被 SWDT 复位后, SWDT_ERR 对应的 ERMU_ESS0 不会被置位。

47.5.16 错误源状态清除 j 寄存器 (ERMU_ESSCj) (j=0~1)

偏移地址 (Offset) : 0x0340+0x0004*j

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	ESSC31	ESSC30	ESSC29	ESSC28	ESSC27	ESSC26	ESSC25	ESSC24	ESSC23	ESSC22	ESSC21	ESSC20	ESSC19	ESSC18	ESSC17	ESSC16	ESSC15	ESSC14	ESSC13	ESSC12	ESSC11	ESSC10	ESSC9	ESSC8	ESSC7	ESSC6	ESSC5	ESSC4	ESSC3	ESSC2	ESSC1	ESSC0	
Access	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
31:0	ESSCk (k=31~0)	错误源状态清除位	错误源状态清除位 i (i = j * 32 + k) 写 1 清除相应的错误源状态位 ERMU_ESSj.ESSk, 写 0 无效; 读出恒为 0 错误源状态清除位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	WO

47.5.17 伪错误源触发 j 寄存器 (ERMU_PETj) (j=0~1)

偏移地址 (Offset) : 0x0380+0x0004*j

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PET31	PET30	PET29	PET28	PET27	PET26	PET25	PET24	PET23	PET22	PET21	PET20	PET19	PET18	PET17	PET16	PET15	PET14	PET13	PET12	PET11	PET10	PET9	PET8	PET7	PET6	PET5	PET4	PET3	PET2	PET1	PET0
Access	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位/位域	标记	位名	功能描述	读写
31:0	PETk (k=31~0)	伪错误源触发位	伪错误源触发位 i (i = j * 32 + k) 写 1 伪错误源触发相应 ERMU_ESSj.ESSk 置 1, 写 0 无效; 读出恒为 0 伪错误源触发位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	WO

47.5.18 复位使能 j 寄存器 (ERMU_REj) (j=0~1)

偏移地址 (Offset) : 0x03C0+0x0004*j

Offset	Bit Position																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	RE31	RE30	RE29	RE28	RE27	RE26	RE25	RE24	RE23	RE22	RE21	RE20	RE19	RE18	RE17	RE16	RE15	RE14	RE13	RE12	RE11	RE10	RE9	RE8	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	REk (k=31~0)	复位使能位	复位使能位 i (i = j * 32 + k) 0b0: 关闭复位功能 0b1: 使能复位功能, ERMU_ESSj.ESSk 的有效位将产生系统复位 复位使能位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	RW

47.5.19 低优先级中断使能 j 寄存器 (ERMU_LPIEj) (j=0~1)

偏移地址 (Offset) : 0x0400+0x0004*j

Offset	Bit Position																																
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	LPIE31	LPIE30	LPIE29	LPIE28	LPIE27	LPIE26	LPIE25	LPIE24	LPIE23	LPIE22	LPIE21	LPIE20	LPIE19	LPIE18	LPIE17	LPIE16	LPIE15	LPIE14	LPIE13	LPIE12	LPIE11	LPIE10	LPIE9	LPIE8	LPIE7	LPIE6	LPIE5	LPIE4	LPIE3	LPIE2	LPIE1	LPIE0	
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	LPIEk (k=31~0)	低优先级使能位	低优先级中断使能位 i ($i = j * 32 + k$) 0b0: 关闭低优先级中断功能 0b1: 使能低优先级中断功能, ERMU_ESSj.ESSk 的有效位将产生低优先级中断 低优先级中断使能位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	RW

47.5.20 高优先级中断使能 j 寄存器 (ERMU_HPIEj) (j=0~1)

偏移地址 (Offset) : 0x0500+0x0004*j

Offset	Bit Position																																
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	HPIE31	HPIE30	HPIE29	HPIE28	HPIE27	HPIE26	HPIE25	HPIE24	HPIE23	HPIE22	HPIE21	HPIE20	HPIE19	HPIE18	HPIE17	HPIE16	HPIE15	HPIE14	HPIE13	HPIE12	HPIE11	HPIE10	HPIE9	HPIE8	HPIE7	HPIE6	HPIE5	HPIE4	HPIE3	HPIE2	HPIE1	HPIE0	
Access	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:0	HPIEk (k=31~0)	高优先级使能位	高优先级中断使能位 i ($i = j * 32 + k$) 0b0: 关闭高优先级中断功能 0b1: 使能高优先级中断功能, ERMU_ESSj.ESSk 的有效位将产生高优先级中断 高优先级中断使能位 i 与错误源 ID 的对应关系请参见 ERMU 错误源列表 。	RW

47.5.21 计数时钟预分频寄存器 (ERMU_CCPS)

Offset	Bit Position																																
0x0600	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	PSS																Reserved																PSE
Access																																	RW

位/位域	标记	位名	功能描述	读写
31:16	PSS	预分频分频系数	计数时钟预分频频系数，经过预分频后的时钟将作为清除定时器、翻转定时器及等待定时器的计数时钟。 预分频后的时钟频率为 $f = f_{MRC} / (PSS + 1)$。  说明 f_{MRC} 是 MRC 频率（8MHz）。	RW
15:1	Reserved	保留	-	-
0	PSE	预分频使能	● 0b0：关闭预分频功能 ● 0b1：使能预分频功能	RW

48 调试控制器 (DBGC)

48.1 简介

本 MCU 的内核是 Cortex-M4，该内核包含用于高级调试功能的硬件，支持嵌入式跟踪宏单元 (ETM)。利用这些调试功能，可以在取指（指令断点）或访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。

提供两个调试接口：

- 串行调试跟踪接口 SWD
- 并行调试跟踪接口 JTAG

48.2 功能说明

48.2.1 功能框图

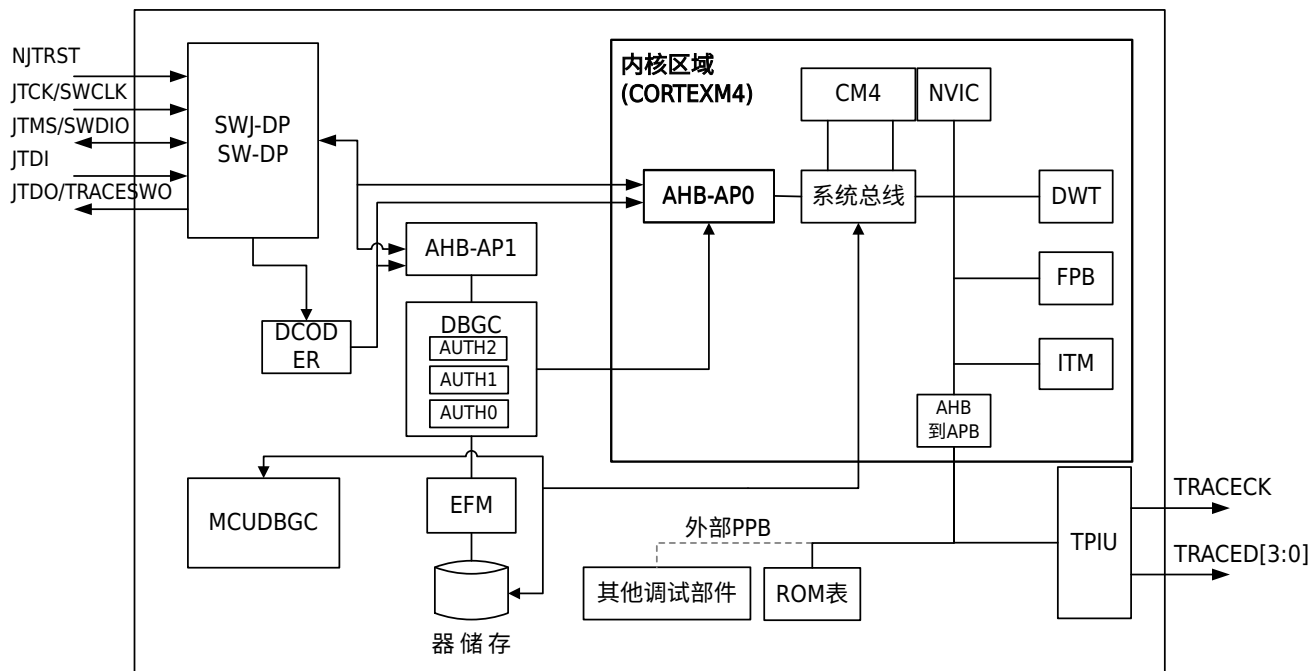


图 48-1 调试控制系统框图

ARM Cortex-M4 内核提供集成片上调试支持。它包括：

- SWJ-DP：SWD/JTAG 调试端口
- AHB-AP：AHB 访问端口
- ITM：指令跟踪单元
- FPB：Flash 指令断点
- DWT：数据断点触发
- TPIU：跟踪端口单元接口（大封装上提供，其中会映射相应引脚）
- 灵活调试引脚分配

说明

有关 ARM Cortex-M4 内核支持的调试功能的更多信息，请参见《Cortex-M4F-r0p1 技术参考手册》和《CoreSight 设计套件 r0p1 技术参考手册》。

48.2.2 参考文档

本产品参考如下 ARM 技术文档：

- Cortex M4 Technical Reference Manual r0p1
- Arm Debug Interface Architecture Specification version 5
- CoreSight Components Technical Reference Manual version 1.0

48.2.3 SWJ-DP 调试端口 (SWD 和 JTAG)

该 MCU 内核集成了 SWD/ JTAG 调试端口 (SWJ-DP)。该端口是 ARM 标准 CoreSight 调试端口，具有 JTAG-DP (5 引脚) 接口和 SW-DP (2 引脚) 接口。

- JTAG 调试端口 (JTAG-DP) 提供用于连接到 AHB-AP 端口的 5 引脚标准 JTAG 接口。
- 串行线调试端口 (SW-DP) 提供用于连接到 AHB-AP 端口的 2 引脚 (时钟+数据) 接口。

在 SWJ-DP 中，SW-DP 的 2 个 JTAG 引脚与 JTAG-DP 的 5 个 JTAG 引脚中的部分引脚复用。

下图中 JTDO 复用了 TRACESWO 与 TDO。这意味着异步跟踪只能在 SW-DP 上实现，不能在 JTAG-DP 上实现。

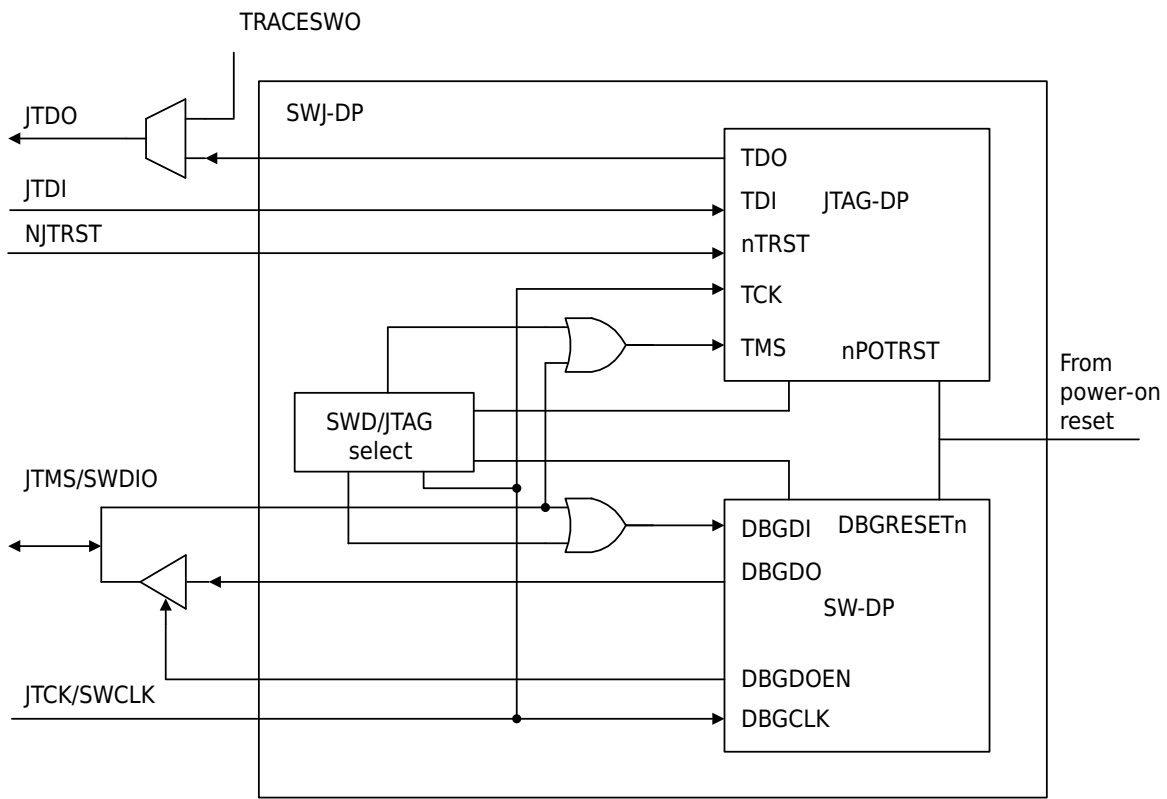


图 48-2 调试控制系统

48.2.3.1 JTAG-DP 或 SW-DP 的切换机制

默认调试接口是 JTAG-DP 接口。

如果调试工具想要切换到 SW-DP，它必须在 JTMS (SWDIO) / JTCK (SWCLK) 上提供专用的 JTAG 序列，用于禁止 JTAG-DP 并使能 SW-DP。这样便可仅使用 SWCLK 和 SWDIO 引脚来访问 SW-DP。

该序列为：

1. 输出超过 50 个 JTCK 周期的 JTMS (SWDIO) =1 信号
2. 输出 16 个 JTMS (SWDIO) 信号 0111_1001_1110_0111 (MSB)
3. 输出超过 50 个 JTCK 周期的 JTMS (SWDIO) =1 信号

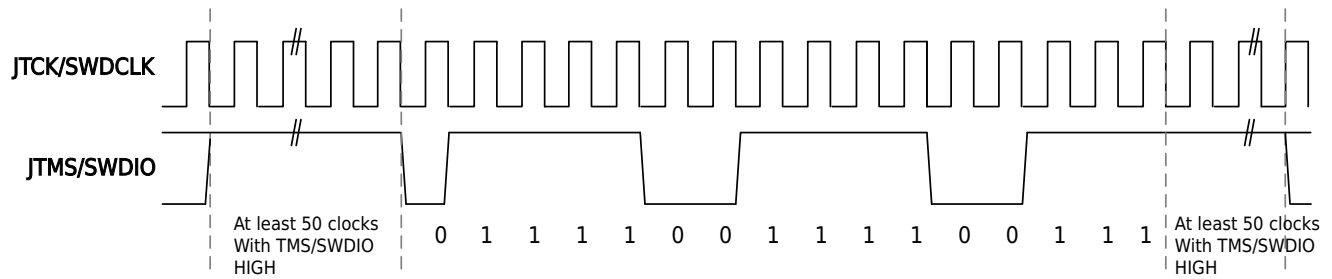


图 48-3 JTAG-DP 到 SW-DP 切换时序

48.2.4 引脚排列和调试端口引脚

根据 MCU 的不同封装有不同的有效引脚数。因此，某些与引脚相关的功能可能随封装而不同。

SWJ 调试端口引脚

MCU 的 5 个普通 I/O 口可用作 SWJ-DP 接口引脚。

表 48-1 SWJ 调试端口引脚

SWJ-DP 引脚名称	JTAG 调试端口		SW 调试端口	
	类型	说明	类型	调试分配
JTMS/SWDIO	输入	JTAG 模式选择	输入/输出	串行线数据输入/输出
JTCK/SWCLK	输入	JTAG 时钟	输入	串行时钟
JTDI	输入	JTAG 数据输入	-	-
JTDO/TRACESWO	输出	JTAG 数据输出	-	TRACESWO（如果使能异步跟踪）
NJTRST	输入	JTAG 复位	-	-

灵活的 SWJ-DP 引脚分配

复位（上电或引脚复位）后，会将用于 SWJ-DP 的全部 5 个引脚指定为专用引脚，可供调试工具立即使用（请注意，若要使用跟踪输出，请配置寄存器 MCUTRACECTL）。但是，MCU 可以禁止部分或全部 SWJ-DP 端口，进而释放相关引脚以用作 GPIO。有关如何禁止 SWJ-DP 端口引脚的更多详细信息，请参见《通用 IO（GPIO）》章节“特殊控制寄存器 PSPCR”。

表 48-2 灵活的 SWJ-DP 引脚分配

可用的调试端口	分配的 SWJ IO 引脚				
	JTMS/ SWDIO	JTCK/ SWCLK	JTDI	JTDO	NJTRST
全部 SWJ（JTAG-DP+SW-DP）-复位状态	✓	✓	✓	✓	✓
禁止 JTAG-DP 和使能 SW-DP	✓	✓	可释放	可释放	可释放
禁止 JTAG-DP 和禁止 SW-DP	可释放	可释放	可释放	可释放	可释放

JTAG 引脚上的内部上拉

根据 JTAG IEEE 标准，必须确保 JTAG 输入引脚不悬空，因为这些引脚直接连接到 MCU 内部用于控制调试功能。还必须特别注意 JTCK/ SWCLK 引脚，该引脚直接用于调试控制时钟功能。为避免 IO 电平浮空，MCU 在 JTAG 引脚上内置有内部上拉电阻：

- NJTRST：内部上拉
- JTDI：内部上拉
- JTMS/ SWDIO：内部上拉
- JTCK/ SWCLK：内部上拉
- JTDO：内部上拉

在未连接调试器状态下，用户软件可以通过设定 GPIO 特殊控制寄存器释放 JTAG IO 作为普通的 I/O 口使用。由于芯片内部上拉为 < 100kΩ 的弱上拉，建议采用外部 10kΩ 的外部上拉。

使用串行接口以及释放未使用的调试引脚以作 GPIO

使用 SWD 时可以释放一些 GPIO，用户软件必须在 GPIO 控制寄存器中更改 GPIO 配置，这样便可释放相应的引脚用作 GPIO。

调试时，主机执行以下操作：

- 在系统复位状态下，分配所有 SWJ 引脚（JTAG-DP+SW-DP）。
- 在系统复位状态下，调试主机发送 JTAG 序列，以从 JTAG-DP 切换到 SW-DP。
- 仍然在系统复位状态下，调试主机在复位地址处设置断点。
- 释放复位信号，内核停止在复位地址处。
- 从此调试端口切换为 SW-DP。然后通过用户软件将其他 JTAG 引脚重新分配为 GPIO。



注意

对于用户软件设计，需要释放调试引脚时，在复位后一直到用户软件释放引脚这段期间，这些引脚仍然处于输入上拉（NJTRST、JTMS、JTDI、JTCK 和 JTDO）。

48.2.5 SW 调试端口-SW 协议简介

同步串行协议使用两个引脚：

- SWCLK：从主机到从机的时钟
- SWDIO：双向

传输数据时，LSB 在前。

对于 SWCLK 和 SWDIO，需要在电路板上对线路进行上拉（建议采用 10kΩ）。

48.2.6 TPIU（跟踪端口接口单元）

48.2.6.1 简介与框图

TPIU 是 ITM 与片上跟踪数据之间的桥接器。

输出数据流封装成跟踪源 ID，然后被跟踪端口分析器（TPA）捕获。

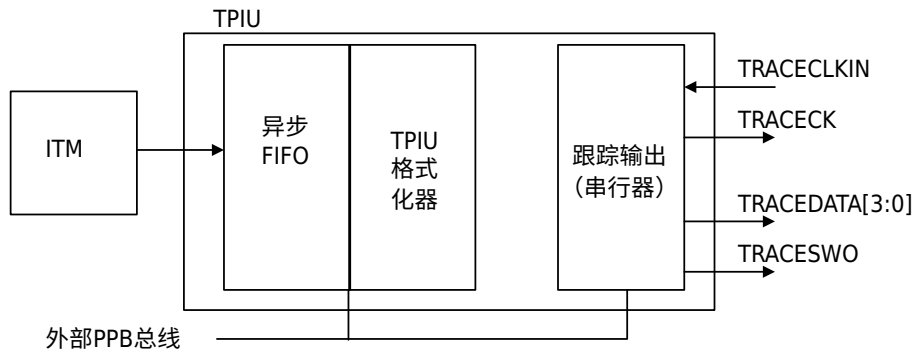


图 48-4 TPIU 框图

48.2.6.2 TRACE 引脚分配

● 异步模式

异步模式需要 1 个额外引脚，并且适用于所有封装。仅在使用串线行模式时异步模式才可用（在 JTAG 模式下不可用）。

TPIU 引脚名称	跟踪异步模式	
	类型	说明
TRACESWO	输出	TRACE 异步数据输出

● 同步模式

同步模式需要 2 到 5 个额外引脚，具体取决于所跟踪数据的长度，并且仅适用于较大型的封装。此外，同步模式在 JTAG 模式和串行模式下均可用，并可提供比异步跟踪更高的带宽输出能力。

TPIU 引脚名称	跟踪同步模式	
	类型	说明
TRACECK	输出	TRACE 时钟
TRACED[3:0]	输出	TRACE 同步数据输出，可以是 1、2 或 4

TPIU TRACE 引脚分配原则

默认情况下，不分配这些引脚。可通过将 MCU 调试组件配置寄存器（MCUTRACECTL）中的 TRACE_IOEN 和 TRACE_MODE 位来配置这些引脚。必须由调试主机或 CPU 来完成此配置。

此外，要分配的引脚数目取决于跟踪配置（异步跟踪或同步跟踪）。

- 异步模式：需要 1 个额外引脚
- 同步模式：需要 5 个额外引脚
 - ▶ TRACECK
 - ▶ TRACED[0]（如果端口数据长度配置为 1、2 或 4）
 - ▶ TRACED[1]（如果端口数据长度配置为 2 或 4）
 - ▶ TRACED[2]（如果端口数据长度配置 4）
 - ▶ TRACED[3]（如果端口数据长度配置 4）

要分配 TRACE 引脚，调试主机必须对 MCU 调试配置寄存器（MCUTRACECTL）的位 TRACE_IOEN 和 TRACE_MODE[1:0]进行配置。默认情况下不分配 TRACE 引脚。

此寄存器映射到外部 PPB 总线上，通过上电（而非引脚复位）复位。可在引脚复位状态下通过调试器写入该寄存器。

TPIU 引脚用途			分配的 TRACE IO 引脚					
跟踪模式	TRACE_IOEN	TRACE_MODE	JTDO/TRACESWO	TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]
无跟踪（默认状态）	0b0	0bXX	释放 ⁽¹⁾	释放	释放	释放	释放	释放
异步跟踪	0b1	0b00	TRACESWO	释放	释放	释放	释放	释放
同步跟踪 1 位	0b1	0b01	释放 ⁽¹⁾	TRACECK	TRACED[0]	释放	释放	释放
同步跟踪 2 位	0b1	0b10	释放 ⁽¹⁾	TRACECK	TRACED[0]	TRACED[1]	释放	释放
同步跟踪 4 位	0b1	0b11	释放 ⁽¹⁾	TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]



说明

- 1. 使用串行模式时，释放此引脚。但使用 JTAG 时，此引脚分配给 TDO。

48.2.6.3 MCU 内部 TRACECLKIN 连接

在本 MCU 中，TPIU 的时钟 TRACECLKIN 连接到内部时钟。MCU 的默认时钟是内部 MRC 振荡器。复位状态下的频率与复位释放后的频率不同。原因是，由于系统复位状态下采用默认的 MRC 校准值，而在每次系统复位释放时会更新该 MRC 校准值。因此，跟踪端口分析器（TPA）在系统复位状态下不应使能跟踪（使用 TRACE_IOEN 位），原因是，在复位状态下的同步帧包的比特宽度与复位后的包不同。

48.2.6.4 TPIU 寄存器使用说明

仅当调试异常和监视控制寄存器（DEMCR）的位 TRCENA 置 1 时才能对 TPIU APB 寄存器进行读写操作。否则，这些寄存器将读为零（此位的输出会使能 TPIU 的时钟）。

48.2.6.5 TPIU 配置示例

- 1. 将调试异常和监视控制寄存器（DEMCR）中的位 TRCENA 置 1
- 2. 将所需值写入 TPIU 当前端口大小寄存器（对于 1 位端口大小，默认值为 0x1）
- 3. 将 0x102 写入 TPIU 格式化器和刷新控制寄存器（默认值）
- 4. 写入 TPIU 选择引脚协议以选择同步模式或异步模式。示例：0x2 表示异步 NRZ 模式（类似于 USART）
- 5. 将 0x00 写入 MCUTRACECTL 控制寄存器（位 TRACE_IOEN），为异步模式分配 TRACE I/O
- 6. 此时发送 TPIU 同步数据包（FF_FF_FF_7F）
- 7. 配置 ITM 并对 ITM 激励寄存器进行写操作以输出值

48.3 寄存器

48.3.1 寄存器总表

基地址：0xE0042000

表 48-6 调试控制器 (DBGC) 寄存器偏移地址

偏移地址	寄存器	描述	位宽	访问主机
0x000C	CHIPID	芯片专属标志寄存器	32	调试 IDE
0x001C	MCUDBGSTAT	DBG 状态寄存器	32	CPU/调试 IDE ⁽¹⁾
0x0020	MCUSTPCTL	外设调试暂停寄存器	32	CPU/调试 IDE ⁽¹⁾
0x0024	MCUTRACECTL	调试组件配置寄存器	32	CPU/调试 IDE ⁽¹⁾
0x0028	MCUSTPCTL2	外设调试暂停寄存器 2	32	CPU/调试 IDE ⁽¹⁾

 说明

1. 寄存器位于 PPB 区域，只能在特权模式下 CPU 才能进行访问。

48.3.2 芯片专属标志寄存器 (CHIPID)

用于表示公司信息及产品型号信息。

Offset	Bit Position																															
0x000C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x584804A8																															
Name	CHIPID																															
Access	RO																															

位/位域	标记	位名	功能描述	读写
31:0	CHIPID	芯片专属标志	CHIPID[31:16]: XH 的对应 ASCII 码, 0x5848 CHIPID[15:0]: 产品型号, 0x04A8	RO

48.3.3 DBG 状态寄存器 (MCUDBGSTAT)

DBG 调试上电状态确认寄存器。

Offset	Bit Position																															
0x001C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														CDBGPWRUPACK	CDBGPWRUPREQ
Access																															RW	RW

位/位域	标记	位名	功能描述	读写
31:2	Reserved	保留	-	-
1	CDBGPWRUPACK	调试器上电反馈	● 0b0: 无反馈 ● 0b1: 调试上电反馈	RW
0	CDBGPWRUPREQ	调试器上电请求	● 0b0: 无上电请求 ● 0b1: 上电请求	RW

48.3.4 外设调试暂停寄存器 (MCUSTPCTL)

当 CPU 处于调试状态时，周边模块暂停控制。

Offset	Bit Position																																
0x0020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x0000003B																																
Name	Reserved						M25STP	M24STP	M23STP	M22STP	M21STP	M20STP	M19STP	M18STP	M17STP	M16STP	M15STP	M14STP	M13STP	M12STP	M11STP	M10STP	M09STP	M08STP	M07STP	M06STP	PVD2STP	PVD1STP	PVD0STP	RTCSTP	WDTSTP	SWDSTP	
Access							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:26	Reserved	保留	-	-
25	M25STP	Timer0_5 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
24	M24STP	Timer0_4 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
23	M23STP	Timer0_3 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
22	M22STP	Timer6_8 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
21	M21STP	Timer6_7 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
20	M20STP	Timer6_6 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
19	M19STP	Timer6_5 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
18	M18STP	Timer6_4 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
17	M17STP	Timer6_3 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
16	M16STP	Timer6_2 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
15	M15STP	Timer6_1 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
14	M14STP	Timer4_3 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
13	M13STP	Timer4_2 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
12	M12STP	Timer4_1 计数暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW

位/位域	标记	位名	功能描述	读写
11	M11STP	Timer2_4 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
10	M10STP	Timer2_3 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
9	M09STP	Timer2_2 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
8	M08STP	Timer2_1 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
7	M07STP	Timer0_2 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
6	M06STP	Timer0_1 计数 暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
5	PVD2STP	PVD2 暂停信号	0b0: 即使内核停止, 仍然产生 PVD2 中断申请或复位 0b1: 内核停止时, 屏蔽 PVD2 中断申请或复位	RW
4	PVD1STP	PVD1 暂停信号	0b0: 即使内核停止, 仍然产生 PVD1 中断申请或复位 0b1: 内核停止时, 屏蔽 PVD1 中断申请或复位	RW
3	PVD0STP	PVD0 暂停信号	0b0: 即使内核停止, 仍然产生 BOR 中断申请或复位 0b1: 内核停止时, 屏蔽 BOR 中断申请或复位  说明 PVD0 用作 BOR 复位。	RW
2	RTCSTP	RTC 计数暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
1	WDTSTP	WDT 计数暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW
0	SWDTSTP	SWDT 计数暂停信号	0b0: 即使内核停止, 计数器仍然计数 0b1: 内核停止时, 计数器暂停计数	RW

48.3.5 调试组件配置寄存器 (MCUTRACECTL)

通过该寄存器配置 TRACE 输出引脚。

Offset	Bit Position																															
0x0024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																												TRACE_IOEN		TRACE_MODE	
Access																													RW		RW	

位/位域	标记	位名	功能描述	读写
31:3	Reserved	保留	-	-
2	TRACE_IOEN	TRACE 引脚输出控制	0b0: 同步跟踪引脚输出禁止 0b1: 同步跟踪引脚输出许可	RW
1:0	TRACE_MODE	TRACED 输出引脚控制	0b00: 异步跟踪 0b01: 同步跟踪 1 位 TRACED[0] 0b10: 同步跟踪 2 位 TRACED[1:0] 0b11: 同步跟踪 4 位 TRACED[3:0]	RW

48.3.6 外设调试暂停寄存器 2 (MCUSTPCTL2)

当 CPU 处于调试状态时，周边模块暂停控制。

Offset	Bit Position																															
0x0028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				M43STP	M42STP	M41STP	M40STP	M39STP	M38STP	M37STP	M36STP	M35STP	M34STP	M33STP	M32STP
Access																					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位/位域	标记	位名	功能描述	读写
31:12	Reserved	保留	-	-
11	M43STP	TimerA_12 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
10	M42STP	TimerA_11 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
9	M41STP	TimerA_10 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
8	M40STP	TimerA_9 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
7	M39STP	TimerA_8 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
6	M38STP	TimerA_7 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
5	M37STP	TimerA_6 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
4	M36STP	TimerA_5 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
3	M35STP	TimerA_4 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
2	M34STP	TimerA_3 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
1	M33STP	TimerA_2 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW
0	M32STP	TimerA_1 计数 暂停信号	0b0: 即使内核停止，计数器仍然计数 0b1: 内核停止时，计数器暂停计数	RW

A 术语表

术语	全称	说明
Byte	-	字节，8 位数据
Half Word	-	半字，16 位数据
Word	-	字，32 位数据
Double Word	-	双字，64 位数据
IAP	-	指可以在用户程序运行期间对微控制器的 Flash 进行重新编程（在应用中编程）
ICP	-	指可以在器件安装于用户应用电路板上时使用 JTAG 协议、SWD 协议或自举程序对微控制器的 Flash 进行编程（在线编程）
AHB	Advanced High-performance Bus	高级高性能总线
APB	Advanced Peripheral Bus	低速外设总线
DMA	Direct Memory Access	直接存储器访问
TIM/TMR	-	定时器