XHSC 小华半导体
XIAOHUA SEMICONDUCTOR

32 位微控制器

EMB 功能使用

应用笔记

Rev1.01　2026 年 07 月

适用对象

产品系列	产品型号	产品系列	产品型号
HC32A448	ALL	**HC32F467**	ALL
HC32A460	ALL	**HC32F472**	ALL
HC32A472	ALL	**HC32F4A0**	ALL
HC32A4A0	ALL	**HC32F4A2**	ALL
HC32A4A8	ALL	**HC32F4A8**	ALL
HC32F334	ALL	**HC32K116**	ALL
HC32F431	ALL	**HC32K118**	ALL
HC32F448	ALL	**HC32M120**	ALL
HC32F451	ALL	**HC32M413**	ALL
HC32F452	ALL	**HC32M441**	ALL
HC32F460	ALL	**HC32M458**	ALL

声　明

★ 小华半导体有限公司（以下简称：“XHSC”）保留随时更改、更正、增强、修改小华半导体产品和/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。

★ 任何带有“®”或“™”标识的图形或字样是 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。

★ 本通知中的信息取代并替换先前版本中的信息。

目 录

1 概述

本文档主要介绍小华半导体 HC32 系列芯片 EMB 功能使用。EMB（Emergency Motor Brake），紧急电机刹车，用于在出现异常情况时控制 PWM 输出，Timer 接到控制事件后，可根据设定将输出端口置为高电平、低电平或高阻态。

1.1 主要特性

1. 检测多种异常情况并产生控制事件：

- 外部端口输入电平
- PWM 输出端口电平发生同相
- 电压比较器比较结果
- 系统错误发生
- 写寄存器软件控制

2. 外部端口输入电平检测滤波功能
3. 自动释放功能

2 EMB 功能介绍

EMB 结构框图如下所示（*此图以 HC32F448 EMB 模块为例，其他芯片 EMB 模块框图请参考对应手册）。

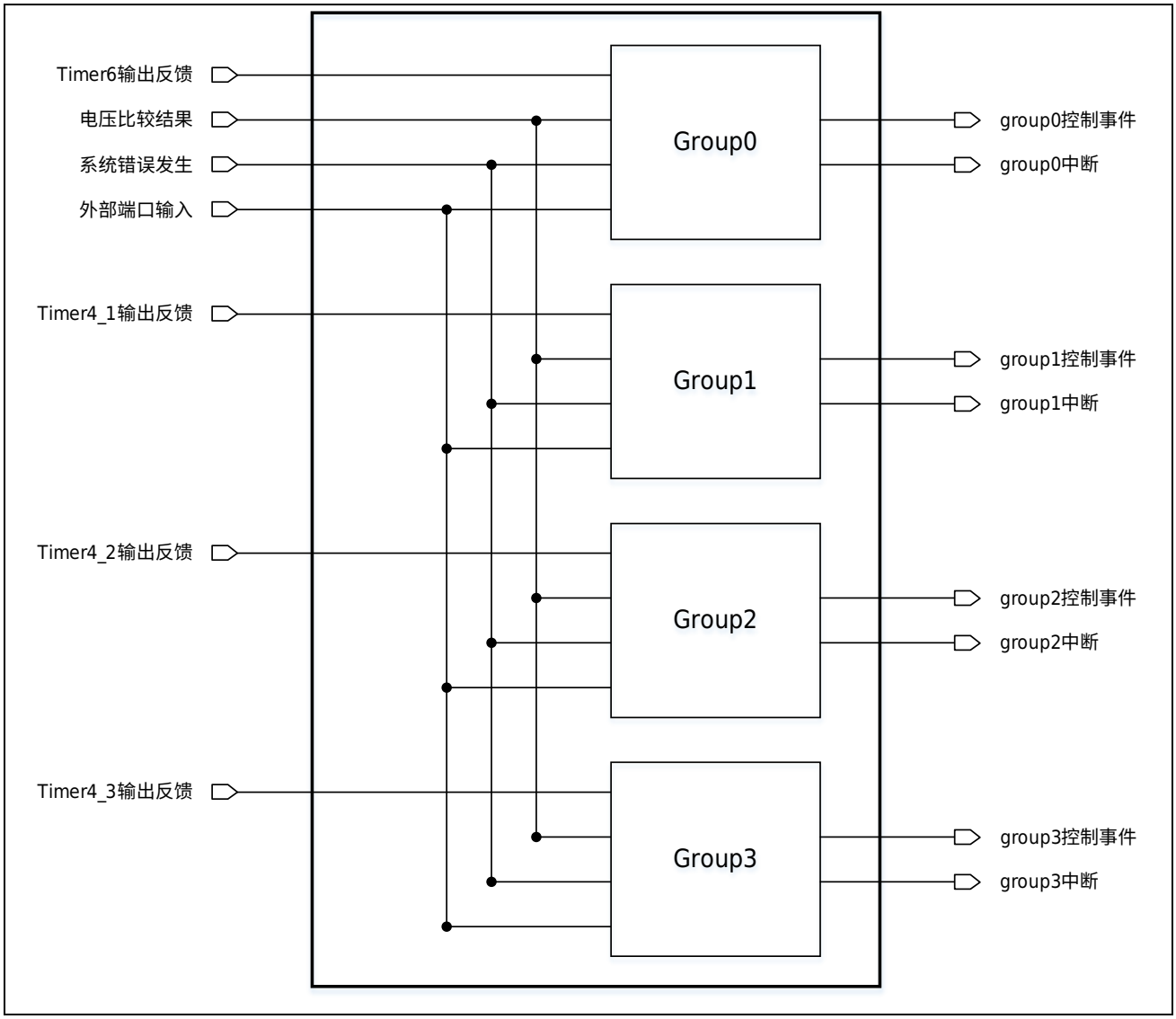

图 2-1 EMB 结构框图*

EMB 通道框图如下所示（*此图以 HC32F448 EMB 模块为例，其他芯片 EMB 模块框图请参考对应手册）。

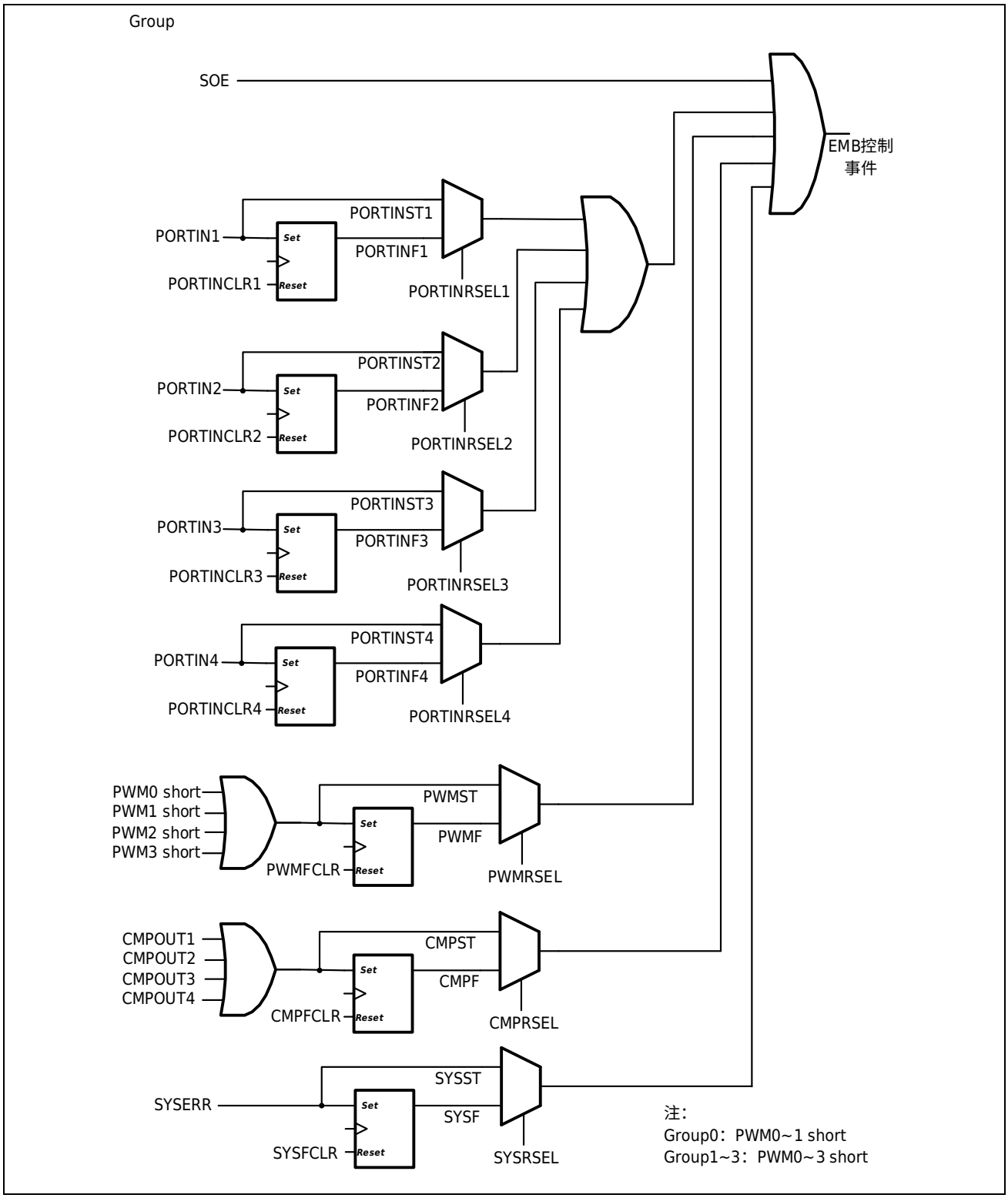

图 2-2 EMB 通道功能框图*

2.1 EMB 触发源矩阵

EMB 模块可以被多种触发源触发，监测不同的异常情况，下表是不同芯片搭载的触发源情况。

表 2-1 各芯片 EMB 模块触发源矩阵

芯片	触发源										
	软件写	外部端口检测	比较器输出结果	Timer 同相检测	XTAL 停止振荡	LockUp 信号	LVD 检测	FLASH ECC 校验错误	SRAM 奇偶校验错误	SRAM ECC 校验错误	FCM 频率测量异常
HC32M120	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32F460	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32A460	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32F4A0	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32A4A0	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32F4A2	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32F467	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32F472	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32A472	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32F448	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32A448	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32M441	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32F334	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
HC32F4A8	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
HC32A4A8	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
HC32K118	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
HC32K116	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
HC32F431	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32M413	✓	✓	✓	✓	✓	×	×	×	×	×	×
HC32M458	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓

2.2 EMB 异常监测功能

2.2.1 外部端口电压监测

EMB 有多个外部端口用于实现当输入电平变化时控制 PWM 信号输出。每组 EMB 能够独立设置 1 个或多个外部端口有效，端口分配详见各芯片手册 EMB 章节。

使用外部端口输入电平变化控制 PWM 信号输出时，首先将使能位置为有效，同时设置在端口高电平或端口低电平时产生控制信号。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被置位，同时端口输入控制标志位被置位。当中断许可寄存器有效时，将产生中断。

2.2.2 PWM 输出端口电平同相监测

EMB 可对多个 Timer 输出 PWM 对实行同相（同高或同低）检测功能。可用于实现 PWM 输出同相时控制 PWM 信号输出。每组 EMB 能够独立设置一对或多对 PWM 同相检测。

使用时将使能位设置为有效，选择需要检测的有效电平。设置完成后，EMB 监测 Timer 的互补 PWM 输出信号，当输出信号出现同高或同低的情况时，EMB 状态寄存器的 PWM 输出状态位被置位，同时 PWM 输出同相控制标志位被置位。当中断许可寄存器有效时，将产生中断。

2.2.3 电压比较器比较结果监测

EMB 的能够根据电压比较器的比较结果向 Timer 发送控制事件信号。电压比较器输出结果的设定参考手册《电压比较器（CMP)》章节相关描述。

使用时将使能位置为有效。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被置位，同时 EMB 电压比较器控制标志位被置位。当中断许可寄存器有效时，将产生中断。

2.2.4 系统错误发生监测

EMB 能够在系统发生错误时向 Timer 发送通知信号。系统错误信号来自外部振荡器停止、SRAM 奇偶校验错误以及 ECC 校验错误、Lockup 锁定、LVD 检测、FLASH ECC 校验错误等信号。

使用时将使能位置为有效，使能需要检测的系统错误源。当选择的系统错误志位被置位时，EMB 状态寄存器的系统错误状态被置位，同时 EMB 系统错误控制标志位被置位。当中断许可寄存器有效时，将产生中断。

2.2.5 软件控制 PWM 信号输出

当用户需要监测的异常情况不被 EMB 模块直接支持时，可以通过 EMB 的软件输出使能控制寄存器直接向 Timer 发送控制信号，实现对 PWM 的控制。软件控制 PWM 输出时不会产生中断请求。

2.3 外部端口滤波功能

EMB 模块使用外部端口输入监测功能时，可以配合滤波器功能，以实现对外部端口尖刺信号的过滤。

滤波器根据滤波时钟（EMB 外设时钟）对输入信号进行采样，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三次一致的电平时，该电平被当作有效电平传送到模块内部；小于三次一致的电平会被当作外部干扰滤掉，不传送到模块内部。

2.4 自动释放功能

EMB 模块包含有自动释放功能，配置自动释放后，当 EMB 异常信号转为无效时，EMB 状态位将自动清零，同时控制事件将立即释放；如果配置手动释放，当 EMB 异常信号转为无效后，需要通过 EMB 状态复位寄存器使 EMB 状态寄存器清零才能将控制事件释放。此功能仅部分芯片搭载有，是否包含有此功能请参考芯片手册。

3 驱动和应用举例

所有搭载高级定时器的 HC32 系列芯片均搭载了 EMB 模块，用于在发生异常情况时控制 PWM 输出。以下将介绍不同方式控制 PWM 输出的配置和相关注意事项。

3.1 外部端口输入电平控制 PWM 输出

3.1.1 结构体定义

```
/**
* @brief EMB monitor EMB port configuration
*/
typedef struct {
uint32_t u32PortState;          /*!< Enable or disable EMB detect port in control function
                                This parameter can be a value of @ref EMB_Port_Selection */
uint32_t u32PortLevel;          /*!< EMB detect port level
                                This parameter can be a value of @ref EMB_Detect_Port_Level */
uint32_t u32PortFilterDiv;      /*!< EMB port filter division
                                This parameter can be a value of @ref EMB_Port_Filter_Clock_Division */
uint32_t u32PortFilterState;    /*!< Enable or disable EMB detect port filter in control function
                                This parameter can be a value of @ref EMB_Port_Filter_Selection */
} stc_emb_monitor_port_config_t;
```

通过结构体可配置选项为：

端口检测使能

```
/**
 * @defgroup EMB_Port_Selection EMB Port Selection
 * @{
 */
#define EMB_PORT1_DISABLE                   (0UL)
#define EMB_PORT2_DISABLE                   (0UL)
#define EMB_PORT3_DISABLE                   (0UL)
#define EMB_PORT4_DISABLE                   (0UL)
#define EMB_PORT1_ENABLE                    (EMB_CTL1_PORTINEN1)
#define EMB_PORT2_ENABLE                    (EMB_CTL1_PORTINEN2)
#define EMB_PORT3_ENABLE                    (EMB_CTL1_PORTINEN3)
#define EMB_PORT4_ENABLE                    (EMB_CTL1_PORTINEN4)
/**
```

```
 * @}
        */
```

端口电平状态

```
/**
 * @defgroup EMB_Detect_Port_Level EMB Detect Port Level
 * @{
 */
#define EMB_PORT1_DETECT_LVL_HIGH              (0UL)
#define EMB_PORT2_DETECT_LVL_HIGH              (0UL)
#define EMB_PORT3_DETECT_LVL_HIGH              (0UL)
#define EMB_PORT4_DETECT_LVL_HIGH              (0UL)
#define EMB_PORT1_DETECT_LVL_LOW               (EMB_CTL1_INVSEL1)
#define EMB_PORT2_DETECT_LVL_LOW               (EMB_CTL1_INVSEL2)
#define EMB_PORT3_DETECT_LVL_LOW               (EMB_CTL1_INVSEL3)
#define EMB_PORT4_DETECT_LVL_LOW               (EMB_CTL1_INVSEL4)
/**
 * @}
        */
```

端口滤波分频

```
/**
 * @defgroup EMB_Port_Filter_Clock_Division EMB Port Filter Clock Division
 * @{
 */
#define EMB_PORT1_FILTER_CLK_DIV1              (0UL << EMB_CTL2_NFSEL1_POS)
#define EMB_PORT1_FILTER_CLK_DIV8              (1UL << EMB_CTL2_NFSEL1_POS)
#define EMB_PORT1_FILTER_CLK_DIV32             (2UL << EMB_CTL2_NFSEL1_POS)
#define EMB_PORT1_FILTER_CLK_DIV128            (3UL << EMB_CTL2_NFSEL1_POS)
#define EMB_PORT2_FILTER_CLK_DIV1              (0UL << EMB_CTL2_NFSEL2_POS)
#define EMB_PORT2_FILTER_CLK_DIV8              (1UL << EMB_CTL2_NFSEL2_POS)
#define EMB_PORT2_FILTER_CLK_DIV32             (2UL << EMB_CTL2_NFSEL2_POS)
#define EMB_PORT2_FILTER_CLK_DIV128            (3UL << EMB_CTL2_NFSEL2_POS)
#define EMB_PORT3_FILTER_CLK_DIV1              (0UL << EMB_CTL2_NFSEL3_POS)
```

```
#define EMB_PORT3_FILTER_CLK_DIV8          (1UL << EMB_CTL2_NFSEL3_POS)
#define EMB_PORT3_FILTER_CLK_DIV32         (2UL << EMB_CTL2_NFSEL3_POS)
#define EMB_PORT3_FILTER_CLK_DIV128        (3UL << EMB_CTL2_NFSEL3_POS)
#define EMB_PORT4_FILTER_CLK_DIV1          (0UL << EMB_CTL2_NFSEL4_POS)
#define EMB_PORT4_FILTER_CLK_DIV8          (1UL << EMB_CTL2_NFSEL4_POS)
#define EMB_PORT4_FILTER_CLK_DIV32         (2UL << EMB_CTL2_NFSEL4_POS)
#define EMB_PORT4_FILTER_CLK_DIV128        (3UL << EMB_CTL2_NFSEL4_POS)
/**
 * @}
      */
```

端口滤波使能

```
* @defgroup EMB_Port_Filter_Selection EMB Port Filter Selection
 * @{
 */
#define EMB_PORT1_FILTER_DISABLE           (0UL)
#define EMB_PORT2_FILTER_DISABLE           (0UL)
#define EMB_PORT3_FILTER_DISABLE           (0UL)
#define EMB_PORT4_FILTER_DISABLE           (0UL)
#define EMB_PORT1_FILTER_ENABLE            (EMB_CTL2_NFEN1)
#define EMB_PORT2_FILTER_ENABLE            (EMB_CTL2_NFEN2)
#define EMB_PORT3_FILTER_ENABLE            (EMB_CTL2_NFEN3)
#define EMB_PORT4_FILTER_ENABLE            (EMB_CTL2_NFEN4)
/**
 * @}
    */
```

3.1.2 应用举例

EMB 检测端口输入电平，当电平满足设定时，EMB 模块将控制 PWM 输出。此处配置检测 PORT4 低电平时，控制 PWM 输出低；当 PORT4 变为高电平后，PWM 恢复输出。

```
/* EMB: initialize EMB pin */
GPIO_SetFunc(EMB_PORT, EMB_PIN, EMB_GPIO_FUNC);
```

```
/* EMB: initialize */
(void)EMB_TMR4_StructInit(&stcEmbInit);
stcEmbInit.stcPort.stcPort4.u32PortState = EMB_PORT4_ENABLE;
stcEmbInit.stcPort.stcPort4.u32PortLevel = EMB_PORT4_DETECT_LVL_LOW;
(void)EMB_TMR4_Init(EMB_GROUP, &stcEmbInit);

/* EMB: enable interrupt */
EMB_IntCmd(EMB_GROUP, EMB_INT_PORT4, ENABLE);

/* EMB: set release PWM condition */
EMB_SetReleasePwmCond(EMB_GROUP, EMB_EVT_PORT4, EMB_RELEASE_PWM_COND_STAT_ZERO);

/* EMB: register IRQ handler && configure NVIC. */
stcIrqConfig.enIRQn = EMB_INT_IRQn;
stcIrqConfig.enIntSrc = EMB_INT_SRC;
stcIrqConfig.pfnCallback = &EMB_IRQ_Handler;
(void)INTC_IrqSignIn(&stcIrqConfig);
NVIC_SetPriority(stcIrqConfig.enIRQn, DDL_IRQ_PRIO_DEFAULT);
NVIC_ClearPendingIRQ(stcIrqConfig.enIRQn);
NVIC_EnableIRQ(stcIrqConfig.enIRQn);
```

具体的现象如下图所示：

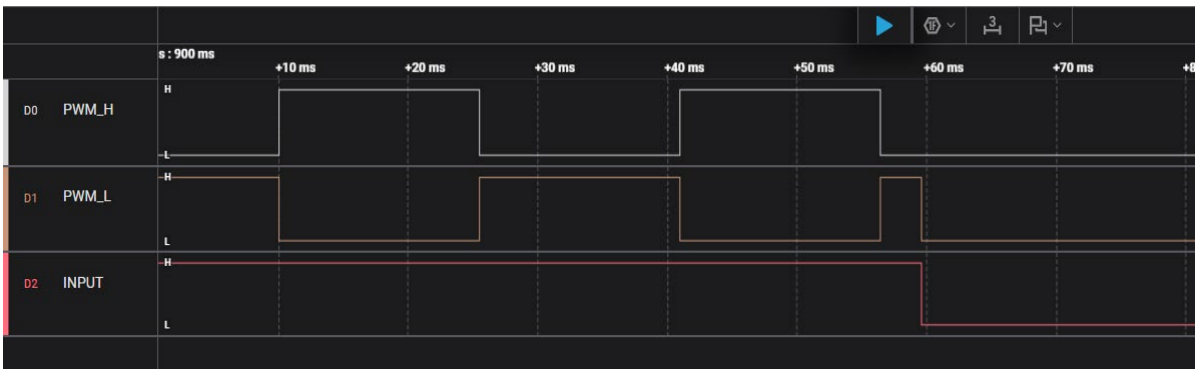

图 3-1　外部端口电平有效时，EMB 控制 PWM 停止输出

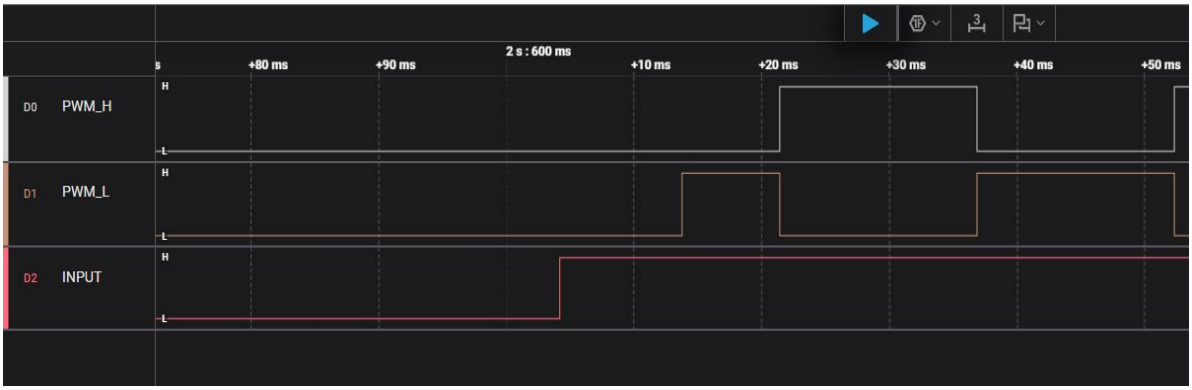

图 3-2　外部端口电平无效时，EMB 控制 PWM 恢复输出

3.2 PWM 输出端口电平同相控制 PWM 信号输出

3.2.1 结构体定义

```
/**
 * @brief EMB monitor PWM configuration
 */
typedef struct {
    uint32_t u32PwmState;    /*!< Enable or disable EMB detect timer same phase function
                              This parameter can be a value of @ref EMB_Detect_PWM state. */
    uint32_t u32PwmLevel;    /*!< Detect timer polarity level
                              This parameter can be a value of @ref EMB_Detect_PWM level */
} stc_emb_monitor_tmr_pwm_t;
```

可选配置为：

使能 PWM 同相检测（此处以 Timer4 为例）

```
/**
 * @defgroup EMB_TMR4_PWM_Selection EMB TMR4 PWM Selection
 * @{
 */
#define EMB_TMR4_PWM_X_DISABLE                  (0UL)
#define EMB_TMR4_PWM_W_DISABLE                  (0UL)
#define EMB_TMR4_PWM_V_DISABLE                  (0UL)
#define EMB_TMR4_PWM_U_DISABLE                  (0UL)
#define EMB_TMR4_PWM_X_ENABLE                   (EMB_CTL1_PWMSEN3)
#define EMB_TMR4_PWM_W_ENABLE                   (EMB_CTL1_PWMSEN0)
#define EMB_TMR4_PWM_V_ENABLE                   (EMB_CTL1_PWMSEN1)
#define EMB_TMR4_PWM_U_ENABLE                   (EMB_CTL1_PWMSEN2)
```

```
/**
 * @}
*/
```

选择检测 PWM 极性

```
/**
 * @defgroup EMB_Detect_TMR4_PWM_Level EMB Detect TMR4 PWM Level
 * @{
 */
#define EMB_DETECT_TMR4_PWM_X_BOTH_LOW          (0UL)
#define EMB_DETECT_TMR4_PWM_W_BOTH_LOW          (0UL)
#define EMB_DETECT_TMR4_PWM_V_BOTH_LOW          (0UL)
#define EMB_DETECT_TMR4_PWM_U_BOTH_LOW          (0UL)
#define EMB_DETECT_TMR4_PWM_X_BOTH_HIGH         (EMB_CTL2_PWMLV3)
#define EMB_DETECT_TMR4_PWM_W_BOTH_HIGH         (EMB_CTL2_PWMLV0)
#define EMB_DETECT_TMR4_PWM_V_BOTH_HIGH         (EMB_CTL2_PWMLV1)
#define EMB_DETECT_TMR4_PWM_U_BOTH_HIGH         (EMB_CTL2_PWMLV2)
/**
 * @}
*/
```

3.2.2 应用举例

EMB 可以检测正在输出 PWM 的端口，当端口发生同相时，EMB 能控制 PWM 停止输出。

```
/* EMB: initialize */
(void)EMB_TMR4_StructInit(&stcEmbInit);
stcEmbInit.stcTmr4.stcTmr4PwmW.u32PwmState = EMB_TMR4_PWM_W_ENABLE;
stcEmbInit.stcTmr4.stcTmr4PwmW.u32PwmLevel = EMB_DETECT_TMR4_PWM_W_BOTH_HIGH;
(void)EMB_TMR4_Init(EMB_GROUP, &stcEmbInit);

/* EMB: enable interrupt */
EMB_IntCmd(EMB_GROUP, EMB_INT_PWMS, ENABLE);

/* EMB: set release PWM condition */
```

```
EMB_SetReleasePwmCond(EMB_GROUP, EMB_EVT_PWMS, EMB_RELEASE_PWM_COND_FLAG_ZERO);


/* EMB: register IRQ handler && configure NVIC. */
(void)INTC_IrqInstallHandle(EMB_INT_IRQn, EMB_INT_SRC, DDL_IRQ_PRIO_DEFAULT, EMB_IRQ_Handler);
```

触发 EMB 后现象如下图所示：

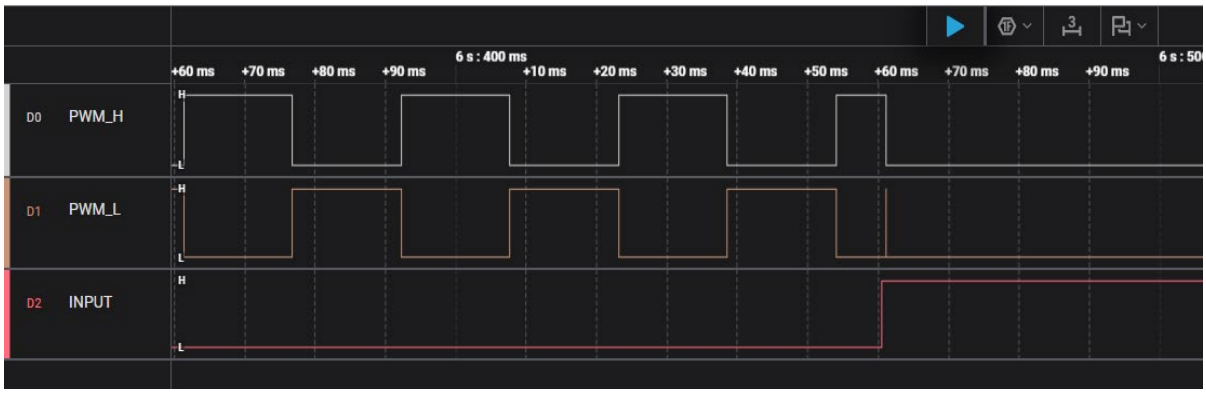

图 3-3 PWM 端口输出同相时，EMB 控制 PWM 停止输出

配置自动释放，则当监测条件失效后，EMB 控制 PWM 恢复输出：

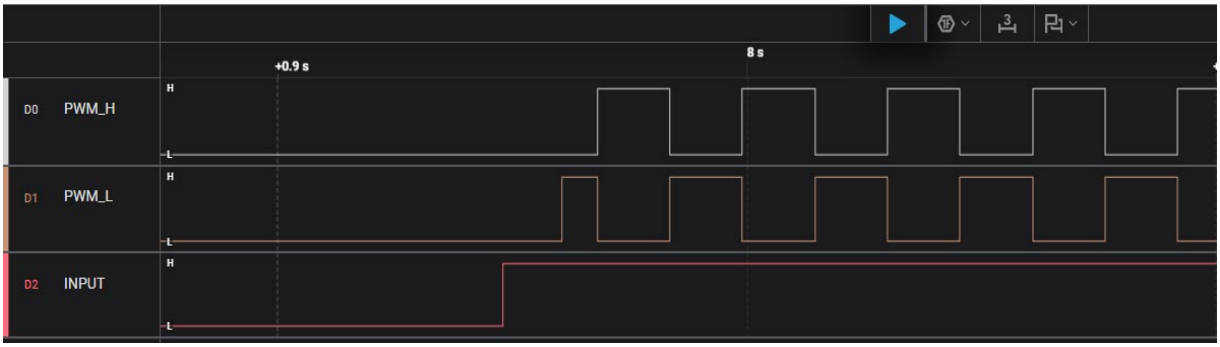

图 3-4 PWM 恢复互补波形后，EMB 控制 PWM 恢复输出

3.2.3 注意事项

在使用 EMB 同相检测功能时，在 PWM 输出使能情况下时，请勿切换 IO 功能；如需切换 IO 功能（GPIO<->PWM），请按如下步骤操作。

GPIO 功能切换到 PWM 输出功能时：

1) 关闭 PWM 输出*

2) 关闭 EMB PWM 输出同相检测中断功能

3) GPIO 功能切换到 PWM 输出功能

4) 清除 EMB PWM 同相检测标志位

5) 打开 PWM 输出*

6) 打开 EMB PWM 输出同相检测中断功能

*如果 *Timer* 功能不支持关闭 *PWM* 输出，请忽略步骤 *1*）和步骤 *5*)。

PWM 输出功能切换到 GPIO 功能：

1) 关闭 EMB PWM 输出同相检测中断功能

2) 关闭 PWM 输出*

3) PWM 输出功能切换到 GPIO 功能

*如果 *Timer* 功能不支持关闭 *PWM* 输出，请忽略步骤 *2*)。

3.3 电压比较器输出控制 PWM 信号输出

3.3.1 结构体定义

```
/**
 * @brief EMB monitor CMP configuration
 */
typedef struct {
    uint32_t u32Cmp1State;          /*!< Enable or disable EMB detect CMP1 result function
                                    This parameter can be a value of @ref EMB_CMP_Selection */
    uint32_t u32Cmp2State;          /*!< Enable or disable EMB detect CMP2 result function
                                    This parameter can be a value of @ref EMB_CMP_Selection */
    uint32_t u32Cmp3State;          /*!< Enable or disable EMB detect CMP3 result function
                                    This parameter can be a value of @ref EMB_CMP_Selection */
    uint32_t u32Cmp4State;          /*!< Enable or disable EMB detect CMP4 result function
                                    This parameter can be a value of @ref EMB_CMP_Selection */
} stc_emb_monitor_cmp_t;
```

可选配置为：

使能 CMP 检测

```
/**
 * @defgroup EMB_CMP_Selection EMB CMP Selection
 * @{
 */
#define EMB_CMP1_DISABLE                (0UL)
#define EMB_CMP2_DISABLE                (0UL)
#define EMB_CMP3_DISABLE                (0UL)
#define EMB_CMP4_DISABLE                (0UL)
#define EMB_CMP1_ENABLE                 (EMB_CTL1_CMPEN1)
#define EMB_CMP2_ENABLE                 (EMB_CTL1_CMPEN2)
#define EMB_CMP3_ENABLE                 (EMB_CTL1_CMPEN3)
```

```
#define EMB_CMP4_ENABLE                         (EMB_CTL1_CMPEN4)
/**
 * @}
*/
```

3.3.2 应用举例

EMB 可以根据 CMP 输出结果控制 PWM 输出。

CMP 配置：

```
stc_cmp_init_t stcCmpInit;
stc_gpio_init_t stcGpioInit;

/* CMP compare voltage pin */
(void)GPIO_StructInit(&stcGpioInit);
stcGpioInit.u16PinAttr = PIN_ATTR_ANALOG;
(void)GPIO_Init(CMP_INP_PORT, CMP_INP_PIN, &stcGpioInit);

/* Enable peripheral Clock */
CMP_FCG_ENABLE();

/* Configuration for normal compare function */
(void)CMP_StructInit(&stcCmpInit);
stcCmpInit.u16PositiveInput = CMP_POSITIVE_INP1;
stcCmpInit.u16NegativeInput = CMP_NEGATIVE_INM3;    /* DAO */
stcCmpInit.u16OutPolarity = CMP_OUT_INVT_OFF;
stcCmpInit.u16OutDetectEdge = CMP_DETECT_EDGS_BOTH;
stcCmpInit.u16OutFilter = CMP_OUT_FILTER_CLK_DIV32;
(void)CMP_NormalModeInit(CMP_UNIT, &stcCmpInit);

/* Enable CMP output */
CMP_CompareOutCmd(CMP_UNIT, ENABLE);

CMP_PinVcoutCmd(CMP_UNIT, ENABLE);
```

EMB 配置：

```
/* EMB: initialize */
(void)EMB_TMR4_StructInit(&stcEmbInit);
stcEmbInit.stcCmp.u32Cmp1State = EMB_CMP1_ENABLE;
(void)EMB_TMR4_Init(EMB_GROUP, &stcEmbInit);

/* EMB: enable interrupt */
EMB_IntCmd(EMB_GROUP, EMB_INT_CMP, ENABLE);

/* EMB: set release PWM condition */
```

```
EMB_SetReleasePwmCond(EMB_GROUP, EMB_EVT_CMP, EMB_RELEASE_PWM_COND_FLAG_ZERO);

/* EMB: register IRQ handler && configure NVIC. */
(void)INTC_IrqInstallHandle(EMB_INT_IRQn, EMB_INT_SRC, DDL_IRQ_PRIO_DEFAULT, EMB_IRQ_Handler);
```

CMP 输出满足条件时，EMB 控制 PWM 停止输出：

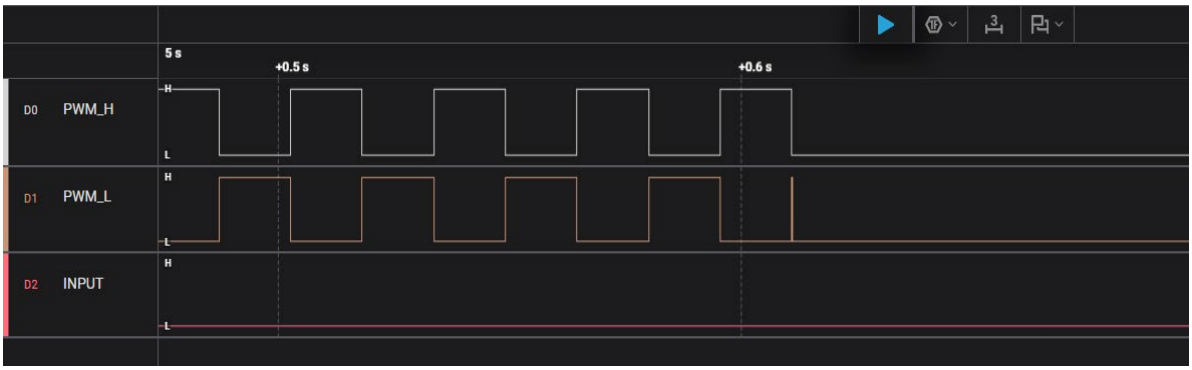

图 3-5　CMP 输出有效时，EMB 控制 PWM 停止输出

配置自动释放功能，当 CMP 输出无效时，EMB 控制 PWM 恢复输出：

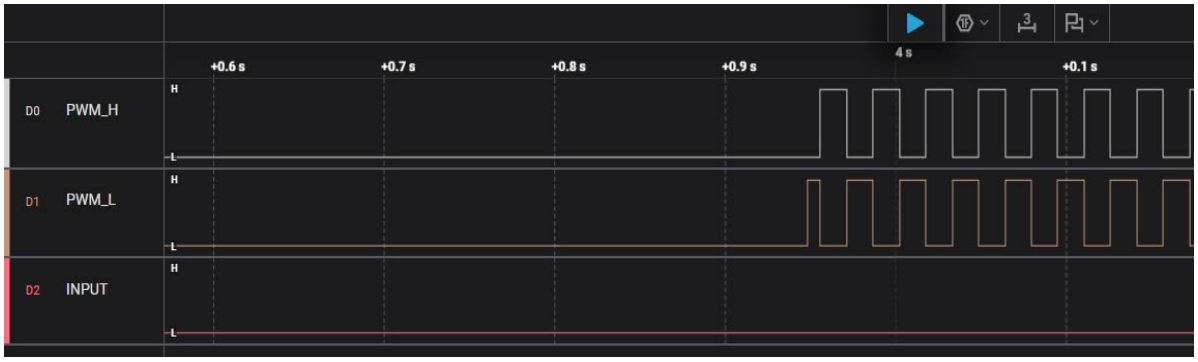

图 3-6　CMP 输出无效时，EMB 控制 PWM 恢复输出

3.4 系统错误控制 PWM 信号输出

3.4.1 结构体定义

```
/**
 * @brief EMB monitor system exception configuration
 */
typedef struct {
    uint32_t u32Osc;                    /*!< Enable or disable EMB detect OSC failure function
                                         This parameter can be a value of @ref EMB_OSC_Selection */
    uint32_t u32SramEccError;           /*!< EMB detect SRAM ECC error function
                                         This parameter can be a value of @ref EMB_SRAM_ECC_Error_Selection */
    uint32_t u32SramParityError;        /*!< EMB detect SRAM parity error function
                                         This parameter can be a value of @ref EMB_SRAM_Parity_Error_Selection */
    uint32_t u32Lockup;                 /*!< EMB detect lockup function
                                         This parameter can be a value of @ref EMB_Lockup_Selection */
```

```
    uint32_t u32Lvd;                    /*!< EMB detect LVD function
                                         This parameter can be a value of @ref EMB_LVD_Selection */
} stc_emb_monitor_sys_t;
```

可选配置：

使能 XTAL 停振检测

```
/**
 * @defgroup EMB_OSC_Selection EMB OSC Selection
 * @{
 */
#define EMB_OSC_DISABLE                         (0UL)
#define EMB_OSC_ENABLE                          (EMB_CTL1_OSCSTPEN)
/**
 * @}
 */
```

使能 SRAMECC 检测

```
/**
 * @defgroup EMB_SRAM_ECC_Error_Selection EMB SRAM ECC Error Selection
 * @{
 */
#define EMB_SRAM_ECC_ERR_DISABLE                (0UL)
#define EMB_SRAM_ECC_ERR_ENABLE                 (EMB_CTL1_SRAMECCERREN)
/**
 * @}
     */
```

使能 SRAM 奇偶检测

```
/**
 * @defgroup EMB_SRAM_Parity_Error_Selection EMB SRAM Parity Error Selection
 * @{
 */
#define EMB_SRAM_PARITY_ERR_DISABLE             (0UL)
#define EMB_SRAM_PARITY_ERR_ENABLE              (EMB_CTL1_SRAMPYERREN)
/**
 * @}
*/
```

使能 Lockup 检测

```
/**
 * @defgroup EMB_Lockup_Selection EMB Lockup Selection
 * @{
 */
#define EMB_LOCKUP_DISABLE                      (0UL)
```

```
#define EMB_LOCKUP_ENABLE                    (EMB_CTL1_LOCKUPEN)
/**
 * @}
 */
```

使能 LVD 检测

```
/**
 * @defgroup EMB_LVD_Selection EMB LVD Selection
 * @{
 */
#define EMB_LVD_DISABLE                      (0UL)
#define EMB_LVD_ENABLE                       (EMB_CTL1_PVDEN)
/**
 * @}
*/
```

3.4.2 应用举例

当发生系统错误时，EMB 模块可以控制 PWM 输出，此处展示为 XTAL 时钟出现故障触发 EMB 控制 PWM 输出功能。

XTAL 配置：

```
stc_clock_xtal_init_t stcXtalInit;

/* Configure XTAL */
GPIO_AnalogCmd(BSP_XTAL_PORT, BSP_XTAL_PIN, ENABLE);
(void)CLK_XtalStructInit(&stcXtalInit);
stcXtalInit.u8State = CLK_XTAL_ON;
(void)CLK_XtalInit(&stcXtalInit);

/* Configure XTAL fault dectect. */
(void)CLK_XtalStdInit(CLK_XTALSTD_ON, CLK_XTALSTD_EXP_TYPE_INT);
```

EMB 配置：

```
/* EMB: initialize */
(void)EMB_TMR4_StructInit(&stcEmbInit);
stcEmbInit.stcOsc.u32OscState = EMB_OSC_ENABLE;
(void)EMB_TMR4_Init(EMB_GROUP, &stcEmbInit);

/* EMB: enable interrupt */
EMB_IntCmd(EMB_GROUP, EMB_INT_OSC, ENABLE);

/* EMB: set release PWM condition */
EMB_SetReleasePwmCond(EMB_GROUP, EMB_INT_OSC, EMB_RELEASE_PWM_COND_FLAG_ZERO);
```

```
/* EMB: register IRQ handler && configure NVIC. */
(void)INTC_IrqInstallHandle(EMB_INT_IRQn, EMB_INT_SRC, DDL_IRQ_PRIO_DEFAULT, EMB_IRQ_Handler);
```

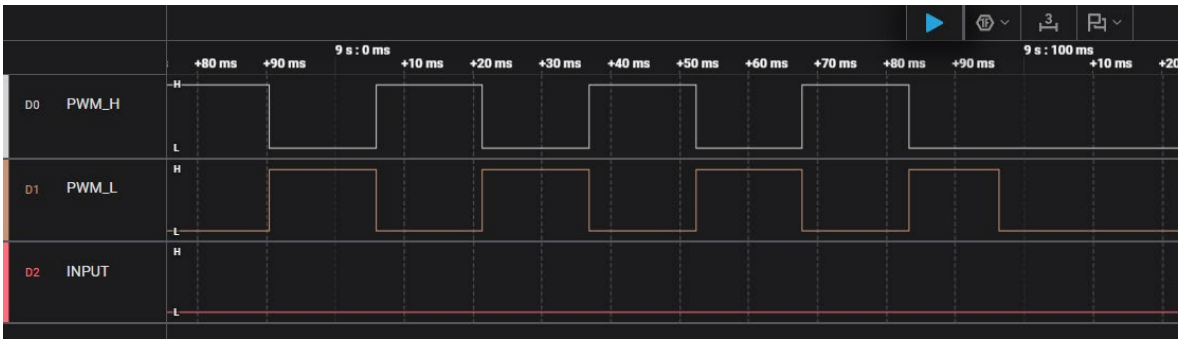

图 3-7　XTAL 振荡器停止时，EMB 控制 PWM 停止输出

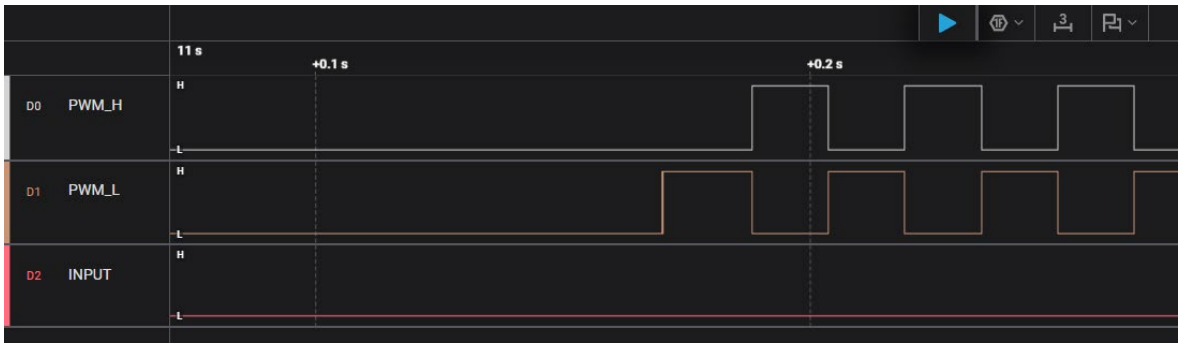

图 3-8　XTAL 振荡器恢复后，EMB 控制 PWM 恢复输出

下图是系统时钟为 XTAL（8MHz），设置 Timer4 输出 PWM，人为制造 XTAL 振荡异常后测试对应波形，实际测试结果小于 1μs。

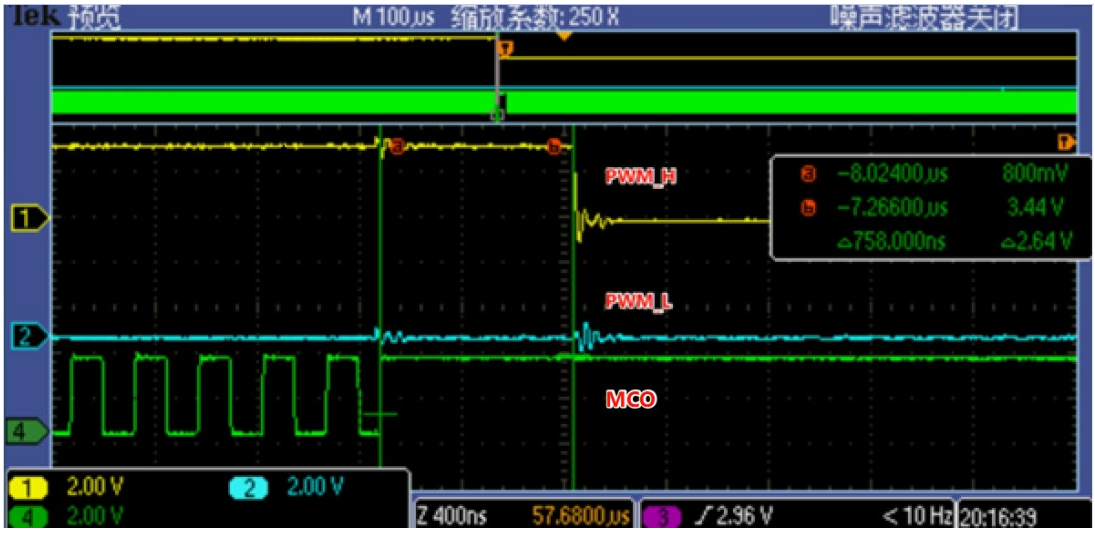

图 3-9　XTAL 故障后停止 PWM 所需时间

3.5 软件控制 PWM 信号输出

3.5.1 接口函数

```
void EMB_SWBrake(CM_EMB_TypeDef *EMBx, en_functional_state_t enNewState);
```

3.5.2 应用举例

EMB 模块也可以通过软件写寄存器的方式控制 PWM 输出。

EMB 配置：

```
/* Wait key pressed */
  while (RESET == KEY_State()) {
  }

/* Software start brake signal */
  EMB_SWBrake(EMB_GROUP, ENABLE);

  /* Wait key pressed */
  while (RESET == KEY_State()) {
  }

  /* Software stop brake signal */
EMB_SWBrake(EMB_GROUP, DISABLE);
```

通过软件写 SOE 寄存器，可以使 EMB 控制 PWM 停止输出：

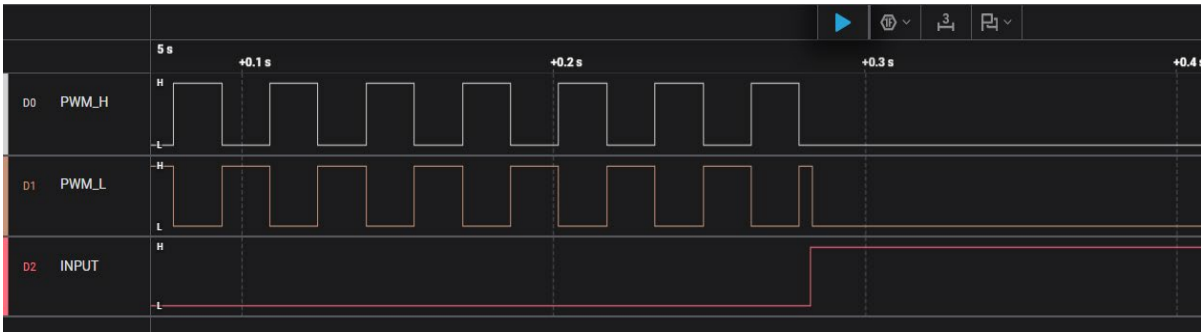

图 3-10 写 SOE 寄存器，EMB 控制 PWM 停止输出

通过软件写 SOE 寄存器，可以使 EMB 控制 PWM 恢复输出：

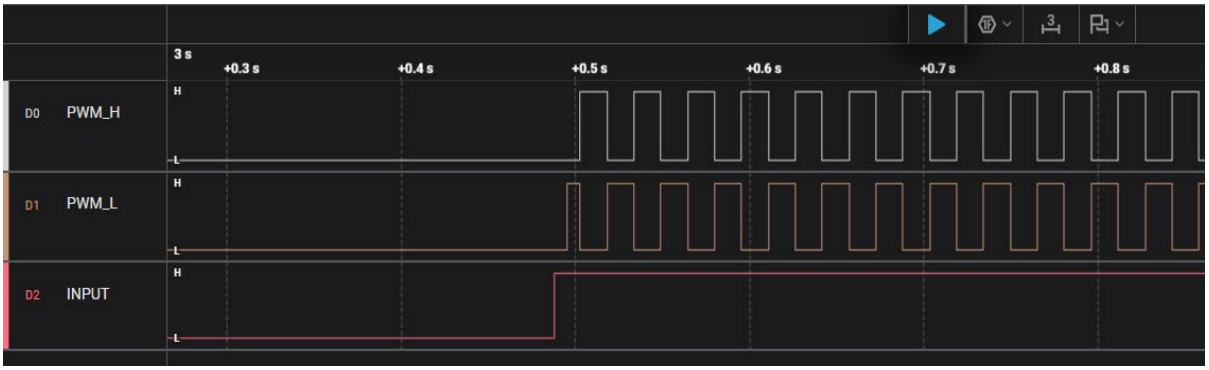

图 3-11 写 SOE 寄存器，EMB 控制 PWM 恢复输出

3.6 其他注意事项

1. EMB 控制寄存器（EMB_CTL1/EMB_CTL2*）写入后仅复位后才能再次写入

2. 写寄存器 EMB_SOEx 产生 EMB 控制信号，不会产生中断请求

3. EMB 检测条件满足后，定时器 PWM 极性更新需 3 个 EMB 模块时钟；如使能了滤波功能，还要加上滤波时间

*部分芯片无 *EMB_CTL2* 寄存器。

4 总结

本文档主要介绍小华半导体 HC32 系列芯片 EMB 功能使用，给客户实际使用提供了参考。

版本修订记录

版本号	修订日期	修订内容
Rev1.00	2025/11/05	初版发布。
Rev1.01	2026/07/23	新增适用对象：HC32F431、HC32M413、HC32M458。