PWM 使用指南

概述

PWM 在电力电子系统中有着至关重要的作用，广泛的应用在电机控制、开关电源及 UPS 使用场景中。

适用范围	
SPC1125 系列	SPC1125，SPC1128
SPC1168 系列	SPC1155，SPC1156，SPC1158，SPC1168，SPD1148，SPD1178，SPD1188，SPD1163，SPM1173
SPC2168 系列	SPC2168，SPC2165，SPC2166，SPC1198
SPC1169 系列	SPC1169，SPD1179，SPD1176
SPC2188 系列	SPC1185，SPC2188

目录

图片列表

表格列表

版本历史

版本	日期	作者	状态	变更
A/0	2023-05-11	Hang Su	Outdated	1. 首次发布。
C/0	2024-08-13	LemengZhou	Released	1. 修改为全系列通用文档。

术语或缩写

术语或缩写	描述
PWM	Pulse Width Modulation

1 特性

PWM 单元有以下特点：

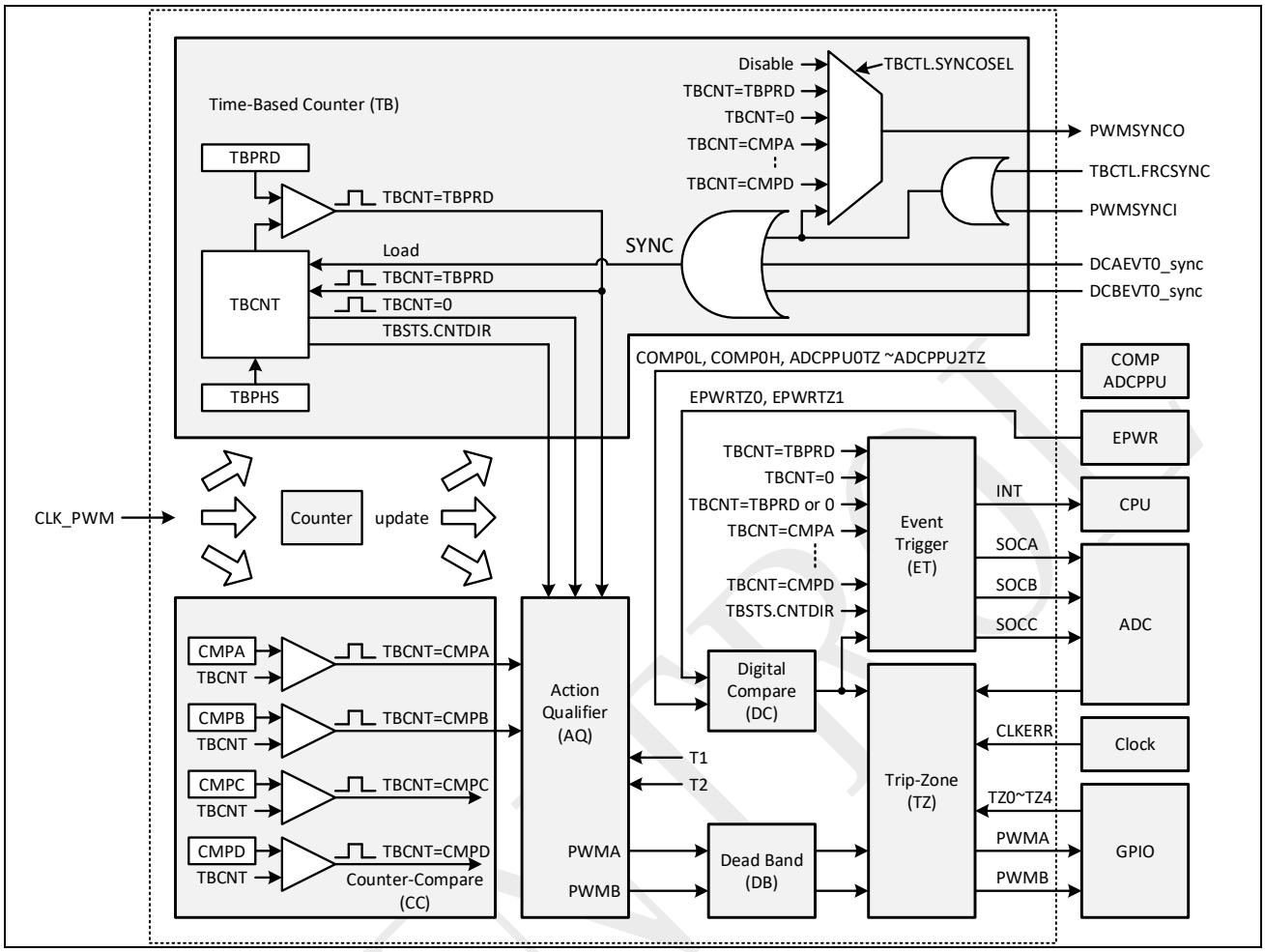
- 周期可配置的 16-bit 时基计数
- 软件直接强制改变 PWM 输出
- 独立可配的相位控制
- 周期性的相位同步
- 上升沿延迟和下降沿延迟独立可配置的死区控制
- 异常事件下可配的周期性或者一次性封锁
- 封锁时可以配置 PWM 输出为高、低或高阻
- 数字比较或者封锁信号输入可产生原始封锁事件或者过滤后的封锁事件
- 所有事件可以用于产生 CPU 中断或者 ADC REQ（开始采样）信号

2 功能描述

PWM 模块由如下子模块组成：

- 时基产生（TB）子模块
- 计数比较（CC）子模块
- 行为限定（AQ）子模块
- 死区控制（DB）子模块
- 信号封锁（TZ）子模块
- 数字比较（DC）子模块
- 事件触发（ET）子模块

图 2-1：PWM 单元功能框图



3 功能实例

3.1 PWM 基础功能

3.1.1 实例 1：单通道固定频率 PWM

3.1.1.1 功能需求

产生一个单通道 PWM 波形，具备以下要求：

- 占空比为 50%；
- 频率为 20KHz；

3.1.1.2 功能实现

需要通过如下步骤实现功能：

1. 确定 PWM_Clock_Freq：本实例以 PWM_Clock_Freq 为 100MHz 为基础，进行后续配置值计算。
2. 计算 TBPRD：

$$TBPRD = PWM\_Clock\_Freq / PWM\_Freq/2$$

其中 PWM_Freq 为需求的 20KHz，经计算 TBPRD 为 2500。

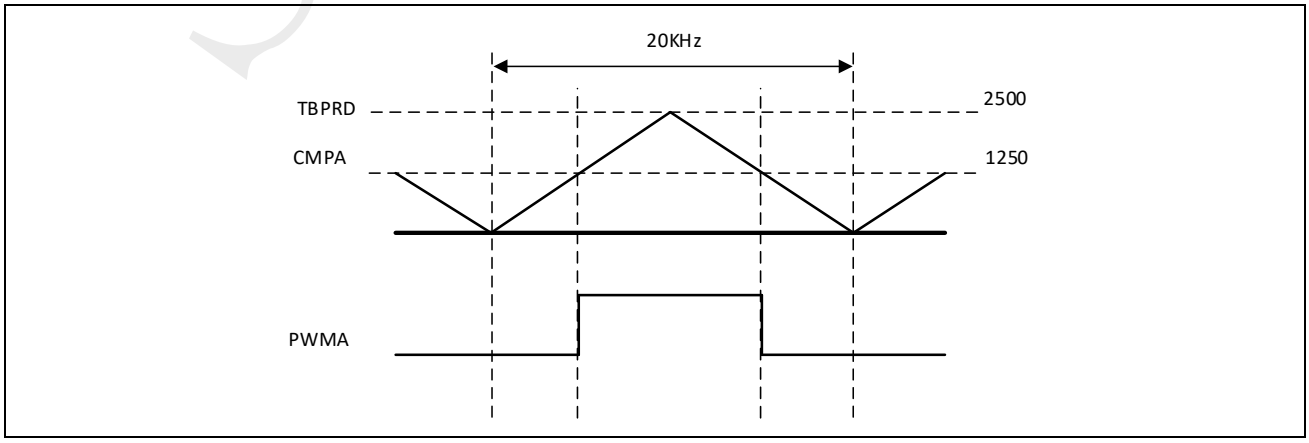
3. 配置 PWM 计数方式为上下计数模式。此计数模式下，当计数器向上计数达到 CMP 时，将波形拉高，向下计数达到 CMP 时，将波形拉低；
4. 计算 CMP：

$$Duty\_Ratio = CMP/TBPRD$$

其中 Duty_Ratio 为需求的 50%，经计算 CMP 为 1250。

5. 配置 GPIO 功能为 PWM 输出。
6. 运行 PWM 模块。
7. 通过以上实现步骤，可产生如下图 3-1 示例波形：

图 3-1：单通道独立 PWM



以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-1：

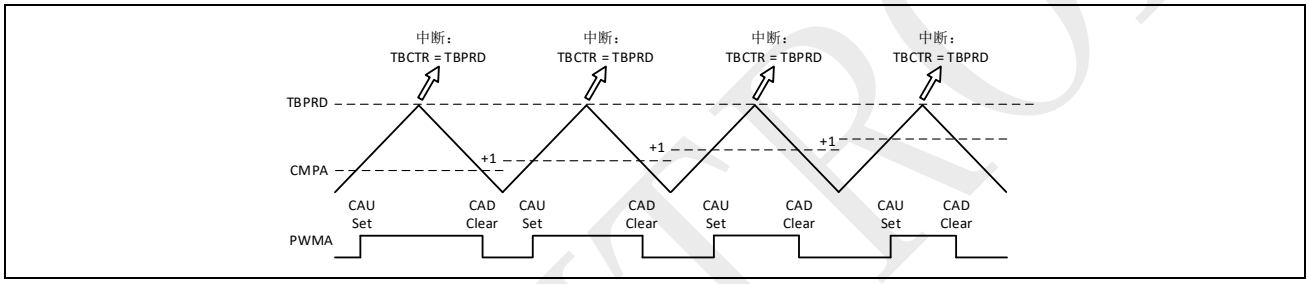
表 3-1：实例 1 代码路径

MCU 产品型号	代码路径
所有型号	SDK 目录\0_Examples\ PWM_Single_Output_With_Up_Down_Counting_Mode

3.1.2 实例 2：波形占空比调节

PWM 可以在时基的计数器值 TBCTR 等于 TBPRD 时刻，产生中断事件。并在中断时间服务函数中通过调整 CMP 影子寄存器的值，以实现调整占空比，如图 3-2 所示：

图 3-2：周期性变化 PWM



3.1.2.1 功能需求

产生一个单通道 PWM 波形，具备以下要求：

- 周期性调整 PWM 占空比从 0~100%。
- 频率为 20KHz；

3.1.2.2 功能实现

需要通过如下步骤实现功能：

1. 确定 PWM_Clock_Freq：本实例以 PWM_Clock_Freq 为 100MHz 为基础，进行后续配置值计算。
2. 计算 TBPRD：

$$TBPRD = PWM\_Clock\_Freq / PWM\_Freq/2$$

其中 PWM_Freq 为需求的 20KHz，经计算 TBPRD 为 2500。

3. 配置 PWM 计数方式为上下计数模式。此计数模式下，当计数器向上计数达到 CMP 时，将波形拉高，向下计数达到 CMP 时，将波形拉低；
4. 计算 CMP：

$$Duty\_Ratio = CMP/TBPRD$$

其中 Duty_Ratio 为需求的 50%，经计算 CMP 为 1250。

5. 配置 GPIO 功能为 PWM 输出。

6. 设置 CMP 影子寄存器的向 CMPA 寄存器的数值加载时机为，计数器值为零时加载。
7. 使能 PWM 在 TBCTR 等于 TBPRD 时刻，产生中断事件。
8. 使能 CPU 处理 PWM 中断事件。
9. 编写中断服务函数，中断服务函数中修改 CMP 值，用以调节占空比。
10. 运行 PWM 模块。

以上实现步骤的示例代码如下：

Example Code

```
int main(void)
{
    CLOCK_InitWithRCO(CLOCK_CPU_100MHZ);

    Delay_Init();

    /*
     * Init the UART
     */
    PIN_SetChannel(PIN_GPIO10, PIN_GPIO10_UART0_TXD);
    PIN_SetChannel(PIN_GPIO11, PIN_GPIO11_UART0_RXD);
    UART_Init(UART0, 38400);

    /* Init PWM1 and output 20kHz waveform on channel A */
    PWM_InitSingleChannel(PWM1, PWM_CHA, PWM_Frequency);

    /* Set PWM1A output 50% duty waveform */
    u32PWMPeriod = PWMPeriod(PWM_Frequency);
    PWM_SetCMPA(PWM1, u32PWMPeriod / 2);

    /* Select PIN_GPIO12 as the channel A output of PWM1 */
    PIN_SetChannel(PIN_GPIO12, PIN_GPIO12_PWM1A);

    /* Set CMPA load time */
    PWM_SetCMPALoadTiming(PWM1, CMPCTL_LOAD_ON_ZERO);

    /* Enable PWM1 TBCTR = TBPRD event */
    PWM_SetTimeEventIntTiming(PWM1, EQU_PERIOD);
    PWM_SetTimeEventIntPeriod(PWM1, ON_1ST_EVENT);
    PWM_EnableTimeEventInt(PWM1);

    /* Enable PWM1 Interrupt in CPU side */
    NVIC_EnableIRQ(PWM1_IRQn);

    /* Start PWM1 */
    PWM_RunCounter(PWM1);

    while (1)
    {

    }
}

void PWM1_IRQHandler(void)
{
```

Example Code

```
    uint16_t u16CMPAVal = 0;

    u16CMPAVal = PWM1->CMPA;

    /* Change PWM duty */
    if((u16CMPAVal +1) >= PWM1->TBPRD)
    {
        PWM_SetCMPA(PWM1,0);
    }
    else
    {
        PWM_SetCMPA(PWM1, u16CMPAVal + 1);
    }

    /* Clear interrupt flag */
    PWM_ClearTimeEventInt(PWM1);
}
```

[1] 示例代码适用于 SPC1169 系列产品，其它系列产品的示例代码会根据实际需求进行补充。

3.1.3 实例 3：两路互补、带死区 PWM 波

在电力电子、开关电源以及电机控制中，往往要应对半桥电路、全桥电路；这种电路的一个基本控制方法，就是用互补的 PWM 波去分别驱动半桥、全桥的上下桥臂开关管，同时互补的 PWM 波还需要带有一定的死区时间。

对于半桥控制中的死区应用，一般是通过将一个 PWM 波增加死区时间后，取反得到带死区的互补 PWM 波。PWM 模块具备灵活的死区配置方式，如下图 3-3 所示。

图 3-3 和图 3-4，展示了如何从一路 PWM 波，最终称为一对带死区互补 PWM 波的过程。

图 3-3：死区控制单元框图

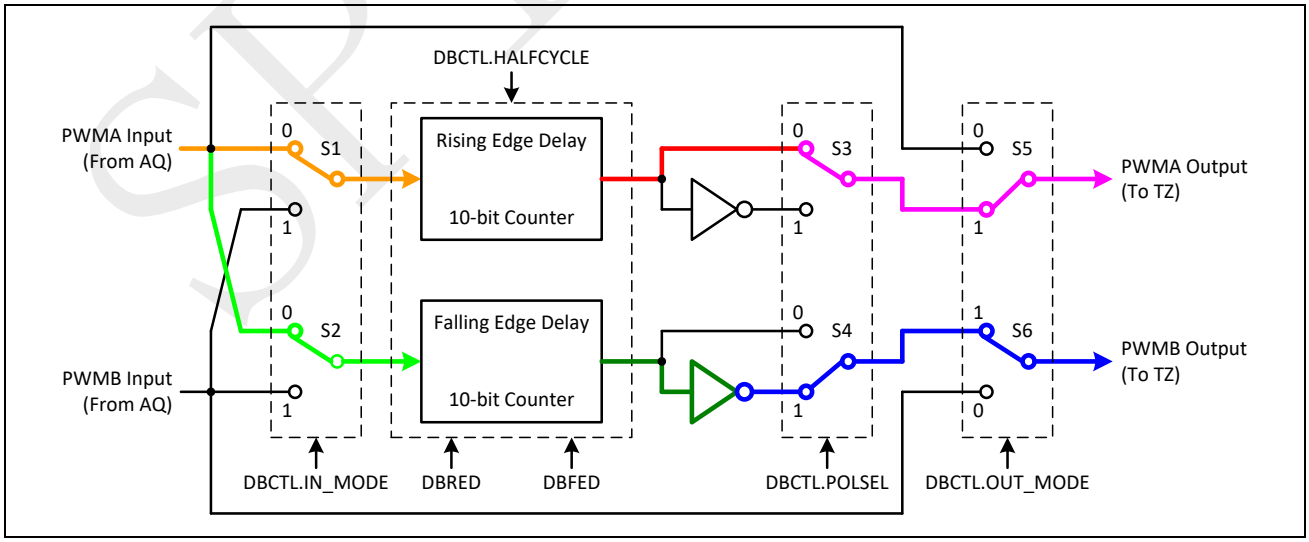
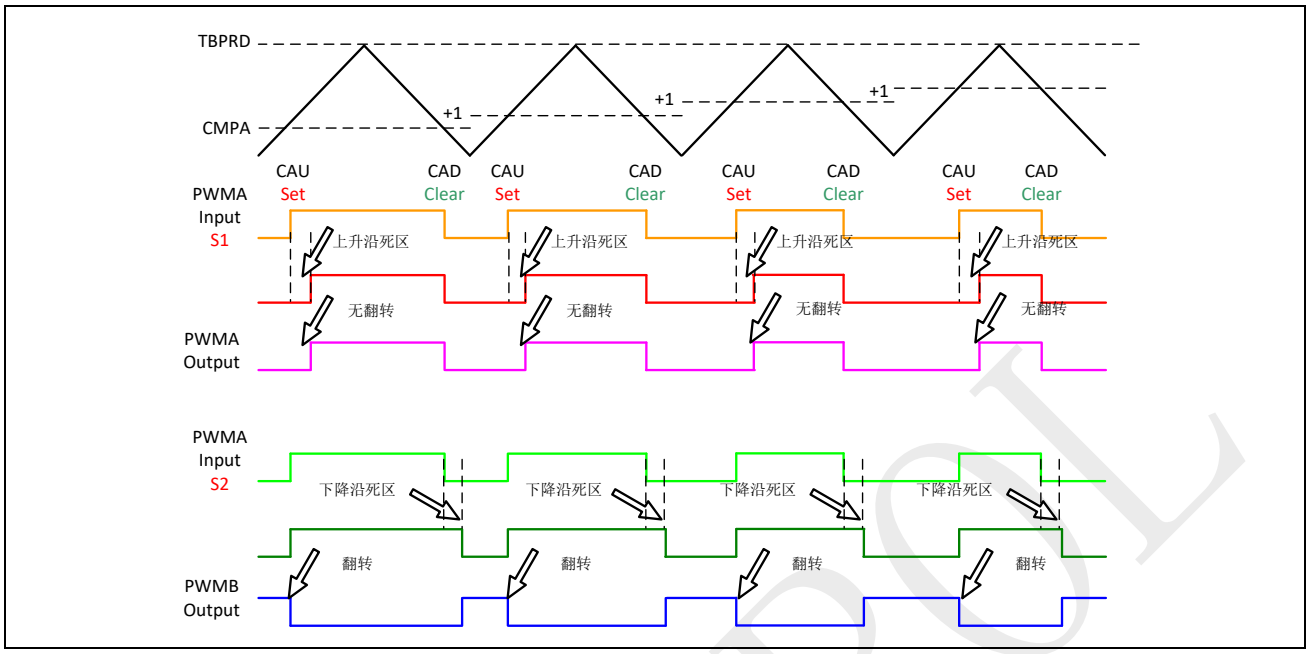


图 3-4：生成带死区互补 PWM 示意图



3.1.3.1 功能需求

产生一个半桥带有死区的 PWM 波。

- PWM 占空比为 75%；
- 频率为 20KHz；
- 死区时间为 1000ns。

3.1.3.2 功能实现

1. 确定 PWM_Clock_Freq：本实例以 PWM_Clock_Freq 为 100MHz 为基础，进行后续配置值计算。
2. 计算 TBPRD：

$$TBPRD = PWM\_Clock\_Freq / PWM\_Freq/2$$

其中 PWM_Freq 为需求的 20KHz，经计算 TBPRD 为 2500。

3. 配置 PWM 计数方式为上下计数模式。此计数模式下，当计数器向上计数达到 CMP 时，将波形拉高，向下计数达到 CMP 时，将波形拉低；
4. 计算 CMP：

$$Duty\_Ratio = CMP/TBPRD$$

其中 Duty_Ratio 为需求的 75%，经计算 CMP 为 725。

5. 配置 PWM 为双通道互补输出。
6. 配置死区时间为 1000ns。
7. 配置 GPIO 功能为 PWM 输出。

8. 运行 PWM 模块。

以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-2。

表 3-2：实例 3 代码路径

MCU 产品型号	代码路径
所有型号	SDK 目录\0_Examples\ PWM_Complementary_Pair_Channel

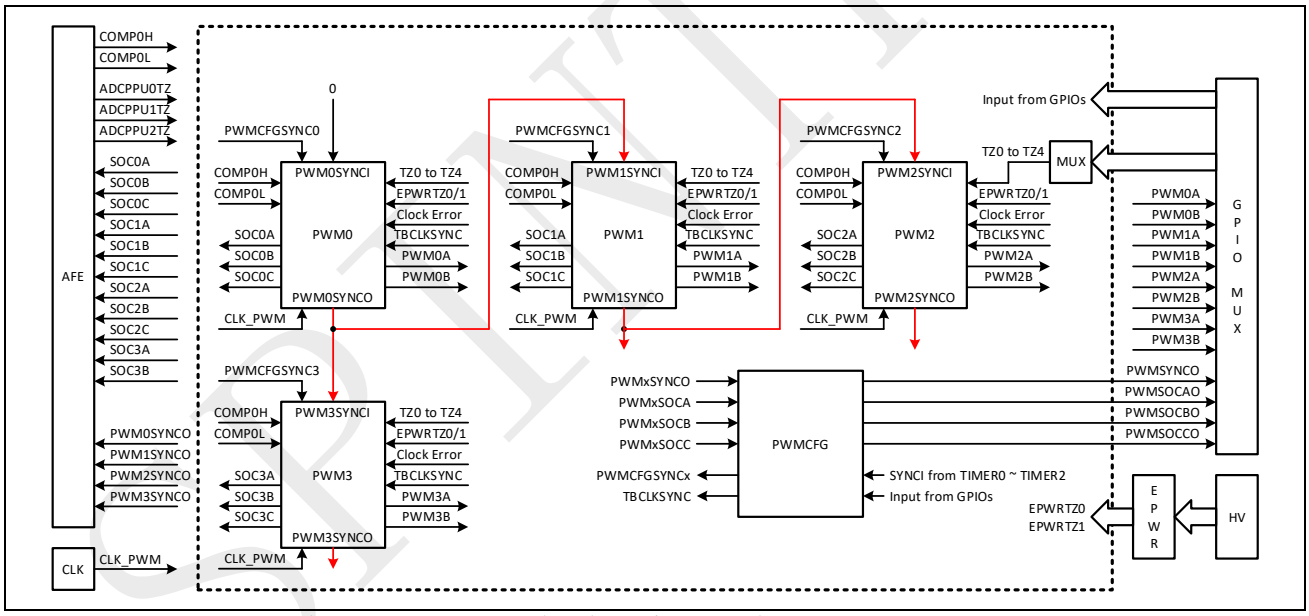
3.2 PWM 同步

在使用 PWM 过程中，有时需要在不同 PWM 间产生一固定相位差，这时可以用 PWM 级联同步或非级联同步实现。

3.2.1 实例 4：级联方式实现 PWM 同步

级联同步指的是同步信号在不同 PWM 模块中依次传导，如图 3-5 所示。

图 3-5：级联关系



[1] 上图以 SPC1169 为例，可以建立级联同步关系的链路为 PWM0->PWM1->PWM2 或 PWM0->PWM3

注意：
1. PWMSYNCI 除了来自 PWMSYNCO，还可以来自 Timer 或 GPIO。
2. 不同型号产品的 PWMx 数量与级联关系不一致，可参考对应型号产品 TRM 手册中的“PWM 系统总览”图确认。

3.2.1.1 功能需求

使用级联功能，将 PWM0、PWM1、PWM2 进行级联同步，输出 PWM 波形。

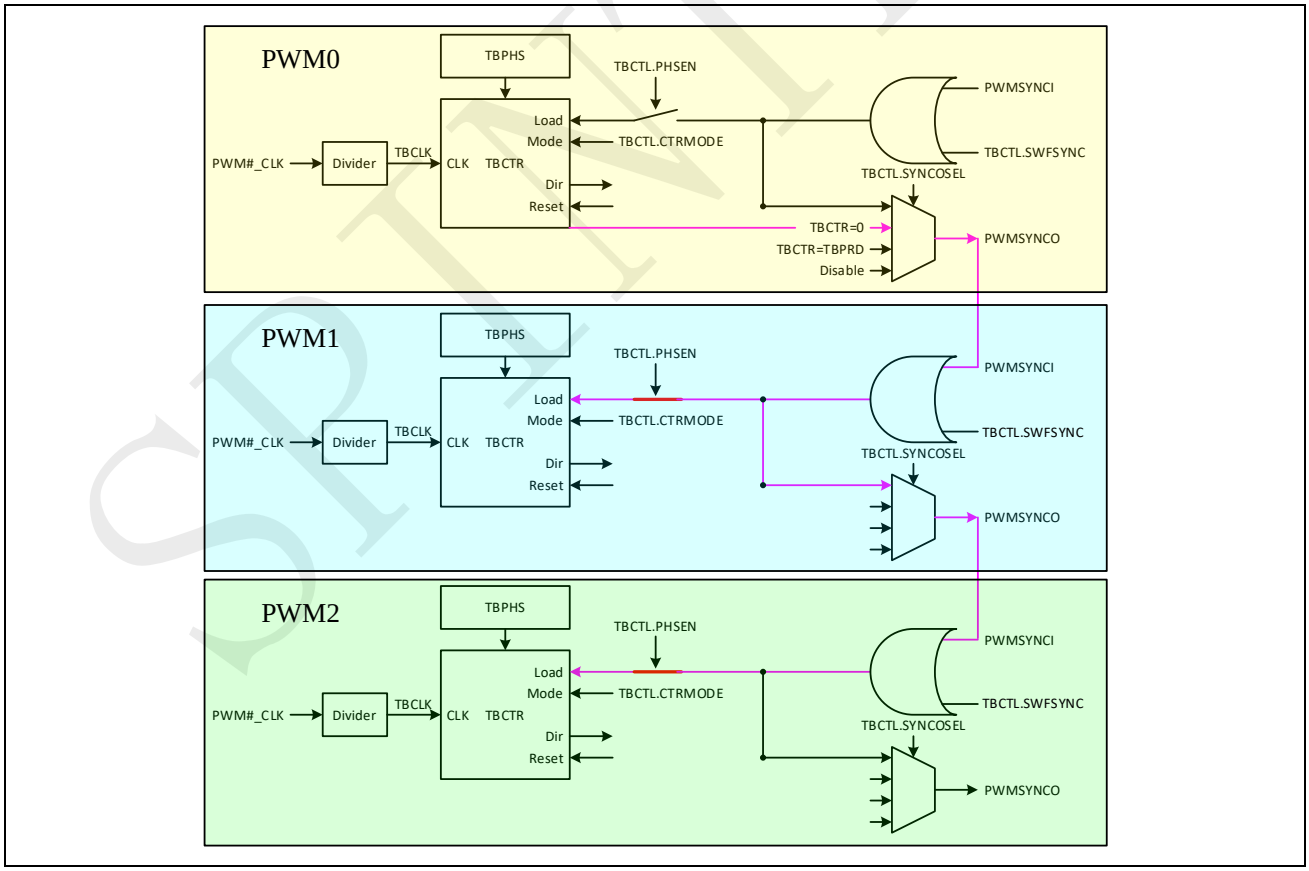
- PWM0、PWM1、PWM2 都为上下计数模式。
- PWM0、PWM1、PWM2 频率都为 20KHz。
- PWM0、PWM1、PWM2 占空比都为 33.33%。
- 当 PWM0 计数器为 0 时级联同步 PWM1、PWM2，使得 PWM0、PMW1、PWM2 产生 120° 相位差。

3.2.1.2 功能实现

需要通过如下步骤实现功能：

1. 确定级联同步的同步关系：
   a) PWM0 使能 TBCTR=0 信号通过 PWMSYNCO 的输出到 PWM1；
   b) PWM1 使能 PWMSYNCI 信号通过 PWMSYNCO 的输出到 PWM2；
   c) 级联同步关系框图，如图 3-6 所示：

图 3-6：级联式同步



2. 确定相位差产生方式：
   a) 同步时基相位值应为计数比较器的值。
   b) PWM1、PWM2 都设置为上下计数模式。

c) PWM1 配置为 TBCNT=CMPA 事件发生且 TBCNT 递增计数时，转换为高电平。
d) PWM1 配置为 TBCNT=CMPA 事件发生且 TBCNT 递减计数时，转换为低电平。
e) PWM2 配置为 TBCNT=CMPA 事件发生且 TBCNT 递增计数时，转换为低电平。
f) PWM2 配置为 TBCNT=CMPA 事件发生且 TBCNT 递减计数时，转换为高电平。
其他基础需求请参考本文 3.1 章节。

3. 以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-2：

表 3-3：实例 4 代码路径

产品型号	代码路径
SPC1125 系列，SPC1169 系列，SPC2188 系列	SDK 目录\0_Examples\PWM_Software_Group_SYNC
SPC1168 系列，SPC2168 系列	SDK 目录 \0_Examples\ PWM_TBCNT_0_SYNC

3.2.2 实例 5：非级联方式实现 PWM 同步

非级联同步指的是同步信号同时加在不同 PWM 模块上。

3.2.2.1 功能需求

使用软件强制同步功能，将 PWM0、PWM1、PWM2 进行级联同步，输出 PWM 波形。

- PWM0、PWM1、PWM2 都为上下计数模式。
- PWM0、PWM1、PWM2 频率都为 20KHz。
- PWM0、PWM1、PWM2 占空比都为 33.33%。
- 软件强制同步施加时：

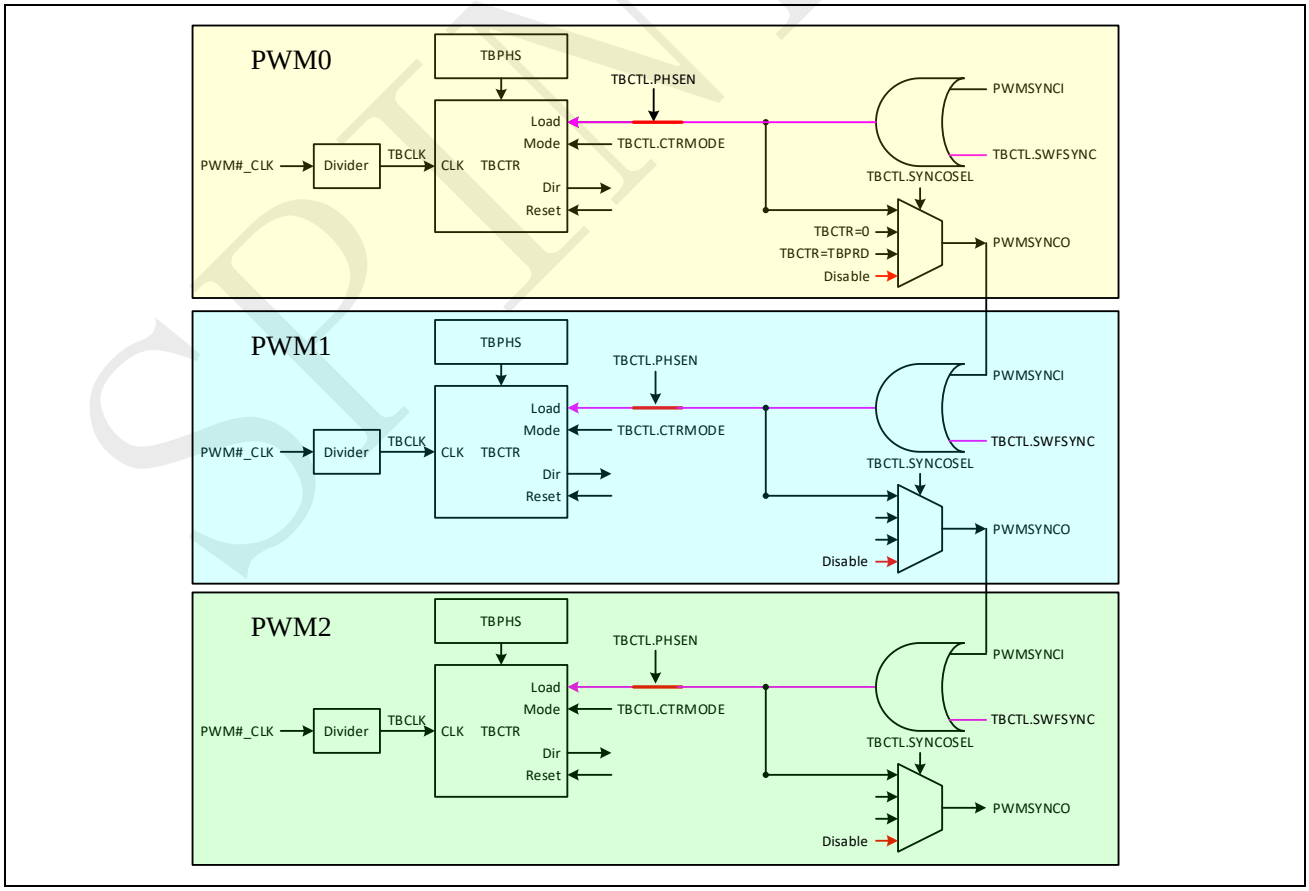
  PWM0、PMW1、PWM2 产生 120° 相位差。

3.2.2.2 功能实现

需要通过如下步骤实现功能：

1. 确认同步关系：
   a) 使用软件强制同步信号 TBCTL.SWFSYNC 同时同步 PWM0、PWM1、PWM2。
   b) 同步关系如下图所示：

图 3-7：非级联式同步



2. 确定时基计数器 TBCNT 等于计数比较器 CMPA 时的电平翻转状态。
   a) PWM0、PWM1、PWM2 都设置为上下计数模式。
   b) PWM0 配置为 TBCNT=CMPA 事件发生且 TBCNT 递增计数时，转换为高电平。
   c) PWM0 配置为 TBCNT=CMPA 事件发生且 TBCNT 递减计数时，转换为低电平。
   d) PWM2 配置为 TBCNT=CMPA 事件发生且 TBCNT 递增计数时，转换为低电平。
   e) PWM2 配置为 TBCNT=CMPA 事件发生且 TBCNT 递减计数时，转换为高电平。
   f) PWM1 需求中没有特殊要求，配置为和 PWM1 一致。
3. 关闭各 PWM 模块的 PWMSYNCO 功能。
4. 采用软件同时改变 PWM0，PWM1，PWM2 的相位。
5. 其他基础需求请参考本文 3.1 章节。

以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-2：

表 3-4：实例 5 代码路径

MCU 产品型号	代码路径
SPC1125 系列，SPC1169 系列，SPC2188 系列	SDK 目录\0_Examples\PWM_Software_Global_SYNC
SPC1168 系列，SPC2168 系列	SDK 目录\0_Examples\PWM_Global_Software_Force_SYNC

3.2.3 观测同步输出信号

可以将 PWM 同步输出信号通过具有 PWMSOCO 功能的引脚送出，输出给示波器观测。

Example Code

```
    /* set the GPIO to observe the SYNC signal*/
    #if defined(SPC1169)
    PIN_SetChannel(PIN_GPIO18, PIN_GPIO18_PWMSYNCO);
    #else
    PIN_SetChannel(PIN_GPIO31, PIN_GPIO31_PWMSYNCO);
    #endif
    PWMCFG->SYNCOCTL = (PWMCFG->SYNCOCTL & (~SYNCOCTL_SYNCO0EN_Msk)) |
SYNCOCTL_SYNCO0EN_ENABLE;
    PWMCFG->SYNCOCTL = (PWMCFG->SYNCOCTL & (~SYNCOCTL_DURATION_Msk)) |
SYNCOCTL_DURATION_32_PWM_CLK;
```

[1] 示例代码适用于 SPC1169 系列，其它系列产品的示例代码会根据实际需求进行补充。

3.3 PWM Trip Zone

在很多应用场景里头，存在类似紧急停车的信号，当这个信号来到时候，需要立即或者安全的停止一切动作。如果把 PWM 的输出看成这种动作或者动作的驱动信号，同样有着立即进行安全停止动作的需求，则可通过如下方式实现：

- 按周期（Cycle-by-Cycle，CBC）封锁 PWM 输出，会在下一个 PWM 周期重新启动（适用于电源控制中的定电流控制，或是步进电机中的恒流细分控制）；
- 一次性封锁（One Shot，OST）PWM 输出，One-shot 则是会直接停止 PWM 输出，直到使用者介入，清除 One-shot flag 之后才重新输出波形。
- PWM Trip Zone 输入信号源有：COMP，ADCPPU，GPIO（TZ0~TZ4）。

图 3-8：周期封锁

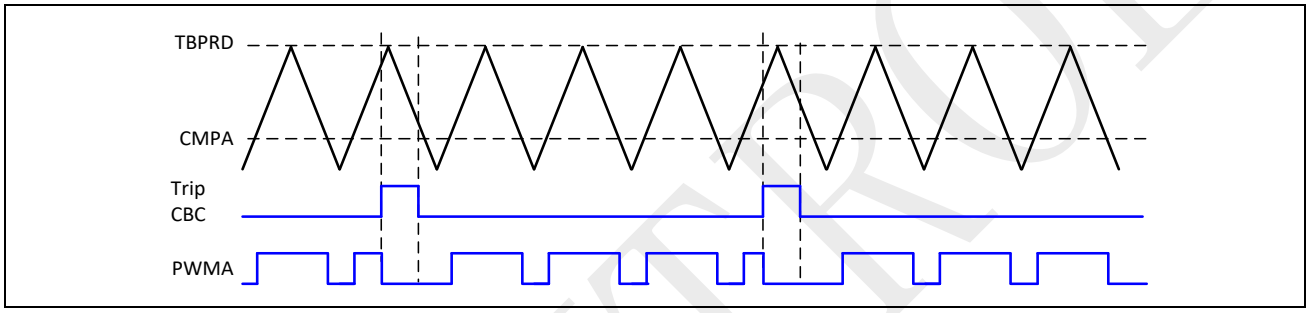
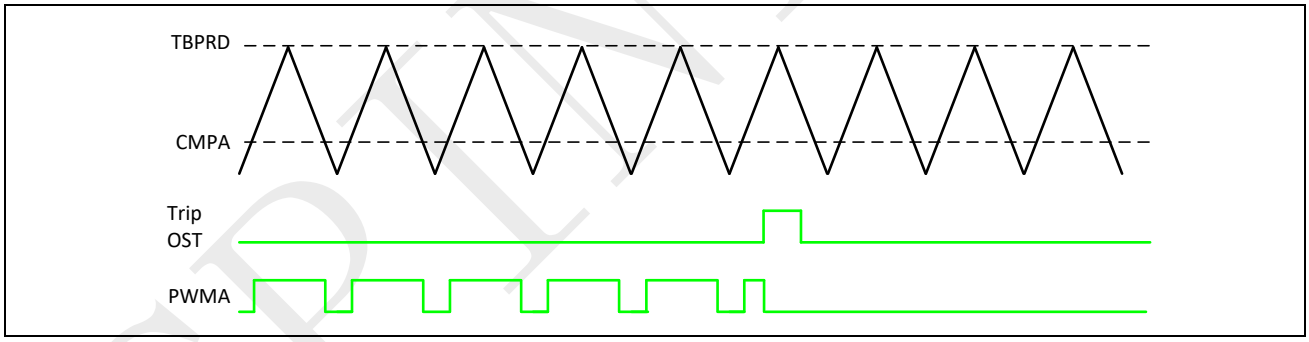


图 3-9：一次性封锁



注意： Trip Zone 按周期和一次性可以同时配置，如果两个信号同时产生，则一次性封锁的信号将覆盖按周期的信号，即一次性封锁优先级高于周期封锁。

3.3.1 实例 6：GPIO 触发封锁

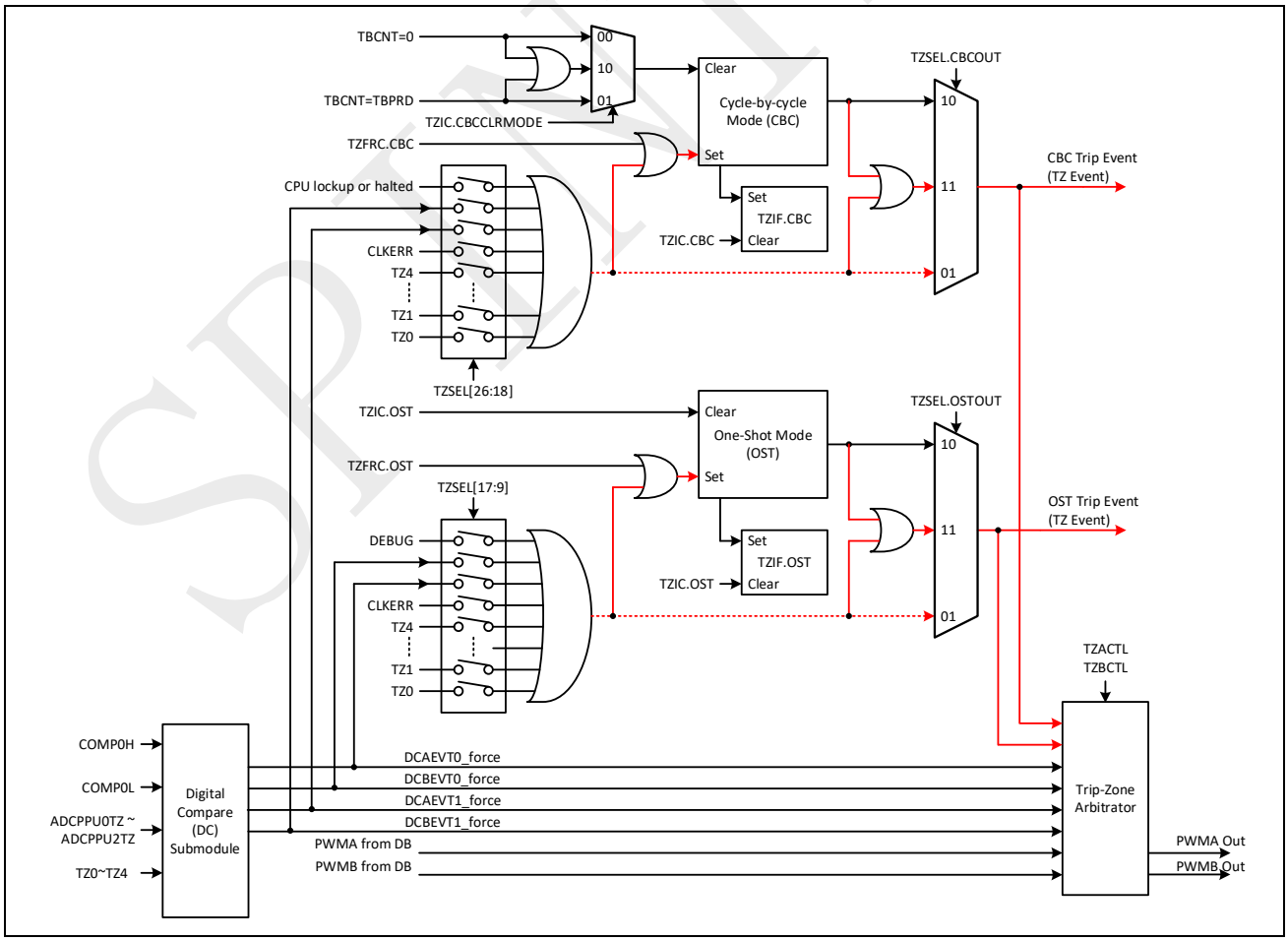
3.3.1.1 功能需求

通过 GPIO 触发信号，产生 TZ 事件，对输出的 PWM 波进行一次性封锁。
通过 GPIO 触发信号，产生 TZ 事件，对输出的 PWM 波进行周期性封锁。

3.3.1.2 功能实现

1. 确定触发信号链路：
   a) GPIO 的触发信号名为 TZ0~TZ4，其既可以通过 CBC 或 OST 直接产生 TZ 事件。
   b) 也可以通过 DC 模块先转换为 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force，再通过 CBC 或 OST 产生 TZ 事件。
   c) 经分析，通常使用上述第一种方法实现，因为其路径较短，如图 3-10 所示。
2. 为不同的 GPIO 设置 TZ0、TZ1、TZ2 信号。
3. 设置 TZ0 为一次性封锁事件，TZ1、TZ2 为周期性封锁事件。
4. 配置行为限定输出控制寄存器 AQCTLx，使得封锁事件产生后 PWM 输出引脚变为高阻态。

图 3-10：Trip Zone 功能框图



以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-2：

表 3-5：实例 6 代码路径

MCU 产品型号	代码路径
SPC1125 系列，SPC1169 系列，SPC2188 系列	SDK 目录\0_Examples\ PWM_GPIO_Trigger_TZ
SPC1168 系列，SPC2168 系列	SDK 目录\0_Examples\PWM_GPIO_Trigger_Trip_Zone

3.3.2 实例 7：COMP 触发封锁

实例适用范围
SPC1125 系列，SPC1169 系列，SPC2188 系列

3.3.2.1 功能需求

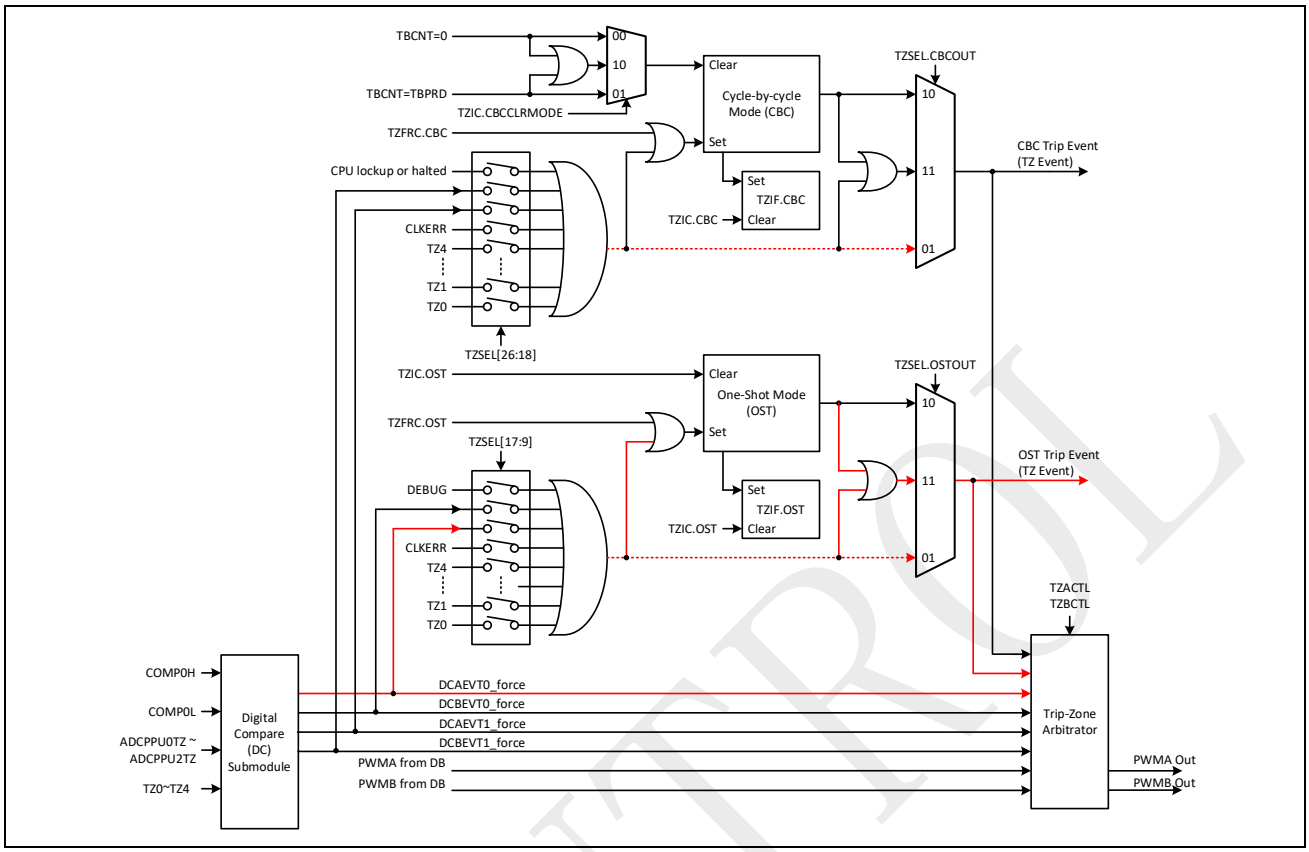
通过 COMP 触发信号，产生 TZ 事件，对输出的 PWM 波进行一次性封锁。

- 使用 COMP 作为电流防护功能，并以以 PWM1 输出为例，当 SPGA 输出超过 2500mV 时，COMP0H 触动 PWM 输出停止，当 SPGA 的电流小于 1500mV，由 COMP0L 触发 PWM 停止。

3.3.2.2 功能实现

1. 确定触发信号链路：
   a) COMP0H 或 COMP0L 可以通过 DC 模块转换为 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force，再通过 CBC 或 OST 产生 TZ 事件，如图 3-11 所示。
   b) 在使用该路信号的时候，要将 TZACTL，TZBCTL 中，和 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force 相关控制位设置为 DO_NOTHING，否则 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force 会不经过 CBC 或 OST，直接到达 Trip-Zone Arbitrator 模块。

图 3-11：Trip Zone 功能框图



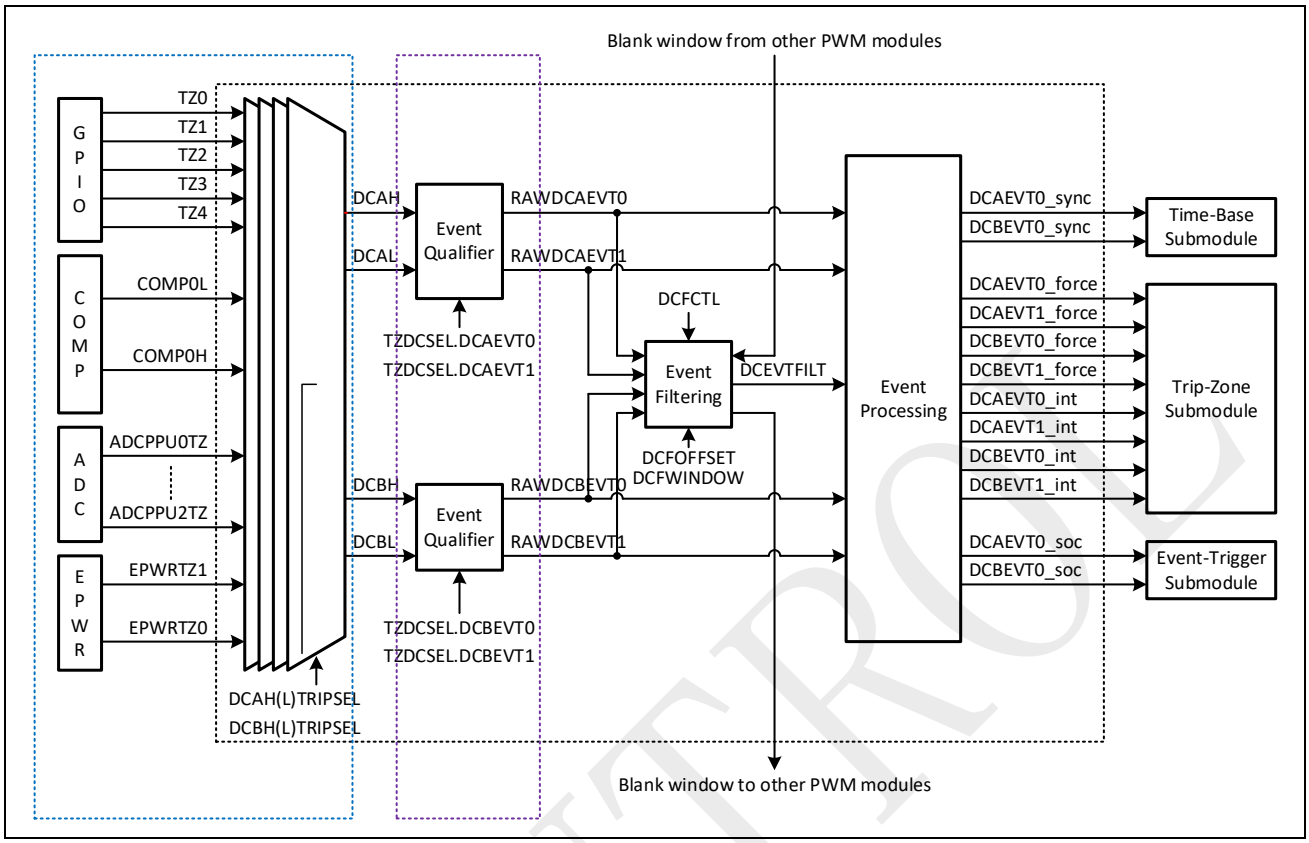
2. 信号过滤功能详述：
   a) 通过 DC 模块进行信号过滤，如图 3-12。
   b) 蓝色虚框区域内为一个事件选择器，通过对 TRIPSEL 寄存器的配置，可以将左侧事件进行逻辑或选择，被选择的事件触发后可产生右侧的 DCAH(L)、DCBH(L)中任一事件。
   c) 紫色虚框内包含两个 Event Qualifier 单元，其中一个 Event Qualifier 单元用于对 DCAH(L)进行逻辑筛选，用以产生 RAWDCxEVT0 或 RAWDCxEVT11。以产生 RAWDCAEVT0 事件为例，进行 Event Qualifier 单元逻辑筛选功能说明，如表 3-6：

表 3-6：Event Qualifier 功能表

逻辑筛选配置	输入事件	触发事件
0	/	禁用事件
1	DCAL 为低	RAWDCAEVT0
2	DCAL 为高	
3	DCAH 为低	
4	DCAH 为高	
5	DCAL 为低且 DCAH 为高	
6	DCAL 为高且 DCAH 为低	
7	DCAL 为高且 DCAH 为高	

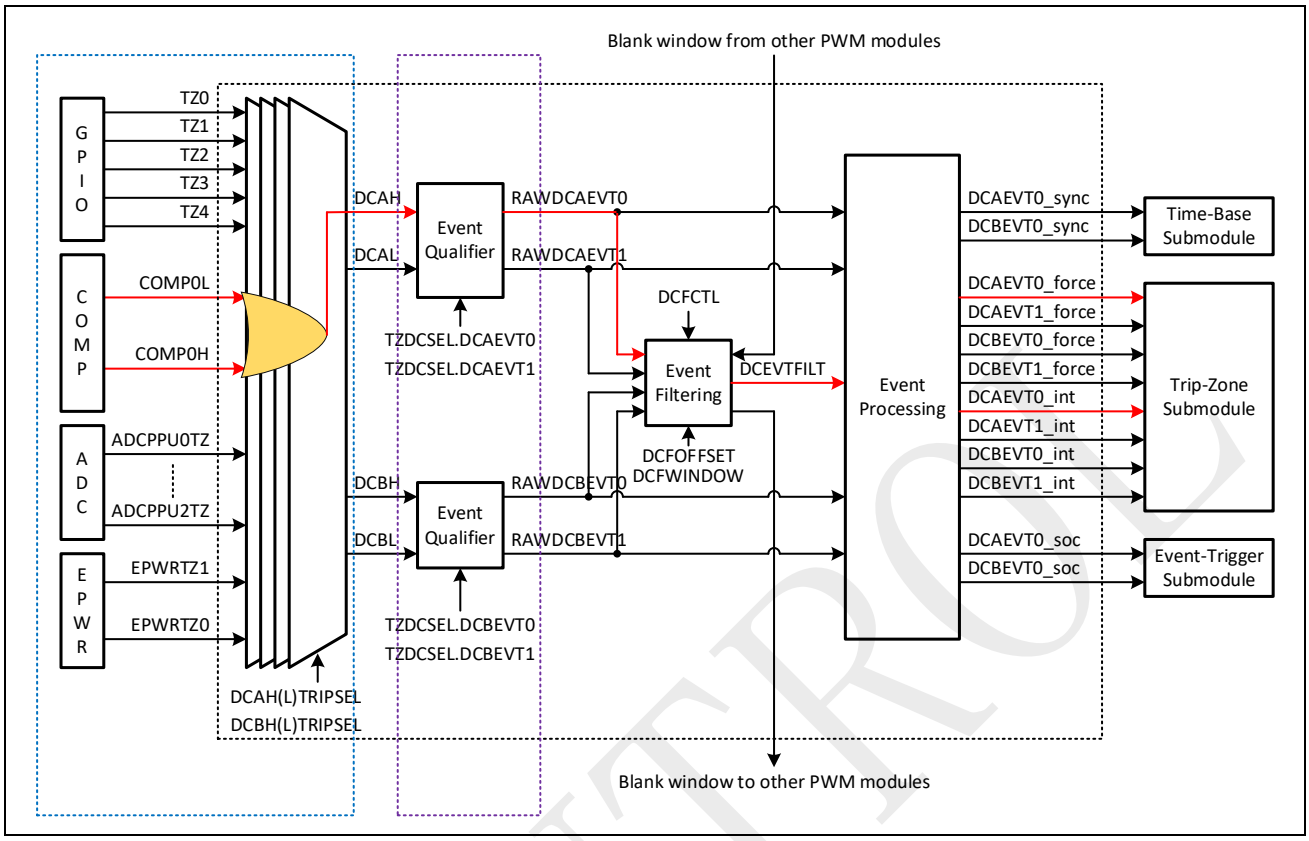
最后 Event Filtering 模块对 RAWDCxEVT0 或 RAWDCxEVT1 信号进行窗口过滤，并通过 Event Processing 模块产生最终输出信号。

图 3-12：DC 功能框图



3. 本实例中信号过滤：
   a) 信号链路如图 3-13 中红线部分。
   b) 通过配置 DCAHTRIPSEL 寄存器，使得 COMP0L 与 COMP0H 任一事件触发，产生 DCAH 事件。
   c) 通过配置 TZSCSEL.DCAEVT0 位段，使得 DCAH 为高时，产生 RAWDCAEVT0 事件。
   d) 通过 Event Filtering 模块对事件信号进行窗口过滤。
   e) 通过 EventProcessing 模块使得 COMP0L 和 COMP0H 任意事件触发后，产生最终输出的 DCAEVT0_force 和 DCAEVT0_int 信号。

图 3-13：DC 功能框图



以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-2：

表 3-7：实例 7 代码路径

MCU 产品型号	代码路径
SPC1125 系列，SPC1169 系列，SPC2188 系列	SDK 目录 \0_Examples\PWM_Current_Protect_Trigger_TZ

3.3.3 实例 8：COMP 触发封锁

实例适用范围
SPC1168 系列，SPC2168 系列

3.3.3.1 功能需求

通过 COMP 触发信号，产生 TZ 事件，对输出的 PWM 波进行一次性封锁。

- 使用 COMP 作为 ADC 采集比较功能，并以 PWM0、PWM1 输出为例。当采集的 ADC 大于 3000mV 时，对正在输出的 PWM0 和 PWM1 进行一次性封锁。由 COMP0H 触发 PWM 停止。

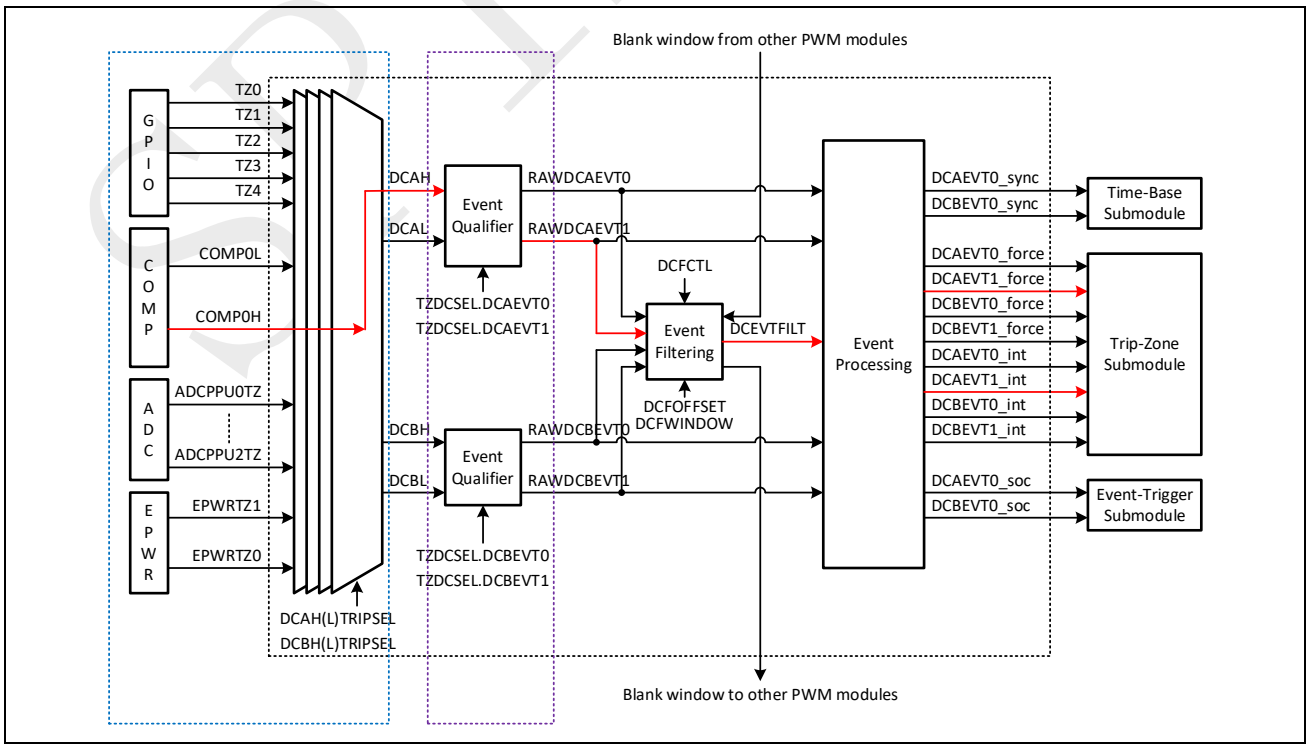
3.3.3.2 功能实现

1. 确定触发信号链路：
   a) COMP0H 可以通过 DC 模块转换为 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force，再通过 CBC 或 OST 产生 TZ 事件，如图 3-11 所示。
   b) 在使用该路信号的时候，要将 TZACTL，TZBCTL 中，和 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force 相关控制位设置为 DO_NOTHING，否则 DCAEVT0_force，DCBEVT0_force，DCAEVT1_force，DCBEVT1_force 会不经过 CBC 或 OST，直接到达 Trip-Zone Arbitrator 模块。

2. 信号过滤功能详述：
   a) 通过 DC 模块进行信号过滤，如图 3-12。
   b) 蓝色虚框区域内为一个事件选择器，通过对 TRIPSEL 寄存器的配置，可以将左侧事件进行逻辑或选择，被选择的事件触发后可产生右侧的 DCAH(L)、DCBH(L)中任一事件。
   c) 紫色虚框内包含两个 Event Qualifier 单元，其中一个 Event Qualifier 单元用于对 DCAH(L)进行逻辑筛选，用以产生 RAWDCxEVT0 或 RAWDCxEVT11。以产生 RAWDCAEVT0 事件为例，进行 Event Qualifier 单元逻辑筛选功能说明，如表 3-6：
   d) 最后 Event Filtering 模块对 RAWDCxEVT0 或 RAWDCxEVT1 信号进行窗口过滤，并通过 Event Processing 模块产生最终输出信号。

3. 本实例中信号过滤：
   a) 信号链路如图 3-14 中红线部分。
   b) 通过配置 DCAHTRIPSEL 寄存器，使得 COMP0H 事件触发，产生 DCAH 事件。
   c) 通过配置 TZSCSEL.DCAEVT0 位段，使得 DCAH 为高时，产生 RAWDCAEVT1 事件。
   d) 通过 Event Filtering 模块对事件信号进行窗口过滤。
   e) 通过 EventProcessing 模块使 COMP0 事件触发后，产生最终输出的 DCAEVT1_force 和 DCAEVT11_int 信号。

图 3-14：DC 功能框图



以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-8：

表 3-8：实例 8 代码路径

产品型号	代码路径
SPC1168 系列，SPC2168 系列	SDK 目录\0_Examples\PWM_Independent_Trigger_TZ

3.4 PWM 触发 ADC 采样

3.4.1 实例 9：PWM 触发 ADC 采样

3.4.1.1 功能需求

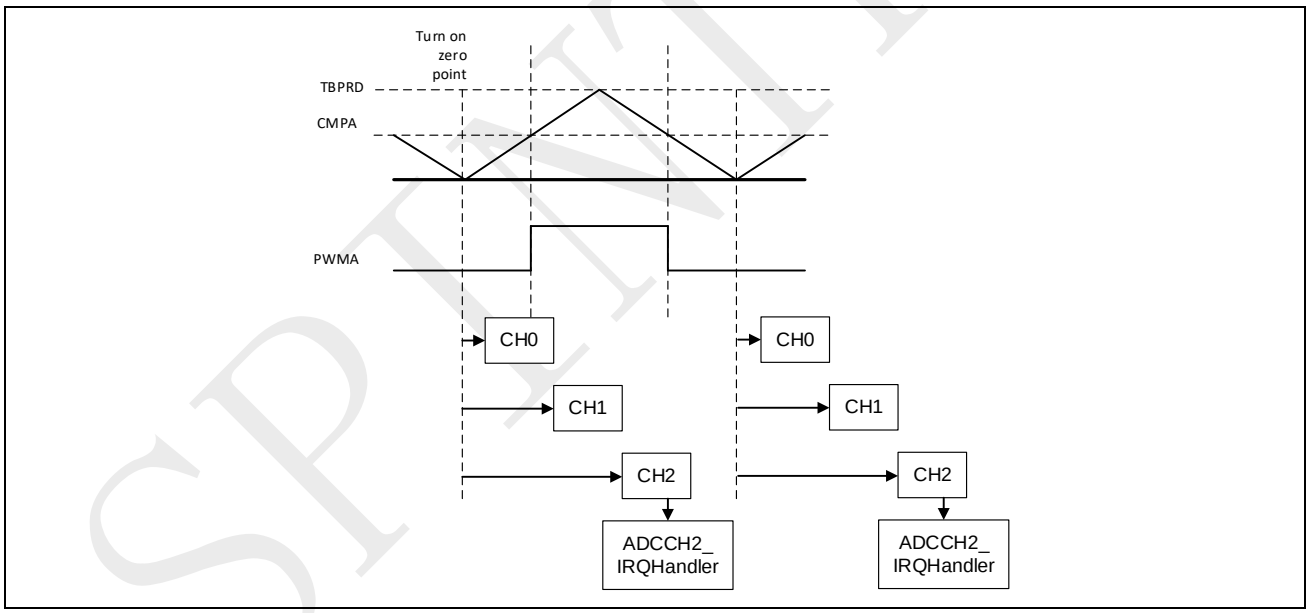
实现 PWM 触发三相电流采样：

- PWM 为上下计数模式，占空比 25%。
- PWM 时基计数器为 0 时触发三相电流转换。

3.4.1.2 功能实现

1. 使用 PWM0 产生 PWM 波形，请参考本文 3.1 章节。
2. 设定 PWM0 之 TBCNT 过零时触发 ADC，一次同时触发 CH0~CH2 分别负责转换三相电流。
3. 当 CH2 转换完之后，进入中断服务程序(ADCCH2_IRQHandler)，获取采样值，如图 3-15 所示。

图 3-15：PWM 触发三相电流采样



以上实现步骤的示例代码可参考 SDK 提供的 Demo，如表 3-9：

表 3-9：实例 9 代码路径

MCU 产品型号	代码路径
SPC1125 系列，SPC1169 系列，SPC2188 系列	SDK 目录 \0_Examples\ PWM_Trigger_ADC_Sample_and_Signal_Observe
SPC1168 系列，SPC2168 系列	SDK 目录 \0_Examples\ PWM_Trigger_ADC_Sample