ACMP Trigger EPWM Brake

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	Use ACMP to trigger EPWM brake
BSP Version	M480 Series BSP CMSIS V3.04.000
Hardware	NuMaker-ETM-M487 v1.2

1 Function Description

1.1 Introduction

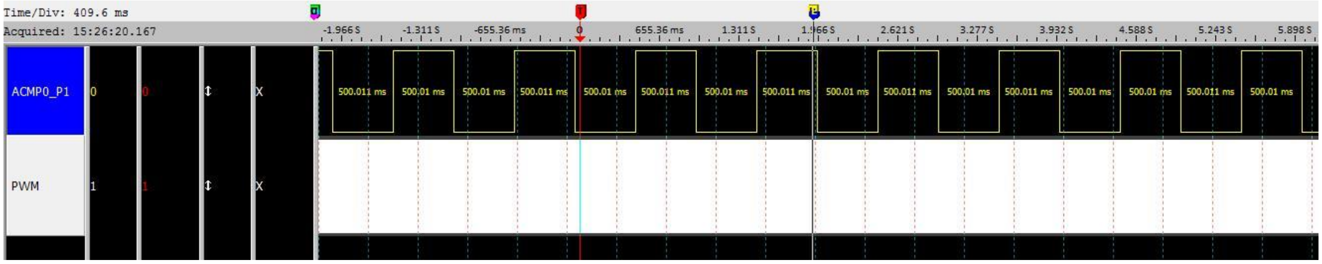
This example code shows how to use ACMP to trigger EPWM brake.

1.2 Principle

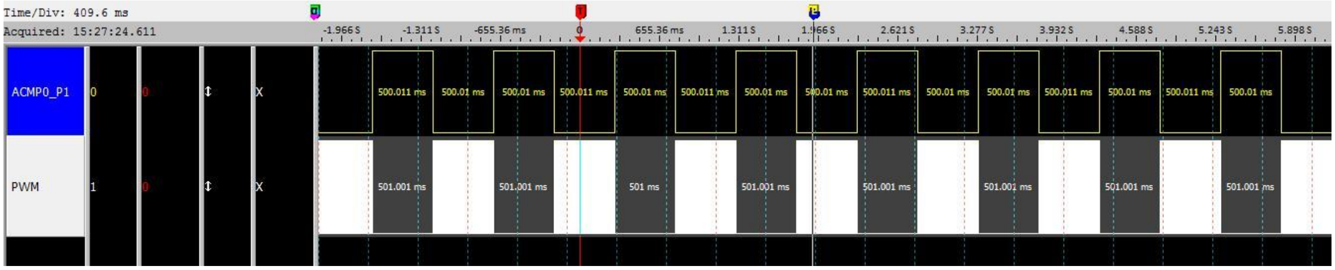
The Brake function is used to prevent EPWM still output waveform in a fault situation. In this example code we use ACMP output as EPWM brake source and demonstrate how EPWM recovery to output waveform when using brake by edge-detect or level-detect.

1.3 Demo Result

- Edge-detect
  The EPWM will recovery to output waveform when user clear edge-detect brake interrupt flag, because this example code clear it at brake interrupt handler so the waveform look likes continuous.



- Level-detect
  The EPWM recovery to output waveform until the brake condition release (ACMP0_P1 < ACMP0_N), so that the waveform only output when ACMP0_P1 is at low-level.



2 Code Description

Select edge-detect or level-detect by uncomment define.

```

#define USE_EDGE_DETECT
//#define USE_LEVEL_DETECT
```

Use PA.5 as EPWM_CH0 and output 100KHz, duty 50% waveform, and enable EPWM brake source from ACMP output. Note the registers to control EPWM brake function is write-protect, need to unlock before rewrite.

```
void epwm_init(void)
{
    /* Set PA.5 as EPWM0_CH0 */
    SYS->GPA_MFPL &= ~(SYS_GPA_MFPL_PA5MFP_Msk);
    SYS->GPA_MFPL  |= (SYS_GPA_MFPL_PA5MFP_EPWM0_CH0);
    /* Config EPWM0_CH0 output 100KHz, duty 50% waveform */
    EPWM_ConfigOutputChannel(EPWM0, 0, 100000, 50);
    EPWM_EnableOutput(EPWM0, 0x01);
    EPWM_Start(EPWM0, 0x01);
    /* Unlock protected registers */
    SYS_UnlockReg();
#ifdef USE_EDGE_DETECT
    /* Set EPWM0_0 brake source from ACMP0, and use edge detect */
    EPWM_EnableFaultBrake(EPWM0, EPWM_CH_0_MASK, 0, EPWM_FB_EDGE_ACMP0);
    /* Enbale EPWM0 edge-detect brake interrupt */
    EPWM_EnableFaultBrakeInt(EPWM0, EPWM_FB_EDGE);
#endif
#ifdef USE_LEVEL_DETECT
    /* Set EPWM0_0 brake source from ACMP0, and use level detect */
    EPWM_EnableFaultBrake(EPWM0, EPWM_CH_0_MASK, 0, EPWM_FB_LEVEL_ACMP0);
    /* Enbale EPWM0 level-detect brake interrupt */
    EPWM_EnableFaultBrakeInt(EPWM0, EPWM_FB_LEVEL);
#endif
    /* Lock protected registers */
    SYS_LockReg();
    /* Enable BRAKE0 interrupt */
    NVIC_EnableIRQ(BRAKE0_IRQn);
}
```

Use PB.2 as ACMP0_P1 input and set ACMP0_N input from CRV.

```
void acmp_init(void)
{
    /* Set PB.2 as ACMP0_P1 */
    SYS->GPB_MFPL &= ~(SYS_GPB_MFPL_PB2MFP_Msk);
    SYS->GPB_MFPL |= (SYS_GPB_MFPL_PB2MFP_ACMP0_P1);

    /* Disable digital input path of analog pin ACMP0_P1 to prevent leakage */
    GPIO_DISABLE_DIGITAL_PATH(PB, (1ul << 2));

    /* Open ACMP0 and set negative source from CRV */
    ACMP_Open(ACMP01, 0, ACMP_CTL_NEGSEL_CRV, ACMP_CTL_HYSTERESIS_DISABLE);

    /* Set CRV voltage source from AVDD */
    ACMP_SELECT_CRV_SRC(ACMP01, ACMP_VREF_CRVSSEL_VDDA);

    /* Set CRV level = 6
     * CRV = AVDD x (1/6 + level /24) = 3.3 x (1/6 + 6/24) = 1.375 V
     */
    ACMP_CRV_SEL(ACMP01, 6);

    /* Select positive input from ACMP0_P1 */
    ACMP_SELECT_P(ACMP01, 0, ACMP_CTL_POSSEL_P1);

    /* Enbale ACMP Interrupt */
    ACMP_ENABLE_INT(ACMP01, 0);
    NVIC_EnableIRQ(ACMP01_IRQn);
}
```

Clear edge-detect interrupt flag or level-detect interrupt flag at brake interrupt handler. Note the brake interrupt flag registers of EPWM are write-protect.

```
void BRAKE0_IRQHandler(void)
{
#ifdef USE_EDGE_DETECT
    if (EPWM_GetFaultBrakeIntFlag(EPWM0, EPWM_FB_EDGE) == 1) {
        /* Unlock protected registers */
        SYS_UnlockReg();
        /* Clear PWM0 brake interrupt flag */
        EPWM_ClearFaultBrakeIntFlag(EPWM0, EPWM_FB_EDGE);
        printf("!!! Edge Brake happened !!!\n");
        /* Lock protected registers */
        SYS_LockReg();
    }
#endif

#ifdef USE_LEVEL_DETECT
    if (EPWM_GetFaultBrakeIntFlag(EPWM0, EPWM_FB_LEVEL) == 1) {
        /* Unlock protected registers */
        SYS_UnlockReg();
        /* Clear PWM0 brake interrupt flag */
        EPWM_ClearFaultBrakeIntFlag(EPWM0, EPWM_FB_LEVEL);
        printf("!!! Level Brake happened !!!\n");
        /* Lock protected registers */
        SYS_LockReg();
    }
#endif
}
```

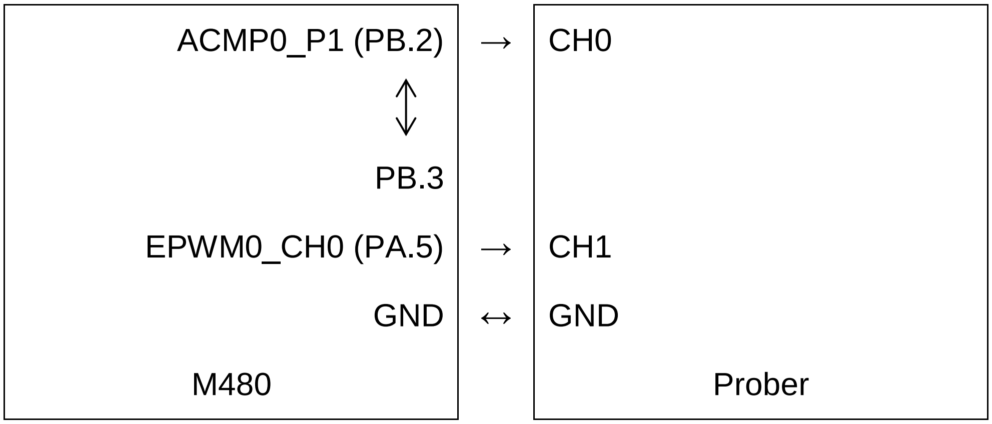
3 Hardware and Software environment

Software Environment

- BSP version
  - M480 Series BSP CMSIS V3.04.000
- IDE version
  - Keil uVersion 5.26

Hardware Environment

- Circuit components
  - NuMaker-ETM-M487
- Diagram



4 Directory Information

EC_M480_ACMP_Trigger_PWM_Brake_V1.00			
	Library		Sample code header and source files
		CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
		Device	CMSIS compliant device header file
		StdDriver	All peripheral driver header and source files
	SamlpeCode		
		ExampleCode	Source file of example code

5 How to execute a sample code

1. Browsing into sample code folder by Directory Information (section 4) and double click ACMP_Trigger_PWM_Brake.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
May. 07, 2019	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, “Insecure Usage”.

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer’s risk, and in the event that third parties lay claims to Nuvoton as a result of customer’s Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.