XT32H05x

XT32 microcontroller Advanced Timer (TIMA) Application notes

Rev 0.0.0

Original Release Date: 26-Oct-2023

Revised :

Revision History

Release	Date	Author	Summary of Change
V0.0.0	26/10/2023	Shirling Liu	Initial

Contents

List of Figures

List of Tables

1 Introduce

This application note serves as a comprehensive guide for software developers, offering essential information on output compare configurations of advanced timer (TIM1 and TIM2). It covers fundamental concepts and provides guidelines to ensure proper utilization of basic timers in software development projects. Whether you're a beginner or an experienced developer, this document will equip you with the necessary knowledge and best practices to effectively configure and utilize timer in your applications.

1.1 Required peripherals

This application involves PADI module, GPIO, and TIM2 module.

Table 1. Modules in example

Sub-module	Peripheral use	Note
PADI	4 ports as GPIO 7 ports as TIM2 output	
TIM2	Advanced timer2 output compare	
GPIO	LEDs show the TIMA interrupt callback state	

1.2 Compatible devices

This example is compatible with the devices in Table 2.

Table 2. Device list

Product	EVB
XT32H050	XB002823

2 Design description

2.1 Feature overview

XT32H0 microcontroller has two advanced timers, TIM1 and TIM2. These timers include the following features:

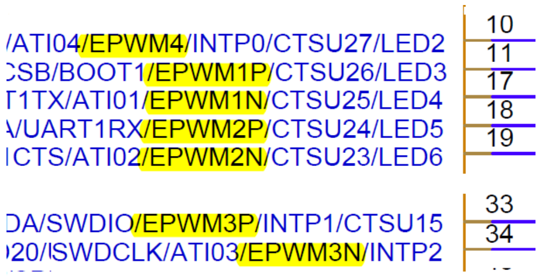
- 16-bit up, down, up and down auto-load counter
- Up to 6 independent channels for PWM/Output compare.
- 4 independent channels for /Input capture.
- Complementary outputs with programmable dead-time
- 2 bidirectional break inputs
- Trigger input for external clock
- Interrupts generator

- Configure by DMA

2.2 Design steps

Here, the example uses the channel 1-4 of advanced timer 2 (TIM2) as output-compare to generate output signal.

1. Set TIM2 source clock and clock divider.
2. Configure base timer parameters of TIM2: counter mode, period, presales, clock-division.
3. Configure the output-compare output port (total 7 ports in this example).
   - PADI_IDX_IO10_ATOUT1_CH1_P, means select and enable the IO10(pin 11) as PWM output of channel 1 of advanced timer1.
   - PADI_IDX_IO15_ATOUT1_CH2_N, means select and enable the IO15(pin 19) as PWM complementary output of channel 2 of advanced timer1.



/ATI04/EPWM4/INTP0/CTSU27/LED2
CSB/BOOT1/EPWM1P/CTSU26/LED3
T1TX/ATI01/EPWM1N/CTSU25/LED4
/UART1RX/EPWM2P/CTSU24/LED5
CTS/ATI02/EPWM2N/CTSU23/LED6
DA/SWDIO/EPWM3P/INTP1/CTSU15
20/SWDCLK/ATI03/EPWM3N/INTP2
10
11
17
18
19
33
34

Figure 1. IO function selection

Note: please refer to XT32H0xxB—reference manual document to find the assignment relationship between pin with IOx

4. Configure output-compare parameter: OC-mode toggle, pulse width, polarity…
5. Enable advanced timer1 (TIM2) Interrupt。
6. Start the advanced timer 1(TIM2) to generate output-compare output signal.

2.3 Design considerations

2.4 Software flowchart

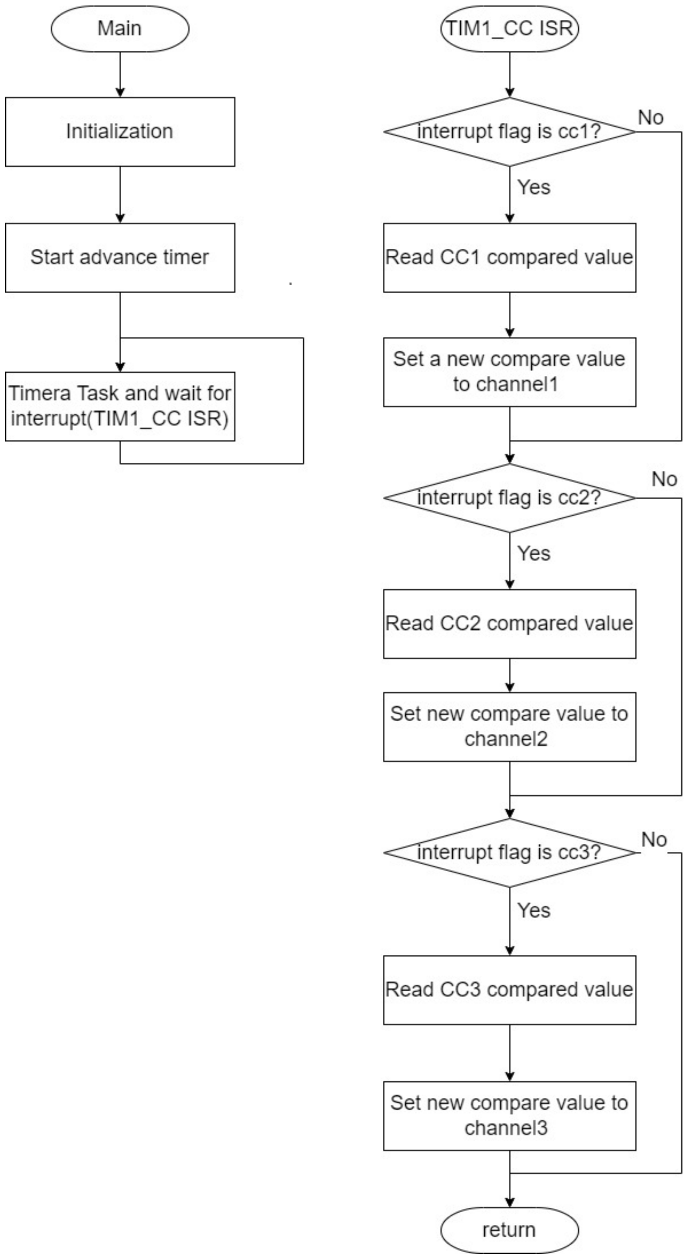


Figure 2. Application flow

2.5 Reference code

Configure Peripheral PAD to select alternate function as TIM2 output-compare output port.

```
void HAL_TIM_OC_MspInit(TIM_HandleTypeDef* htim_pwm)
{

  if(htim_pwm->Instance==TIM2)
  {
        /* Peripheral clock enable */
   __HAL_RCC_TIMA_CLK_ENABLE();
        /* Configure output pin */
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_1,   TIM2_CH1P_OUT_PIN_IDX)
;
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_1,   TIM2_CH1N_OUT_PIN_IDX)
;
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_2,   TIM2_CH2P_OUT_PIN_IDX)
;
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_2,   TIM2_CH2N_OUT_PIN_IDX)
;
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_3,   TIM2_CH3P_OUT_PIN_IDX)
;
   HAL_TIM_OutputPortConfig(htim_pwm,  TIM_PORT_CHANNEL_3,   TIM2_CH3N_OUT_PIN_IDX)
;
  }
}
```

Enable TIM2 CC interrupt code:

```
static void XT_Nvic_Init(void)
{
#if defined(XT32H0xxB)
    HAL_NVIC_SetPriority(TIM1_CC_IRQn, 2, 0);
    HAL_NVIC_EnableIRQ(TIM1_CC_IRQn);
#endif /* XT32H0xxB */
}
```

Configure Peripheral TIM2 using HAL_TIM_OC_Init.

```
    /* Initialize TIMA */
    htima2.Instance = TIM2;
    htima2.Init.Prescaler = APP_PRESCALER_VALUE;
    htima2.Init.Period    = APP_PERIOD_VALUE_1MS;
    htima2.Init.CounterMode = TIM_COUNTERMODE_UP;
    htima2.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
    htima2.Init.RepetitionCounter = 0;
    htima2.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
    if (HAL_TIM_OC_Init(&htima2) != HAL_OK)
    {
        Error_Handler();
    }
```

```
    /* -4- Configure output compare parameter configuration */
    sConfigOC.OCMode = TIM_OCMODE_TOGGLE;
    sConfigOC.Pulse  = APP_TIM_OC_PULSE1_STEP;
    sConfigOC.OCPolarity   = TIM_OCPOLARITY_HIGH;
    sConfigOC.OCNPolarity  = TIM_OCNPOLARITY_HIGH;
    sConfigOC.OCFastMode   = TIM_OCFAST_DISABLE;
    sConfigOC.OCIdleState  = TIM_OCIDLESTATE_RESET;
    sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
    sConfigOC.OCDeadTime  = 0;
    sConfigOC.OCNDeadTime = 0;
    if (HAL_TIM_OC_ConfigChannel(&htima2, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
    {
        Error_Handler();
    }


    sConfigOC.OCMode = TIM_OCMODE_TOGGLE;
    sConfigOC.Pulse  = APP_TIM_OC_PULSE2_STEP;
    sConfigOC.OCPolarity    = TIM_OCPOLARITY_HIGH;
    sConfigOC.OCNPolarity   = TIM_OCNPOLARITY_HIGH;
    sConfigOC.OCFastMode    = TIM_OCFAST_DISABLE;
    sConfigOC.OCIdleState   = TIM_OCIDLESTATE_RESET;
    sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
    sConfigOC.OCDeadTime   = 0;
    sConfigOC.OCNDeadTime = 0;
    if (HAL_TIM_OC_ConfigChannel(&htima2, &sConfigOC, TIM_CHANNEL_2) != HAL_OK)
    {
        Error_Handler();
    }


    sConfigOC.OCMode = TIM_OCMODE_TOGGLE;
    sConfigOC.Pulse  =APP_TIM_OC_PULSE3_STEP;
    sConfigOC.OCPolarity    = TIM_OCPOLARITY_HIGH;
    sConfigOC.OCNPolarity   = TIM_OCNPOLARITY_HIGH;
    sConfigOC.OCFastMode    = TIM_OCFAST_DISABLE;
    sConfigOC.OCIdleState   = TIM_OCIDLESTATE_RESET;
    sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
    sConfigOC.OCDeadTime = 0;
    sConfigOC.OCNDeadTime = 0;
    if (HAL_TIM_OC_ConfigChannel(&htima2, &sConfigOC, TIM_CHANNEL_3) != HAL_OK)
    {
        Error_Handler();
    }
```

Start the advanced timer 2 (TIM2) to generate output signal.

```
void XT_TIM2_Start(void )
{
    if (HAL_TIM_OC_Start_IT(&htima2, TIM_CHANNEL_1) != HAL_OK)
    {
    /*Error_Handler*/
```

```c
    }

    if (HAL_TIM_OC_Start_IT(&htima2, TIM_CHANNEL_2) != HAL_OK)
    {
    /*Error_Handler*/
    }

    if (HAL_TIM_OC_Start_IT(&htima2, TIM_CHANNEL_3) != HAL_OK)
    {
    /*Error_Handler*/
    }
}
```

The Output compare interrupt callback set new compare value to channel of advanced timer

```c
void HAL_TIM_OC_DelayElapsedCallback(TIM_HandleTypeDef *htim)
{
    uint16_t autoreload_value;
    uint32_t u4CompareValue = 0;
    autoreload_value = __HAL_TIM_GET_AUTORELOAD(htim);

   if ((__HAL_TIM_GET_FLAG(htim, TIM_FLAG_CC1) !=RESET)||
     (HAL_TIM_GetActiveChannel(htim)== HAL_TIM_ACTIVE_CHANNEL_1))
    {
      __HAL_TIM_CLEAR_FLAG(htim, TIM_FLAG_CC1);
      u4CompareValue = __HAL_TIM_GET_COMPARE(htim,TIM_CHANNEL_1);
      if (u4CompareValue +APP_TIM_OC_PULSE1_STEP < (autoreload_value/2))
       __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_1, u4CompareValue
+APP_TIM_OC_PULSE1_STEP);
      else
        __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_1,0 );
    }
    if ((__HAL_TIM_GET_FLAG(htim, TIM_FLAG_CC2) != RESET)||
       (HAL_TIM_GetActiveChannel(htim)== HAL_TIM_ACTIVE_CHANNEL_2))
    {
      __HAL_TIM_CLEAR_FLAG(htim, TIM_FLAG_CC2);
      u4CompareValue = __HAL_TIM_GET_COMPARE(htim,TIM_CHANNEL_2);
      if (u4CompareValue + APP_TIM_OC_PULSE2_STEP < autoreload_value/2)
         __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_2, u4CompareValue +
APP_TIM_OC_PULSE2_STEP);
      else
         __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_2,0);
    }


    if ((__HAL_TIM_GET_FLAG(htim, TIM_FLAG_CC3) != RESET) ||
     (HAL_TIM_GetActiveChannel(htim)== HAL_TIM_ACTIVE_CHANNEL_3))
    {
      __HAL_TIM_CLEAR_FLAG(htim, TIM_FLAG_CC3);
      u4CompareValue = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_3);

      if (u4CompareValue + APP_TIM_OC_PULSE3_STEP < autoreload_value)
          __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_3, u4CompareValue +
APP_TIM_OC_PULSE3_STEP);
      else
          __HAL_TIM_SET_COMPARE(htim, TIM_CHANNEL_3,0);
    }
    u1Tim_cbState = CB_TIM2_OC_DELAY_ELAPSED;
}
```

2.6 Additional resources

- XT32H0xxB--reference manual