Application note-AN23130D

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 one-pulse mode 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 TIM1 module.

Table 1. Modules in example

Sub-module	Peripheral use	Note
PADI	4 ports as GPIO 1 input and 1 output of timer	
TIM1	Advanced timer1 one pulse mode	
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 of advanced timer 1 (TIM1) as input-capture to capture the input signal and channel 2 of TIM1 as output-capture.

1. Set TIM1 source clock and clock divider.
2. Configure base timer parameters of TIM1: counter mode, period, prescaler, clock-division.
3. Configure the input signal port and output port.
    - ➢ PADI_IDX_IO13_ATIN1_CH1, means select and enable the IO13(pin 17) as input of channel 1 of advanced timer1.
    - ➢ PADI_IDX_IO14_ATOUT1_CH2_P means select and enable the IO14(pin 18) as output of channel 1 of advanced timer1

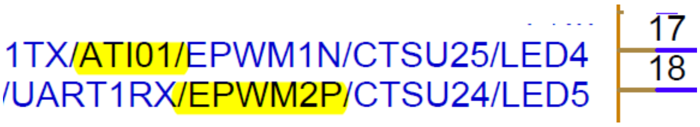

Figure 1. IO function selection

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

4. Configure input-capture parameter: polarity, direction
5. Configure output compare parameter: PWM-mode, pulse width, polarity…
6. Enable advanced timer1 (TIM1) Interrupt。
7. Start the advanced timer 1(TIM1) to capture input signal and output compare output.

2.3 Design considerations

2.4 Software flowchart

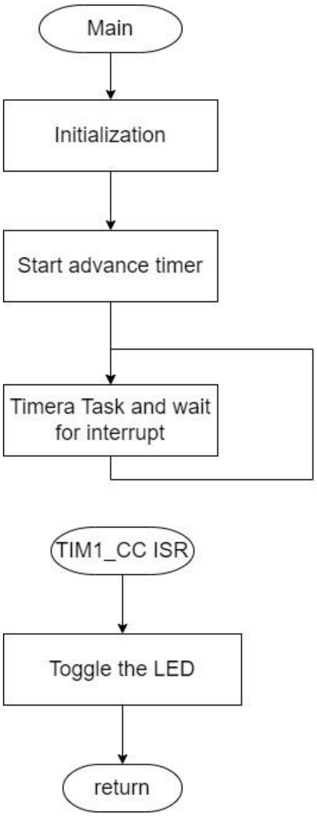

Figure 2. Application flow

2.5 Reference code

Configure Peripheral PAD to select alternate function as TIM1 input port and output port.

```
void HAL_TIM_OnePulse_MspInit(TIM_HandleTypeDef *htim)
{
  if(htim->Instance==TIM1)
  {
   __HAL_RCC_TIMA_CLK_ENABLE();
   LL_RCC_SetAdvancedTimerInput(LL_RCC_TIMA1_CH1IN,SET);
   HAL_TIM_InputPortConfig(htim,  TIM_PORT_CHANNEL_1 ,  TIM1_CH1_IC_PIN_IDX);
   HAL_TIM_OutputPortConfig(htim, TIM_PORT_CHANNEL_2,   TIM1_CH1_IC_PIN_IDX);
  }
}
```

Enable TIM1 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 TIM1 using HAL_TIM_OnePulse_Init.

```
      /* Initialize TIMA */
    htima1.Instance = TIM1;
    htima1.Init.Prescaler = APP_PRESCALER_VALUE;
    htima1.Init.Period    = APP_PERIOD_VALUE_1MS;
    htima1.Init.CounterMode = TIM_COUNTERMODE_UP;
    htima1.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
    htima1.Init.RepetitionCounter = 0;
    htima1.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
    if (HAL_TIM_OnePulse_Init(&htima1,TIM_OPMODE_SINGLE) != HAL_OK)
    {
        Error_Handler();
    }
```

```
  /* -4- Configure one pulse parameter configuration */
  sOpmConfig.OCMode = TIM_OCMODE_PWM1; //TIM_OCMODE_PWM1
  sOpmConfig.Pulse  = APP_OC_PULSE2_VALUE;
  sOpmConfig.OCPolarity   = TIM_OCPOLARITY_HIGH;      //TIM_OCPOLARITY_LOW;//
  sOpmConfig.OCNPolarity  = TIM_OCNPOLARITY_HIGH;     //TIM_OCNPOLARITY_LOW;//
  sOpmConfig.OCIdleState  = TIM_OCIDLESTATE_RESET;
  sOpmConfig.OCNIdleState = TIM_OCNIDLESTATE_RESET;

  sOpmConfig.ICPolarity   = TIM_ICPOLARITY_RISING;
  sOpmConfig.ICSelection  =  TIM_ICSELECTION_INDIRECTTI;//TIM_ICSELECTION_DIRECTTI
  sOpmConfig.ICFilter     = 0;

    if (HAL_TIM_OnePulse_ConfigChannel(&htima1, &sOpmConfig, TIM_CHANNEL_2,
TIM_CHANNEL_1) != HAL_OK)
    {
      /* Configuration Error */
      Error_Handler();
    }
```

Start the one pulse of advanced timer 1(TIM1) .

```
void XT_TIM1_Start(void )
```

```
{
    if (HAL_TIM_OnePulse_Start_IT(&htima1, TIM_CHANNEL_2) != HAL_OK)
    {
    /*Error_Handler*/
    }
}
```

2.6 Additional resources

- XT32H0xxB--reference manual