SDK应用_eTmr应用案例_同一个eTMR instance PWM输出和输入捕获

版本：

SDK YTM32B1LE0_SDK_RTM_1.0.3

前言：

PWM输出与脉冲输入捕获是eTMR的常见应用，本文将介绍使用同一个eTRM的不同通道分别实现PWM输出和输入捕获的用法，并给出应用时的注意事项。

1. 模块配置与功能实现

本实例中使用eTMR0_ch2作为PWM输出通道，使用eTMR0_ch7作为输入捕获通道，使用最简单的配置实现功能需求。

1.1 初始化eTMR0模块

代码块

```
status_t eTMR_DRV_Init(uint32_t instance, const etmr_user_config_t *info, etmr_state_t *state);
/* Initialize eTMR0 instance */
eTMR_DRV_Init(eTMR_INST, &eTMR_Config, &etmrState);
```

instance表示初始化的目标实例，eTMR0对应此处为0；

etmr_user_config_t 表示eTMR的初始化结构体配置，主要配置项如下：

```
/* etmr sync configuration */
etmr_pwm_sync_t eTMR_PwmSyncConfig = {
    .regSyncFreq = 0,
    .regSyncSel = REG_SYNC_WITH_MOD,
    .cntInitSyncSel = CNT_SYNC_WITH_REG,
    .maskOutputSyncSel = CHMASK_SYNC_WITH_REG,
    .regSyncTrigSrc = DISABLE_TRIGGER,
    .cntInitSyncTrigSrc = DISABLE_TRIGGER,
    .maskOutputSyncTrigSrc = DISABLE_TRIGGER,
};

/* etmr user configuration */
etmr_user_config_t eTMR_Config = {
    .etmrClockSource = eTMR_CLOCK_SOURCE_INTERNALCLK,
    .etmrPrescaler = 3,          // divided by 3
    .debugMode = false,
    .syncMethod = &eTMR_PwmSyncConfig,
    .outputTrigConfig = NULL,
    .isTofIntEnabled = false,
};
```

配置ETMR模块同步加载点为MOD点

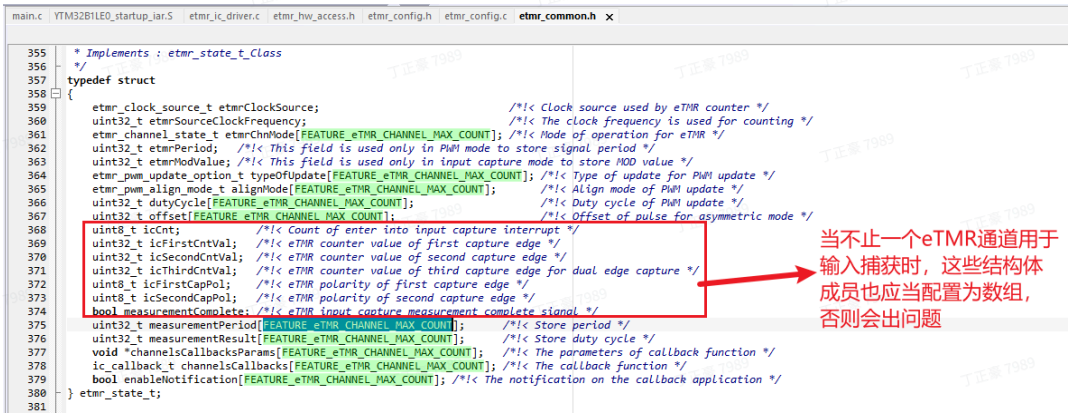
配置ETMR时钟源为内部时钟FIRC 48Mhz

配置ETMR模块时钟分频为3

配置ETMR模块同步加载点

不使能溢出中断

etmr_state_t 为eTMR模块的状态机参数，其成员包含PWM的状态及输入捕获事件发生时对应时刻的计数值和脉冲极性，这几个参数是计算捕获脉冲频率、占空比的核心参数，这些参数会在eTMR_DRV_InputCaptureHandler函数中被更新。



1.2 初始化PWM通道

代码块

```
status_t eTMR_DRV_InitPwm(uint32_t instance, const etmr_pwm_param_t *param);
/* Initialize eTMR0_ch2(PTC2) as PWM instance */
eTMR_DRV_InitPwm(eTMR_INST, &eTMR_PwmConfig);

```

instance含义同上。

etmr_pwm_param_t 为PWM基础参数，详细配置如下图。

```
/* etmr independent/complementary channel pwm config */
etmr_pwm_ch_param_t eTMR_PwmIndependentChannelConfig[1] = {
    {
        .hwChannelId = 2,                    使能eTMR0_ch2作为PWM通道
        .polarity = eTMR_POLARITY_NORMAL,
        .align = eTMR_PWM_CENTER_ALIGN,
        .channelInitVal = 0,
        .dutyCycle = 0x4000U,                配置占空比为0x4000（50%）
        .enableSecondChannelOutput = false,              //false->independent
    },

};

/* etmr pwm config */
etmr_pwm_param_t eTMR_PwmConfig = {
    .mode = eTMR_PWM_MODE,                   配置为PWM模式
    .nNumPwmChannels = 1,
    .uFrequencyHZ = 600U,                    配置频率为600Hz
    .pwmChannelConfig = eTMR_PwmIndependentChannelConfig,
    .faultConfig = &eTMR_PwmFaultConfig,
};
```

1.3 初始化InputCapture通道

代码块

```
status_t eTMR_DRV_InitInputCapture(uint32_t instance, const etmr_ic_param_t *param);
/* Initialize eTMR0_ch7(PTE7) as Input_Capture instance */
eTMR_DRV_InitInputCapture(eTMR_INST, &eTMR_InputCaptureConfig);
```

instance含义同上。

etmr_pwm_param_t 为PWM基础参数，详细配置如下图。

```
/* etmr input capture channel parameters configuration */
etmr_ic_ch_param_t eTMR_InputCaptureChannelConfig[1] = {
  {
    .hwChannelId = 7,                        使能eTMR0_ch7作为输入捕获通道
    .edge = eTMR_DUAL_EDGES,                 一般需要计算占空比要配置为捕捉双边沿
    .enableNotification = false,
  },
};

/* etmr input capture parameters configuration */
etmr_ic_param_t eTMR_InputCaptureConfig = {
    .numChannels = 1,
    .countValue = 0xFFFF,                    计数值配置为65535
    .inputChConfig = eTMR_InputCaptureChannelConfig,
};
```

1.4 测试代码

此外，需要打开eTMR0的中断及捕获通道的中断，并启动eTMR0。

代码块

```
INT_SYS_EnableIRQ(eTMR0_CH_0_7_IRQn);          //enable eTMR0 int
eTMR_DRV_EnableChnInt(eTMR_INST, 7);           //enable channel7 int
```

```
3    eTMR_DRV_Enable(eTMR_INST);                    //start eTMR0
```

通过串口打印周期、占空比计算所需的参数，

```
while(1)
{
    if (eTMR_DRV_IsCaptureComplete(eTMR_INST))   查询捕获事件完成后打印状态参数
    {
        PRINTF("icFirstCapPol: %d\n", etmrState.icFirstCapPol);
        PRINTF("icFirstCntVal: %d\n", etmrState.icFirstCntVal);
        PRINTF("icSecondCntVal: %d\n", etmrState.icSecondCntVal);
        PRINTF("icThirdCntVal: %d\n", etmrState.icThirdCntVal);
        PRINTF("measurePeriod: %d\n", etmrState.measurementPeriod[7]);
        PRINTF("measureDutyCycle: %d\n", etmrState.measurementResult[7]);
        PRINTF("\n");

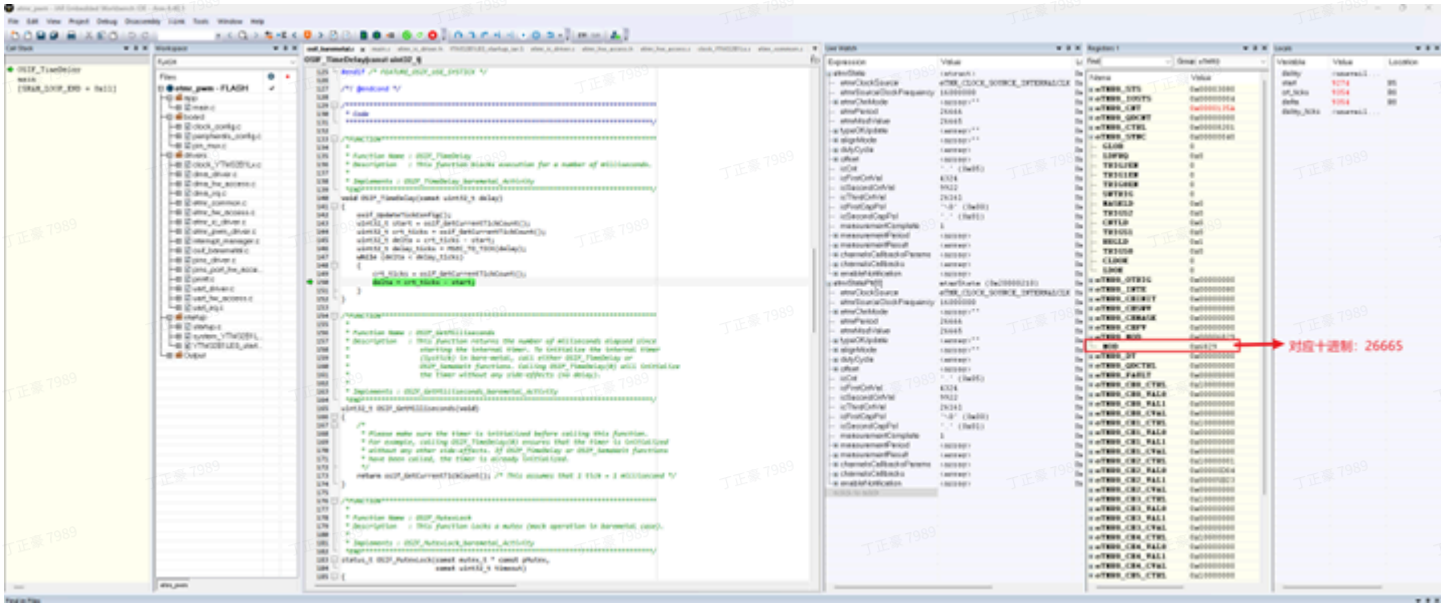
        OSIF_TimeDelay(1000);
        eTMR_DRV_ClearCaptureComplete(eTMR_INST);
```

1.5 补充说明

由于一个eTMR所有通道共用一个MOD寄存器，共用一个MOD值，因此使用同一个eTMR去做输入捕获和PWM输出时需要注意以下问题。

（1）注意PWM输出与输入捕获参数的匹配。

例如，配置eTMR0模块时钟为16Mhz，PWM输出600Hz的脉冲时，eTMR0对应的MOD值16M/600约为26666（实际寄存器的值为26665），此时对于输入捕获通道而言，捕获一个脉冲周期对应的计数值应当小于26666，也就是说适用于捕获频率高于600Hz的脉冲，否则可能导致频率、占空比测量不准。



例如使用另一个demo板输出704Hz，占空比约为75%的方波进行测试，串口打印的数据如下图所示，

占空比 = 16241 / 21843 ≈ 74.35%　计算频率 = 1 / ((1/16Mhz) * 21843) ≈732.5hz，测试基本准确。




```
icFirstCapPol: 1
icFirstCntVal: 8011
icSecondCntVal: 24252
icThirdCntVal: 3189
measurePeriod: 21843
measureDutyCycle: 16241
```

客户可以此为参考，根据实际需求选择是否应当使用同一个eTMR同时做PWM输出与输入捕获应用。

（2）同理，在输入捕获期间，PWM波形的占空比可以调整，但不能改变PWM的频率，因为这会改变MOD值，进而导致输入捕获结果不准。

1.6 结论

当资源充足时，建议使用不同的eTMR分别实现PWM脉冲输出和输入捕获功能；必须使用同一个eTMR时，可参考本文进行具体评估。