<http://blog.sina.com.cn/s/blog_63d525fd0100u9o3.html>

## stm32的PWM输入捕获模式

 (2011-08-22 02:22:10)

[[http://simg.sinajs.cn/blog7style/images/common/sg_trans.gif](javascript:;)转载▼](javascript:;)

|  |  |
| --- | --- |
| 标签： [杂谈](http://search.sina.com.cn/?c=blog&q=%D4%D3%CC%B8&by=tag) | 分类： [stm32](http://blog.sina.com.cn/s/articlelist_1674913277_5_1.html) |

一、概念理解

PWM输入捕获模式是输入捕获模式的特例，自己理解如下

1. 每个定时器有四个输入捕获通道IC1、IC2、IC3、IC4。且IC1 IC2一组，IC3 IC4一组。并且可是设置管脚和寄存器的对应关系。

2. 同一个TIx输入映射了两个ICx信号。

3. 这两个ICx信号分别在相反的极性边沿有效。

4. 两个边沿信号中的一个被选为触发信号，并且从模式控制器被设置成复位模式。

5. 当触发信号来临时，被设置成触发输入信号的捕获寄存器，捕获“一个PWM周期（即连续的两个上升沿或下降沿）”，它等于包含TIM时钟周期的个数（即捕获寄存器中捕获的为TIM的计数个数n）。

6. 同样另一个捕获通道捕获触发信号和下一个相反极性的边沿信号的计数个数m，即（即高电平的周期或低电平的周期）

7. 由此可以计算出PWM的时钟周期和占空比了

    frequency=f（TIM时钟频率）/n。

    duty cycle=（高电平计数个数/n），

    若m为高电平计数个数，则duty cycle=m/n

    若m为低电平计数个数，则duty cycle=（n-m）/n

注：因为计数器为16位，所以一个周期最多计数65535个，所以测得的 最小频率= TIM时钟频率/65535。

二、程序设计与分析

1. 程序概述：选择TIM3作为PWM输入捕获。IC2设置为上升沿，并设置为有效的触发输入信号。所以IC2的捕获寄存器捕获PWM周期，

                   IC1的捕获寄存器捕获PWM的高电平周期。

2.程序代码如下：

  RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM3, ENABLE);      //时钟配置

  RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE);

  GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_7;                               //GPIO配置  
  PIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_IN\_FLOATING;  
  GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

  GPIO\_Init(GPIOA, &GPIO\_InitStructure);

  NVIC\_InitStructure.NVIC\_IRQChannel = TIM3\_IRQn;                     //NVIC配置   
  NVIC\_InitStructure.NVIC\_IRQChannelPreemptionPriority = 0;  
  NVIC\_InitStructure.NVIC\_IRQChannelSubPriority = 1;  
  NVIC\_InitStructure.NVIC\_IRQChannelCmd = ENABLE;  
  NVIC\_Init(&NVIC\_InitStructure);

  TIM\_ICInitStructure.TIM\_Channel = TIM\_Channel\_2;                   //通道选择  
  TIM\_ICInitStructure.TIM\_ICPolarity = TIM\_ICPolarity\_Rising;       //上升沿触发  
  TIM\_ICInitStructure.TIM\_ICSelection = TIM\_ICSelection\_DirectTI;    //管脚与寄存器对应关系  
  TIM\_ICInitStructure.TIM\_ICPrescaler = TIM\_ICPSC\_DIV1;           //输入预分频。意思是控制在多少个输入周期做一次捕获，如果

//输入的信号频率没有变，测得的周期也不会变。比如选择4分频，则每四个输入周期才做一次捕获，这样在输入信号变化不频繁的情况下，

//可以减少软件被不断中断的次数。  
  TIM\_ICInitStructure.TIM\_ICFilter = 0x0;                            //滤波设置，经历几个周期跳变认定波形稳定0x0～0xF

  TIM\_PWMIConfig(TIM3, &TIM\_ICInitStructure);                 //根据参数配置TIM外设信息

  TIM\_SelectInputTrigger(TIM3, TIM\_TS\_TI2FP2);                //选择IC2为始终触发源

  TIM\_SelectSlaveMode(TIM3, TIM\_SlaveMode\_Reset);//TIM从模式：触发信号的上升沿重新初始化计数器和触发寄存器的更新事件

  TIM\_SelectMasterSlaveMode(TIM3, TIM\_MasterSlaveMode\_Enable); //启动定时器的被动触发

  TIM\_Cmd(TIM3, ENABLE);                                 //启动TIM2

  TIM\_ITConfig(TIM3, TIM\_IT\_CC2, ENABLE);     //打开中断

中断处理函数

void TIM3\_IRQHandler(void)  
{  
   
  TIM\_ClearITPendingBit(TIM3, TIM\_IT\_CC2);                //清楚TIM的中断待处理位

  IC2Value = TIM\_GetCapture2(TIM3);                         //读取IC2捕获寄存器的值，即为PWM周期的计数值

  if (IC2Value != 0)  
  {  
     
    DutyCycle = (TIM\_GetCapture1(TIM3) \* 100) / IC2Value;         //读取IC1捕获寄存器的值，并计算占空比

    Frequency = 72000000 / IC2Value;                                          //计算PWM频率。  
  }  
  else  
  {  
    DutyCycle = 0;  
    Frequency = 0;  
  }  
}

注（一）：若想改变测量的PWM频率范围，可将TIM时钟频率做分频处理

 TIM\_TimeBaseStructure.TIM\_Period = 0xFFFF;     //周期0～FFFF  
  TIM\_TimeBaseStructure.TIM\_Prescaler = 5;       //时钟分频，分频数为5+1即6分频  
  TIM\_TimeBaseStructure.TIM\_ClockDivision = 0;   //时钟分割  
  TIM\_TimeBaseStructure.TIM\_CounterMode = TIM\_CounterMode\_Up;//模式  
  TIM\_TimeBaseInit(TIM2, &TIM\_TimeBaseStructure);//基本初始化  
注注（二）：定时器TIM的倍频器X1或X2。在APB分频为1时，倍频值为1，否则为2