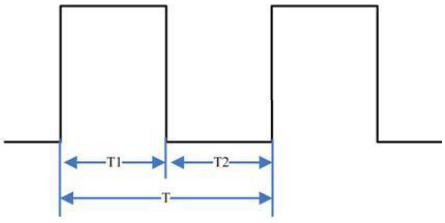
前言：

在 PWM 输出那一节我们已经讲过 STM32 有三种定时器，分别为基本定时器，通用定时器和高级定时器。基本定时器只能用来实现定时功能，而通用定时器 TIM2--TIM5 比基本定时器功能强大了不少，除了定时之外还可以用来实现测量输入脉冲的频率，脉冲宽度与输出 PWM 的场合。

通用定时器的计时功能与基本定时器几乎一样，都是把时钟源经过预分频器输出到脉冲宽度计数器 TIMx_CNT 累加，溢出时产生中断或 DMA 请求，这里不再赘述。

PWM 输入过程的分析：

首先我们需要把定时器配置为输入功能，这个在后面会有详细介绍，我们先假设我们已经配置好了，往下看。众所周知 PWM 是方波，那么就有跳变沿（上升沿和下降沿），我们想要测量脉冲的宽度，占空比还有频率，其实就是抓取一个周期的跳变沿。简单的来讲，



当 T1 这个区间的上升沿到来时，我们开始计数，因为这个周期开始了，对应到单片机的操作也就是将 TIMx_CNT 的值清零并开始向上累加，而当 T1 的下降沿到来时，这时我们已经测出了高电平的时间，因此我们将这个时间存储一下，存到 TIMx_CCR2 这个寄存器里，这个时候注意，TIMx_CNT 的值不能清零，要继续累加，因为我们还没测出一整个周期的时间，那可能有同学要问了，直到什么时候 TIMx_CNT 的值停止累加再次清零呢，直到 T2 的上升沿（T2 的右端点值）到来时，这个时候我们将 TIMx_CNT 的值存储到 TIMx_CCR1 这个寄存器里，如此一来我们就已经完成了测量脉冲宽度的操作，是不是很简单呢，如果想要算出占空比那只需要用

（（TIMx_CCR2+1）/（TIMx_CCR1+1））×100%就可以了

那么下面我们来看看代码的实现吧

通过前面的学习我们知道，STM32 要使用什么样的功能，首先都要初始化，这就像跟他打个招呼，你不跟他提前说好了他不给你干活。

我们要用到 TIM 定时器，当然肯定也要用到 IO 口（毕竟你不能直接把线插 TIM 上），那么自然是要初始化 GPIO 和定时器啦。

首先 我们用到的函数首先是 GPIO 初始化

```
static void GENERAL_TIM_GPIO_Config(void)
{
  GPIO_InitTypeDef GPIO_InitStructure;

  // 输入捕获通道 GPIO 初始化
  RCC_APB2PeriphClockCmd(GENERAL_TIM_CH1_GPIO_CLK, ENABLE);//使能TIM定时器通道1的时钟
  GPIO_InitStructure.GPIO_Pin =  GENERAL_TIM_CH1_PIN;        //确定通道1的管脚号
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;      //设置IO口模式
  GPIO_Init(GENERAL_TIM_CH1_PORT, &GPIO_InitStructure);       //载入结构体
}
```

首先肯定是定义 GPIO 的结构体

然后开时钟，然后单片机得知道哪个口能用吧，你告诉他通道 1 的管脚号，然后再就是设置 IO 口的模式，这里我们设置为 floating 也就是浮空输入的意思，因为你要测量脉冲，当然不能给内部上拉或者下拉了。（理解不了的话可以理解为输入的高电平就是高电平，低电

平就是低电平，外部输入是啥里面接受就是啥）

再然后是初始化时钟 你用哪个时钟自然初始化哪个就好了

首先大家应该明白定时器的时钟来源，这里因为篇幅限制我不多讲解，只需要知道我们得到的时钟是 72M 即可。

```
static void GENERAL_TIM_Mode_Config(void)
{

    TIM_TimeBaseInitTypeDef  TIM_TimeBaseStructure;//声明时基结构体

   TIM_ICInitTypeDef TIM_ICInitStructure;//声明输入捕获结构体
  // 开启定时器时钟,即内部时钟CK_INT=72M
  GENERAL_TIM_APBxClock_FUN(GENERAL_TIM_CLK,ENABLE);

/*--------------------时基结构体初始化--------------------------*/

  // 自动重装载寄存器的值，累计TIM_Period+1个频率后产生一个更新或者中断
  TIM_TimeBaseStructure.TIM_Period=GENERAL_TIM_PERIOD;
  // 驱动CNT计数器的时钟 = Fck_int/(psc+1)
  TIM_TimeBaseStructure.TIM_Prescaler= GENERAL_TIM_PSC;
  // 时钟分频因子
  TIM_TimeBaseStructure.TIM_ClockDivision=TIM_CKD_DIV1;
  // 计数器计数模式，设置为向上计数
  TIM_TimeBaseStructure.TIM_CounterMode=TIM_CounterMode_Up;
  // 重复计数器的值，没用到不用管
  TIM_TimeBaseStructure.TIM_RepetitionCounter=0;
  // 初始化定时器
  TIM_TimeBaseInit(GENERAL_TIM, &TIM_TimeBaseStructure);
```

首先也是声明结构体 这里会有两个 一个是时基结构体也就是所有定时器都有的结构体，再往后是我们这次要用到的与输入捕获有关的结构体。

时基结构体首先配置 period 的值 也就是一个计时周期是多长

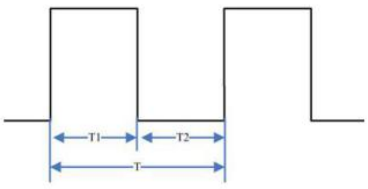
然后是计数器 CNT 的时钟 这个很重要 直接决定了你最后计算脉冲时间的参数，为了方便我们取分频因子为 72-1（因为时钟频率 72M 这样好算数） 也就是计数器蹦一下是 1/(72M/(72-1+1))秒

然后我们设置分频因子为 CKD_DIV_1 他的意思是不分频 也就是时钟是 72M

然后设置计数模式为向上计数

最后初始化定时器

以上我们已经配置好了我们的定时器，那么现在我们需要在定时器的中断里实现我们自己的操作 也就是我上面提到的



当 T1 这个区间的上升沿到来时，我们开始计数，因为这个周期开始了，对应到单片机的操作也就是将 TIMx_CNT 的值清零并开始向上累加，而当 T1 的下降沿到来时，这时我们已经测出了高电平的时间，因此我们将这个时间存储一下，存到 TIMx_CCR2 这个寄存器里，这个时候注意，TIMx_CNT 的值不能清零，要继续累加，因为我们还没测出一整个周期的时间，那可能有同学要问了，直到什么时候 TIMx_CNT 的值停止累加再次清零呢，直到 T2 的上升沿（T2 的右端点值）到来时，这个时候我们将 TIMx_CNT 的值存储到 TIMx_CCR1 这个寄存器里，如此一来我们就已经完成了测量脉冲宽度的操作，是

好了 首先来看上升沿怎么捕捉

```
// 当要被捕获的信号的周期大于定时器的最长定时时，定时器就会溢出，产生更新中断
// 这个时候我们需要把这个最长的定时周期加到捕获信号的时间里面去
if ( TIM_GetITStatus ( GENERAL_TIM, TIM_IT_Update) != RESET )
{
  TIM_ICUserValueStructure.Capture_Period ++;
  TIM_ClearITPendingBit ( GENERAL_TIM, TIM_FLAG_Update );
}

// 上升沿捕获中断
if ( TIM_GetITStatus (GENERAL_TIM, GENERAL_TIM_IT_CCx ) != RESET)
{
  // 第一次捕获
  if ( TIM_ICUserValueStructure.Capture_StartFlag == 0 )
  {
    // 计数器清0
    TIM_SetCounter ( GENERAL_TIM, 0 );
    // 自动重装载寄存器更新标志清0
    TIM_ICUserValueStructure.Capture_Period = 0;
    // 存捕获比较寄存器的值的变量的值清0
    TIM_ICUserValueStructure.Capture_CcrValue = 0;

    // 当第一次捕获到上升沿之后，就把捕获边沿配置为下降沿
    GENERAL_TIM_OCxPolarityConfig_FUN(GENERAL_TIM, TIM_ICPolarity_Falling);
    // 开始捕获标准置1
    TIM_ICUserValueStructure.Capture_StartFlag = 1;
  }
```

if 里面的操作是

如果这个脉冲太长了 有多长呢 比我们计数的一个周期还长，那我就得记录完一个周期以后将周期数加一也就是里面的 TIM_ICUserValueStructure.Capture_Period 这个变量加一 来记录周期数 然后我们要做的是 把计数器清零重新开始计

好了 下面进行上升沿的捕捉 首先如果捉到了第一个上升沿好 我们把计数器 CNT 清 0 周期数也清零包括记录脉冲宽度的寄存器都清零 开始计数

然后 我们下一个需要抓取的脉冲是下降沿对吧 哎 我们设置一下

把捕获的边沿配置成下降沿 同时这个时候把捕获的标志位改为 1 这是为了让下面的抓取下降沿能顺利运行 请看

```
  // 下降沿捕获中断
  else // 第二次捕获
  {
    // 获取捕获比较寄存器的值，这个值就是捕获到的高电平的时间的值
    TIM_ICUserValueStructure.Capture_CcrValue =
    GENERAL_TIM_GetCapturex_FUN (GENERAL_TIM);

    // 当第二次捕获到下降沿之后，就把捕获边沿配置为上升沿，好开启新的一轮捕获
    GENERAL_TIM_OCxPolarityConfig_FUN(GENERAL_TIM, TIM_ICPolarity_Rising);
    // 开始捕获标志清0
    TIM_ICUserValueStructure.Capture_StartFlag = 0;
    // 捕获完成标志置1
    TIM_ICUserValueStructure.Capture_FinishFlag = 1;
  }
```

当下降沿到来时再次触发定时器中断 这个时候因为标志位写的是 1 所以我们进入了 else 这个分支 对吧 也就是抓取下降沿

首先这个时候我把下降沿到来时 CNT 的值存一下 存到哪？

寄存器里 就是那个 Ccr_Value

这个东西存好了之后 我们已经抓到了下降沿 可以算出来高电平的时间了

然后把捕获标志位清 0 完成标志位置 1 证明我们抓完了（这里只列举了抓取高电平的方法 我相信看了这些以后你能自己写出抓取整个脉冲宽度的方法）

TIM_ClearITPendingBit (GENERAL_TIM,GENERAL_TIM_IT_CCx);

最后在中断里清除一下计数器 毕竟人家还在计数 等会还是要回来继续抓脉冲的（这里必须注意手动计数器清 0 ）

```c
int main(void)
{
  uint32_t time;
  // TIM 计数器的驱动时钟
  uint32_t TIM_PscCLK = 72000000 / (GENERAL_TIM_PSC+1);

  /* 定时器初始化 */
  GENERAL_TIM_Init();
  /*数码管初始化*/
  hc595_init();

  while(1)
  {
    if(TIM_ICUserValueStructure.Capture_FinishFlag == 1)
    {
      // 计算高电平时间的计数器的值
      time = TIM_ICUserValueStructure.Capture_Period * (GENERAL_TIM_PERIOD+1) +
            (TIM_ICUserValueStructure.Capture_CcrValue+1);

            //在数码管上显示高电平脉宽时间
      Dispaly_M(time/TIM_PscCLK,time%TIM_PscCLK);

      TIM_ICUserValueStructure.Capture_FinishFlag = 0;
    }
  }
}
```

这里提一句 我是用的数码管显示 所以用到了数码管的初始化 具体等讲数码管的学长会给大家详细介绍 我们这里略过不提。

在主函数里

当采集标志位是 1 的时候（采集完毕） 这个时候在数码管上显示一下我们的采到的脉冲时间 最后显示完了再把标志位清成 0 为了下次不要误判 （毕竟你已经显示过了 你要告诉单片机说这是旧的了）