同一定时器不同通道的输入捕获实验理解

原创

nidie508 于 2019-09-03 17:14:18 发布 8920 收藏 7

分类专栏： stm32 stm32基础

本文链接：https://blog.csdn.net/nidie508/article/details/99056033


stm32 同时被 2 个专栏收录

20 篇文章 7 订阅

订阅专栏


stm32基础

12 篇文章 0 订阅

订阅专栏

提前说说

暑期学习截止到明天就彻底结束了，顺便把这些天学到的东西做一个小结，至少这一个月也不能白过。

同定时器不同通道的输入捕获实验理解

首先，看看一**个定时器一个通道下**的输入捕获实验

main.c

```
#include "sys.h"
#include "usart.h"
#include "delay.h"
#include "led.h"
#include "timer.h"
#include "sys.h"
#include "key.h"
//ALIENTEK Mini STM32开发板范例代码1
//跑马灯实验
//技术支持：www.openedv.com
//广州市星翼电子科技有限公司
extern u8 TIM2CH1_CAPTURE_STA;
extern u16 TIM2CH1_CAPTURE_VAL;

int main(void)
{
 u32 t,temp=0;
 Stm32_Clock_Init(9);  //系统时钟设置
 uart_init(72,9600);
 delay_init(72);       //延时初始化
 LED_Init();       //初始化与LED连接的硬件接口
 TIM2_Cap_Init(0XFFFF,72-1);
 while(1)
 {
  if(TIM2CH1_CAPTURE_STA&0X80)
  {
   temp=TIM2CH1_CAPTURE_STA&0X3F;
   temp*=65536;
   temp+=TIM2CH1_CAPTURE_VAL;
   printf("HIGH:%d us\r\n",temp);
   TIM2CH1_CAPTURE_STA=0;
  }
 }
}
```

timer.c

```
#include "timer.h"
#include "led.h"

void TIM2_Cap_Init(u16 arr,u16 psc)
{
 RCC->APB1ENR|=1<<0;
 RCC->APB2ENR|=1<<2;

 GPIOA->CRL&=0XFFFFFFF0;
 GPIOA->CRL|=0X00000008;
 GPIOA->ODR|=0<<0;

 TIM2->ARR=arr;
 TIM2->PSC=psc;

 TIM2->CCMR1|=1<<0;
 TIM2->CCMR1|=1<<4;
 TIM2->CCMR1|=0<<10;

 TIM2->CCER|=0<<1;
 TIM2->CCER|=1<<0;
```

```
 TIM2->CCER|=1<<0;

 TIM2->DIER|=1<<1;
 TIM2->DIER|=1<<0;
 TIM2->CR1|=0x01;
 MY_NVIC_Init(2,0,TIM2_IRQn,2);
}

u8 TIM2CH1_CAPTURE_STA=0;
u16 TIM2CH1_CAPTURE_VAL;

void TIM2_IRQHandler(void)
{
 u16 tsr;
 tsr=TIM2->SR;
 if((TIM2CH1_CAPTURE_STA&0X80)==0)
 {
  if(tsr&0X01)
  {
   if(TIM2CH1_CAPTURE_STA&0X40)
   {
    if((TIM2CH1_CAPTURE_STA&0X3F)==0X3F)
    {
     TIM2CH1_CAPTURE_STA|=0X80;
     TIM2CH1_CAPTURE_VAL=0XFFFF;
    }else TIM2CH1_CAPTURE_STA++;
   }
  }
  if(tsr&0x02)
  {
   if(TIM2CH1_CAPTURE_STA&0X40)
   {
    TIM2CH1_CAPTURE_STA|=0X80;
    TIM2CH1_CAPTURE_VAL=TIM2->CCR1;
    TIM2->CCER&=~(1<<1);
   }else
   {
    TIM2CH1_CAPTURE_VAL=0;
    TIM2CH1_CAPTURE_STA=0X40;
    TIM2->CNT=0;
    TIM2->CCER|=1<<1;
   }
  }
 }
 TIM2->SR=0;
}
```

这里就不详细赘述了，各个手册和资料记录的也十分详细，一个通道下的输入捕获也是比较简单。

那么同一定时器下不同通道的输入捕获实验又是怎样呢？

首先，让我们不修改源码，直接把另一个通道的代码复制粘贴过去，再在主函数打印出其各自高低电平时间。（这里我直接用超声波做实验了，要是拿按键感觉加的代码更多而且还不够直观）

test.c

```
#include "sys.h"
#include "usart.h"
#include "delay.h"
#include "led.h"
#include "timer.h"
#include "sys.h"


extern u8 TIM2CH1_CAPTURE_STA;
extern u16 TIM2CH1_CAPTURE_VAL;

extern u8 TIM2CH2_CAPTURE_STA;
extern u16 TIM2CH2_CAPTURE_VAL;

int main(void)
{
 u32 temp1,temp2;
 Stm32_Clock_Init(9);  //系统时钟设置
 uart_init(72,9600);
 delay_init(72);       //延时初始化
 TIM2_Cap_Init(0XFFFF,72-1);
 while(1)
 {
  printf("hello world\r\n");
  PBout(6)=1;                        //打开TRIG
  PBout(7)=1;
  delay_us(20);
  PBout(6)=0;
  PBout(7)=0;
  if(TIM2CH1_CAPTURE_STA&0X80)   //成功捕获到了一次高电平
  {
   temp2=TIM2CH1_CAPTURE_STA&0X3F;
   temp2*=65536;      //溢出时间总和
   temp2+=TIM2CH1_CAPTURE_VAL;
   temp2*=0.017;          //得到距离
   printf("TIM2_CH1_LENGTH:%d cm\r\n",temp2);  //打印距离
    TIM2CH1_CAPTURE_STA=0;    //开启下一次捕获
   }
  if(TIM2CH2_CAPTURE_STA&0X80)   //成功捕获到了一次高电平
  {
   temp1=TIM2CH2_CAPTURE_STA&0X3F;
   temp1*=65536;      //溢出时间总和
   temp1+=TIM2CH2_CAPTURE_VAL;
   temp1*=0.017;          //得到距离
   printf("TIM2_CH2_LENGTH:%d cm\r\n",temp1);  //打印距离
    TIM2CH2_CAPTURE_STA=0;    //开启下一次捕获
   }
 }
}
```

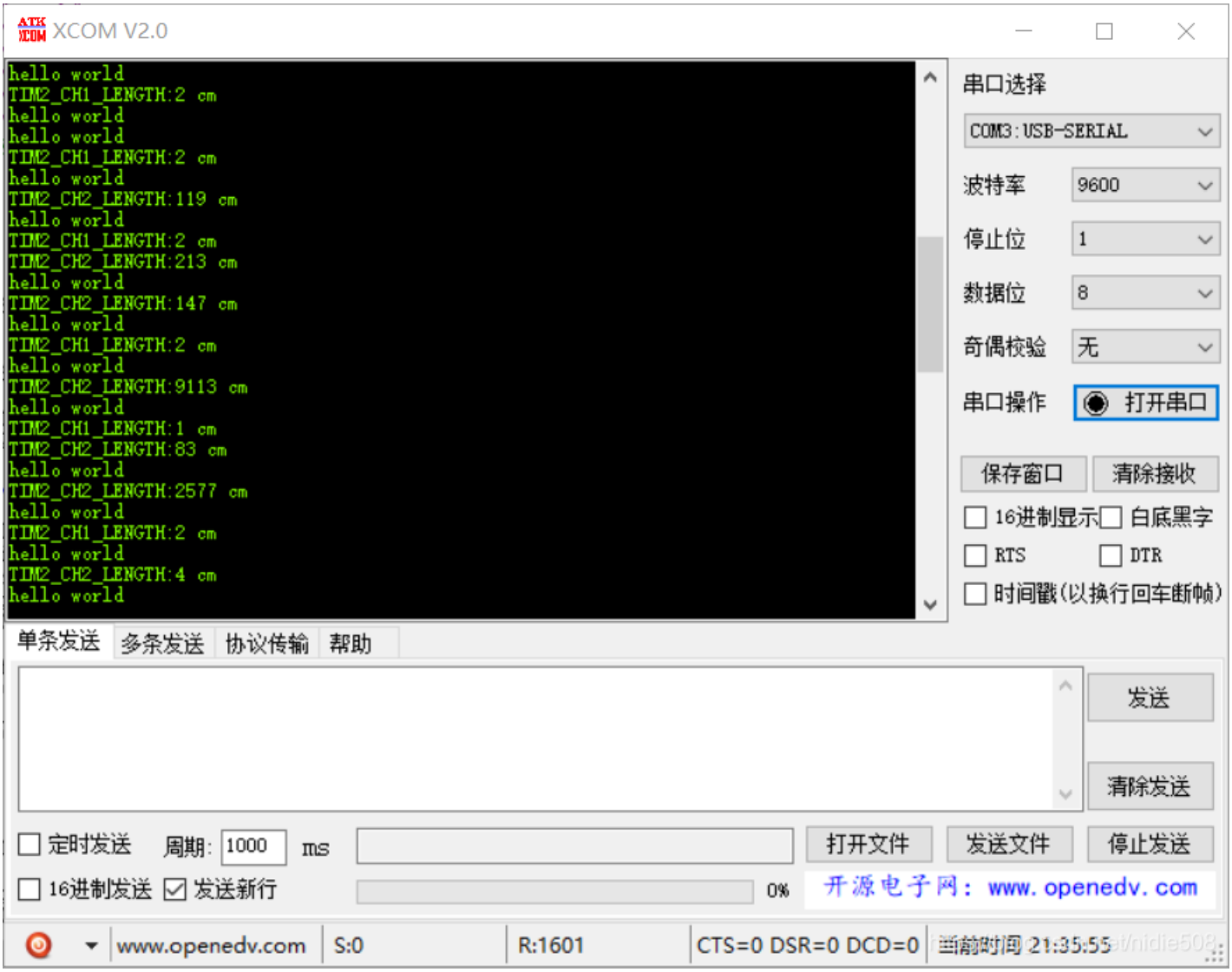
timer.c

```
#include "timer.h"
#include "led.h"
#include "sys.h"
void TIM2_Cap_Init(u16 arr,u16 psc)
{
 RCC->APB1ENR|=1<<0;
```

```
 RCC->APB1ENR|=1<<0;
 RCC->APB2ENR|=1<<2;
 RCC->APB2ENR|=1<<3;     //使能PORTB时钟

 GPIOA->CRL&=0XFFFFFF00;   //定时器2通道1
 GPIOA->CRL|=0X00000088;   //定时器2通道2
 GPIOA->ODR|=0<<0;
 GPIOA->ODR|=1<<1;


  GPIOB->CRL&=0X00FFFFFF;//PB7清除之前设置
  GPIOB->CRL|=0X33000000;//PB7推挽输出
  GPIOB->ODR|=1<<7;      //PB7 输出高
 GPIOB->ODR|=1<<6;

 TIM2->ARR=arr;
 TIM2->PSC=psc;

 TIM2->CCMR1|=1<<0;
 TIM2->CCMR1|=1<<4;
 TIM2->CCMR1|=0<<2;

 TIM2->CCMR1|=1<<8;
 TIM2->CCMR1|=1<<12;
 TIM2->CCMR1|=0<<10;

 TIM2->CCER|=0<<1;
 TIM2->CCER|=1<<0;

 TIM2->CCER|=0<<5;
 TIM2->CCER|=1<<4;

 TIM2->DIER|=1<<1;
 TIM2->DIER|=1<<2;
 TIM2->DIER|=1<<0;
 TIM2->CR1|=0x01;
 MY_NVIC_Init(2,0,TIM2_IRQn,2);
}

u8 TIM2CH1_CAPTURE_STA=0;
u16 TIM2CH1_CAPTURE_VAL;

u8 TIM2CH2_CAPTURE_STA=0;
u16 TIM2CH2_CAPTURE_VAL;

void TIM2_IRQHandler(void)
{
 u16 tsr;
 tsr=TIM2->SR;
 if((TIM2CH1_CAPTURE_STA&0X80)==0)
 {
  if(tsr&0X01)
  {
   if(TIM2CH1_CAPTURE_STA&0X40)
   {
    if((TIM2CH1_CAPTURE_STA&0X3F)==0X3F)
    {
     TIM2CH1_CAPTURE_STA|=0X80;
     TIM2CH1_CAPTURE_VAL=0XFFFF;
    }else TIM2CH1_CAPTURE_STA++;
```

```
  }
 }
 if(tsr&0x02)
 {
  if(TIM2CH1_CAPTURE_STA&0X40)
  {
   TIM2CH1_CAPTURE_STA|=0X80;
   TIM2CH1_CAPTURE_VAL=TIM2->CCR1;
   TIM2->CCER&=~(1<<1);
  }else
  {
   TIM2CH1_CAPTURE_VAL=0;
   TIM2CH1_CAPTURE_STA=0X40;
   TIM2->CNT=0;
   TIM2->CCER|=1<<1;
  }
 }
}
if((TIM2CH2_CAPTURE_STA&0X80)==0)
{
 if(tsr&0X01)
 {
  if(TIM2CH2_CAPTURE_STA&0X40)
  {
   if((TIM2CH2_CAPTURE_STA&0X3F)==0X3F)
   {
    TIM2CH2_CAPTURE_STA|=0X80;
    TIM2CH2_CAPTURE_VAL=0XFFFF;
   }else TIM2CH2_CAPTURE_STA++;
  }
 }
 if(tsr&0x04)
 {
  if(TIM2CH2_CAPTURE_STA&0X40)
  {
   TIM2CH2_CAPTURE_STA|=0X80;
   TIM2CH2_CAPTURE_VAL=TIM2->CCR2;
   TIM2->CCER&=~(1<<5);
  }else
  {
   TIM2CH2_CAPTURE_VAL=0;
   TIM2CH2_CAPTURE_STA=0X40;
   TIM2->CNT=0;
   TIM2->CCER|=1<<5;
  }
 }
}
TIM2->SR=0;
}
```

上述代码基本上是在一个超声波测距的情况下加上另一个超声波（即多设置一个输入捕获），当然这种方案是不可行的，以下是实验结果。



因为录不了视频，但是大致就是我在两个超声波前均放了障碍物，距离超声波5cm左右，从图中可以看出，距离并不准确，而且跳动范围很大。

分析上述整个错误程序我发现，虽然两个通道在进行输入捕获时并没有干扰，但是他们是在同一定时器下，即他们所用的计数器是相同的。

在一个输入捕获实验中，TIM2_CH1 来捕获高电平脉宽，也就是要先设置输入捕获为上升沿检测，记录发生上升沿的时候 TIM2_CNT 的值。然后配置捕获信号为下降沿捕获，当下降沿到来时，发生捕获，并记录此时的 TIM2_CNT 值。这样，前后两次 TIM2_CNT 之差，就是高电平的脉宽，同时 TIM2 的计数频率我们是知道的，从而可以计算出高电平脉宽的准确时间，从而算出距离。

而两个通道中，他们共用的是同一定时器。如果他们同时发生中断，各自进入各自的函数中，那么问题就来了，以下代码示：

```
else
{
  TIM2CH1_CAPTURE_VAL=0;
  TIM2CH1_CAPTURE_STA=0X40;
  TIM2->CNT=0;
  TIM2->CCER|=1<<1;
}
```

这是中断函数中的一段代码，最重要的是第五行TIM2->CNT=0，试想如果两个通道同时发生中断，进入各自的函数中，均会执行这行代码，这行代码就是一个清零计数器，就像你得先把你的秒表清空了，才可以计时，对吧？但是两者在执行各自代码中，都将计数器清零！这样就会导致计数的错乱，从而显示的数字就会杂乱无章，没有规律。

解决方法

一开始问题出来的时候我也挺头痛的，最后看了一下学长的代码，才慢慢理解，自己又写了写，改了改，感觉始终没有学长的好用，所以不如直接用他的吧。

STM32单个定时器四通道输入捕获