**STM32运用总结**

主要分为IO口，定时器的PWM和QEI，中断，ADC，DAC和DMA介绍。在STM32的运用中第一步一般是使能相应模块的时钟，然后配置IO口，最后配置相应的寄存器。

# IO口

STM32的IO口非常多，而且与其它外设模块通常是复用的。在不同的外设中IO口的设置是不一样的。这一部分介绍普通的数值IO口。IO口有A-G共7组，每组16口。

1. IO口在时钟总线ＡＨＢ１上，使能对应端口的时钟。在寄存器RCC->AHB1ENR中。
2. 配置IO口的模式，普通的IO口配置为普通的输入输出模式。配置ＩＯ口是悬空还是上拉或者下拉。以上两步分别在寄存器GPIOx->MODER和GPIOx-> PUPDR（x=A,B,C,D,E,F,G）
3. 其中配置为输出模式时还要设置速度和相应的输出方式，开漏或者推挽，以上两步分别在寄存器GPIOx-> OSPEEDR和GPIOx->OTYPER（x=A,B,C,D,E,F,G）。
4. 设置IO口的高低电平。在寄存器GPIOx->BSRRH中置相应的位为1就是将相应的位置0，在寄存器GPIOx->BSRRL中置相应的位为1就是将相应的位置1.另外还可以设置GPIOx\_ODR寄存器来设置输出电平以及读取GPIOx\_IDR寄存器来获取输入电平。

# PWM

STM32的定时器也非常之多，用到的主要是两个部分：用定时器产生PWM和定时触发ADC，这里一部分介绍PWM。（高级定时器的配置和这差不多，由于在STM32F103里面已经尝试过在STM32F407里面就没有再写）

1. 配置IO口。我们说过STM32的外设模块主要是和IO口复用的，因此在使用外设模块时首先配置好相应的IO口。比如使用A口的PA1作为定时器Timer2的PWM输出。则应按照如下的步骤来配置PA1。
2. 使能A口的时钟。在寄存器RCC->AHB1ENR中。
3. 配置PA1为复用功能。在寄存器GPIOA->MODER中。
4. 配置PA1的上拉下拉或者悬空。在寄存器GPIOA->PUPDR中。
5. 配置PA1的速度。在GPIOA->OSPEEDR中。
6. 配置PA1的复用功能是和Timer2对应的。在GPIOA->AFR[0]中。（相对应的复用对应表数据手册上有）。
7. 配置定时器模块
8. 使能相应的定时器模块时钟（注意不同的定时器在不同的时钟总线上）。Timer2在APB1总线上。所以在RCC->APB1ENR中使能Timer2.
9. 设置定时器的预分频系数，周期值。在寄存器TIMx->PSC和TIMx->ARR中设置（对应不同的寄存器x取不同的值，比如用Timer2，则x=2）。
10. 软件产生一次更新事件将刚才设置的值导入到周期寄存器中。在TIMx->EGR中设置。
11. 使能自动重装载功能与否，使能相应的事件，配置计数方式。在TIMx->CR1中设置。
12. 配置Timer2的捕获比较通道。共有四路捕获比较通道，我们这里用的是CC2通道。将CC2配置为输出比较模式（OC），PWM2方式，预装载使能等。在TIMx->CCMR1（CC1/OC1和CC2/OC2）和TIMx->CCMR2（CC3/OC3和CC4/OC4）.
13. 设置匹配值，用来控制占空比。在TIMx->CCR2中。
14. 使能输出比较使能或者比较捕获功能。在TIMx->CCER中。
15. 清空计数值。TIMx->CNT=0。
16. 使能计数。在TIMx->CR1中。注意把这一项放在最后写，否者会由于寄存器之间的互锁写保护产生一些问题。

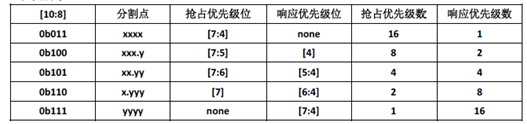
# QEI

QEI的使用主要是对照着数据手册上的说明来，具体没有什么多说的。程序里面有详细的注释。

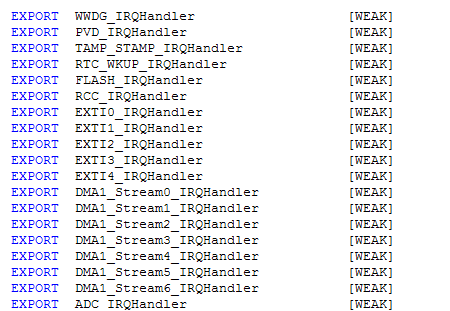
# 中断系统

STM32的中断主要设置中断优先级的配置，中断服务函数的命名，以及中断服务函数的处理等几个方面。

在STM32的中断系统中有抢占优先级和次占（响应）优先级的概念。其中数值越低代表优先级越高。高优先级的抢占可以打断低优先级的抢占。同优先级的抢占中，不同的次占（响应）优先级不可以相互打断，次占（响应）只是表示同时发生时先执行优先级高的次占。抢占优先级和次占（响应）优先级共占有4位，可以根据不同的设置来改变抢占优先级的次占（响应）优先级的位数，一个程序中抢占优先级的次占（响应）优先级的位数的设置只能有一次，且不能再改变。抢占优先级和次占（响应）优先级的配置之前要写入一个合适的序列才能实现配置。在SCB->AIRCR寄存器中先写入 0x05FA0000，然后根据下表中SCB->AIRCR[10:8]位数值的不同来配置抢占和次占（响应）的位数。例如SCB->AIRCR =0x05FA0000 | 0x400，表示抢占3为，次占（响应）1位。



1. 配置相应的中断优先级。首先要找到相应的中断的IP号（即在启动文件中的入口地址，如下图所示第一个为0，依次往后面数即可，在数据手册上也给出来了）。然后在寄存器NVIC->IP[n]中配置优先级，该寄存器共8位，其中低四位已被清0，不给用户使用，我们使用的是高四位。比如DMA2-Stream0的中断优先级配置：NVIC->IP[56]=0xB0。



1. 中断服务函数的命名。在上图中的启动文件里面也可以看到相关中断服务函数的命名的规则。就将相应的名称抄写下来即可。若要改名字，则也要改动启动文件。
2. 使能相应的中断源使能位和NVIC与之对应的中断使能位。NVIC->ISER[0:2]共三个，每个32位，对应IP号从0~81的中断。比如IP号为56的在NVIC->ISER[1]的（56-32）位。
3. 中断服务函数的处理。进入中断服务函数首要的任务是将不能硬件清除中断标志位的中断，清除其标志位。然后进行相应的处理，有必要关中断的要关中断。

# ADC

对于ADC来说主要是按照老师的要求完成配置任务，在开发板上实现的是运用ADC1和ADC2两个AD转换器实现4个模拟量两个一组同时采集。没有滤波，定时采样，采用DMA方式传送数据。并采用DA输出。所以这一部分介绍的内容较多，把AD和DA一起介绍了。

1. 配置ADC的IO口。采用C口的PC1,PC2,PC4,PC5作为4个模拟量的输入。所以使能C口的时钟。在RCC->AHB1ENR中。然后配置这4个IO口为模拟模式。在GPIOC->MODER中。最后配置上拉下拉或者悬空，这里要选择悬空模式。在GPIOC->PUPDR中配置。
2. 配置DAC的IO口。采用A口的PA4,PA5作为DAC的模拟输出。所以使能A口的时钟。在RCC->AHB1ENR中。然后配置这两个IO口为模拟模式。在GPIOA->MODER中。最后配置上拉下拉或者悬空，这里选择悬空。在GPIOA->PUPDR中配置。
3. 配置TIM2。使能TIM2的时钟。在RCC->APB1ENR中。设置分频系数和周期。在TIM2->PSC和TIM2->ARR中。产生更新事件，配置TIM2向上计数，使能更新事件配置PWM等。与（二）中的配置差不多。这里主要是运用TIM2的CC2通道事件来触发ADC转换。
4. 配置ADC。首先使能ADC1和ADC2的时钟。在RCC->APB2ENR寄存器中。设置ADC1和ADC2的总共的转换路数，这里我们转换的是每个ADC个两路。在ADCx->SQR1中设置（x=1,2）。配置每个ADC中每一通道转换的循序。在ADCx->SQRn（n=1,2,3,4）中设置。配置ADC的公共控制寄存器，主要是ADC时钟的分频，DMA的工作方式，ADC1和ADC2的工作模式等。在ADC->CCR寄存器中配置。配置采样周期。在ADCx->SMPRm（m=1,2）中设置。使能ADC的扫描模式。在ADCx->CR1中设置（扫描模式是指转换完一个通道接着转化下一个通道）。使能ADC1的TIM2的CC2上升沿触发，使能DMA方式，而且给ADC1上电。同时不使能ADC2的外部触发方式（作为从设备，由ADC1提供触发信号），但要使能ADC2 的DMA方式，而且给ADC2上电。
5. 配置DAC。使能DAC的时钟。在RCC->APB1ENR寄存器中设置。使能DAC1和DAC2。在DAC->CR中配置。
6. 配置DMA。使能DMA2时钟（我们用的就是DMA2）。在RCC->AHB1ENR中配置。主要配置外设地址，内存地址，传输数据的大小，外设数据的格式，内存数据的格式，内存地址递增模式，循环模式（即完成一次传输，内存地址重头开始放）。
7. 配置中断。主要是针对DMA2\_Stream0的配置，主要的说明前面的中断说明中提到过这里不再说了。
8. 使能定时器。
9. 在中断服务函数中，将DMA从ADC1和ADC2数据寄存器中运送回来的数据赋给DAC->DHR12R1和DAC->DHR12R2中再输出。

# USART

1.使能IO口时钟， RCC->AHB1ENR。设置相应的端口为为复用功能（包括TX和RX），其中TX还要设置输出速度，上拉下来；RX只要设置为复用模式就可以不管了

2.使能USART时钟，RCC->APB1ENR。设置波特率，USARTx->BRR；设置USARTx->CR1寄存器，包括USART使能，发射使能，接受使能，发射中断，接收中断等。（按需求来配置）

3.在USARTx->SR寄存器中。查询TC位（或者TXE位，发送完成；当读取此位并有后续数据写入时自动清0，或者软件清0）和RXNE位（接收完成；读取数据，自动清0或者软件清0）或者是中断处理。（要清除相应的中断标志位，在USARTx->CR1寄存器中）

4.在使用DMA的时候要在USARTx->CR3中设置DMAT和DMAR位。

● CC1S= ‘01’ (TIMx\_CCMR1 register, TI1FP1 mapped on TI1)

● CC2S= ‘01’ (TIMx\_CCMR2 register, TI2FP2 mapped on TI2)

● CC1P= ‘0’, CC1NP = ‘0’, IC1F =’0000’ (TIMx\_CCER register, TI1FP1 noninverted,

TI1FP1=TI1)

● CC2P= ‘0’, CC2NP = ‘0’, IC2F =’0000’ (TIMx\_CCER register, TI2FP2 noninverted,

TI2FP2=TI2)

● SMS= ‘011’ (TIMx\_SMCR register, both inputs are active on both rising and falling edges)

● CEN = 1 (TIMx\_CR1 register, Counter is enabled)

# 互补PWM

**互补PWM的配置很简单，按照寄存器的说明来进行即可，要注意相互之间的数量关系以及引脚的配置。**