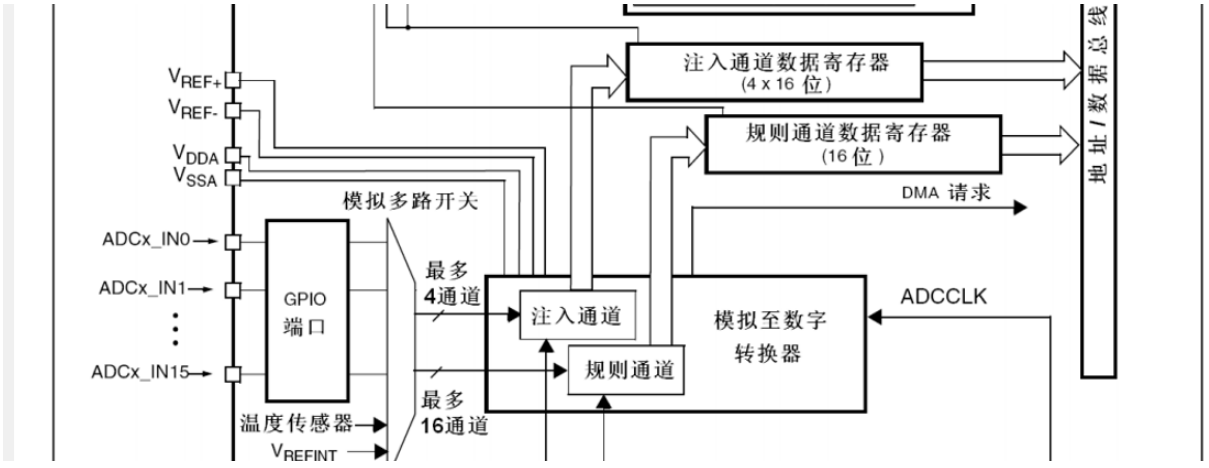
江协科技-2

DAC 应用场景被PWM占据，因为PWM的功耗更低；

ADC0809，12位4096，逐次逼近，二分法比较得到数字量；从高位到低位依次判断是否为0，判断12次即可得到结果。参考电压由电源电压决定；

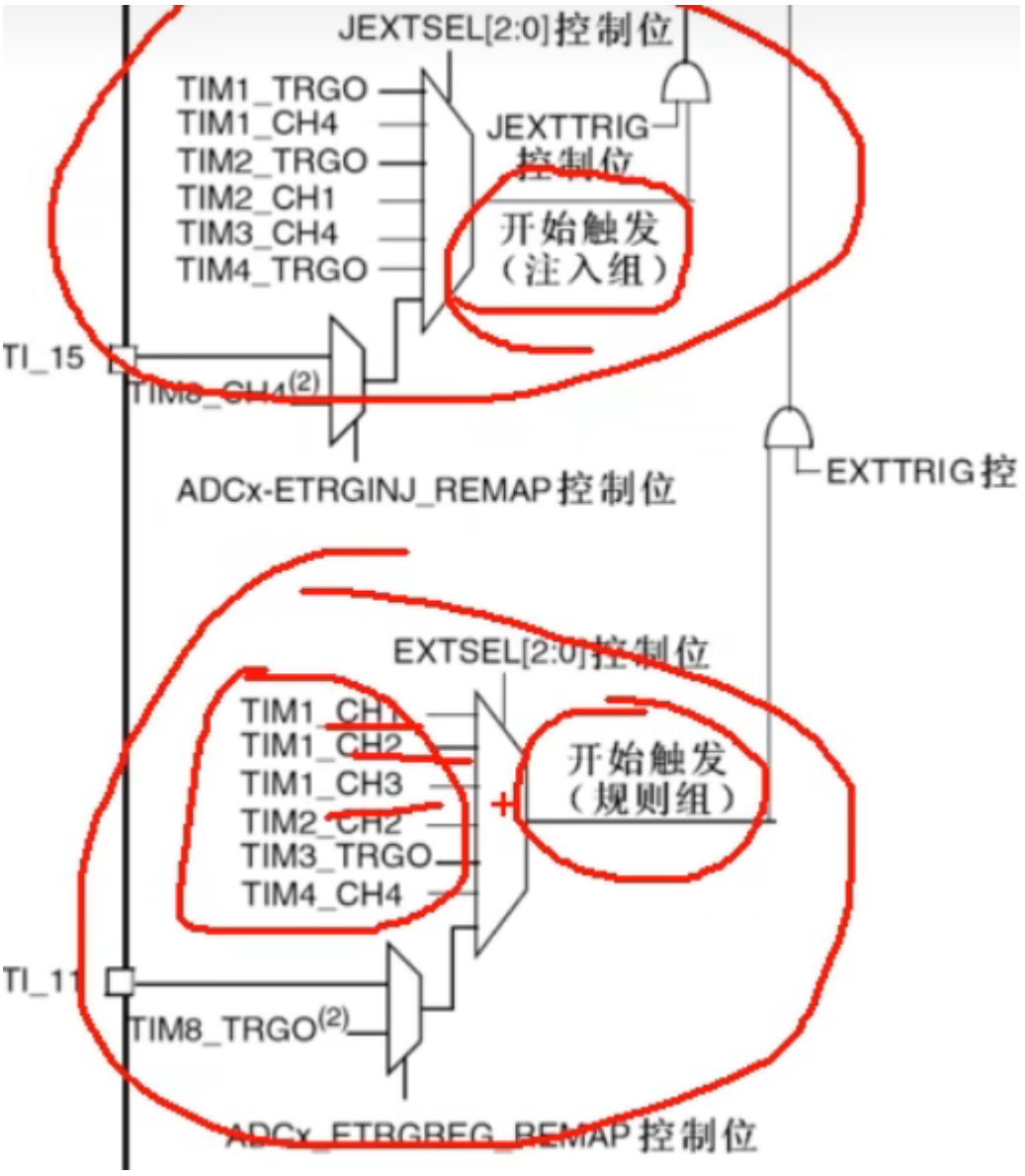
EOC end of convert

STM32的ADC



规则组： 可以同时测得16个数据，但是寄存器只能存储1个，需要配合DMA进行存储；

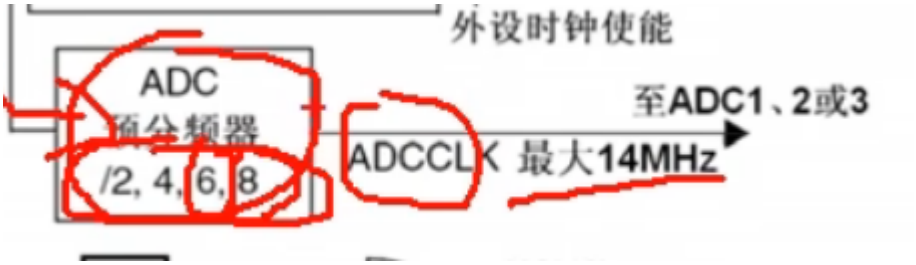
注入通道 可以同时测4个，使用较少；



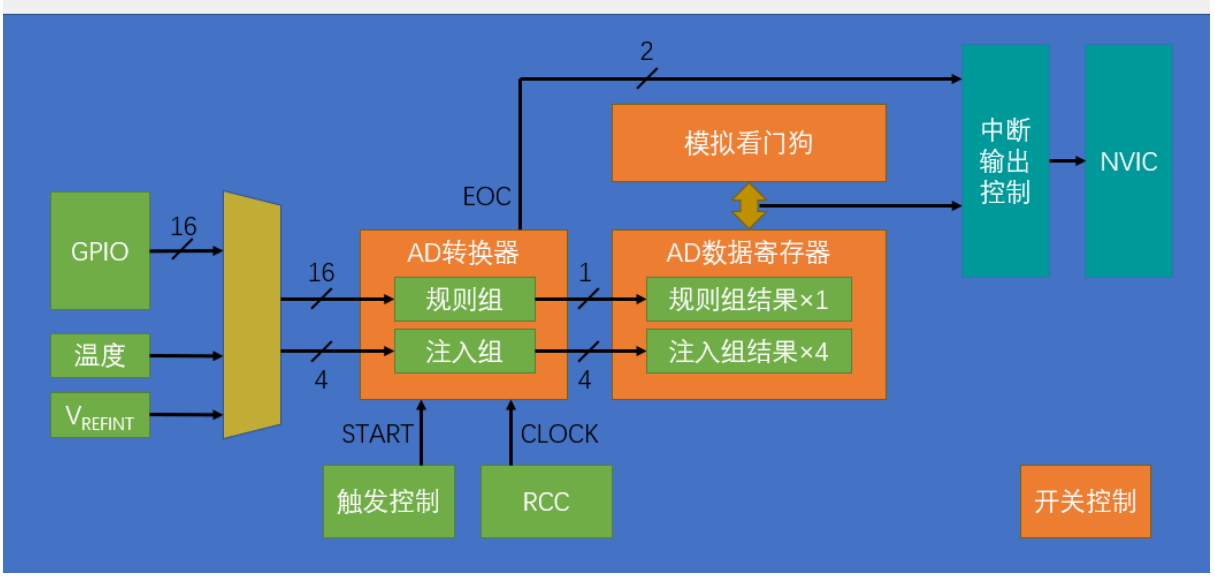
硬件事件：可以直接触发ADC，不再使用 定时器的中断；

（固定时间的中断影响主程序的执行，影响其他的中断；

TIM_TRGO：计时完成之后不产生中断，直接进入电路，开启ADC；



ADC的预分频器 只能选 6（12MHz）与8（9MHz）；



ADC_cmd() 开关控制；

通道	ADC1	ADC2	ADC3
通道0	PA0	PA0	PA0
通道1	PA1	PA1	PA1
通道2	PA2	PA2	PA2
通道3	PA3	PA3	PA3
通道4	PA4	PA4	PF6
通道5	PA5	PA5	PF7
通道6	PA6	PA6	PF8
通道7	PA7	PA7	PF9
通道8	PB0	PB0	PF10
通道9	PB1	PB1	

双ADC模式；

连续转换：转换完成之后，将会重新开始，只用触发一次（单次转换）

扫描模式：可以指定多个通道，同时进行转换；

表64　ADC1和ADC2用于规则通道的外部触发

触发源	类型	EXTSEL[2:0]
TIM1_CC1事件	来自片上定时器的内部信号	000
TIM1_CC2事件		001
TIM1_CC3事件		010
TIM2_CC2事件		011
TIM3_TRGO事件		100
TIM4_CC4事件		101
EXTI线11/TIM8_TRGO事件(1)(2)	外部引脚/来自片上定时器的内部信号	110
SWSTART	软件控制位	111

改变寄存器的值，修改外部触发方法；

对应的库函数定义如下：详见ADC_InitStruct.ADC_ExternalTrigConv

```
#define ADC_ExternalTrigConv_T1_CC1
#define ADC_ExternalTrigConv_T1_CC2
#define ADC_ExternalTrigConv_T2_CC2
#define ADC_ExternalTrigConv_T3_TRGO
#define ADC_ExternalTrigConv_T4_CC4
#define ADC_ExternalTrigConv_Ext_IT11_T

#define ADC_ExternalTrigConv_T1_CC3
#define ADC_ExternalTrigConv_None

#define ADC_ExternalTrigConv_T3_CC1
#define ADC_ExternalTrigConv_T2_CC3
#define ADC_ExternalTrigConv_T8_CC1
#define ADC_ExternalTrigConv_T8_TRGO
#define ADC_ExternalTrigConv_T5_CC1
#define ADC_ExternalTrigConv_T5_CC3
```

None 就是 软件触发方式；

数据右对齐：12位存到16位的寄存器中；

AD转换的步骤：采样，保持，量化，编码，ADC最快转换频率 1MHz，大约需要 1us；

ADC需要进行校准；

迟滞比较器：状态在高于上阈值、低于下阈值时发生改变，适用于数据不断抖动的情况，详见于施密特触发器；

GPIO端口的模拟输入模式 AIN，是一种专为ADC设计的输入模式。

```
ADC_GetSoftwareStartConvStatus( ADC1) //用来 检测 ADC软件触发转换
ADC_GetFlagStatus( ADC1, ADC_FLAG_EOC) //得到 ADC转换是否完成，很
```

```
ADC_GetConversionValue( ADC1);
//会 直接清除中断标志位EOC， 所以读取之后 可以不用清除中断标志位。

ADC_RegularChannelConfig(ADC1,ADC_Channel_1,1, ADC_SampleTime
//设置指定 ADC 的规则组通道，设置它们的转化顺序和采样时间
```

连续转换模式：

```
ADC_InitStruct->ADC_ContinuousConvMode = ENABLE;
```

此时 只需要 **一次软件触发**，ADC转换 将不断刷新寄存器，不断读出即可刷新；

如果想要在 单次转换、非扫描方式下（不使用DMA）展示多个ADC通道的计算结果，可以按照以下方式设计函数：

```
uint16_t adc_conversion( uint8_t ADC_Channel){

    ADC_RegularChannelConfig(ADC1,ADC_Channel,1, ADC_SampleTi

    ADC_SoftwareStartConvCmd(ADC1,ENABLE);
    while(ADC_GetFlagStatus( ADC1, ADC_FLAG_EOC) == RESET );
    return ADC_GetConversionValue( ADC1);//转换结束标志EOC将会自
}
//指定 已经使能的通道， 将这个通道的单次转换结果 保存在 寄存器中；
```

@July 31, 2024