ADC

使用STM32cubeMAX配置ADC博客：(91条消息) 【STM32】HAL库 STM32CubeMX教程九---ADC_stm32cubemx adc_Z小旋的博客-CSDN博客

ADC的转换顺序：

- 规则组转换

  某个ADC的通道按照一定的顺序进行转换，最多允许16个输入通道转换

- 注入组转换

  注入是打断原来的状态，相当于中断；比如当前规则组正在执行转换但突然注入组打断规则组，需要等待注入组转换完成后才能继续规则组的转换，注入组最多允许4个输入通道转换

ADC可以通过软件或者硬件触发转换。软件触发就是通过代码触发，而硬件触发有定时器和输入引脚触发，在我看来，硬件触发一般适用于周期性的触发方式。

当我们对采样率有要求的时候，ADC可以通过定时器触发，通过设置定时器的频率即可得到响应的采样率。

对于*STM32F4*系列的板子，*ADC*的输入电压一定要保持在*0~3.3V*之间

HAL库中关于ADC的函数：

```
//ADC启动转换函数
HAL_StatusTypeDef HAL_ADC_Start(ADC_HandleTypeDef *hadc);
//等待ADC规则组转换完成函数，一般和启动函数连用
HAL_StatusTypeDef HAL_ADC_PollForConversion(ADC_HandleTypeDef
*hadc,uint32_t Timeout);
//获取常规ADC转换值函数
uint32_t HAL_ADC_GetValue(ADC_HandleTypeDef *hadc);
```

ADC软件抗干扰：

- 多次采样取平均值

HAL库中关于DMA方式的ADC采样：

```
//启动DMA传输方式
HAL_Status TypeDef HAL_DMA_Start(DMA_HandleTypeDef *hdma,uint32_t
SrcAddress, uint32_t DstAddress, uint32_t DataLength);
```

ADC的设置模式(Mode):

- 独立模式
- 多ADC模式：多个ADC协同运行，如果多个ADC同时采集一个通道的信号，则采样速率是单通道的两倍

ADC转换模式:

- 单次转换

  单通道，ADC只执行一次转换，可以设置为外部触发或软件触发，转换完成后如果设置了中断就触发中断

- 连续转换

  单通道，ADC执行多次转换(连续不断转换)，可以设置为外部触发或软件触发，如果设置了中断就表示每次转换结束后都会触发一次中断

- 扫描模式

  多通道，按顺序对规则组和注入组执行一次转化，如果设置了中断，那么每个通道转换完成后都会触发中断

- 连续扫描模式

  多个通道+连续转换，和上面相同

ADC可以配置一下三种形式:

- 单通道
- 单通道+DMA
- 多通道+DMA：适用于高速采样的情况

说明:

- 如果设置了连续采样之后，并且触发方式为软件触发，这个时候我们不需要在代码中多次执行采样函数，因为是连续转换，我们只需要对其进行一次使能即可让ADC一直进行采样。
- 在进行多通道DMA传输时，在数组中数据的分布为[A,B,A,B...]，A代表第一个通道采集的数据，B代表第二个通道采集的数据。

单通道+单次转换+软件触发：

```
while(1)
{
    HAL_ADC_Start(ADC_HandleTypeDef *hadc);//启动转换
    HAL_ADC_PollForConversion(ADC_HandleTypeDef *hadc,uint32_t
Timeout);//等待转换结束
    val = HAL_ADC_GetValue(ADC_HandleTypeDef *hadc);//获取转换后的值

}
```

单次转换需要不断的执行HAL_ADC_Start函数来进行连续的采样。如果设置了连续转换，那么HAL_ADC_Start函数只需要执行一次，如果需要停止ADC可以使用函数HAL_ADC_Stop函数。

单通道+单次转换+定时器触发：

```
HAL_TIM_Base_Start(&htim3);//启动定时器
HAL_ADC_Start(&hadc1);//启动转换
while(1)
{
    val = HAL_ADC_GetValue(&hadc1);//获取转换后的值
}
```

定时器触发的特点是具有一定的频率，并且我们可以人为的设置该频率。

注意：在开启定时器后必须使用HAL_ADC_Start(&hadc1)函数来启动ADC，否则ADC不会工作。

单通道+连续转换+软件触发：

```
HAL_ADC_Start(ADC_HandleTypeDef *hadc);//启动转换
while(1)
{
    val = HAL_ADC_GetValue(ADC_HandleTypeDef *hadc);//获取转换后的值

}
```

在实验过程中发现如果以上述方式设置连续转换，则采样值是固定不变的，和单次转换没有区别

单通道+连续转换+DMA传输 + normal模式：

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)  //回调函数判断转换是否完成
{
    if(hadc->Instance == ADC1)  //在回调函数中将转换完成标志位置为1
        state = 1;
}

HAL_ADC_Start_DMA(&hadc1,(uint32_t *)buf,10);//启动DMA
while(1)
{
    if(state == 1)
    {
        //数据处理
        state = 0;
        HAL_ADC_Start_DMA(&hadc1,(uint32_t *)buf,10);//由于设置为normal模式需要重新启动
    }
}
```

开启连续转换后，会一直执行ADC采样，并且通过DMA传输，一般的实现方式是，通过DMA传输完成的中断函数将转换完成标志置为1，然后再主循环判断该标志是否为1，如果为1执行数据处理程序，否则等待传输完成

如果为软件触发不论是单次转换还是多次转换都需要使用HAL_ADC_Start函数来开启ADC，如果为定时器触发，需要开启定时器和ADC两个

在连续转换模式中，如果要重新进行采样和DMA传输时，需要先把当前的ADC采样和DMA传输关闭然后再开启采样和DMA传输。

注意：存放DMA传送数据的数组应该设置为16位的，如果设置为32会导致误差。

单通道+连续转换+DMA传输 + circular模式：

此时HAL_ADC_Start_DMA函数只需要在主循环外执行一次即可。

多通道+单次转换+非DMA：

```
while(1)
{
    for(i=0,i<通道数,i++)
    {
    HAL_ADC_Start(ADC_HandleTypeDef *hadc);//启动转换
    HAL_ADC_PollForConversion(ADC_HandleTypeDef *hadc,uint32_t
Timeout);//等待转换结束
    val = HAL_ADC_GetValue(ADC_HandleTypeDef *hadc);//获取该通道转换
后的值
    HAL_ADC_Stop(ADC_HandleTypeDef *hadc);//停止转换
    }
}
```

多通道 + 连续转换 + DMA

此时和单通道类似，只不过需要注意各个通道在数组中的存放位置

ADC采样结束回调函数：

在单通道+单次转换时，如果开启了ADC中断，则在每次转换结束后都会调用该中断函数

如果是使用了DMA方式，那么会在每次DMA转换结束后调用该函数

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
{
    if(hadc->Instance == ADC1)  //在回调函数中将转换完成标志位置为1
        state = 1;
}
```

回调函数的使用通常是进行数据处理。我们可以在回调函数中将一标志位置为1，然后在主循环中根据该标志位进行数据处理。