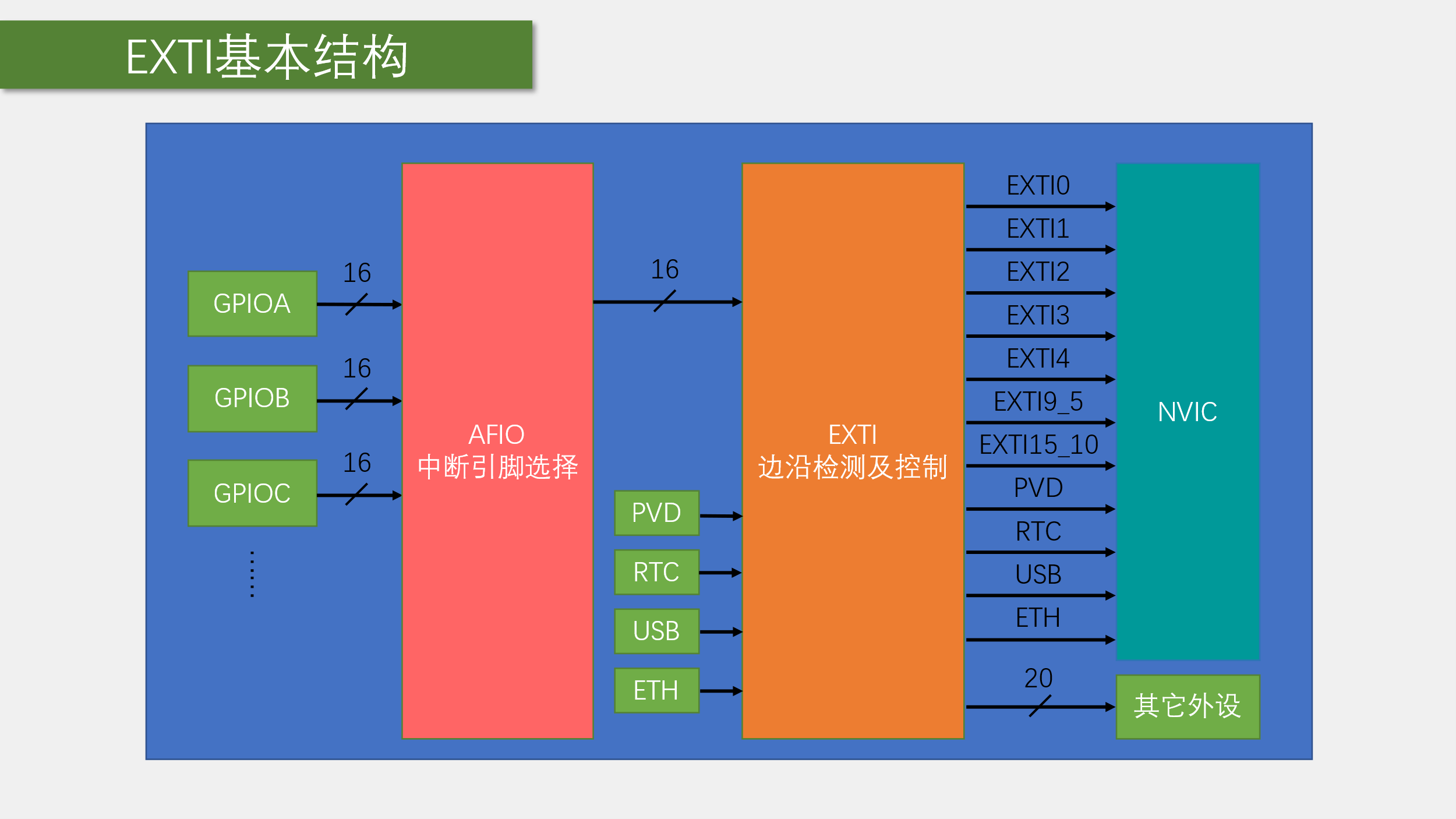
GPIO中断结构参见下图。



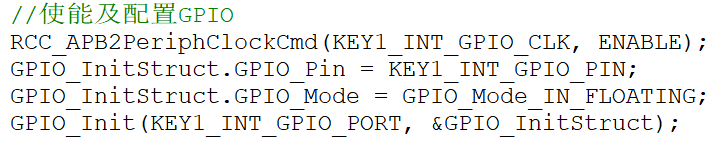
**GPIO中断的基本流程是：**

首先GPIO产生一个电信号，这个电信号被EXTI捕捉、识别，并发送信号给内核。内核中由NVIC接受EXTI发送来的中断，对其排序，通知CPU响应中断。

在这个过程中，需要操作：GPIO，AFIO，EXTI，NVIC，以及编写对应的响应函数。其介绍及配置方法如下。

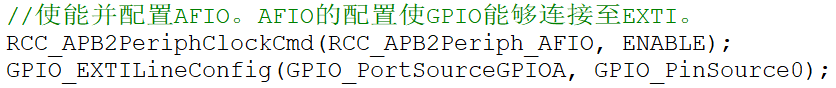
**1、GPIO。(stm32f10x\_gpio.h)**

GPIO外设需要使能，并初始化响应的端口、引脚，将其配置为输入模式。



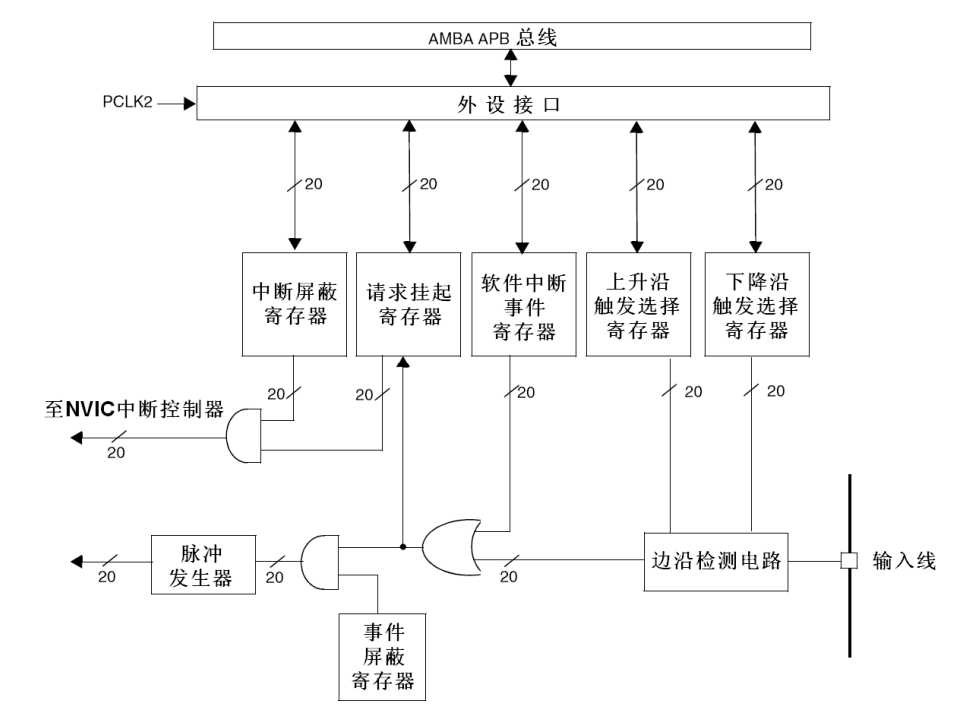
**2、AFIO。(stm32f10x\_gpio.h)**

AFIO在STM32中有两个作用，一负责引脚复用，二负责管理GPIO的中断是否被放出。使用前需要对时钟使能。

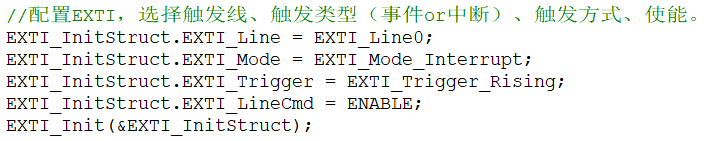


**3、EXTI。(stm32f10x\_exti.h)**

EXTI负责管理所有GPIO产生的中断信号，并负责向内核发送中断请求。它的设置包括：哪条线路产生了中断，中断触发方式（上升or下降or both），触发类型（事件or中断），使能中断屏蔽寄存器。其对应框图如下。

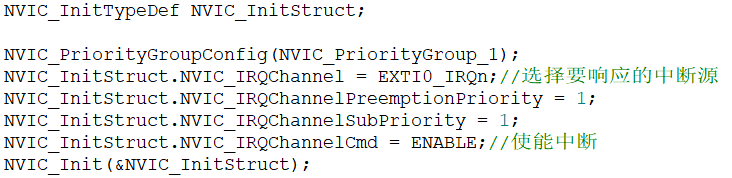


其配置方式为：



**4、NVIC。(msic.h)**

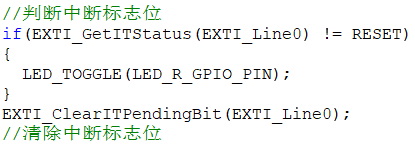
NVIC负责响应所有外部中断，包括除了GPIO之外的，例如RCC或定时器等产生的中断，它是一个内核中存在的通用的中断管理器。使用时需要配置优先级分组，并选择要响应的中断源；配置父子优先级，最后使能该中断。其代码配置如下。



**NB:**在选择要响应的中断源时，需要留意EXTI5~9，10~15使用同一条线路。因此没有例如EXTI10\_IRQn的写法。

**5、中断函数编写。(stm32f10x\_it.h)**

中断函数名称需要严格遵循startup\_stm32f10x\_hd.s中的函数名称，并将函数编写至stm32f10x\_it.c文件中。在函数中，首先要判断中断标志位，其次在函数结束后要清除中断标志位，否则程序将陷入中断中。



**6、内核判断中断来源的方法。**

1）、由于4中NB所述，EXTI5~9，10~15使用同一条线路，因此内核接收到中断之后需要对其判断分辨时哪条线来的中断，即调用EXTI\_GetITStatus方法，可以获知某一条线路是否进入中断。

2）、若GPIOA2、GPIOB2都有中断产生，由于EXTI将A~G口同脚位的信号打包经由EXTI2发送给NVIC（见下图），因此NVIC并不知道EXTI2\_IRQn的中断具体来自A~G的哪个端口。此时就需要在中断程序中再判断是由哪个口产生的中断信号。具体可以读取GPIO输入数据寄存器中的值。

