Interrupt

> FROM: **https://zhuanlan.zhihu.com/p/196452953**

1. 单片机中的中断机制

以`cortex-M3`内核为例

中断向量表

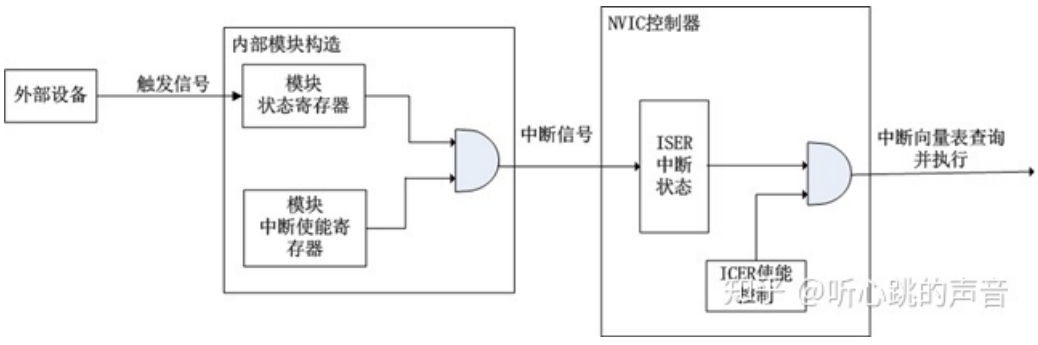
对于Cortex-M3内核，支持最大编号包含0~255的中断类型

- **0~15为系统异常，**主要处理系统执行中产生的复位，错误，主动触发的SVC，异常
- **编号16~255则是由芯片设计厂商自定义设计，**用于满足芯片功能需求的中断(芯片厂商可以自由定制, 不超过最大编号且不重复即

反映在软件中就是`startup_xxx.s`启动文件中定义的中断向量表

系统中断是在内核定义时确定的，外部中断在芯片设计时被确定

中断流程



1. 外设或者模块触发 - 模块内部状态变化 - 修改状态标志位
2. 模块将中断信号提交到中断向量控制器(也就是NVIC)
3. NVIC根据中断信号定义的编号信息，置位ISER中的相应中断Pending位，在配合ICER的使能状态

原理

`SCB->VTOR`寄存器的值对应中断向量表的首地址， 而中断的编号和其在向量表中的位置是**一一对应**的。

以中断向量表的首地址为`start`，中断编号为`num`为例，则中断的入口地址为：

```
Saddr = Start + num*4;
```

内核将**修改后的地址赋值给`PC`指针**，即可实现软件代码的执行路径改变

中断优先级

图 7.6　3 位优先级，从比特 1 处分组

优先级等级

决定中断执行顺序的等级，M3规定优先级值越小，优先级越高

中断嵌套

当一个中断打断另一个中断的执行流程时，优先执行时，被称为中断嵌套

抢占优先级

优先级的寄存器高位`bit[7:n]`，声明抢占优先级高的中断可以打断低优先级中断，进行中断嵌套。

亚优先级

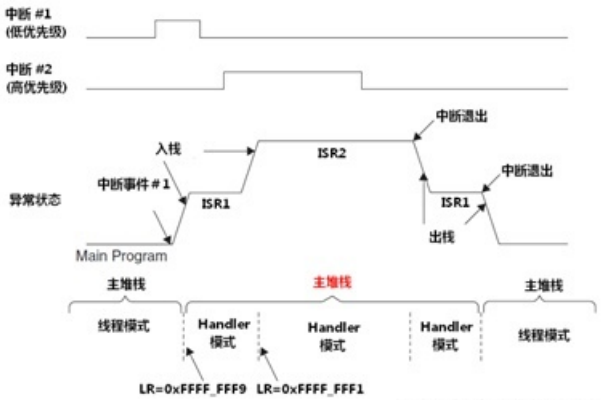


图9.4　LR的值在异常期间被设置为EXC_RETURN（线程模式使用主堆栈）

优先级的寄存器高位`bit[n-1:m]`，声明亚优先级高的中断，能够优先执行，但不能打断其它中断，需要等低优先级中断结束后才能执行

1. 中断`#1`判断为触发，进行入栈操作，同时系统模式从线程切换到`Handler`模式，开始执行中断服务函数`ISR1`
2. 当优先级更高的中断`#2`判断为触发，则打断中断`#1`的执行，再次执行入栈操作，系统模式不变，执行中断服务函数`ISR2`
3. `ISR2`执行完毕后，进行出栈操作，执行`ISR1`的后续部分，系统模式不变

4. `ISR1`执行完毕后，进行出栈操作，后续进行被打断的主循环中，继续执行，同时系统模式从`Handler`切换为线程模式，整个中断嵌套的流程执行完毕。

2. 嵌入式Linux中的中断机制

未完待续