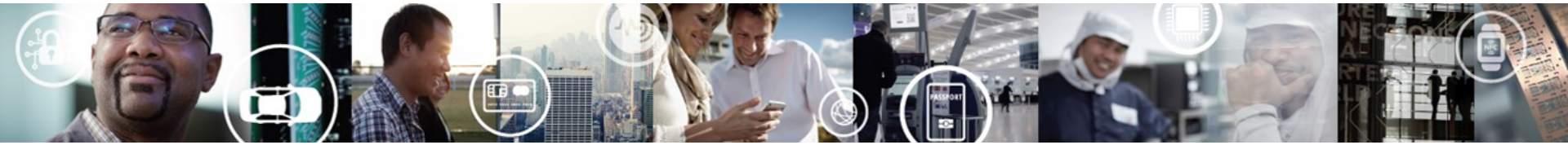
LPC82X 培训资料

外部引脚中断

MAY, 2016



内容

- PININT模块功能简介
- PININT单引脚中断
- PININT模式匹配引擎

PININT模块功能简介

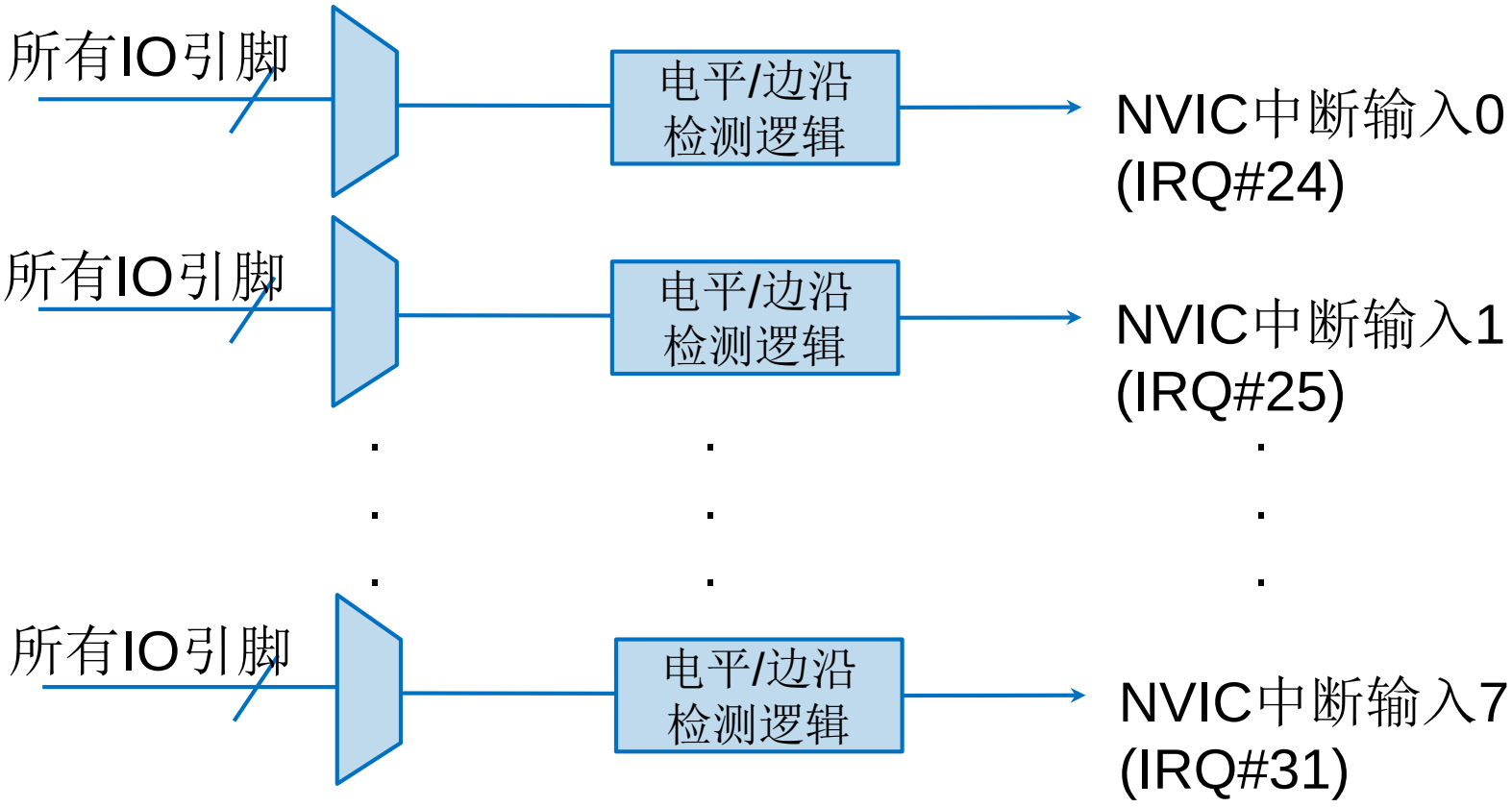
PININT模块功能简介

- 有8路引脚信号输入，每路输入可以与任何一个IO引脚配接
  - 这意味着每个IO脚都能产生外部中断，但最多选出8个脚
  - 8路输入可以按两种方式产生中断：引脚中断 与 模式匹配中断
  - 单个引脚可以配接到多路输入
  - 注意：即使某个IO脚不是当作GPIO使用，也能配接到PININT的中断源上。
- (常用)引脚(Pin)中断：单路信号产生中断，也就是常说的 “外部中断”。
  - 触发条件可配置为：
    - 电平触发：高电平、低电平
    - 边沿触发：上升沿、下降沿、双边沿
  - 8路输入每路都在NVIC中有自己专用的中断号，共**8个**
- (高级应用)模式匹配(Pattern match)中断：1-8路信号经过布尔位运算为真时产生中断 ，每路信号亦可以选择电平或刚刚的边沿来触发
  - 例如，三人意见表决器，以少数服从多数原则：(A&B)|(B&C)|(C&A)

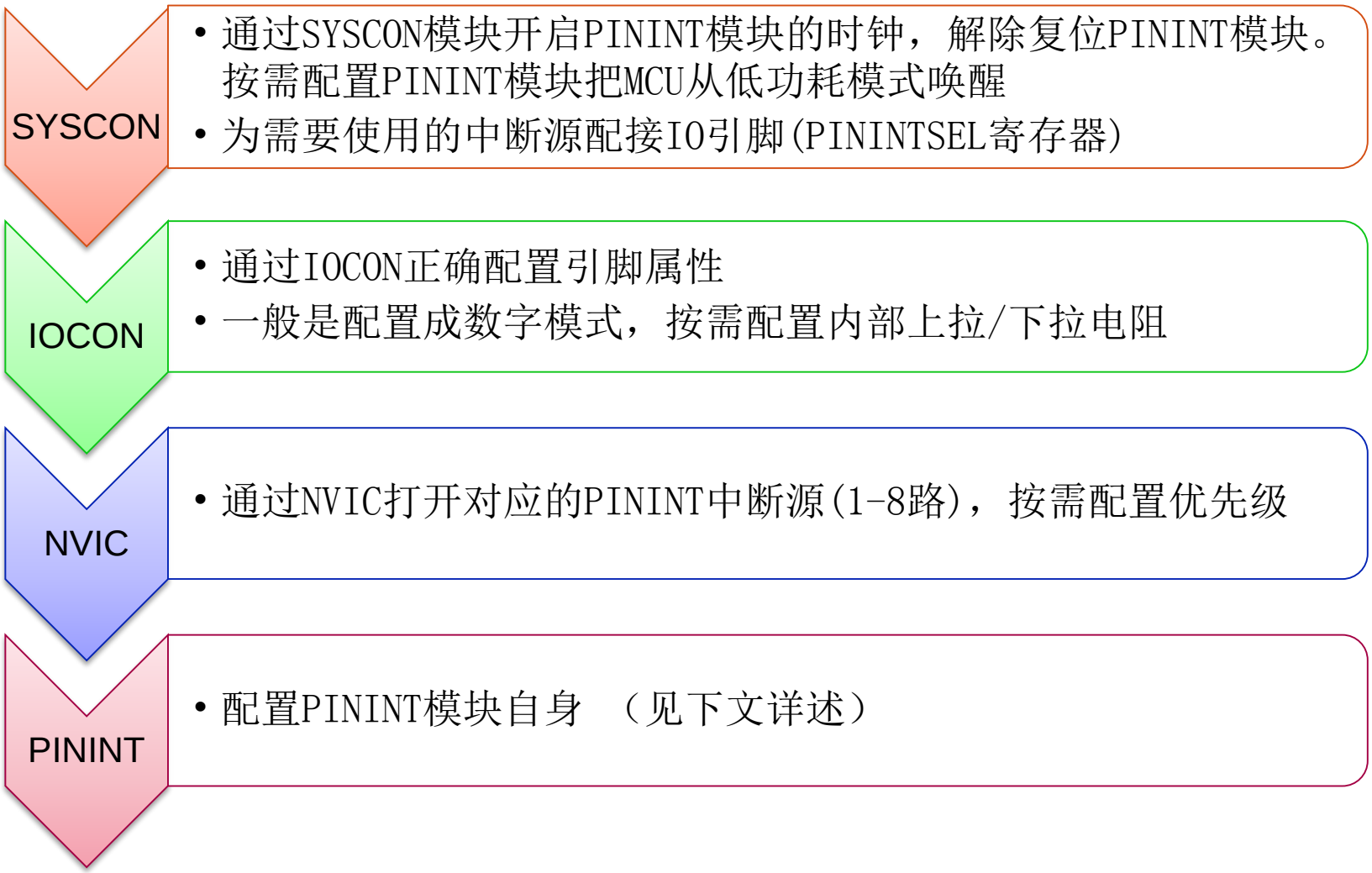
PININT单引脚中断

单引脚中断的功能框图

- 共8路独立工作的引脚输入中断信号，每路都可配接到任何IO引脚，而且不限于用作GPIO的脚

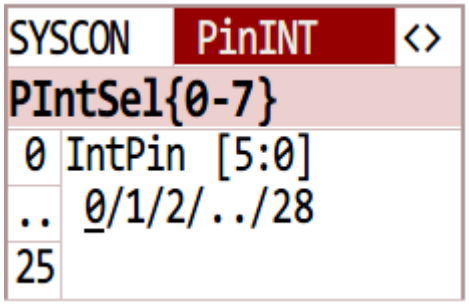


使用PININT模块的步骤



PININT中断源与IO引脚的配接 (IO不是GPIO)

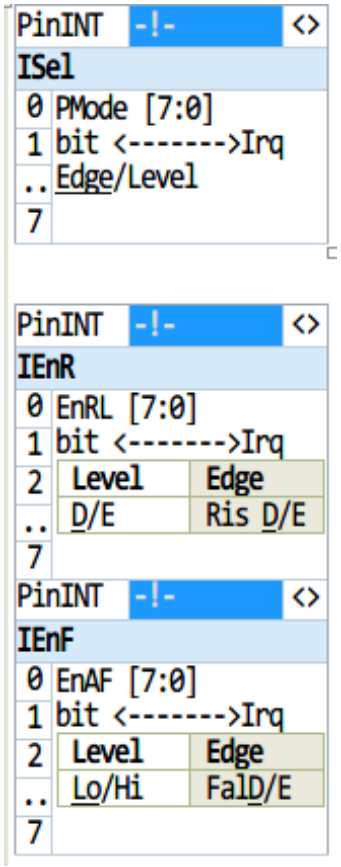
- 在SYSCON模块中，有一组寄存器PINTSEL[8]，用于配接每个PINTSEL对应一路中断源
- 寄存器的值决定配接到哪个IO口
  - 端口号：PINTSEL# / 32
  - 引脚号：PINTSEL# % 32
- 例：把P0_20配接到5号中断源
  - SYSCON.PINTSEL[5] = 20



- 即使某个引脚并不作为GPIO使用，也能配接到PININT的中断源
  - 例如：要自动探测4个I2C接口中哪一个被连接到主机上，即可以把它们4个的SCL线所在的IO脚分别配接到4路PININT中断上。通信时，发生哪个中断，就认为连接到了哪个I2C接口
  - 上例中，IO引脚并非作为GPIO使用，而是作为I2C的SCL信号

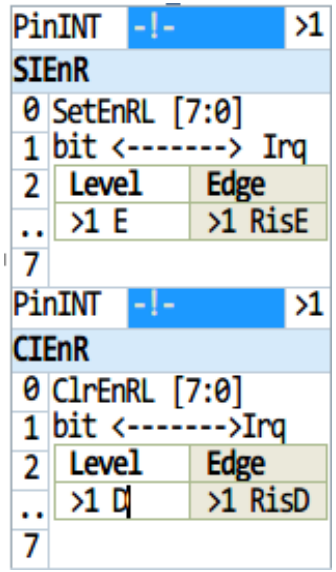
PININT单引脚中断的寄存器配置

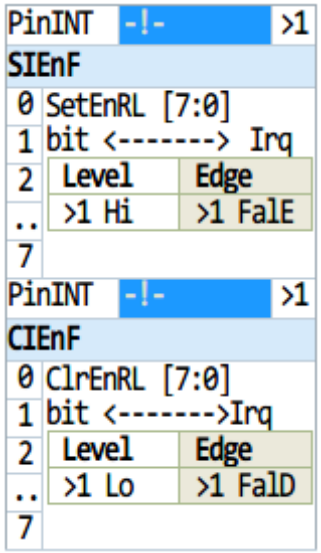
- 所有寄存器都是每个位(bit)对应一路引脚中断源
- 通过 ISEL寄存器决定信号的触发是电平还是边沿
  - Chip_PININT_SetPinModeEdge()：配置为边沿触发
  - Chip_PININT_SetPinModeLevel()：配置为电平触发
- 配置IENR和IENF寄存器以决定触发方式的细节
  - 电平和边沿共享这一对配置寄存器，但是有各自的用法
    - 电平触发：分别为每路输入设定是否允许电平触发和选择高/低电平
    - 边沿触发：分别为每路输入设定是否分别允许上升沿和下降沿触发
      - 如果同时允许两个，就实现了双边沿触发。
    - 相关API (既用于电平触发，也用于边沿触发)
      - Chip_PININT_EnableIntHigh() : 使能高电平或上升沿
      - Chip_PININT_DisableIntHigh() : 除能高电平或上升沿
      - Chip_PININT_EnableIntLow() : 使能低电平或下降沿
      - Chip_PININT_DisableIntLow() : 除能高电平或上升沿



用于化简“读-改-写”为单次操作的伴侣寄存器

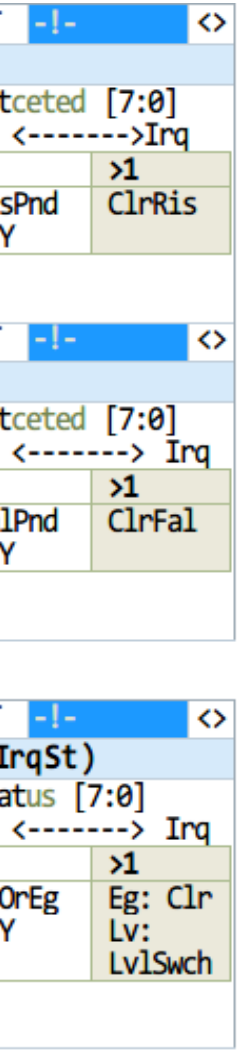
- 为了免除“读-改-写”的麻烦，IENR和IENF都有对应的两个伴侣寄存器，分别用于置1和置0
  - IENR的伴侣寄存器是 SIENR, CIENR; IENF的是SIENF和CIENF
  - 伴侣寄存器都是写1有效的，写0无效，避免了读-改-写
  - 往置1寄存器(SIENR, SIENF)的位里写1，主寄存器的相应位被置1
  - 往置0寄存器(CIENR, CIENF)的位里写1，主寄存器的相应位被置0





单引脚中断的配置(续)

- 边沿在检测后被锁存，因此提供了边沿检测与清除的功能，由寄存器”RISE”和”FALL”负责
  - –被读取时，反映上升沿/下降沿是否已检测待处理
    - Chip_PININT_GetRiseStates() , Chip_PININT_GetFallStates()
  - –被写1时（写0无效），清除对应的边沿检测锁存标志
    - Chip_PININT_ClearRiseStates() , Chip_PININT_ClearFallStates()
- 通过”IST”寄存器要查看各路中断请求的状态，并快速执行常用操作
  - –被读取时，反映当前各路中断是否已请求(触发)
    - Chip_PININT_GetIntStatus()
  - –被写入时，只能写1，写0无效
    - 对于边沿触发，同时清除双边沿检测的锁存标志
    - 对于电平触发，切换触发电平
    - Chip_PININT_ClearIntStatus()
    - 注意：电平触发的中断由硬件在有效电平消失时自动清除标志，没有软件清除电平中断标志的操作——会被解释为切换触发电平！



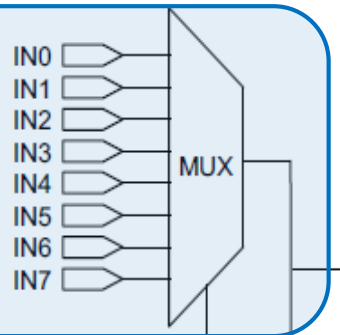
PININT模式匹配引擎

模式匹配引擎功能简介 引脚中断功能的扩展

- 模式匹配引擎是PININT模块的一部分，扩展了单引脚中断的功能。把多个引脚的状态进行逻辑运算后再产生中断
  - 支持“与”，“或”，“非”，不原生支持“异或”
- 实现方式是，创建一个或多个布尔表达式，每个布尔表达式都可以产生中断请求
  - 共有8个输入变量，每个变量可以与任一个PININT输入配接
  - 同一变量可以在一个表达式中出现多次，
  - 例如(三人表决器): A&B | B&C | C&A

模式匹配引擎的8个输入变量(Slice)

模式匹配引擎的每路变量输入可以配接到PININT外设的任一路输入上
通过**PMSRC**寄存器配置



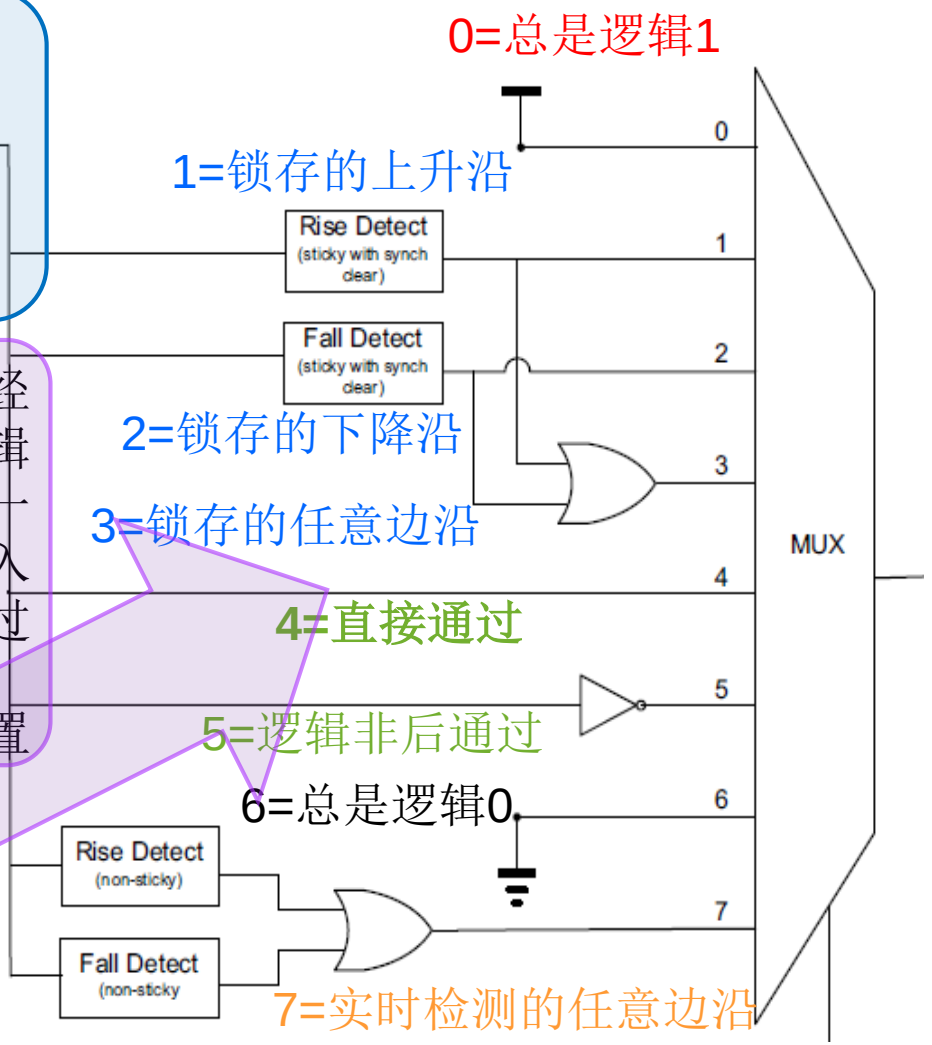
PinINT Ptn	
PMSrc	
0	>0
..	
7	
8	Src0 [2:0]
9	SelFromPinINTSel:
10	0/1/2/3/4/5/6/7
11	Src1 [2:0]
12	
13	
14	Src2 [2:0]
15	

… 每路3位

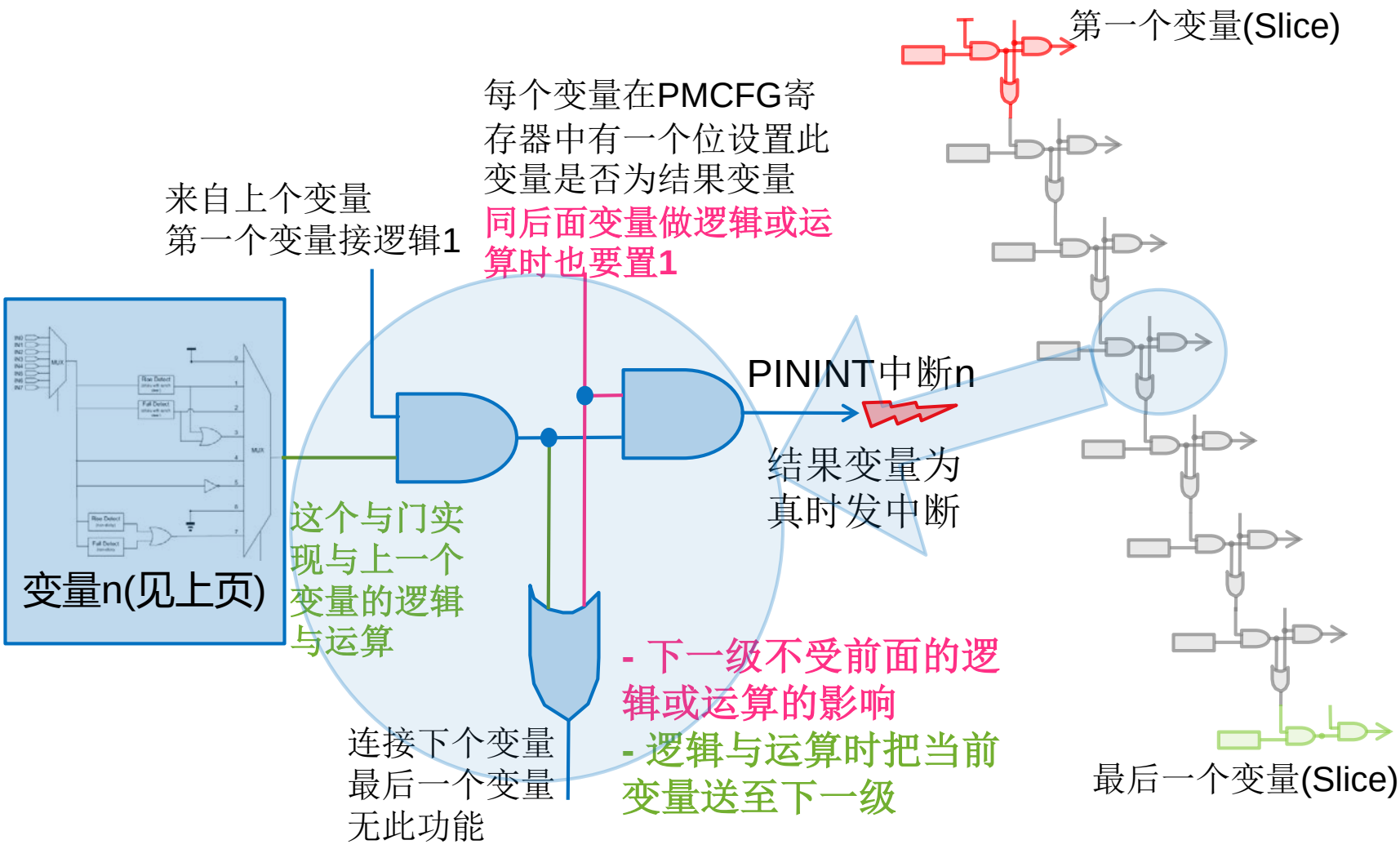
28	
29	Src7 [2:0]
30	
31	

PinINT Ptn	
PMCfg	
0	Prod_Endpts[7:0]
1	bit<----->slice
2	0=Slice 是中间变量
3	
4	1=Slice 是结果变量
5	(Endpoint)
6	
7	结果变量(Endpoint) 表达式为真时发中断
8	3bits <----> slc
9	
10	0=Always 1
11	1=StickyRisingEdge
12	2=StickyFalingEdge
13	3=StickyAnyEdge
14	4=HiLevel
15	5=LoLevel
16	6=Always 0
17	7=RealtimeAnyEdge

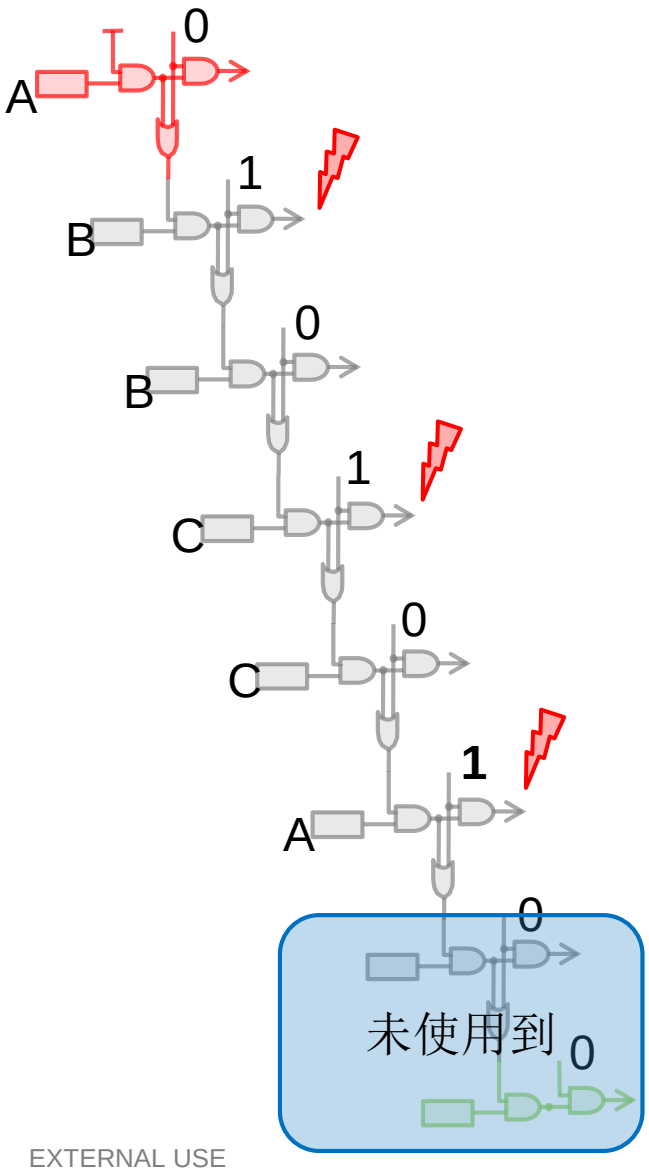
选出后再经过一种逻辑变换成为一个真正输入变量，通过**PMCFG**寄存器配置



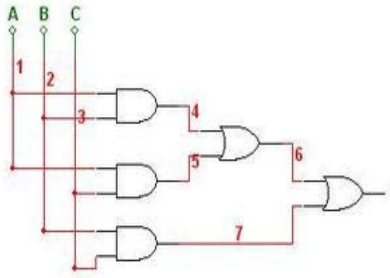
变量布尔运算的实现逻辑



示例 三人表决器



- A,B,C三人投票表决，多数赞成时通过，Y= A&B | B&C | C&A，可如下设置
  - 变量0: A, 直通, 中间变量
  - 变量1: B, 直通, 结果变量
  - 变量2: B, 直通, 中间变量
  - 变量3: C, 直通, 结果变量
  - 变量4: C, 直通, 中间变量
  - 变量5: A, 直通, 结果变量



- 需要响应PININT的1号，3号，5号中断
- 核心寄存器操作示意代码如下
  - // 1. 配置PININT以模式匹配引擎的方式工作
  - PMCFG = 1;
  - // 2. 为所需变量(共6个)选择输入信号
  - PMSRC = 0<<8|1<<11|1<<14|2<<17|2<<20|0<<23;
  - // 3. 设置中间变量(后面做逻辑与)和结果变量(后面做逻辑或，或者为表达式的完结点)
  - PMCFG = 0<<0|1<<1|0<<2|1<<3|0<<4|1<<5;
  - // 4. 为每个变量输入选择直通后送出
  - PMCFG |= 4<<8|4<<11|4<<14|4<<17|4<<20|4<<23;

SECURE CONNECTIONS
FOR A SMARTER WORLD