EE351
微机原理与微系统

姜俊敏

电子与电子工程系

第七章 中断

本章学习目标

掌握单片机中断系统

掌握单片机中断处理过程

掌握中断程序设计

理解中断使用过程中需要注意的问题

中断的概念

- 中断是计算机中的一个很重要的技术，它既和硬件有关，也和软件有关。
- 正因有了中断技术，计算机的控制功能才更加灵活、效率更高、计算机的发展和应用才更好，中断功能的强弱已成为衡量一台计算机功能完善与否的重要指标。
- 例如，下面的情况下，就需要采用中断技术。
  - 当计算机正在正常运行一个程序段的时候，若有一紧急事件出现，又必须要立即处理这个紧急的事件；
  - 计算机边工作边随时准备处理一个事件，但又不确定该事件出现的确切时刻，像处理防火防盗事件一样。

中断的概念

- 所谓中断是指CPU在正常执行程序的过程中，出现某些事件需要立即处理时，CPU暂时中止正在执行的程序，转去执行对某种请求的处理程序(中断处理(服务)程序)；
- 当处理程序执行完毕后，CPU再回到先前被暂时中止的程序继续执行。
- 实现这种功能的部件称为中断系统，请求CPU中断的请求源称为中断源。
- 中断源向CPU发出中断申请，CPU暂停当前工作转去处理中断源事件称为中断响应。对整个事件的处理过程称为中断服务。
- 事件处理完毕CPU返回到被中断的地方称为中断返回。

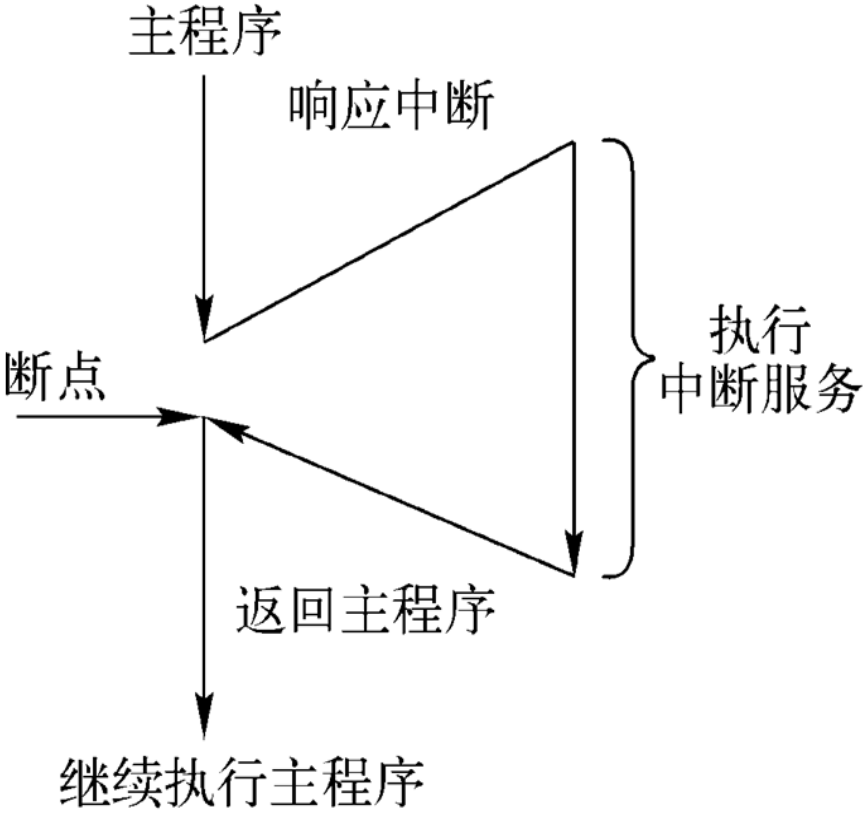
中断的概念

◆中断过程:

◆多个中断源同时向CPU请求中断时，就出现了CPU应该先响应哪个中断请求的问题。

◆计算机往往根据中断源引发事件的轻重缓急为其设置不同的优先级，优先级是计算机对中断源响应次序的规定。

◆优先级高的中断请求先响应，优先级低的中断请求后响应。

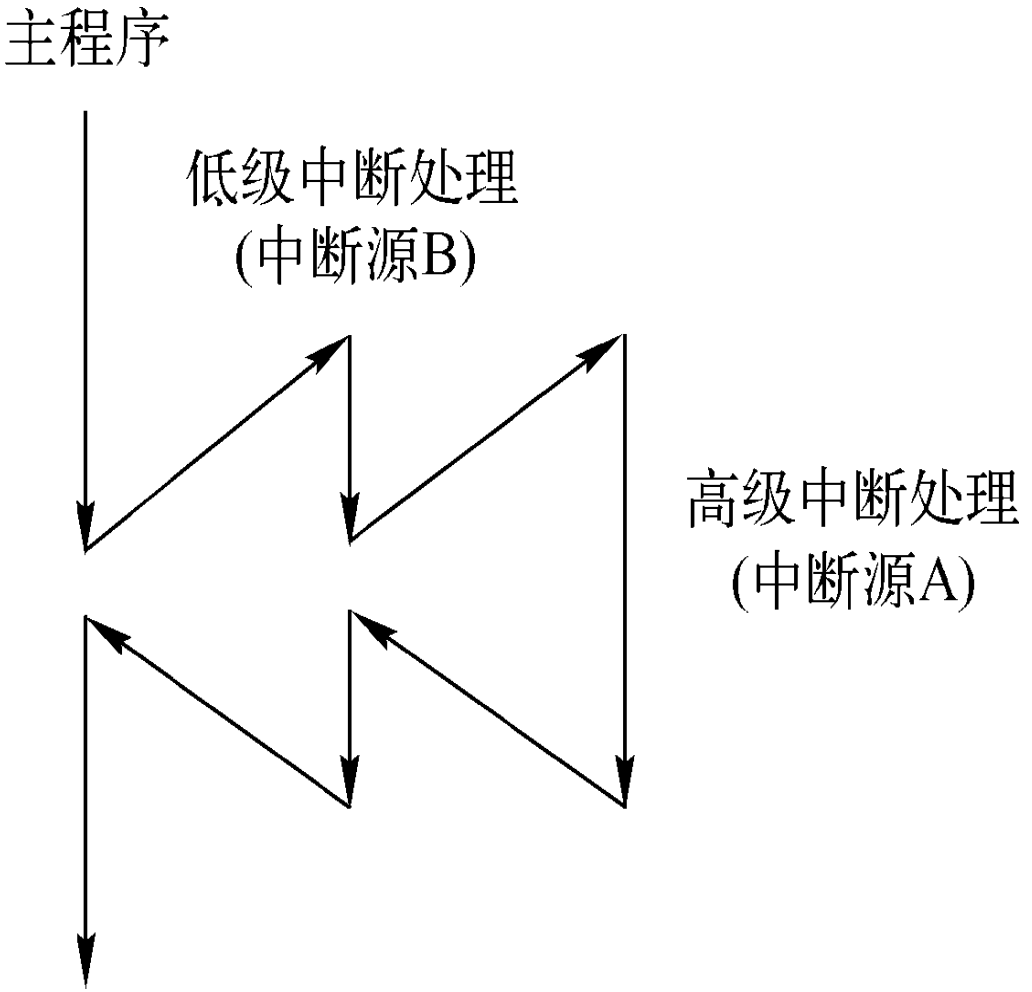


中断过程示意图

中断的嵌套

CPU 在进行中断响应时可以响应更高级的中断请求，这种情况称为中断的嵌套。

需注意：引起中断嵌套的中断源的优先级一定要高于当前响应中断源的优先级，同优先级或低优先级中断源的中断请求不能引起中断嵌套。



中断嵌套示意图

中断的概念

◆计算机采用中断技术, 大大提高了工作效率和处理问题的灵活性, 主要表现在3个方面：

- 可及时处理控制系统中许多随机发生的事件；
- 解决了快速CPU和慢速外设之间的矛盾, CPU和外设可并行工作;
- 具备了处理故障的能力，提高了系统自身的可靠性。

◇ 中断类似主程序调用子程序，但它们又有区别:

中断和调用子程序之间的主要区别

中　断	调用子程序
产生时刻是随机的	程序事先安排好的
既保护断点(自动), 又保护现场(中服程序)	可只保护断点(自动)
处理程序的入口地址是单片机硬件确定的, 用户不能改变(中断矢量表: 0003H~00BBH)	子程序的入口地址是程序编排的

Flash程序存储器

中断的概念

在中断系统中，还有以下几个相关概念：

1．开中断和关中断

中断的开放(称为开中断或中断允许)和中断的关闭(称为关中断或中断禁止)可通过指令设置相关特殊功能寄存器的内容来实现，这是CPU能否接受中断请求的关键。只在开中断的情况下，才能接受中断源的请求。

2．堆栈

进入子程序或中断处理程序后还要保护要用到的寄存器中的值，叫做保护现场；子程序返回或中断处理返回前，还要能够恢复这些寄存器中的值，叫做恢复现场。

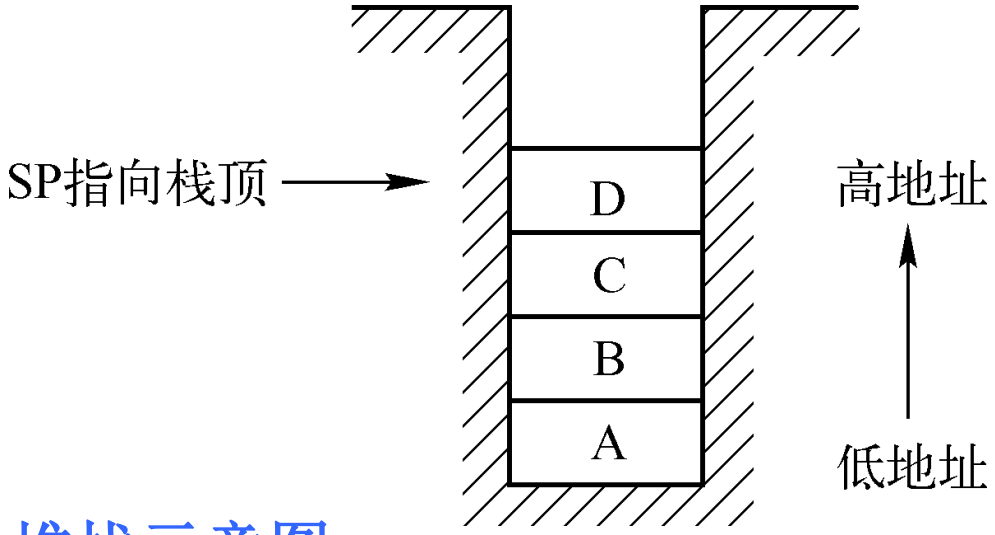
中断的概念

2．堆栈

保护现场和恢复现场都通过堆栈操作来实现。其中通用寄存器的保存和恢复需要由堆栈操作指令来完成;

返回地址保存与恢复(断点保护和恢复)的堆栈操作都在相应的子程序调用(LCALL)和返回指令(RET,RETI)的操作中自动完成的，无需用专门的堆栈操作指令。

```
PUSH  ACC
PUSH  PSW
PUSH  0H; R0压栈
  …
POP   0H; R0弹栈
POP   PSW
POP   ACC
```



堆栈示意图

中断的概念

3．中断的响应

单片机响应中断源请求时，由中断系统硬件控制CPU从主程序转去(0003H~00BBH)执行中断服务程序，同时把断点地址自动送入堆栈进行保护。

4．中断的撤除

在响应中断请求后，返回主程序之前，该中断请求标志应该撤除，否则，单片机执行完中断服务程序会误判为又发生了中断请求而错误地再次进入中断服务程序。

单片机中有些中断请求标志会自动撤除，有些不能自动撤除，必须靠用户使用相应的指令撤除。

单片机的中断系统及其管理

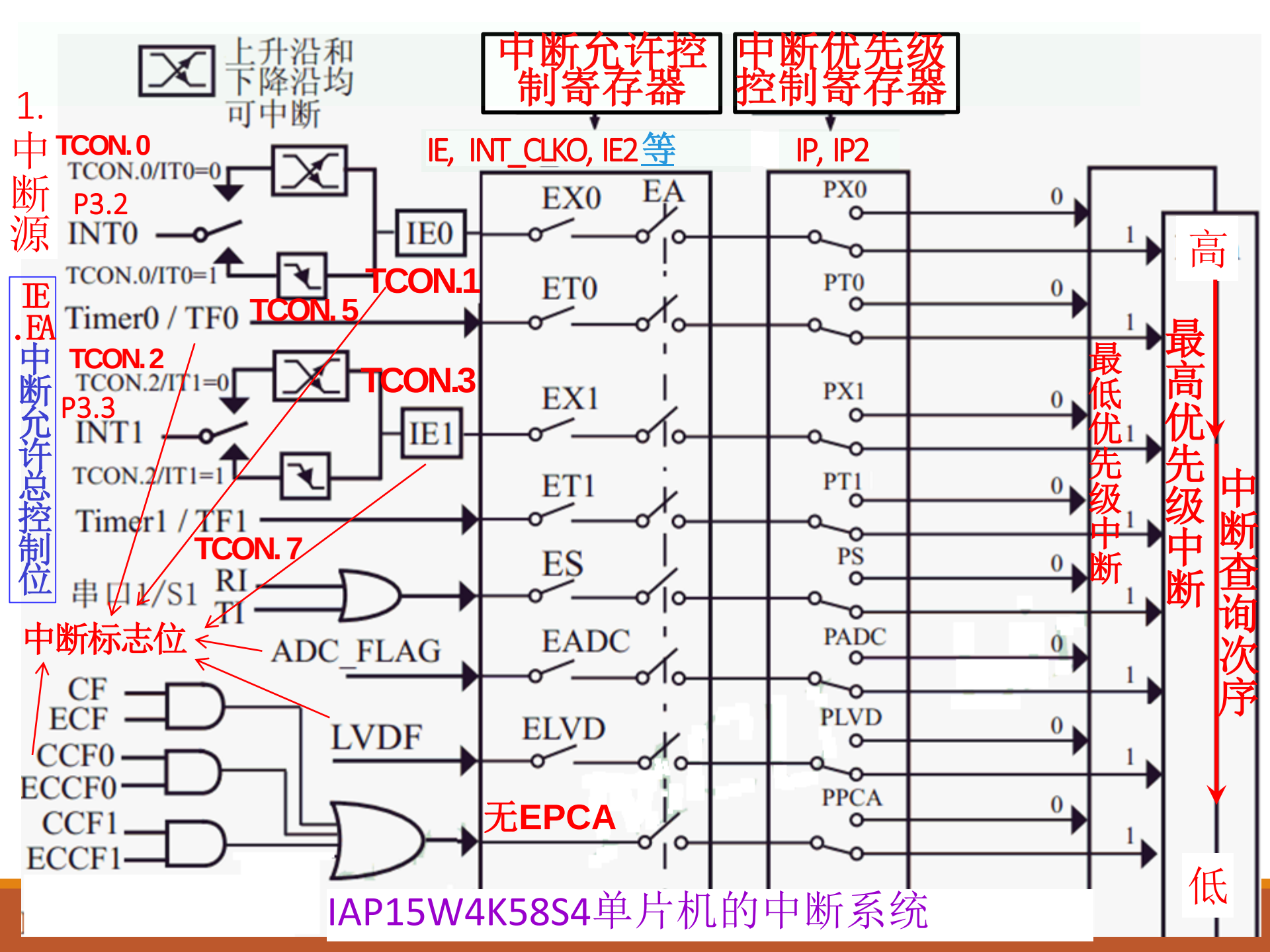
中断源及其优先级管理

1、中断源

中断源是指能发出中断请求，引起中断的装置或事件。

IAP15W4K58S4单片机提供21个中断请求源:

- 5个外部中断请求
- 5个片内定时/计数器溢出中断请求
- 4个片内异步串行口(UART)中断请求
- 1个ADC中断
- 1个低电压检测中断
- 1个PCA中断。
- 1个SPI中断
- 1个比较器中断
- 1个PWM中断以及1个PWM异常检测中断。


1.
中断源
上升沿和下降沿均可中断
中断允许控制寄存器
中断优先级控制寄存器
IE, INT_CLKO, IE2等
IP, IP2
TCON.0
TCON.0/IT0=0
P3.2
INT0
TCON.0/IT0=1
Timer0 / TF0
TCON.1
TCON.5
TCON.2
TCON.2/IT1=0
P3.3
INT1
TCON.2/IT1=1
Timer1 / TF1
TCON.3
TCON.7
串口1/S1
RI
TI
中断标志位
ADC_FLAG
CF
ECF
CCF0
ECCF0
CCF1
ECCF1
LVDF
无EPCA
IE0
IE1
EX0
EA
ET0
EX1
ET1
ES
EADC
ELVD
PX0
PT0
PX1
PT1
PS
PADC
PLVD
PPCA
0
1
高
最高优先级中断
最低优先级中断
中断查询次序
低
IE.EA中断允许总控制位
IAP15W4K58S4单片机的中断系统

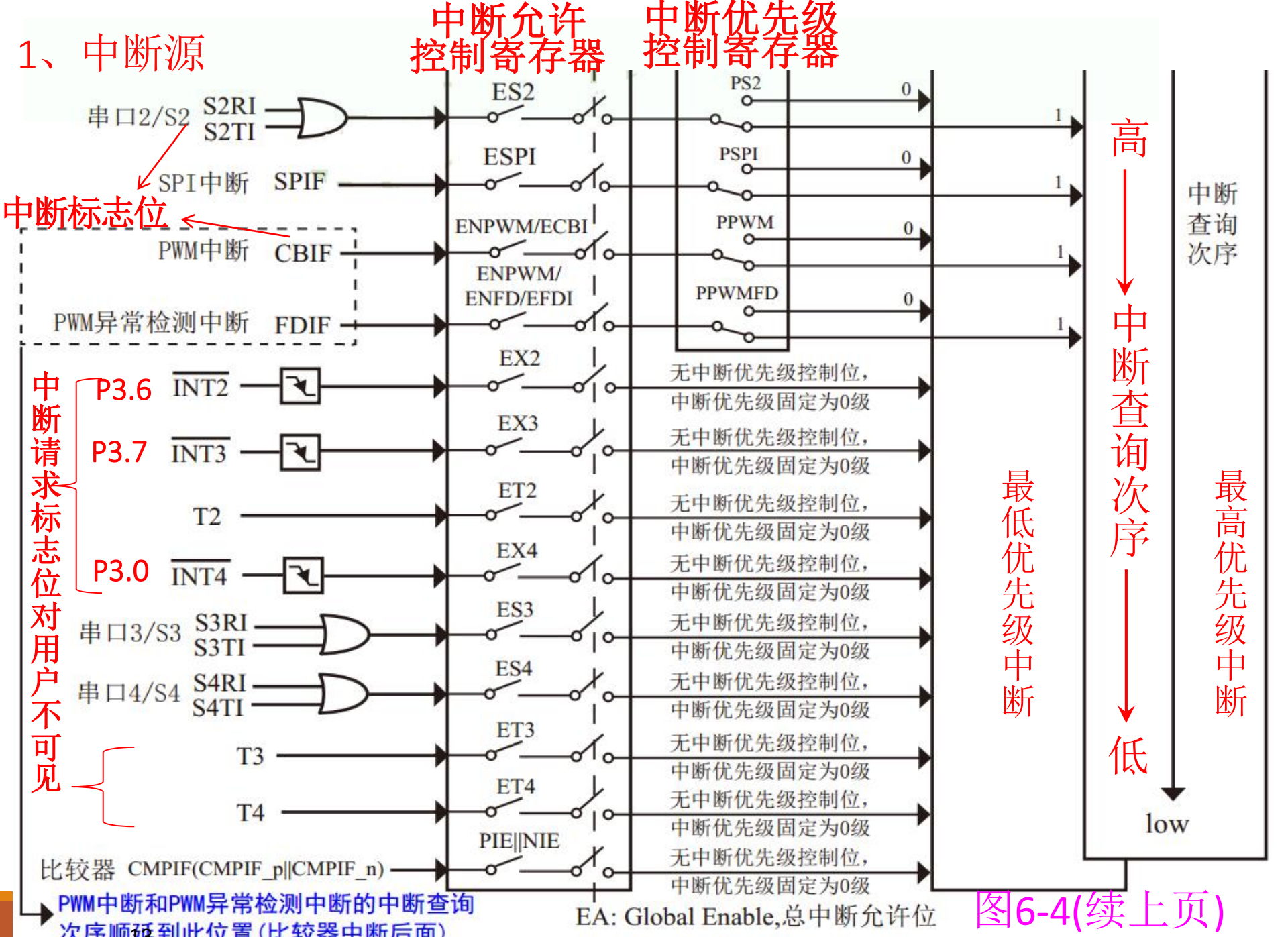

1、中断源
中断允许
控制寄存器
中断优先级
控制寄存器
串口2/S2
S2RI
S2TI
SPI中断
SPIF
中断标志位
PWM中断
CBIF
PWM异常检测中断
FDIF
中断请求标志位对用户不可见
P3.6
INT2
P3.7
INT3
T2
P3.0
INT4
串口3/S3
S3RI
S3TI
串口4/S4
S4RI
S4TI
T3
T4
比较器
CMPIF(CMPIF_p||CMPIF_n)
ES2
ESPI
ENPWM/ECBI
ENPWM/
ENFD/EFDI
EX2
EX3
ET2
EX4
ES3
ES4
ET3
ET4
PIE||NIE
PS2
PSPI
PPWM
PPWMFD
0
1
无中断优先级控制位，
中断优先级固定为0级
高
中断查询次序
低
low
中断
查询
次序
最低优先级中断
最高优先级中断
PWM中断和PWM异常检测中断的中断查询
次序顺延到此位置（比较器中断后面）
EA: Global Enable,总中断允许位

图6-4(续上页)

举例：电源控制寄存器PCON

电源控制寄存器PCON（地址为87H，复位值为30H）

寄存器的各位定义如下：

位号	D7	D6	D5	D4	D3	D2	D1	D0
位名称	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL

- 与中断有关的位是LVDF。
- LVDF是低电压检测标志位，同时也是低电压检测中断请求标志位。

PWM中断的标志——PWM中断标志寄存器PWMIF

位号	D7	D6	D5	D4	D3	D2	D1	D0
位名称	-	CBIF	C7IF	C6IF	C5IF	C4IF	C3IF	C2IF

PWM中断的标志保存在PWM中断标志寄存器PWMIF中，包括PWM计数器中断标志位CBIF和PWM2~PWM7通道的PWM中断标志位C2IF~C7IF。

CBIF为PWM计数器回0中断标志位，当PWM计数器回0时由硬件自动置1，申请中断；

CnIF（n = 2~7）为第n通道的PWM中断标志位，当PWM发生翻转时由硬件自动置1，申请中断。响应中断后，这些中断标志需要由软件清0。

比较器中断请求标志——比较控制寄存器CMPCR1

位号	D7	D6	D5	D4	D3	D2	D1	D0
位名称	CMPEN	CMPIF	PIE	NIE	PIS	NIS	CMPOE	CMPRES

比较器中断请求标志保存在比较控制寄存器CMPCR1中。比较器中断标志CMPIF=(CMPIF_p||CMPIF_n), 其中CMPIF_p是比较器上升沿中断标志, CMPIF_n是比较器下降沿中断标志;

当CPU读取CMPIF时会读到(CMPIF_p ||CMPIF_n); 当CPU对CMPIF写“0”后, CMPIF_p及CMPIF_n会被自动设置为“0”。

因此，当比较器的比较结果由LOW变成HIGH时, 那么CMPIF_p会被设置成1, 比较器中断标志位CMPIF也会被设置成1, 如果比较器上升沿中断被允许, 即PIE(CMPCR1.5)已被设置成1, 则向CPU请求中断, 单片机转去执行比较器上升中断程序;

2、中断的允许、禁止及优先级

Interrupt Enable

（1）中断的允许和禁止

- IAP15W4K58S4没有专门的开中断和关中断指令，中断的允许和禁止是通过设置IE，IE2，外部中断使能和时钟输出寄存器INT_CLKO等寄存器的相应位实现的。
- **单片机对中断源的允许和禁止由两级允许控制组成，即总控制(EA位)和对每个中断源的分别控制。总控制(EA位)用于决定整个中断系统是允许还是禁止，当整个中断系统禁止时，CPU不响应任何中断请求。**
- 与中断允许有关的特殊功能寄存器及其控制位见表

例如:中断允许寄存器IE(地址为A8H，复位值为00H)

位号	D7	D6	D5	D4	D3	D2	D1	D0
位名称	EA	ELVD	EADC	ES	ET1	EX1	ET0	EX0

（1）中断的允许和禁止

⑥ PWM控制寄存器PWMCR

ECBI为PWM计数器回0中断允许控制位。

⑦ PWM外部异常控制寄存器PWMFDCR

EFDI为PWM异常检测中断允许控制位。

⑧ 比较器控制寄存器1CMPCR1

PIE为比较器上升沿中断允许控制位。

NIE为比较器下降沿中断允许控制位。

（1）中断的允许和禁止

IAP15W4K58S4单片机复位后，各中断允许寄存器控制位均被清“0”，即禁止所有中断。如果需要允许某些中断，可在程序中将相应中断控制位置为1。

可通过对IE、IE2和INT_CLKO等寄存器的相应位进行置1或清0的操作来允许或禁止各中断源的中断申请。

禁止某个中断源申请中断，也称为屏蔽某个中断源。欲使某中断源允许中断，必须同时EA=1，即必须使整个中断系统允许中断。所以EA是中断允许的“总开关”。

- **例如，若要允许外部中断0中断，需将EX0和EA都置1。**
  若要允许A/D转换中断，则需要将EADC和EA都置1。

（2）中断的优先级

除了外部中断2~4、定时器T2~T4、串口3~4及比较器中断，不能设置为高优先级外;

其他中断源通过特殊功能寄存器(IP和IP2)中的相应位(表6-4)，可设为高、低二级优先级，实现二级中断嵌套，与传统8051单片机两级中断优先级完全兼容。

- **IAP15W4K58S4单片机对中断优先级的处理原则**
  - 低优先级中断可被高优先级中断所中断, 反之不能。
  - 任何一种中断(不管是高优先级还是低优先级)，一旦得到响应，不会再被它的同级中断所中断。

中断优先级有关的特殊功能寄存器及其控制位

寄存器	地址	D7	D6	D5	D4	D3	D2	D1	D0	复位值
IP	B8H	PPCA	PLVD	PADC	PS	PT1	PX1	PT0	PX0	0000 0000B
IP2	B5H					PPWMFD	PPWM	PSPI	PS2	XXXX 0000B

STC15F2K60S2无

◇ **IP, IP2一起确定IAP15W4K58S4单片机的二个中断优先级 。**

1）中断优先级寄存器IP：各位的定义如下：

每一位对应其相应的中断源的优先级控制位。

1：高优先级；

0：低优先级。

（2）中断的优先级

IAP15W4K58S4单片机的中断源及其相关控制

中断源	中断入口地址	中断号	优先级控制位	低优先级	高优先级	中断请求标志位	中断允许控制位
外部中断INT0	0003H	0	PX0	0	1	IE0	EX0/EA
T0溢出中断	000BH	1	PT0	0	1	TF0	ET0/EA
外部中断INT1	0013H	2	PX1	0	1	IE1	EX1/EA
T1溢出中断	001BH	3	PT1	0	1	TF1	ET1/EA
串口UART1	0023H	4	PS	0	1	RI + TI	ES/EA
ADC中断	002BH	5	PADC	0	1	ADC_FLAG	EADC/EA
LVD中断	0033H	6	PLVD	0	1	LVDF	ELVD/EA
PCA中断	003BH	7	PPCA	0	1	CF+CCF0+CCF1	(ECF+ECCF0+ECCF1+ELVD)/EA
串口UART2	0043H	8	PS2	0	1	S2RI+S2TI	ES2/EA
SPI中断	004BH	9	PSPI	0	1	SPIF	ESPI/EA
外部中断2	0053H	10		0			EX2/EA
外部中断3	005BH	11		0			EX3/EA
T2溢出中断	0063H	12		0			ET2/EA
外部中断 4	0083H	16		0			EX4/EA
串口UART3	008BH	17		0		S3RI+S3TI	ES3/EA
串口UART4	0093H	18		0		S4RI+S4TI	ES4/EA
T3溢出中断	009BH	19		0			ET3/EA
T4溢出中断	00A3H	20		0			ET4/EA
比较器中断	00ABH	21		0		CMPIF	(PIE+NIE)/EA
PWM中断	00B3H	22	PPWM	0	1	CBIF	(EPWM+ECBI)/EA
PWM异常中断	00BBH	23	PPWMFD	0	1	FDIF	(EPWM+EFD+EFDI)/EA

单片机中断处理过程

1、单片机响应中断的条件和过程

当中断源向CPU发出中断请求时，如果中断条件满足，CPU将进入中断响应周期。单片机响应中断的条件是：

- ①中断源有请求。(相应中断标志位为1)
- ②相应的中断允许位设置为1。
- ③CPU中断允许总控制位开放 (EA=1);
- ④无同级或高级中断正在处理。(二级中断嵌套)

- **在每个指令周期的最后一个时钟周期，CPU对各中断源采样，并设置相应中断标志位。**
- **CPU在下一指令周期的最后一个时钟周期按优先级顺序查询各中断标志，若查到某中断标志为1，将在下一指令周期按优先级的高低顺序响应中断并进行处理。**

1、单片机响应中断的条件和过程

CPU响应中断时，将执行如下操作：

- ①当前正被执行的指令执行完毕；
- ②(下一条指令)PC值被压入堆栈；（断点保护(自动)）
- ③现场保护（压栈保护寄存器）；
- ④阻止同级别其他中断；
- ⑤将中断服务程序(对应的)入口地址(中断向量地址,对应8个单元,一般存储指令LJMP)装载到程序计数器PC；
- ⑥执行相应的中断服务程序ISR。
- ⑦中断服务程序ISR中,汇编以RETI(中断返回)指令结束,C语言以最外层}结束。将PC值从堆栈中取回,之后从程序的断点处继续执行。

1、单片机响应中断的条件和过程

每个中断服务程序入口地址之间只相隔8个单元，一般中断服务程序的长度都超过8个字节，这时可将中断服务程序存放到存储器的其他区域，然后在中断入口处安排一条转移指令LJMP，转向中断服务程序。

```
例如：    ORG    0003H      ;外部中断0入口地址
          LJMP   X0_ISR
                 …                    ;其他程序代码
X0_ISR:                               ;外部中断0服务程序
                 …
                 RETI
```

1、单片机响应中断的条件和过程

- 使用C语言编写单片机中断应用程序时,用中断号区分每一个中断。例如,

```
void X0_ISR(void) interrupt 0{ } //外部中断0中断函数
void T0_ISR (void) interrupt 1{ } //定时器T0 中断函数
void X1_ISR(void) interrupt 2{ } //外部中断1中断函数
void T1_ISR (void) interrupt 3{ } //定时器T1中断函数
void UART1_ISR (void) interrupt 4{} //串行口1中断函数
void ADC_ISR (void) interrupt 5{ } //ADC中断函数
void LVD_ISR (void) interrupt 6{ } //低电压检测LVD中断函数
```

C语言编写中断应用程序时，用中断号区分中断

```
void UART3_ISR (void) interrupt 17{}  //串口3中断函数
void UART4_ISR (void) interrupt 18{}  //串口4中断函数
void T3_ISR (void) interrupt 19{}     //定时器T3中断函数
void T4_ISR (void) interrupt 20{}     //定时器T4中断函数
void CMP_ISR (void) interrupt 21{}    //比较器中断函数
void PWM_ISR (void) interrupt 22{}    //PWM中断函数
void PWMFD_ISR (void) interrupt 23{} //PWM异常检测中断函数
```

1、单片机响应中断的条件和过程

(相应中断标志位为1)

在程序的运行过程中，并不是任何时刻都可以响应中断请求。出现下列情况时，CPU不会响应中断请求：

- 中断允许总控制位EA＝0或发出中断请求的中断源所对应的中断允许控制位为0。
- CPU正在执行一个同级或高级的中断服务程序。
- 当前执行指令时刻不是指令周期的最后一个时钟周期。
- 正在执行的指令是中断返回指令RETI或者是访问IE或IP的指令时，CPU至少要再执行一条指令才能响应中断请求。

2、中断服务

中断服务程序从入口地址开始执行，直到执行返回指令RETI为止。RETI指令表示中断服务程序的结束。

中断服务程序由四个部分组成，即保护现场、中断服务、恢复现场以及中断返回。

由于在主程序中一般都会用到累加器A和程序状态字寄存器PSW，所以在现场保护时一般都需要保护A和PSW，其他寄存器根据使用情况决定是否需要保护。

在C语言程序中不需要(用户)进行现场保护。

2、中断服务

在编写中断服务程序时应注意以下3点：

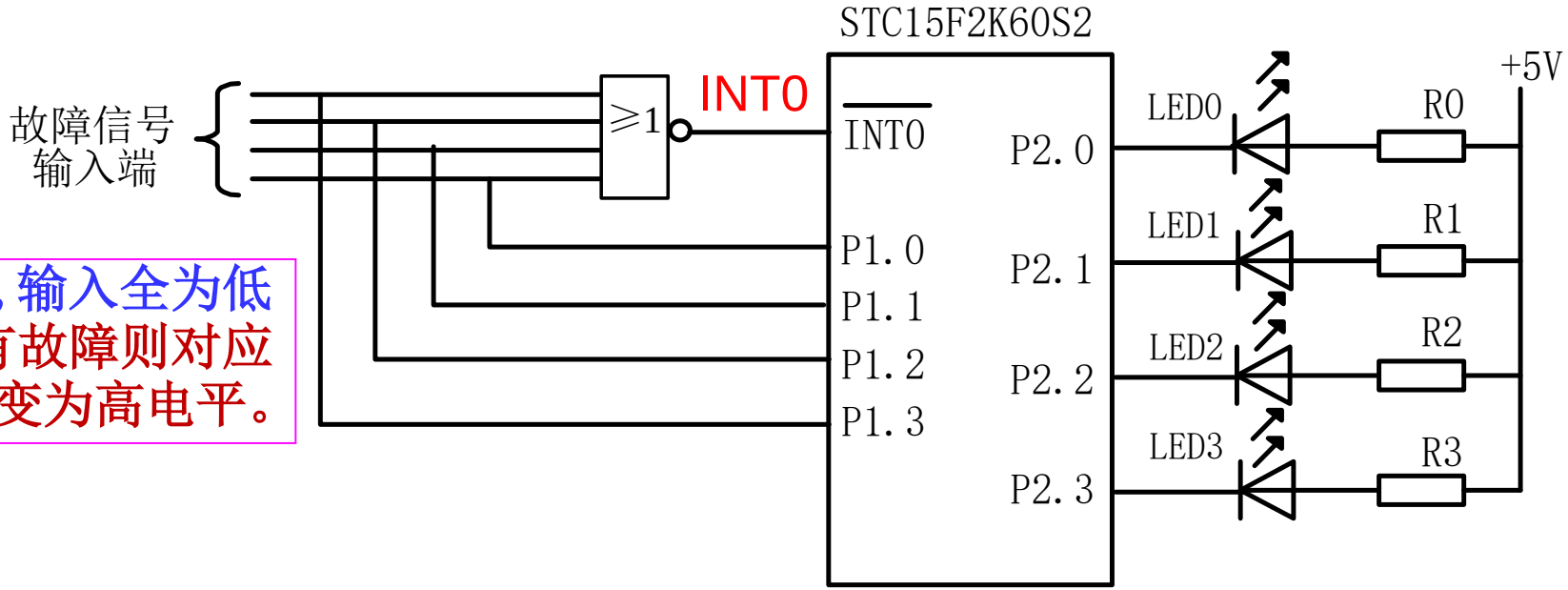
- ①单片机响应中断后，不会自动关闭中断系统(EA=1)，还会继续响应其他中断(更高优先级)。

➢ **若用户程序不希望出现中断嵌套，则必须在中断服务程序的开始处关闭中断(EA=0)，禁止更高优先级的中断请求中断当前的服务程序。**

- ②为了保证保护现场和恢复现场能够连续进行，可在保护现场和恢复现场之前先关中断，当现场保护或现场恢复结束后，再根据实际需要决定是否需要开中断(EA=1)。

【例】外部中断源扩展

- 当外部中断源多于两个时，可用硬件申请与软件查询结合的方法，把多个中断源由硬件“线或”或经或非门引入外中断源输入端(例INT0)，同时又连到某I/O口。
- 这样，每个源都可能引起中断，在中断服务程序中通过软件查询便可确定哪一个是正在申请的中断源，其查询的次序则由中断源优先级决定，这就可实现多个外部中断源的扩展。



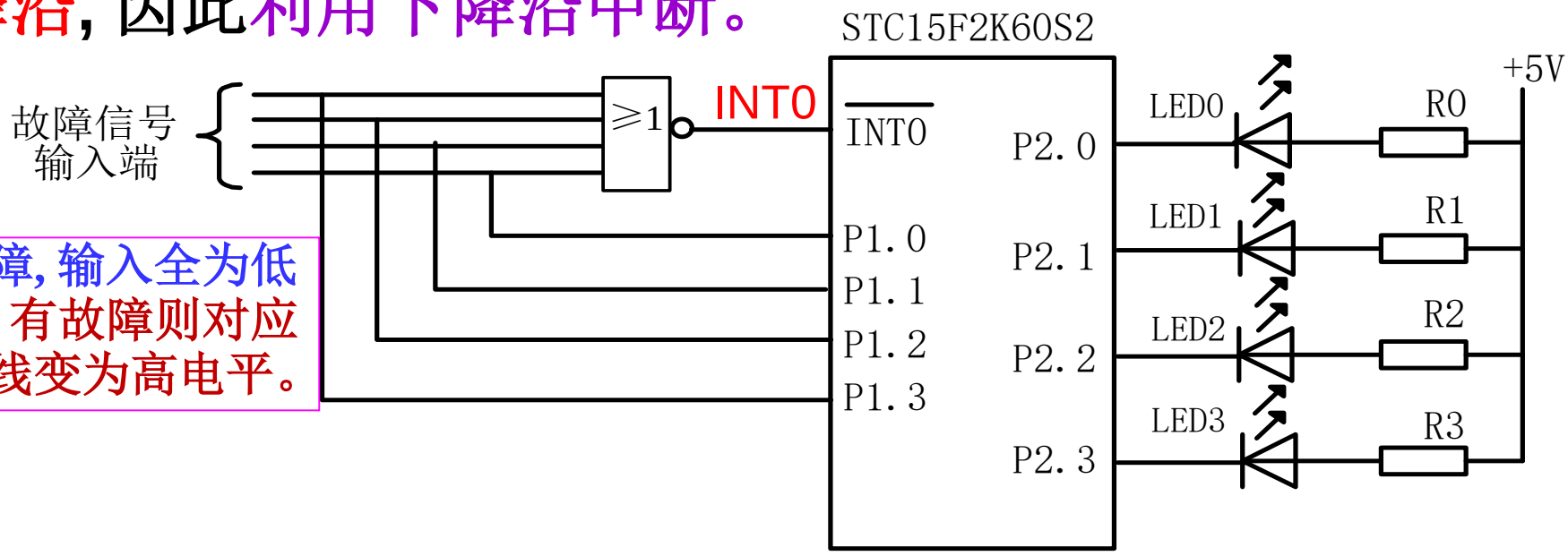
无故障，输入全为低电平，有故障则对应输入线变为高电平。

【例】外部中断源扩展。

如图所示中断线路可实现系统的故障显示，当系统工作正常时，4个故障源输入全为低电平，指示灯全熄灭。

若当某部分出现故障，则对应的输入线由低电平变为高电平，从而引起单片机中断，试设计判定故障源的程序，并进行相应的灯光显示。

故障源是上升沿，INT0由高电平变为低电平，即出现下降沿，因此利用下降沿中断。



无故障，输入全为低电平，有故障则对应输入线变为高电平。

中断使用过程中需要注意的问题

- 在嵌入式系统中，中断是一种很有效的事件处理方式。但是，若使用中断不当，往往会出现一些意想不到的结果。
- 为了获得正确的结果可能要花费大量的调试时间，而且中断服务子程序的错误是比较难于被发现和纠正的。
- 为了避免发生类似的问题，下面介绍中断使用过程中需要注意的问题。

2、使用中断常出现的问题

中断程序尽量短小

- 中断处理子程序应尽量短小，这样其执行速度更快。
- 例如，接收串行中断的处理程序应该从SBUF中读一个字节，并且将其复制到用户定义的临时缓冲区中，然后退出中断程序；
- 而缓冲区中数据的进一步处理应由主程序来处理。
- 中断的时间消耗越少，那么在中断发生时就可以更快的响应和处理其他的中断。

2、使用中断常出现的问题

注意中断标志的清除问题

- 某些中断的中断标志不是在响应相应中断时由硬件自动清除的，
- 用户需要在中断处理程序返回前，使用指令将中断标志位清“0”，
- 否则，中断返回后，还将产生一次新的中断。
- 例如串行通信中断、ADC中断、SPI中断、低电压检测中断以及PCA中断。