本章分为三节，主要介绍：

# 5.1 80C51的中断系统

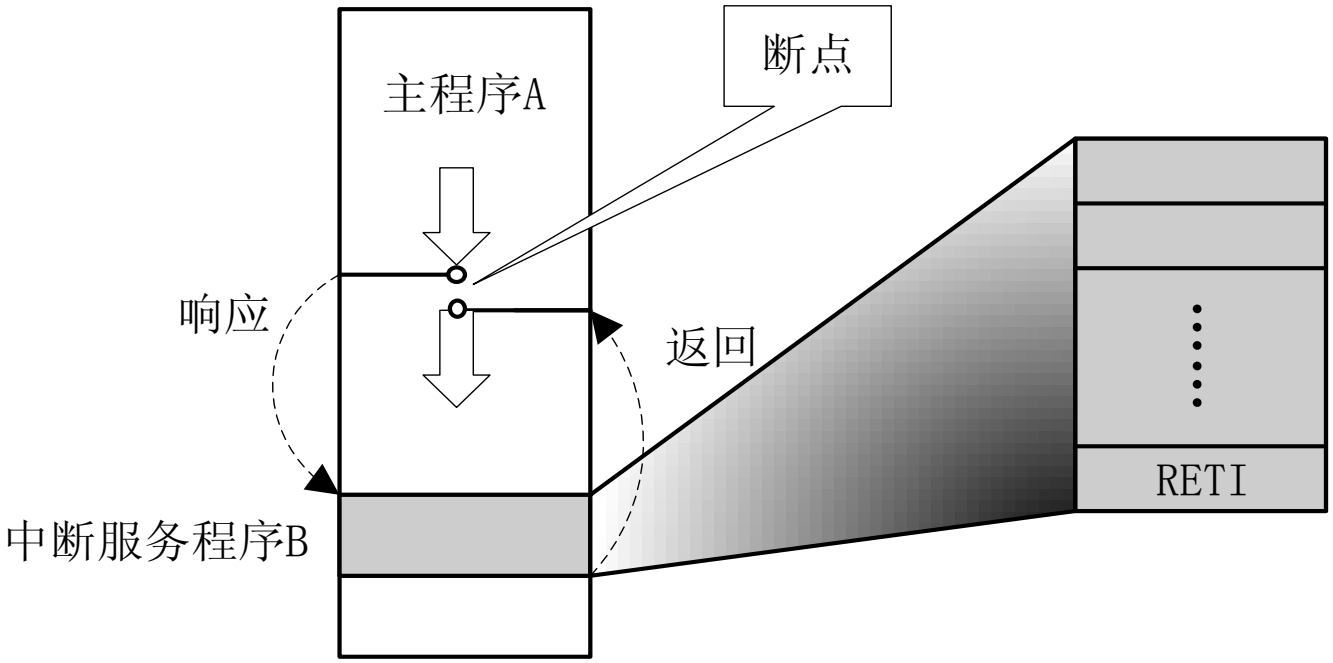
## 5.1.1 80C51的中断系统结构

### 一、中断的概念

CPU在处理某一事件A时，发生了另一事件B请求CPU迅速去处理（中断发生）；

CPU暂时中断当前的工作，转去处理事件B（中断响应和中断服务）；

待CPU将事件B处理完毕后，再回到原来事件A被中断的地方继续处理事件A（中断返回），这一过程称为中断 。




引起CPU中断的根源，称为中断源。中断源向CPU提出的中断请求。CPU暂时中断原来的事务A，转去处理事件B。对事件B处理完毕后，再回到原来被中断的地方（即断点），称为中断返回。实现上述中断功能的部件称为中断系统（中断机构）。

随着计算机技术的应用，人们发现中断技术不仅解决了快速主机与慢速I/O设备的数据传送问题，而且还具有如下优点：

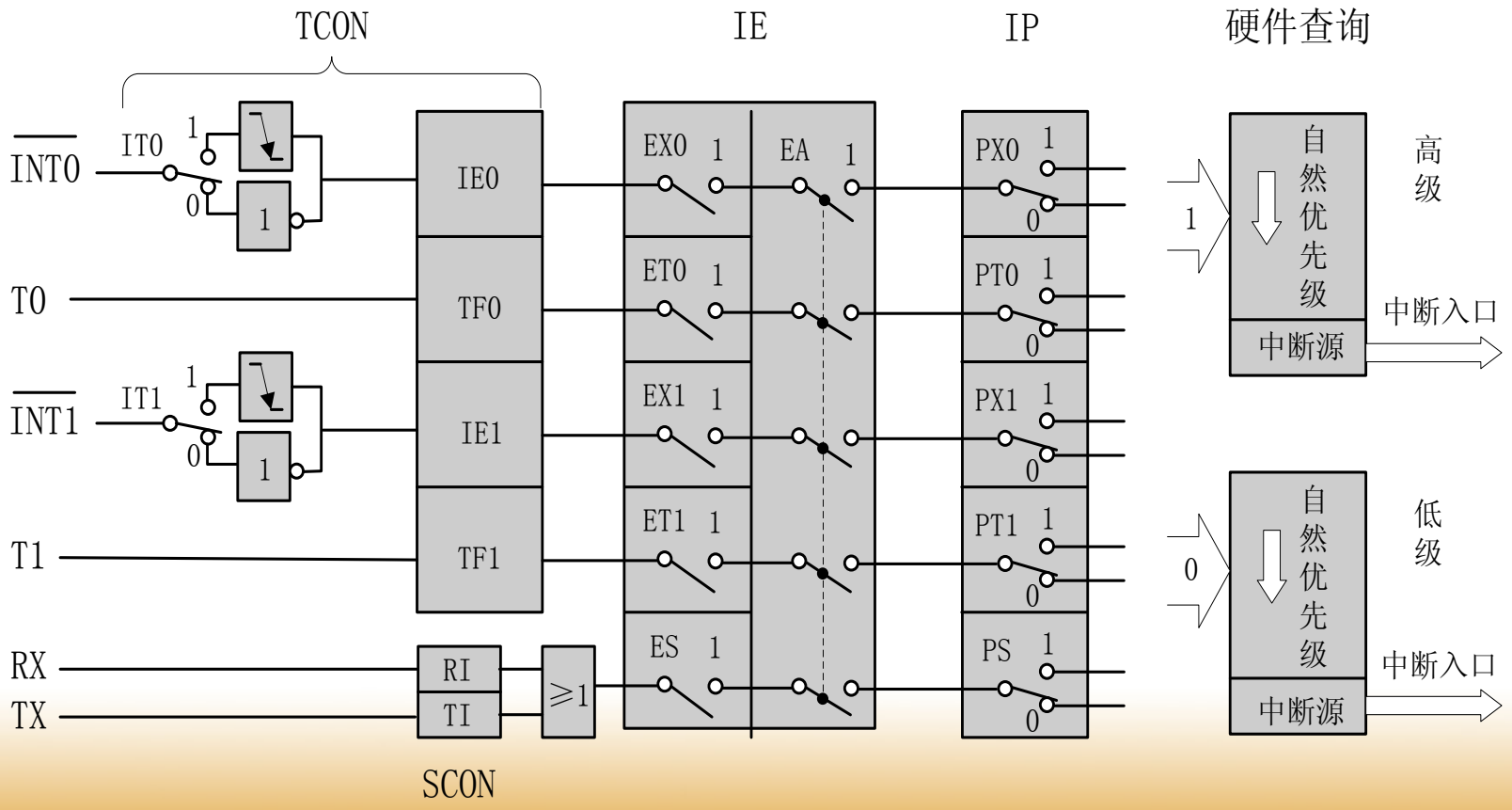
- 分时操作。CPU可以分时为多个I/O设备服务，提高了计算机的利用率；

- 实时响应。CPU能够及时处理应用系统的随机事件，系统的实时性大大增强；

- 可靠性高。CPU具有处理设备故障及掉电等突发性事件能力，从而使系统可靠性提高。

# 二、80C51中断系统的结构

**80C51的中断系统有5个中断源，2个优先级，可实现二级中断嵌套 。**



# 5.1.2 80C51的中断源

## 一、中断源

1、$\overline{\text{INT0}}$（P3.2）。可由IT0(TCON.0)选择其为低电平有效还是下降沿有效。当CPU检测到P3.2引脚上出现有效的中断信号时，中断标志IE0(TCON.1)置1，向CPU申请中断。

2、$\overline{\text{INT1}}$（P3.3）。可由IT1(TCON.2)选择其为低电平有效还是下降沿有效。当CPU检测到P3.3引脚上出现有效的中断信号时，中断标志IE1(TCON.3)置1,向CPU申请中断。

3、TF0（TCON.5），片内定时/计数器T0溢出中断请求标志。当定时/计数器T0发生溢出时，置位TF0，并向CPU申请中断。

4、TF1（TCON.7），片内定时/计数器T1溢出中断请求标志。当定时/计数器T1发生溢出时，置位TF1，并向CPU申请中断。

5、RI（SCON.0）或TI（SCON.1），串行口中断请求标志。当串行口接收完一帧串行数据时置位RI或当串行口发送完一帧串行数据时置位TI，向CPU申请中断。

# 二、中断请求标志

## 1、TCON的中断标志

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：88H | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 | TCON |

**IT0（TCON.0），外部中断0触发方式控制位。**

**当IT0=0时，为电平触发方式。**

**当IT0=1时，为边沿触发方式（下降沿有效）。**

**IE0（TCON.1），外部中断0中断请求标志位。**

**IT1（TCON.2），外部中断1触发方式控制位。**

**IE1（TCON.3），外部中断1中断请求标志位。**

**TF0（TCON.5），定时/计数器T0溢出中断请求标志位。**

**TF1（TCON.7），定时/计数器T1溢出中断请求标志位。**

# 2、SCON的中断标志

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：98H | | | | | | | TI | RI | SCON |

- **RI（SCON.0），串行口接收中断标志位。当允许串行口接收数据时，每接收完一个串行帧，由硬件置位RI。同样，RI必须由软件清除。**
- **TI（SCON.1），串行口发送中断标志位。当CPU将一个发送数据写入串行口发送缓冲器时，就启动了发送过程。每发送完一个串行帧，由硬件置位TI。CPU响应中断时，不能自动清除TI，TI必须由软件清除。**

# 5.1.3　80C51中断的控制

## 一、中断允许控制

CPU对中断系统所有中断以及某个中断源的开放和屏蔽是由中断允许寄存器IE控制的。

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：A8H | EA | | | ES | ET1 | EX1 | ET0 | EX0 | IE |

- EX0(IE.0)，外部中断0允许位；
- ET0(IE.1)，定时/计数器T0中断允许位；
- EX1(IE.2)，外部中断0允许位；
- ET1(IE.3)，定时/计数器T1中断允许位；
- ES（IE.4)，串行口中断允许位；
- EA (IE.7)， CPU中断允许（总允许）位。

# 二、中断优先级控制

**80C51单片机有两个中断优先级，即可实现二级中断服务嵌套。每个中断源的中断优先级都是由中断优先级寄存器IP中的相应位的状态来规定的 。**

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：B8H | | | | PS | PT1 | PX1 | PT0 | PX0 | IP |

- **PX0（IP.0），外部中断0优先级设定位；**
- **PT0（IP.1），定时/计数器T0优先级设定位；**
- **PX1（IP.2），外部中断0优先级设定位；**
- **PT1（IP.3），定时/计数器T1优先级设定位；**
- **PS （IP.4），串行口优先级设定位。**

同一优先级中的中断申请不止一个时，则有中断优先权排队问题。同一优先级的中断优先权排队，由中断系统硬件确定的自然优先级形成，其排列如所示：

各中断源响应优先级及中断服务程序入口表

| 中断源 | 中断标志 | 中断服务程序入口 | 优先级顺序 |
|---|---|---|---|
| 外部中断 0（$\overline{\text{INT0}}$） | IE0 | 0003H | 高 |
| 定时/计数器 0（T0） | TF0 | 000BH | ↓ |
| 外部中断 1（$\overline{\text{INT1}}$） | IE1 | 0013H | ↓ |
| 定时/计数器 1（T1） | TF1 | 001BH | ↓ |
| 串行口 | RI 或 TI | 0023H | 低 |

# 80C51单片机的中断优先级有三条原则：

- CPU同时接收到几个中断时，首先响应优先级别最高的中断请求。
- 正在进行的中断过程不能被新的同级或低优先级的中断请求所中断。
- 正在进行的低优先级中断服务，能被高优先级中断请求所中断。

为了实现上述后两条原则，中断系统内部设有两个用户不能寻址的优先级状态触发器。其中一个置1，表示正在响应高优先级的中断，它将阻断后来所有的中断请求；另一个置1，表示正在响应低优先级中断，它将阻断后来所有的低优先级中断请求。

# 5.2 80C51单片机中断处理过程

## 5.2.1 中断响应条件和时间

### 一、中断响应条件

- 中断源有中断请求；
- 此中断源的中断允许位为1；
- CPU开中断（即EA=1）。

同时满足时，CPU才有可能响应中断。

中断服务的进入：

CPU执行程序过程中，在每个机器周期的S5P2期间，中断系统对各个中断源进行采样。这些采样值在下一个机器周期内按优先级和内部顺序被依次查询。

如果某个中断标志在上一个机器周期的S5P2时被置成了1，那么它将于现在的查询周期中及时被发现。接着CPU便执行一条由中断系统提供的硬件LCALL指令，转向被称作中断向量的特定地址单元，进入相应的中断服务程序。

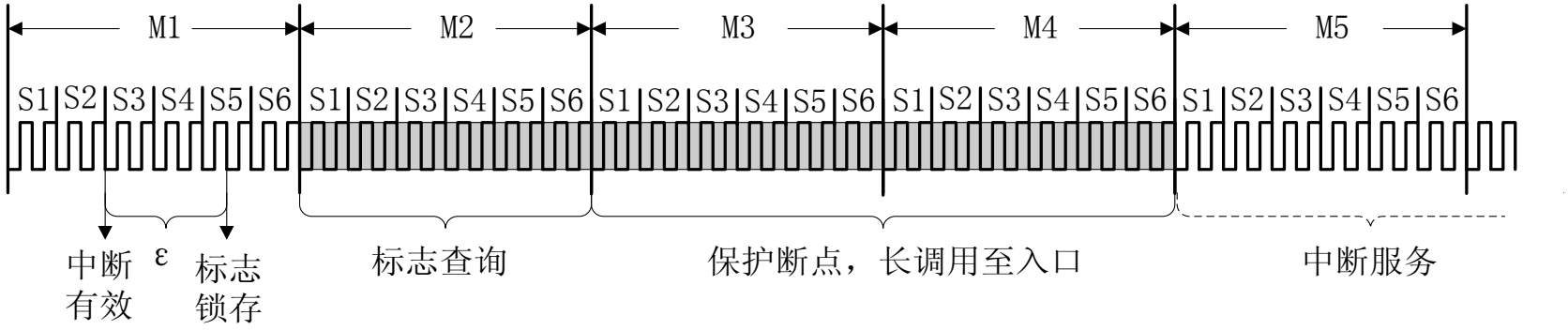
遇以下任一条件，硬件将受阻，不产生LCALL指令：

- CPU正在处理同级或高优先级中断；
- 当前查询的机器周期不是所执行指令的最后一个机器周期。即在完成所执行指令前，不会响应中断，从而保证指令在执行过程中不被打断；
- 正在执行的指令为RET、RETI或任何访问IE或IP寄存器的指令。即只有在这些指令后面至少再执行一条指令时才能接受中断请求。

若由于上述条件的阻碍中断未能得到响应，当条件消失时该中断标志却已不再有效，那么该中断将不被响应。就是说，中断标志曾经有效，但未获响应，查询过程在下个机器周期将重新进行。

# 二、中断响应时间

某中断的响应时序如图：




- 若M1周期的S5P2前某中断生效，在S5P2期间其中断请求被锁存到相应的标志位中去；M2恰逢指令的最后一个机器周期，且该指令不是RETI或访问IE、IP的指令。于是，M3和M4便可以执行硬件LCALL指令，M5周期将进入了中断服务程序。
- 80C51的中断响应时间（从标志置1到进入相应的中断服务），至少要3个完整的机器周期。

## 5.2.2 中断响应过程

- 将相应的优先级状态触发器置1（以阻断后来的同级或低级的中断请求）。
- 执行一条硬件LCALL指令，即把程序计数器PC的内容压入堆栈保存，再将相应的中断服务程序的入口地址送入PC。
- 执行中断服务程序。

中断响应过程的前两步是由中断系统内部自动完成的，而中断服务程序则要由用户编写程序来完成。

## 5.2.3 中断返回

**RETI指令的具体功能是：**

- 将中断响应时压入堆栈保存的断点地址从栈顶弹出送回PC，CPU从原来中断的地方继续执行程序；
- 将相应中断优先级状态触发器清0，通知中断系统，中断服务程序已执行完毕。

注意，不能用RET指令代替RETI指令。在中断服务程序中PUSH指令与POP指令必须成对使用，否则不能正确返回断点 。

若外部中断定义为电平触发方式，中断标志位的状态随CPU在每个机器周期采样到的外部中断输入引脚的电平变化而变化，这样能提高CPU对外部中断请求的响应速度。但外部中断源若有请求，必须把有效的低电平保持到请求获得响应时为止，不然就会漏掉；而在中断服务程序结束之前，中断源又必须撤消其有效的低电平，否则中断返回之后将再次产生中断。

电平触发方式适合于外部中断输入以低电平输入且中断服务程序能清除外部中断请求源的情况。例如，并行接口芯片8255的中断请求线在接受读或写操作后即被复位，因此，以其去请求电平触发方式的中断比较方便。

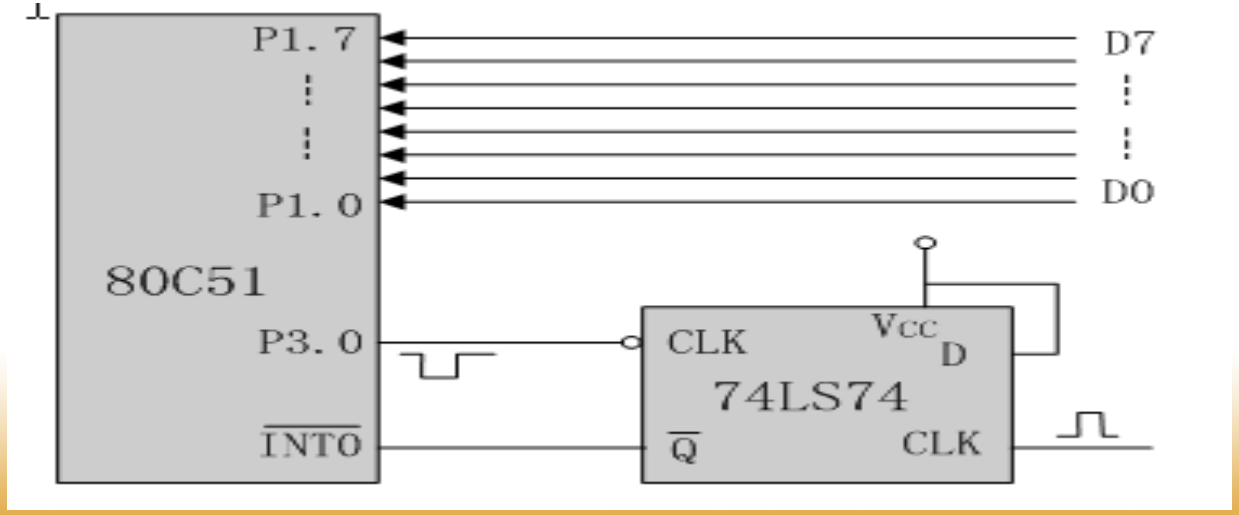
若外部中断定义为边沿触发方式，在相继连续的两次采样中，一个周期采样到外部中断输入为高电平，下一个周期采样到为低电平，则在IE0或IE1中将锁存一个逻辑1。即便是CPU暂时不能响应，中断申请标志也不会丢失，直到CPU响应此中断时才清零。这样，为保证下降沿能被可靠地采样到，外中断引脚上的高低电平（负脉冲的宽度）均至少要保持一个机器周期（若晶振为12MHz时，为1微秒）。

边沿触发方式适合于以负脉冲形式输入的外部中断请求，如ADC0809的转换结束标志信号EOC为正脉冲，经反相后就可以作为80C51的中断输入。

# 5.2.4 中断程序举例

例　　单外部中断源示例。

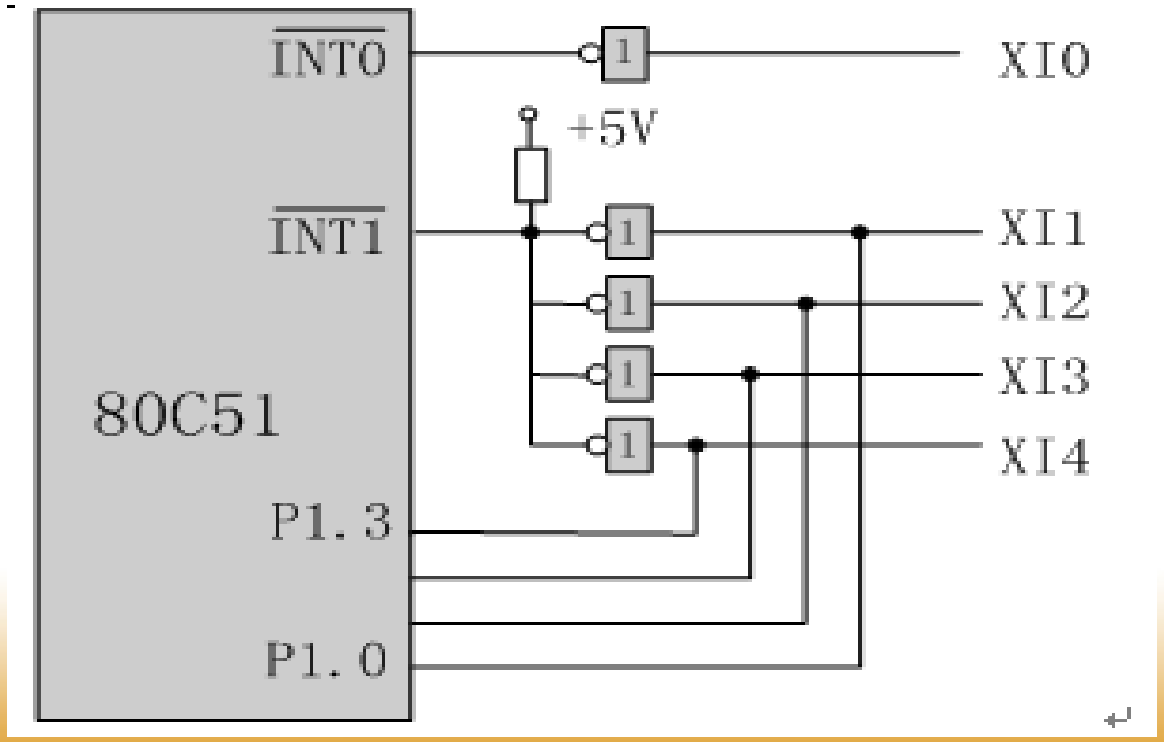
图为采用单外部中断源的数据采集系统示意图。将 P1 口设置成数据输入口，外围设备每准备好一个数据时，发出一个选通信号（正脉冲），使 D 触发器 Q 端置 1，经 $\overline{Q}$ 端向 $\overline{INT0}$ 送入一个低电平中断请求信号。如前所述，采用电平触发方式时，外部中断请求标志 IE0（或 IE1）在 CPU 响应中断时不能由硬件自动清除。因此，在响应中断后，要设法撤除 $\overline{INT0}$ 的低电平。撤除 $\overline{INT0}$ 的方法是，将 P3.0 线与 D 触发器复位端相连，只要在中断服务程序中，由 P3.0 输出一个负脉冲，就能使 D 触发器复位，$\overline{INT0}$ 无效，从而清除 IE0 标志。



```
        ORG 0000H
START:  LJMP MAIN          ；跳转到主程序
        ORG 0003H
        LJMP INTO          ；转向中断服务程序
        ORG 0030H          ；主程序
MAIN:   CLR IT0            ；设为电平触发方式
        SETB EA            ；CPU开放中断
        SETB EX0           ；允许中断
        MOV  DPTR，#1000H  ；设置数据区地址指针
        … …
        ORG 0200H          ；中断服务程序
INT0:   PUSH PSW           ；保护现场
        PUSH ACC
        CLR P3.0           ；由P3.0输出0
        NOP
        NOP
        SETB P3.0          ；由P3.0输出1，撤除
        MOV A，P1          ；输入数据
        MOVX  @DPTR，A     ；存入数据存储器
        INC   DPTR         ；修改数据指针，指向下一个单元
        … …
        POP ACC            ；恢复现场
        POP PSW
        RETI               ；中断返回
```

例　多外部中断源的系统示例。

设有5个外部中断源，中断优先级排队顺序为：XI0、XI1、XI2、XI3、XI4。试设计它们与80C51单片机的接口。



```
        ORG  0003H
         LJMP INSE0      ；转外部中断0服务程序入口
        ORG  0013H
         LJMP INSE1      ；转外部中断1服务程序入口
        … …
        … …
INSE0：  PUSH PSW        ；XI0中断服务程序
        PUSH ACC
         … …
         … …
         POP  ACC
         POP  PSW
         RETI
```

```
INSE1：PUSH PSW        ；中断服务程序
       PUSH ACC
       JB P1.0，DV1    ；P1.0为1，转XI1中断服务程序
       JB P1.1，DV2    ；P1.1为1，转XI2中断服务程序
       JB P1.2，DV3    ；P1.2为1，转XI3中断服务程序
       JB P1.3，DV4    ；P1.3为1，转XI4中断服务程序
INRET：POP ACC
       POP PSW
       RETI
  DV1：… …            ；XI1中断服务程序
       AJMP INRET
  DV2：… …            ；XI2中断服务程序
       AJMP INRET
  DV3：… …            ；XI3中断服务程序
       AJMP INRET
  DV4：… …            ；XI4中断服务程序
       AJMP INRET
```

# 5.3 80C51的定时/计数器

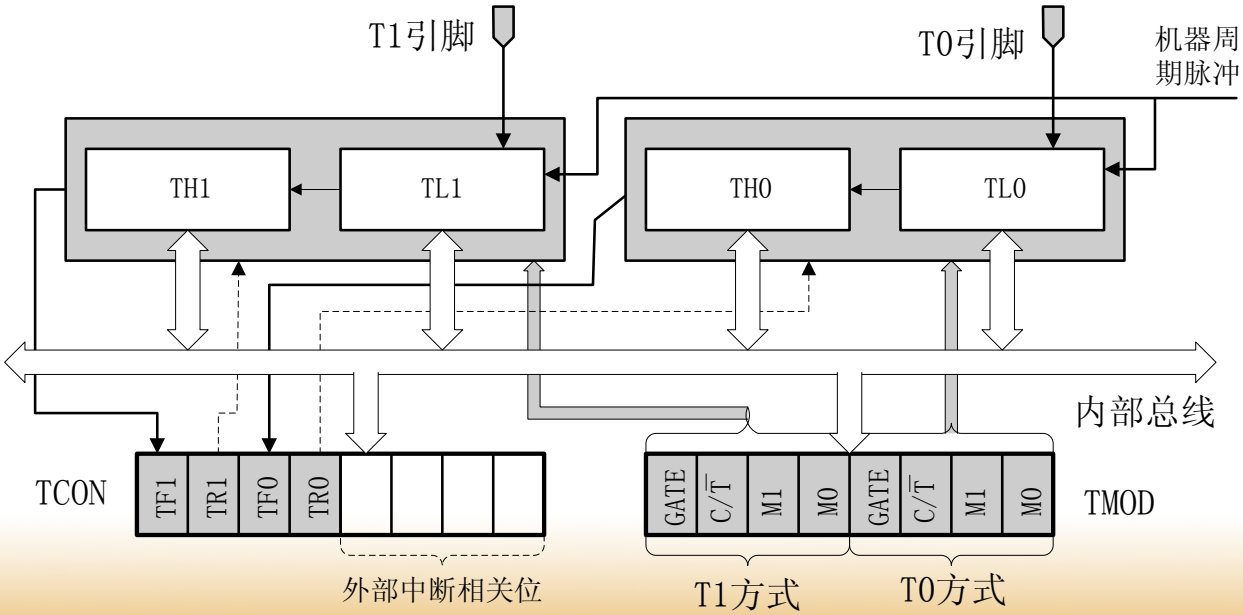
实现定时功能，比较方便的办法是利用单片机内部的定时/计数器。也可以采用下面三种方法：

- 软件定时：软件定时不占用硬件资源，但占用了CPU时间，降低了CPU的利用率。
- 采用时基电路定时：例如采用555电路，外接必要的元器件（电阻和电容），即可构成硬件定时电路。但在硬件连接好以后，定时值与定时范围不能由软件进行控制和修改，即不可编程。
- 采用可编程芯片定时：这种定时芯片的定时值及定时范围很容易用软件来确定和修改，此种芯片定时功能强，使用灵活。在单片机的定时/计数器不够用时，可以考虑进行扩展。

# 5.3.1 定时/计数器的结构和工作原理

## 一、定时/计数器的结构

定时/计数器的实质是加1计数器（16位），由高8位和低8位两个寄存器组成。TMOD是定时/计数器的工作方式寄存器，确定工作方式和功能；TCON是控制寄存器，控制T0、T1的启动和停止及设置溢出标志。



# 二、定时/计数器的工作原理

加1计数器输入的计数脉冲有两个来源,一个是由系统的时钟振荡器输出脉冲经12分频后送来；一个是T0或T1引脚输入的外部脉冲源。每来一个脉冲计数器加1，当加到计数器为全1时，再输入一个脉冲就使计数器回零，且计数器的溢出使TCON中TF0或TF1置1，向CPU发出中断请求（定时/计数器中断允许时）。如果定时/计数器工作于定时模式，则表示定时时间已到；如果工作于计数模式，则表示计数值已满。

可见，由溢出时计数器的值减去计数初值才是加1计数器的计数值。

•设置为定时器模式时，加1计数器是对内部机器周期计数（1个机器周期等于12个振荡周期，即计数频率为晶振频率的1/12）。计数值N乘以机器周期Tcy就是定时时间t 。

•设置为计数器模式时，外部事件计数脉冲由T0或T1引脚输入到计数器。在每个机器周期的S5P2期间采样T0、T1引脚电平。当某周期采样到一高电平输入，而下一周期又采样到一低电平时，则计数器加1，更新的计数值在下一个机器周期的S3P1期间装入计数器。由于检测一个从1到0的下降沿需要2个机器周期，因此要求被采样的电平至少要维持一个机器周期。当晶振频率为12MHz时，最高计数频率不超过1/2MHz，即计数脉冲的周期要大于2 μs。

## 5.3.2 定时/计数器的控制

80C51单片机定时/计数器的工作由两个特殊功能寄存器控制。TMOD用于设置其工作方式；TCON用于控制其启动和中断申请。

### 一、工作方式寄存器TMOD

工作方式寄存器TMOD用于设置定时/计数器的工作方式，低四位用于T0，高四位用于T1。其格式如下：

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：89H | GATE | $C/\overline{T}$ | M1 | M0 | GATE | $C/\overline{T}$ | M1 | M0 | TMOD |

**GATE**：门控位。**GATE＝0**时，只要用软件使**TCON**中的**TR0**或**TR1**为**1**，就可以启动定时/计数器工作；**GATA＝1**时，要用软件使**TR0**或**TR1**为**1**，同时外部中断引脚或也为高电平时，才能启动定时/计数器工作。即此时定时器的启动条件，加上了或引脚为高电平这一条件。

$C/\overline{T}$ :定时/计数模式选择位。$C/\overline{T}$ ＝**0**为定时模式;$C/\overline{T}$**=1**为计数模式。

**M1M0**：工作方式设置位。定时/计数器有四种工作方式，由**M1M0**进行设置。

定时/计数器工作方式设置表

| M1M0 | 工作方式 | 说　　明 |
|---|---|---|
| 00 | 方式 0 | 13 位定时/计数器 |
| 01 | 方式 1 | 16 位定时/计数器 |
| 10 | 方式 2 | 8 位自动重装定时/计数器 |
| 11 | 方式 3 | T0 分成两个独立的 8 位定时/计数器；T1 此方式停止计数 |

# 二、控制寄存器TCON

TCON的低4位用于控制外部中断,已在前面介绍。TCON的高4位用于控制定时/计数器的启动和中断申请。其格式如下：

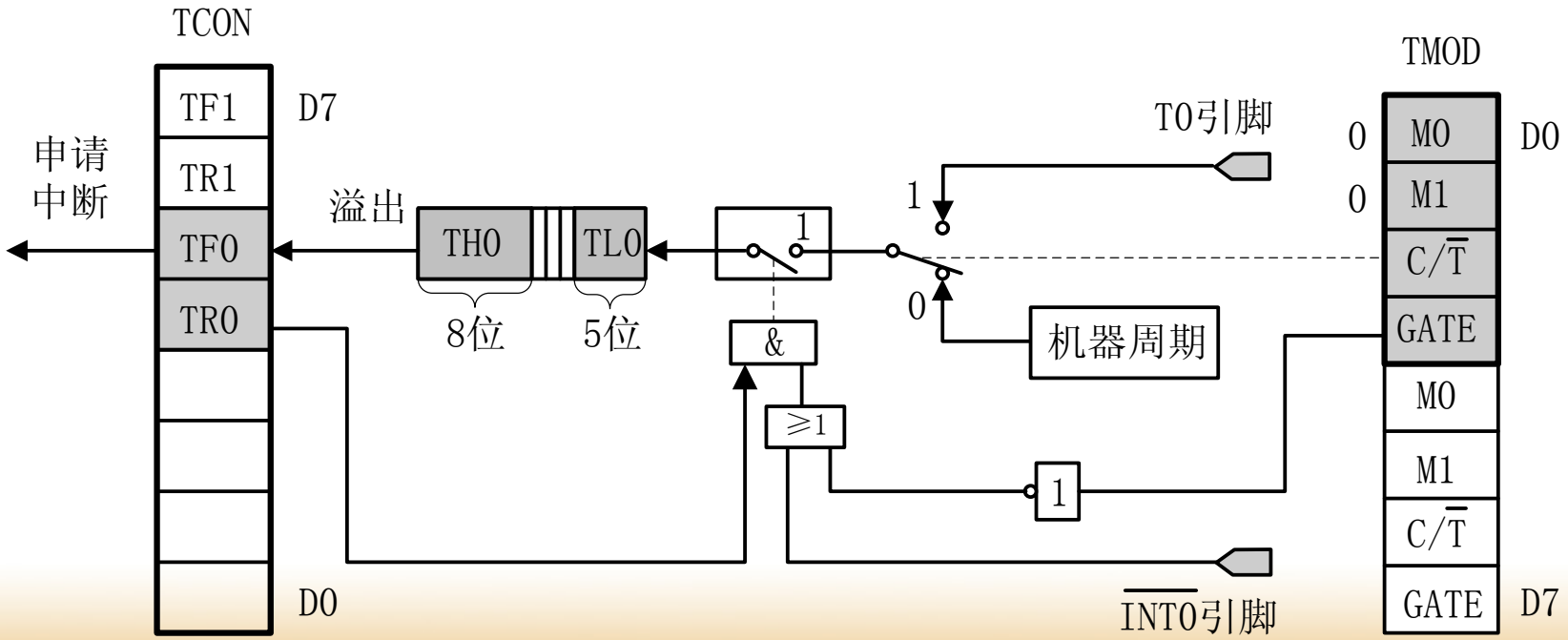
| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 字节地址：88H | TF1 | TR1 | TF0 | TR0 | | | | | TCON |

- **TF1**（TCON.7）：T1溢出中断请求标志位。T1计数溢出时由硬件自动置TF1为1。CPU响应中断后TF1由硬件自动清0。T1工作时，CPU可随时查询TF1的状态。所以，TF1可用作查询测试的标志。TF1也可以用软件置1或清0，同硬件置1或清0的效果一样。
- **TR1**（TCON.6）：T1运行控制位。TR1置1时，T1开始工作；TR1置0时，T1停止工作。TR1由软件置1或清0。所以，用软件可控制定时/计数器的启动与停止。
- **TF0**（TCON.5）：T0溢出中断请求标志位，其功能与TF1类同。
- **TR0**（TCON.4）：T0运行控制位，其功能与TR1类同。

# 5.3.3 定时/计数器的工作方式

## 一、方式0

方式0为13位计数，由TL0的低5位（高3位未用）和TH0的8位组成。TL0的低5位溢出时向TH0进位，TH0溢出时，置位TCON中的TF0标志，向CPU发出中断请求。



定时器模式时有:N＝t/ Tcy

计数初值计算的公式为：$X=2^{13}-N$

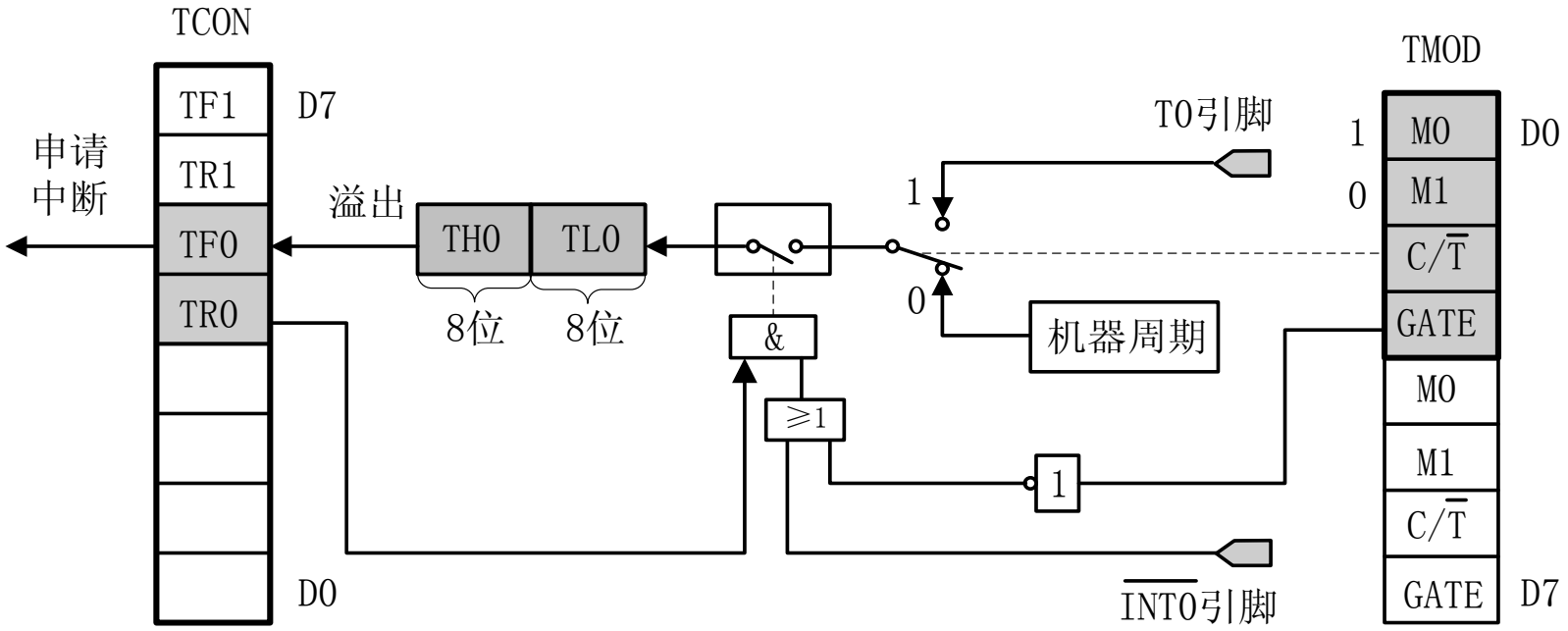
定时器的初值还可以采用计数个数直接取补法获得。

计数模式时，计数脉冲是T0引脚上的外部脉冲。

门控位GATE具有特殊的作用。当GATE=0时，经反相后使或门输出为1，此时仅由TR0控制与门的开启，与门输出1时，控制开关接通，计数开始；当GATE=1时，由外中断引脚信号控制或门的输出，此时控制与门的开启由外中断引脚信号和TR0共同控制。当TR0=1时，外中断引脚信号引脚的高电平启动计数，外中断引脚信号引脚的低电平停止计数。这种方式常用来测量外中断引脚上正脉冲的宽度。

# 二、方式1

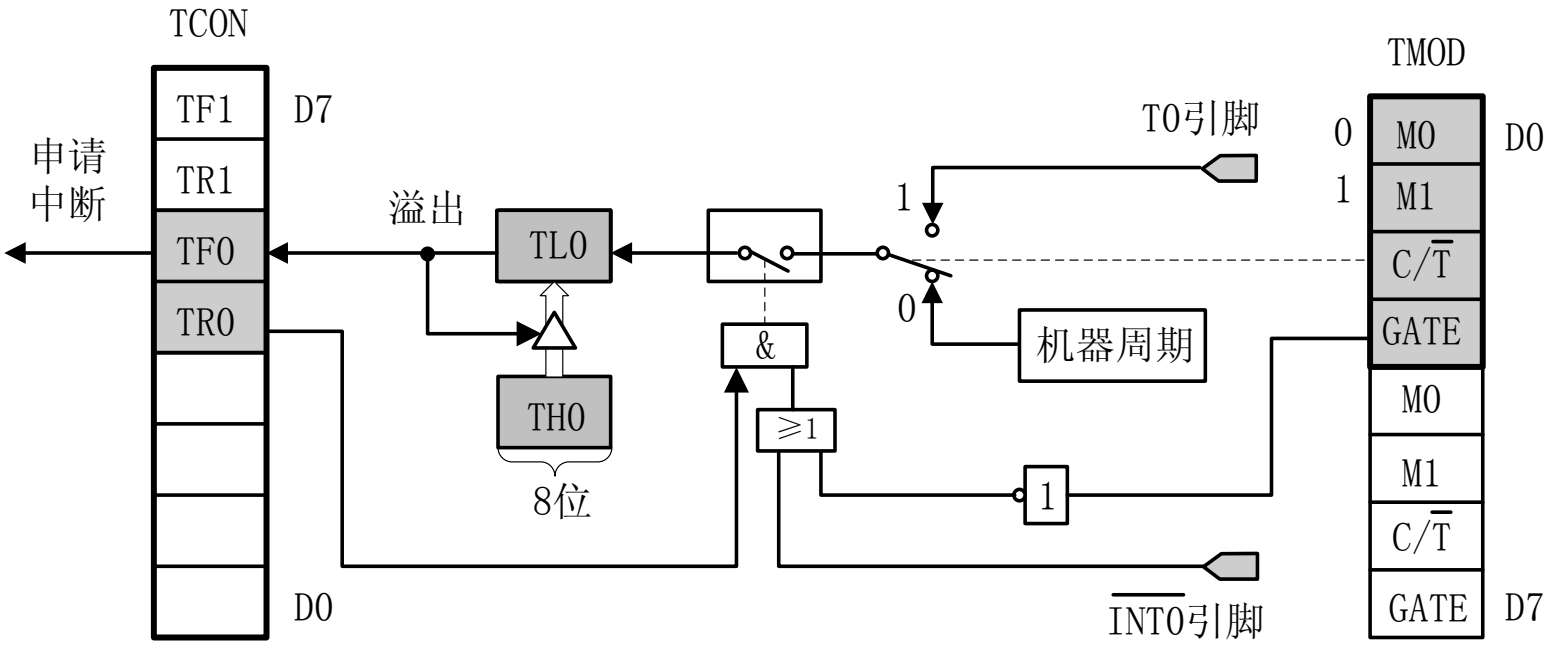
**方式1的计数位数是16位，由TL0作为低8位、TH0作为高8位，组成了16位加1计数器 。**




计数个数与计数初值的关系为： $X=2^{16}-N$

# 三、方式2

## 方式2为自动重装初值的8位计数方式。

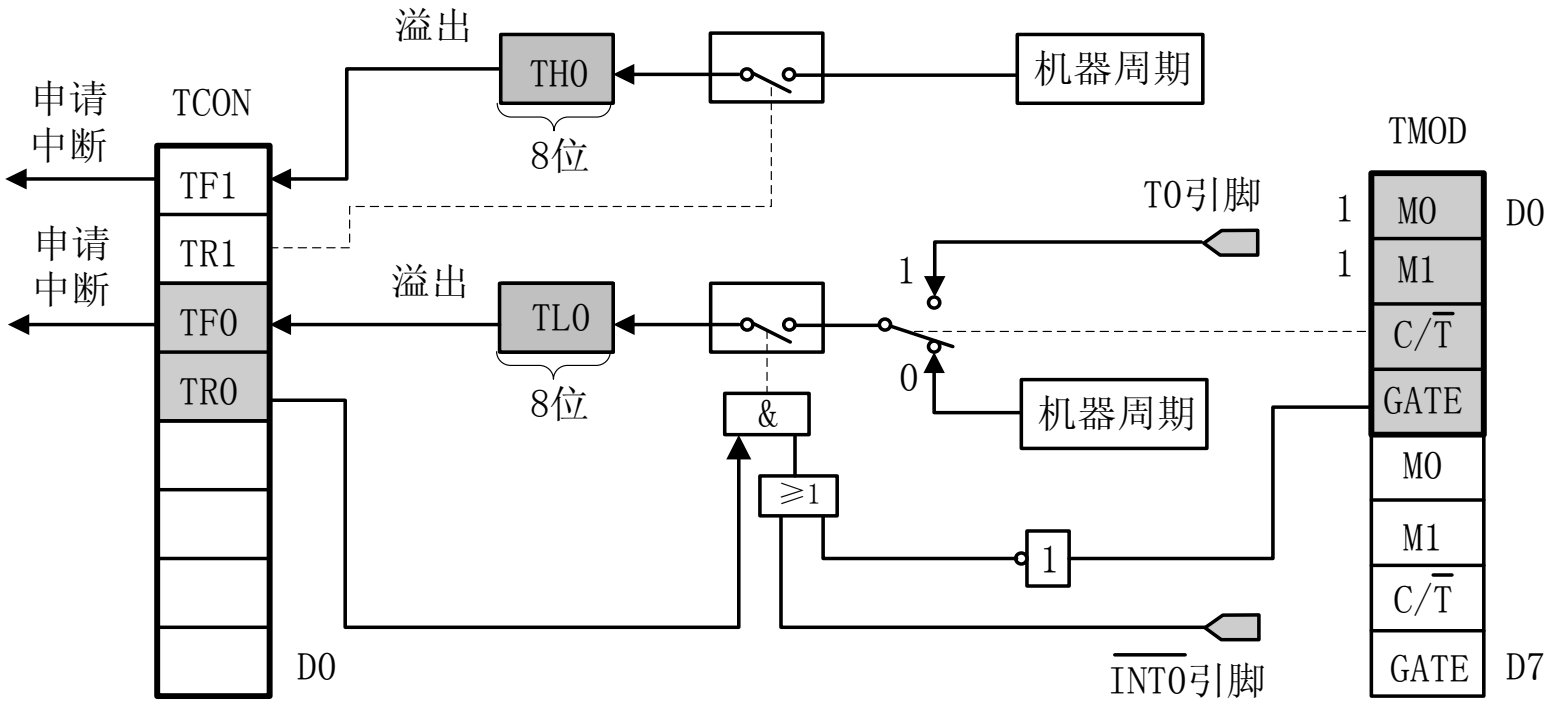



计数个数与计数初值的关系为：$X=2^8-N$

工作方式2特别适合于用作较精确的脉冲信号发生器。

# 四、方式3

方式3只适用于定时/计数器T0，定时器T1处于方式3时相当于TR1=0，停止计数。




工作方式3将T0分成为两个独立的8位计数器TL0和TH0 。

# 5.3.4 定时/计数器用于外部中断扩展

扩展方法是，将定时/计数器设置为计数器方式，计数初值设定为满程，将待扩展的外部中断源接到定时/计数器的外部计数引脚。从该引脚输入一个下降沿信号，计数器加1后便产生定时/计数器溢出中断。

例如，利用T0扩展一个外部中断源。将T0设置为计数器方式，按方式2工作，TH0、TL0的初值均为0FFH，T0允许中断，CPU开放中断。其初始化程序如下：

```
MOV  TMOD，#06H      ；置T0为计数器方式2
MOV  TL0，#0FFH      ；置计数初值
MOV  TH0，#0FFH
SETB TR0             ；启动T0工作
SETB EA              ；CPU开中断
SETB ET0             ；允许T0中断
```

## 5.3.5 定时/计数器应用举例

初始化程序应完成如下工作：

- 对TMOD赋值，以确定T0和T1的工作方式。
- 计算初值,并将其写入TH0、TL0或TH1、TL1。
- 中断方式时，则对IE赋值，开放中断。
- 使TR0或TR1置位，启动定时/计数器定时或计数。

例 利用定时/计数器T0的方式1，产生10ms的定时，并使P1.0引脚上输出周期为20ms的方波，采用中断方式，设系统时钟频率为12 MHz。

解：1、计算计数初值X：

由于晶振为12 MHz，所以机器周期Tcy为1 μs。

所以：

N＝t/ Tcy ＝10×10－3/1×10－6＝10000

X＝65536－10000＝55536＝D8F0H

即应将D8H送入TH0中，F0H送入TL0中

2、求T0的方式控制字TMOD：

M1M0=01，GATE=0，C/T=0，可取方式控制字为01H；

```
        ORG   0000H
        LJMP  MAIN          ；跳转到主程序
        ORG   000BH         ；T0的中断入口地址
        LJMP  DVT0          ；转向中断服务程序
        ORG   0100H
MAIN：  MOV   TMOD，#01H ；置T0工作于方式1
        MOV   TH0，#0D8H      ；装入计数初值
        MOV   TL0，#0F0H
        SETB  ET0           ；T0开中断
        SETB  EA            ；CPU开中断
        SETB  TR0           ；启动T0
        SJMP $              ；等待中断
DVT0：  CPL  P1.0           ；P1.0取反输出
        MOV  TH0，#0D8H       ；重新装入计数值
        MOV  TL0，#0F0H
        RETI                ；中断返回
        END
```

# 思考题及习题

1、80C51有几个中断源？各中断标志是如何产生的？又是如何复位的？CPU响应各中断时，其中断入口地址是多少？

2、某系统有三个外部中断源1、2、3，当某一中断源变低电平时便要求CPU处理，它们的优先处理次序由高到低为3、2、1，处理程序的入口地址分别为2000H、2100H、2200H。试编写主程序及中断服务程序（转至相应的入口即可）。

3、外部中断源有电平触发和边沿触发两种触发方式，这两种触发方式所产生的中断过程有何不同？怎样设定？

4、定时/计数器工作于定时和计数方式时有何异同点？

5、定时/计数器的4种工作方式各有何特点？

6、要求定时/计数器的运行控制完全由TR1、TR0确定和完全由、高低电平控制时，其初始化编程应作何处理？

7、当定时/计数器T0用作方式3时，定时/计数器T1可以工作在何种方式下？如何控制T1的开启和关闭？

8、利用定时/计数器T0从P1.0输出周期为1s，脉宽为20ms的正脉冲信号，晶振频率为12MHz。试设计程序。

9、要求从P1.1引脚输出1000Hz方波，晶振频率为12MHz。试设计程序。

10、试用定时/计数器T1对外部事件计数。要求每计数100，就将T1改成定时方式，控制P1.7输出一个脉宽为10ms的正脉冲，然后又转为计数方式，如此反复循环。设晶振频率为12MHz。

11、利用定时/计数器T0产生定时时钟,由P1口控制8个指示灯。编一个程序,使8个指示灯依次一个一个闪动，闪动频率为20次/秒(8个灯依次亮一遍为一个周期)。

12、若晶振频率为12MHz，如何用T0来测量20～1s之间的方波周期？又如何测量频率为0.5MHz左右的脉冲频率？