51单片机汇编语言教程：第18课-单片机中断系统

（基于 HJ-1G、HJ-3G 实验板）

有关单片机中断系统的概念：什么是中断，我们从一个生活中的例程引入。你正在家中看书，突然电话铃响了，你放下书本，去接电话，和来电话的人交谈，然后放下电话，回来继续看你的书。这就是生活中的“中断”的现象，就是正常的工作过程被外部的事件打断了。仔细研究一下生活中的中断，对于我们学习单片机的中断也很有好处。

第一、什么可以引起中断，生活中很多事件能引起中断：有人按了门铃了，电话铃响了，你的闹钟闹响了，你烧的水开了….等等诸如此类的事件，我们把能引起中断的称之为中断源，单片机中也有一些能引起中断的事件，8031中一共有5个：两个外部中断，两个计数/定时器中断，一个串行口中断。

第二、中断的嵌套与优先级处理：设想一下，我们正在看书，电话铃响了，同时又有人按了门铃，你该先做那样呢？如果你正是在等一个很重要的电话，你一般不会去理会门铃的，而反之，你正在等一个重要的客人，则可能就不会去理会电话了。如果不是这两者（即不等电话，也不是等人上门），你可能会按你常常的习惯去处理。总之这里存在一个优先级的问题，单片机中也是如此，也有优先级的问题。优先级的问题不仅仅发生在两个中断同时产生的情况，也发生在一个中断已产生，又有一个中断产生的情况，比如你正接电话，有人按门铃的情况，或你正开门与人交谈，又有电话响了情况。考虑一下我们会怎么办吧。

第三、中断的响应过程：当有事件产生，进入中断之前我们必须先记住现在看书的第几页了，或拿一个书签放在当前页的位置，然后去处理不一样的事情（因为处理完了，我们还要回来继续看书）：电话铃响我们要到放电话的地方去，门铃响我们要到门那边去，也说是不一样的中断，我们要在不一样的地点处理，而这个地点常常还是固定的。计算机中也是采用的这种办法，五个中断源，每个中断产生后都到一个固定的地方去找处理这个中断的程序，当然在去之前首先要保存下面将执行的指令的地址，以便处理完中断后回到原来的地方继续往下执行程序。具体地说，中断响应能分为以下几个步骤：1、保护断点，即保存下一将要执行的指令的地址，就是把这个地址送入堆栈。2、寻找中断入口，根据5个不一样的中断源所产生的中断，查找5个不一样的入口地址。以上工作是由计算机自动完成的，与编程者无关。在这5个入口地址处存放有中断处理程序（这是程序编写时放在那儿的，如果没把中断程序放在那儿，就错了，中断程序就不能被执行到）。3、执行中断处理程序。4、中断返回：执行完中断指令后，就从中断处返回到主程序，继续执行。究竟单片机是怎么样找到中断程序所在位置，又怎么返回的呢？我们稍后再谈。

MCS-51单片机中断系统的结构：

5个中断源的符号、名称及产生的条件如下。

INT0：外部中断0，由 P3．2端口线引入，低电平或下跳沿引起。

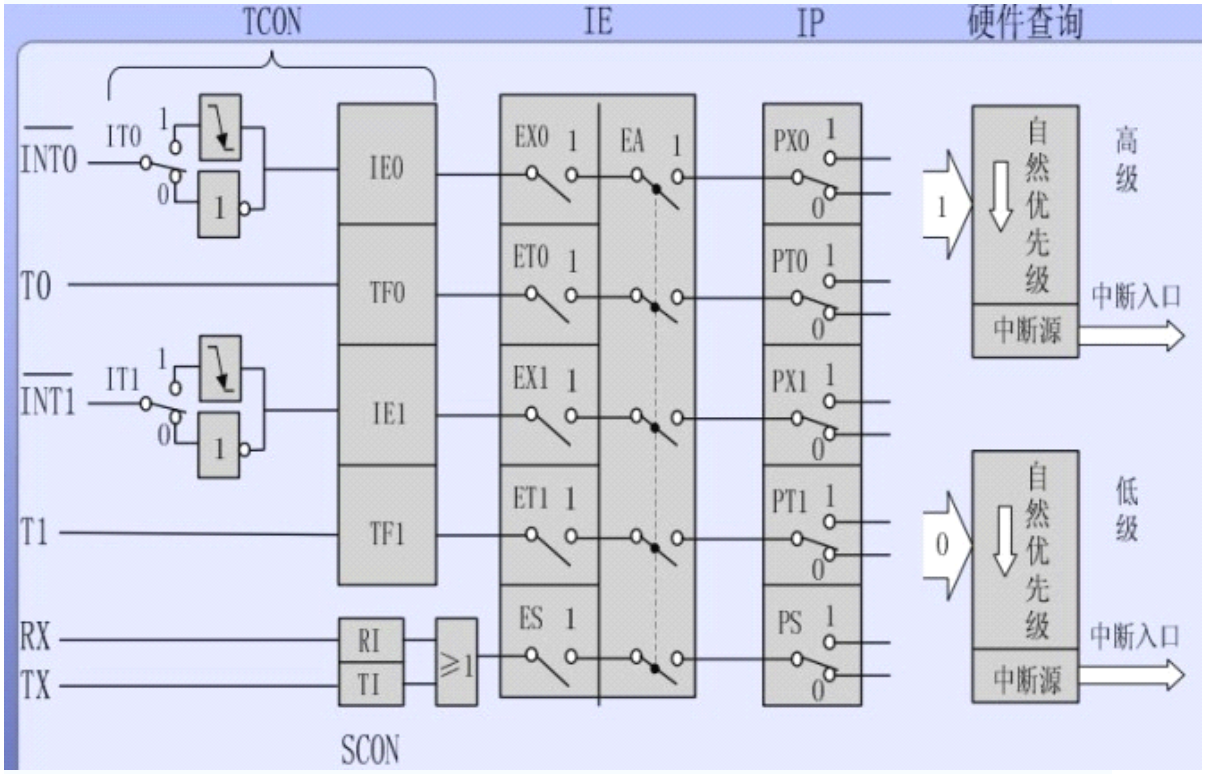
INT1：外部中断1，由 P3．3端口线引入，低电平或下跳沿引起。

T0：定时器／计数器0中断，由 T0计满回零引起。

T1：定时器／计数器 1 中断，由 T1计满回零引起。

TI／RI：串行 I／O 中断，串行端口完成一帧字符发送／接收后引起。

整个中断系统的结构框图见下图一所示。



<51单片机中断系统结构>

如图所示，由与中断有关的特殊功能寄存器、中断入口、次序查询逻辑电路等组成，包括5个中断请求源，4个用于中断控制的寄存器 IE、IP、ECON 和 SCON 来控制中断类弄、中断的开、关和各种中断源的优先级确定。

中断请求源：

（1）外部中断请求源：即外中断0和1，经由外部管脚引入的，在单片机上有两个管脚，名称为 INT0、INT1，也就是 P3.2、P3.3这两个管脚。在内部的 TCON 中有四位是与外中断有关的。IT0：INT0触发方式控制位，可由软件进和置位和复位，IT0=0，INT0为低电平触发方式，IT0=1，INT0为负跳变触发方式。这两种方式的差异将在以后再谈。IE0：INT0中断请求标志位。当有外部的中断请求时，这位就会置1（这由硬件来完成），在 CPU 响应中断后，由硬件将 IE0清0。IT1、IE1的用途和 IT0、IE0相同。（2）内部中断请求源 TF0：定时器 T0的溢出中断标记，当 T0计数产生溢出时，由硬件置位 TF0。当 CPU 响应中断后，再由硬件将 TF0清0。TF1：与 TF0类似。TI、RI：串行口发送、接收中断，在串行口中再讲解。2、中断允许寄存器 IE 在 MCS－51中断系统中，中断的允许或禁止是由片内可进行位寻址的8位中断允许寄存器 IE 来控制的。见下表 EAX

其中 EA 是总开关，如果它等于0，则所有中断都不允许。ES－串行口中断允许 ET1－定时器1中断允许 EX1－外中断1中断允许。ET0－定时器0中断允许 EX0－外中断0中断允许。如果我们要设置允许外中断1，定时器1中断允许，其它不允许，则 IE 能是 EAX

即8CH，当然，我们也能用位操作指令 SETB EA

SETB ET1SETB EX1

来实现它。3、五个中断源的自然优先级与中断服务入口地址外中断0：0003H 定时器0：000BH 外中断1：0013H 定时器1：001BH 串行口：0023H 它们的自然优先级由高到低排列。写到这里，大家应当明白，为什么前面有一些程序一始我们这样写：

ORG 0000HLJMP START

ORG 0030H

START:。

这样写的目的，就是为了让出中断源所占用的向量地址。当然，在程序中没用中断时，直接从0000H 开始写程序，在原理上并没有错，但在实际工作中最好不这样做。优先级：单片机采用了自然优先级和人工设置高、低优先级的策略，即能由程序员设定那些中断是高优先级、

哪些中断是低优先级，由于只有两级，必有一些中断处于同一级别，处于同一级别的，就由自然优先级确定。

开机时，每个中断都处于低优先级，我们能用指令对优先级进行设置。看表2中断优先级中由中断优先级寄存器 IP 来高置的，IP 中某位设为1，对应的中断就是高优先级，不然就是低优先级。

XX

X

PS

PT1

PX1

PT0

PX0

例：设有如下要求，将 T0、外中断1设为高优先级，其它为低优先级，求 IP 的值。IP 的首3位没用，可任意取值，设为000，后面根据要求写就能了 XX

因此，最终，IP 的值就是06H。例：在上例中，如果5个中断请求同时发生，求中断响应的次序。响应次序为：定时器0－＞外中断1－＞外中断0－＞实时器1－＞串行中断。

MCS－51的中断响应过程：

1、中断响应的条件：讲到这儿，我们依然对于计算机响应中断感到神奇，我们人能响应外界的事件，是因为我们有多种“传感器“－－眼、耳能接受不一样的信息，计算机是如何做到这点的呢？其实说穿了，一点都不希奇，MCS51工作时，在每个机器周期中都会去查询一下各个中断标记，看他们是否是“1“，如果是1，就说明有中断请求了，所以所谓中断，其实也是查询，不过是每个周期都查一下而已。这要换成人来说，就相当于你在看书的时候，每一秒钟都会抬起头来看一看，查问一下，是不是有人按门铃，是否有电话。。。。很蠢，不是吗？可计算机本来就是这样，它根本没人聪明。了解了上述中断的过程，就不难解中断响应的条件了。在下列三种情况之一时，CPU 将封锁对中断的响应：

CPU 正在处理一个同级或更高级别的中断请求。

现行的机器周期不是当前正执行指令的最后一个周期。我们知道，单片机有单周期、双周期、三周期指令，当前执行指令是单字节没有关系，如果是双字节或四字节的，就要等整条指令都执行完了，才能响应中断（因为中断查询是在每个机器周期都可能查到的）。

当前正执行的指令是返回批令（RETI）或访问 IP、IE 寄存器的指令，则 CPU 至少再执行一条指令才应中断。这些都是与中断有关的，如果正访问 IP、IE 则可能会开、关中断或改变中断的优先级，而中断返回指令则说明本次中断还没有处理完，所以都要等本指令处理结束，再执行一条指令才能响应中断。

2、中断响应过程 CPU 响应中断时，首先把当前指令的下一条指令（就是中断返回后将要执行的指令）的地址送入堆栈，然后根据中断标记，将对应的中断入口地址送入 PC，PC 是程序指针，CPU 取指令就根据 PC 中的值，PC 中是什么值，就会到什么地方去取指令，所以程序就会转到中断入口处继续执行。这些工作都是由硬件来完成的，不必我们去考虑。这里还有个问题，大家是否注意到，每个中断向量地址只间隔了8个单元，如0003－000B，在如此少的空间中如何完成中断程序呢？很简单，你在中断处安排一个 LJMP 指令，不就能把中断程序跳转到任何地方了吗？一个完整的主程序看起来应该是这样的：

```
ORG 0000HLJMP START
ORG 0003H
LJMP INT0 ；转外中断0ORG 000BH
RETI ；没有用定时器0中断，在此放一条 RETI，万一 “不小心“产生了中断，也不会有太大的后果。。
```

中断程序完成后，一定要执行一条 RETI 指令，执行这条指令后，CPU 将会把堆栈中保存着的地址取出，送回 PC，那么程序就会从主程序的中断处继续往下执行了。注意：CPU 所做的保护工作是很有限的，只保护了一个地址，而其它的所有东西都不保护，所以如果你在主程序中用到了如 A、PSW 等，在中断程序中又要用它们，还要保证回到主程序后这里面的数据还是没执行中断以前的数据，就得自己保护起来。

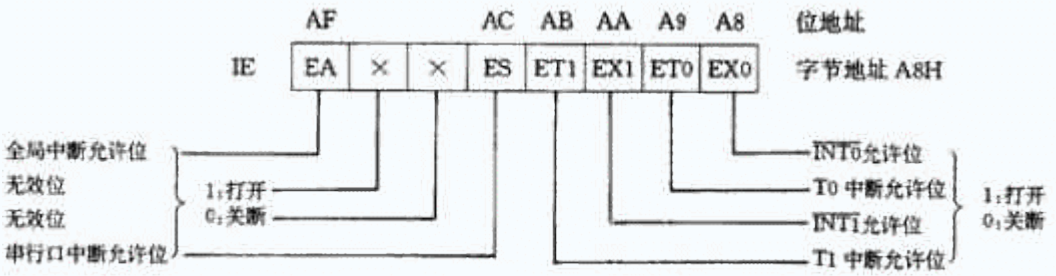
中断系统的控制寄存器：

中断系统有两个控制寄存器 IE 和 IP，它们分别用来设定各个中断源的打开 / 关闭和中断优先级。此外，在 TCON 中另有4位用于选择引起外部中断的条件并作为标志位。

1．中断允许寄存器--IE

IE 在特殊功能寄存器中，字节地址为 A8H，位地址(由低位到高位)分别是 A8H-AFH。

IE 用来打开或关断各中断源的中断请求，基本格式如下图二所示：



图二、中断允许寄存器——IE

EA：全局中断允许位。EA＝0，关闭全部中断；EA＝1，打开全局中断控制，在此条件下，由各个中断控制位确定相应中断的打开或关闭。

×：无效位。

ES：串行 I / O 中断允许位。ES＝1，打开串行 I / O 中断；ES＝0，关闭串行 I / O 中断。

ET1；定时器 / 计数器1中断允许位。ET1＝1，打开 T1中断；ET1＝0，关闭 T1中断。

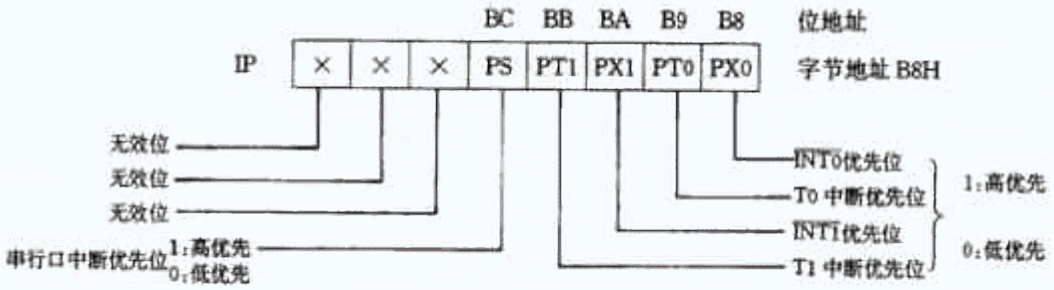
EX1：外部中断 1 中断允许位。EX1＝1，打开 INT1；EX1＝0，关闭 INT1。

ET0：定时器 / 计数器0中断允许位。ET0＝1，打开 T0中断；ET0＝0，关闭 T0 中断。

EX0：外部中断0中断允许位。Ex0＝1，打开 INT0;EX0=0,关闭 INT0.

中断优先寄存器--IP：

IP 在特殊功能寄存器中，字节地址为 B8H，位地址(由低位到高位)分别是 B8H 一 BFH,IP 用来设定各个中断源属于两级中断中的哪一级，IP 的基本格式如下图三所示：



图三、中断优先寄存器——IP

×：无效位。

PS：串行 I / O 中断优先级控制位。PS＝1，高优先级；PS＝0，低优先级。

PT1：定时器 / 计数器1中断优先级控制位。PT1＝1，高优先级；PT1＝0，低优先级。

Px1：外部中断1中断优先级控制位。Px1＝1，高优先级；PX1＝0，低优先级。

PT0：定时器 / 计数器 o 中断优先级控制位。PT0＝1，高优先级；PT0＝0，低优先级。

Px0：外部中断0中断优先级控制位。Px0＝1，高优先级；Px0＝0，伤优先级。

在 MCS-51单片机系列中，高级中断能够打断低级中断以形成中断嵌套；同级中断之间，或低级对高级中断则不能形成中断嵌套。若几个同级中断同时向 CPU 请求中断响应，则 CPU 按如下顺序确定响应的先后顺序：

INT0一 T0---INT1一 T1一 RI / T1.

中断的响应过程

若某个中断源通过编程设置，处于被打开的状态，并满足中断响应的条件，而且①当前正在执行的那条指令已被执行完

1、当前末响应同级或高级中断

2、不是在操作 IE，IP 中断控制寄存器或执行 REH 指令则单片机响应此中断。

在正常的情况下，从中断请求信号有效开始，到中断得到响应，通常需要3个机器周期到8个机器周期。中断得到响应后，自动清除中断请求标志(对串行 I / O 端口的中断标志，要用软件清除)，将断点即程序计数器之值(PC)压入堆栈(以备恢复用)；然后把相应的中断入口地址装入 PC，使程序转入到相应的中断服务程序中去执行。

各个中断源在程序存储器中的中断入口地址如下：

中断源 入口地址

INT0(外部中断0) 0003H

TF0(T0 中断) 000BH

INT1(外部中断1) 0013H

TF1(T1中断) 001BH

RI / TI(串行口中断) 0023H

由于各个中断入口地址相隔甚近，不便于存放各个较长的中断服务程序，故通常在中断入口地址开始的二三个单元中，安排一条转移类指令，以转入到安排在那儿的中断服务程序。以T1中断为例，其过程下如图四所示。

由于5个中断源各有其中断请求标志0，TF0，IE1，TF1 以及 RI / TI，在中断源满足中断请求的条件下，各标志自动置1，以向 CPU 请求中断。如果某一中断源提出中断请求后，CPU 不能立即响应，只要该中断请求标志不被软件人为清除，中断请求的状态就将一直保持,直到 CPU 响应了中断为止,对串行口中断而言,这一过程与其它4个中断的不同之处在于;即使 CPU 响应了中断，其中断标志 RI / TI 也不会自动清零，必须在中断服务程序中设置清除 RI / TI

的指令后，才会再一次地提出中断请求。

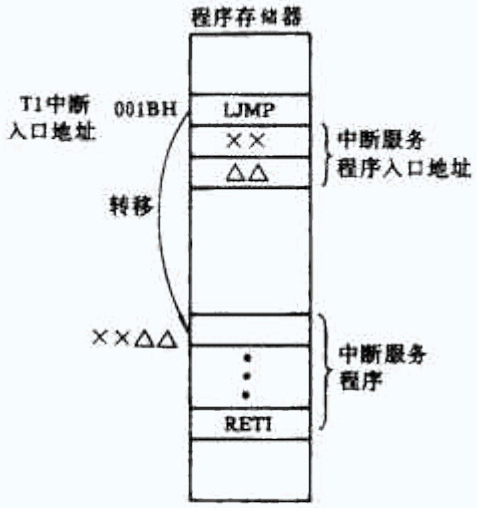
CPU 的现场保护和恢复必须由被响应的相应中断服务程序去完成，当执行 RETI 中断返回指令后，断点值自动从栈顶2字节弹出，并装入 PC 寄存器，使 CPU 继续执行被打断了的程序。

下面给出一个应用定时器中断的实例。

现要求编制一段程序，使 P1．0端口线上输出周期为2ms 的方波脉冲。设单片机晶振频率 Fosc＝6MHZ.

1、方法：利用定时器 T0作1ms 定时，达到定时值后引起中断，在中断服务程序中，使 P1.0的状态取一次反，并再次定时1ms。

2、定时初值：机器周期 MC＝12/fosc＝2us。所以定时 1ms 所需的机器周期个数为500D，亦即01F4H。设 T0为工作方式1(16位方式)，则定时初值是(01F4H)求补＝FE0CH



图四

由中断入口地址进入中断服务程序

```
START:  MOV TMOD,#01H   ; T0为定时器状态，工作方式1
        MOV TL0, #0CH   ; T0的低位定时初值
        MOV TH0,#0FEH   ; T0的高位定时初值
        MOV TCON,#10H   ; 打开 T0
        SETB ET0        ; 1ET0，即允许 T0中断
        SETB EA         ; 1EA，即充许全局中断
        AJMP $          ; 动态暂存
000BH:  AJMP IST0       ; 转入 T0中断服务程序入口地址 IST0
```

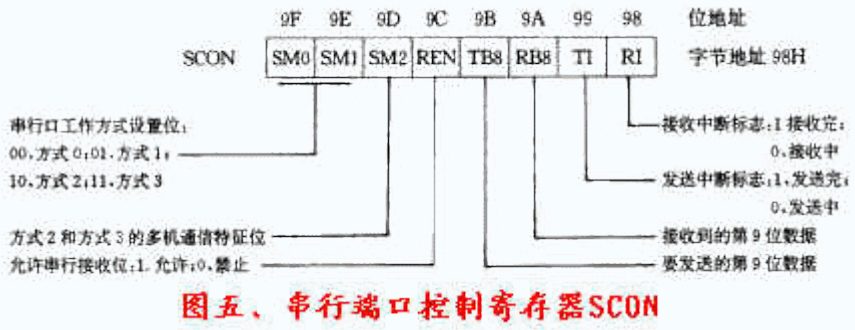
```
IST0:   MOV TL0, #0CH   ; 重置定时器初值
        MOV TH0,#0FEH   ; 重置定时器初值
        CPL P1.0        ; P1.0取反
        RET1            ; 中断返回
```

串行端口的控制寄存器：

串行端口共有2个控制寄存器 SCON 和 PCON，用以设置串行端口的工作方式、接收 / 发送的运行状态、接收 / 发送数据的特征、波特率的大小，以及作为运行的中断标志等。

①串行口控制寄存器 SCON

SCON 的字节地址是98H，位地址(由低位到高位)分别是98H 一9FH。SCON 的格式如图五所示。



图五、串行端口控制寄存器SCON

SMo，SM1：

串行口工作方式控制位。

00--方式0；01--方式1；

10--方式2；11--方式3。

SM2：

仅用于方式2和方式3的多机通讯控制位

发送机 SM2＝1(要求程控设置)。

当为方式2或方式3时：

接收机 SM2＝1时，若 RB8＝1，可引起串行接收中断；若 RB8＝0，不引起串行接收中断。SM2＝0时，若 RB8＝1，可引起串行接收中断；若 RB8＝0，亦可引起串行接收中断。

REN:

串行接收允许位。

0--禁止接收；1--允许接收。

TB8：

在方式2，3中，TB8是发送机要发送的第9位数据。

RB8：

在方式2，3中，RB8是接收机接收到的第9位数据，该数据正好来自发

送机的 TB8。

TI：

发送中断标志位。发送前必须用软件清零，发送过程中 TI 保持零电平，

发送完一帧数据后，由硬件自动置1。如要再发送，必须用软件再清零。

RI：

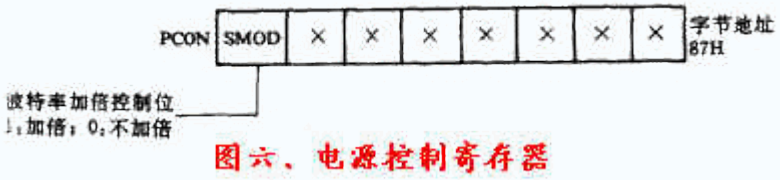
接收中断标志位。接收前，必须用软件清零，接收过程中 RI 保持零电

平，接收完一帧数据后，由片内硬件自动置1。如要再接收，必须用软件

再清零。

电源控制寄存器 PCON

PCON 的字节地址为87H，无位地址，PCON 的格式如图六所示。需指出的是，对80C31单片机而言，PCON 还有几位有效控制位。



图六、电源控制寄存器

SMOD：波特率加倍位。在计算串行方式1，2，3的波特率时；0---不加倍；1---加倍。

串行中断的应用特点：

8031单片机的串行 I／O 端口是一个中断源，有两个中断标志 RI 和 TI，RI 用于接收，TI 用于发送。

串行端口无论在何种工作方式下，发送／接收前都必须对 TI／RI 清零。当一帧数据发送／接收完后，TI/RI 自动置1，如要再发送／接收，必须先用软件将其清除。

在串行中断被打开的条件下，对方式0和方式1来说，一帧数据发送／接收完后，除置位 TI／RI 外，还会引起串行中断请求，并执行串行中侧目务程序。但对方式2和方式3的接收机

而言，还要视 SM2和 RB8的状态，才可确定 RI 是否被置位以及串行中断的开放：

SM2 RB8 接收机中断标志与中断状态

0 1 激活 RI，引起中断

1 0 不激活 RI，不引起中断

1 1 激活 RI，引起中断

单片机正是利用方式2，3的这一特点，实现多机间的通信。串行端口的常用应用方法见相关章节。

波特率的确定：

对方式0来说，波特率已固定成 fosc / 12，随着外部晶振的频率不同，波特率亦不相同。常用的 fosc 有12MHz 和6MHz，所以波特率相应为1000×103和500×103位 / s。在此方式下，数据将自动地按固定的波特率发送 / 接收，完全不用设置。

对方式2而言，波特率的计算式为2SMOD • fosc / 64。当 SMOD＝0时，波特率为 fm / 64；当 SMOD＝1时，波特率为 fosc / 32。在此方式下，程控设置 SMOD 位的状态后，波特率就确定了，不需要再作其它设置。

对方式1和方式3来说，波特率的计算式为2SMOD / 32×T1溢出率，根据 SMOD 状态位的不同，波特率有 T1 / 32溢出率和 T1 / 16溢出率两种。由于 T1溢出率的设置是方便的，因而波特率的选择将十分灵活。

前已叙及，定时器 T1 有4种工作方式，为了得到其溢出率，而又不必进入中断服务程序，往往使 T1设置在工作方式2的运行状态，也就是8位自动加入时间常数的方式。由于在这种方式下，T1的溢出率(次 / 秒)计算式可表达成：

$$\text{T1 溢出率}=\frac{f_{osc}}{12(256-x)}$$

式中 x 为设置的定时初值，于是波特率(位/秒)表达式为：

$$BR=\frac{2^{SMOD}}{32}\times\frac{f_{osc}}{12(256-x)}$$

由上式可见，选取不同的 x 初值，就可得到不同的波特率，

把上式变换一下形式，就可根据所要求的波特率 BR，算出 T1 定时器初值的大小来，其计算式为：

$$x=256-\frac{2^{SMOD}\cdot f_{osc}}{384\cdot BR}$$

例如，以产生 1200 位/秒为例，求定时器 T1 定时初值 x 的大小。

设 $f_{osc}=6MHz$，SMOD=0，则

$$1200/秒=\frac{1}{32}\times\frac{6\times10^{6}Hz}{12(256-x)}$$

解得

$$x=243D=F3H$$

下面一段主程序和中断服务程序，是利用串行方式 1 从数据00H 开始连续不断增大地串行发

送一片数据的程序例。设单片机晶振的频率为6MHZ，波特率为1200位 / 秒。

```
ORG 2000H       ;1200位/秒的定时器初值
MOV
TL1,#0F3H
MOV
                ;使 SMOD=0
TH1,#0F3H
MOV
                ;T1方式2
PCON,#00H
MOV
TMOD,#20H
SETB EA
CLR ET1         ;关闭 T1中断
SETB ES         ;开串行中断
SETB TR1        ;开 T1定时
MOV
                ;串行方式1
SCON,#40H
CLR A
MOV SBUF,A      ;串行发送
JNB T1,$        ;等待发送完
CLR T1,         ;清标志
SJMP $
ORG 0023H       ;串行中断入口地址
MOV SBUF,A      ;连续发送
JNB T1,$
INC A
CLR T1
RET1            ;中断返回
```

51 单片机汇编语言教程-慧净电子会员收集整理 （全部 28 课）

51 实验板推荐(点击下面的图片可以进入下载资料链接)

HJ-1G	HJ-3G