

## 单片机原理及应用

### 第一章 绪论

1.什么叫单片机？其主要特点有哪些？

在一片集成电路芯片上集成微处理器、存储器、I/O 接口电路，从而构成了单芯片微型计算机，即单片机。

特点：控制性能和可靠性高、体积小、价格低、易于产品化、具有良好的性价比。

### 第二章 80C51 的结构和原理

#### 1.80C51 的基本结构

##### a.CPU 系统

- 8 位 CPU，含布尔处理器；
- 时钟电路；
- 总线控制逻辑。

##### b.存储器系统

- 4K 字节的程序存储器（ROM/EPROM/FLASH，可外扩至 64KB）；
- 128 字节的数据存储器（RAM，可外扩至 64KB）；
- 特殊功能寄存器 SFR。

##### c.I/O 口和其他功能单元

- 4 个并行 I/O 口；
- 2 个 16 位定时/计数器；
- 1 个全双工异步串行口；
- 中断系统（5 个中断源，2 个优先级）

#### 2.80C51 的应用模式

##### a.总线型单片机应用模式

- ◆ 总线型应用的“三总线”模式；
- ◆ 非总线型应用的“多 I/O”模式

#### 3.80C51 单片机的封装和引脚

##### a.总线型 DIP40 引脚封装

- $\overline{\text{RST}}/\text{V}_{\text{PO}}$ ：复位信号输入引脚/备用电源输入引脚；
- $\text{ALE}/\overline{\text{PROG}}$ ：地址锁存允许信号输出引脚/编程脉冲输入引脚；

- $\overline{\text{EA}}/\text{V}_{\text{PP}}$ ：内外存储器选择引脚/片内 EPROM 编程电压输入引脚；

- $\overline{\text{PSEN}}$ ：外部程序存储器选通信号输出引脚

##### b.非总线型 DIP20 封装的引脚

- $\overline{\text{RST}}$ ：复位信号输入引脚

#### 4.80C51 的片内存储器

增强型单片机片内数据存储器为 256 字节，地址范围是 00H~FFH。低 128 字节的配情况与基本型单片机相同。高 128 字节一般为 RAM，仅能采用寄存器间接寻址方式询问。注意：与该地址范围重叠的特殊功能寄存器 SFR 空间采用直接寻址方式询问。

#### 5.80C51 的时钟信号

晶振周期为最小的时序单位。一个时钟周期包含 2 个晶振周期。晶振信号 12 分频后形成机器周期。即一个机器周期包含 12 个晶振周期或 6 个时钟周期。

#### 6.80C51 单片机的复位

定义：复位是使单片机或系统中的其他部件处于某种确定的初始状态。

##### a.复位电路

两种形式：一种是上电复位；另一种是上电与按键均有效的复位。

##### b.单片机复位后的状态

单片机的复位操作使单片机进入初始化状态。初始化后，程序计数器  $\text{PC}=0000\text{H}$ ，所以程序从 0000H 地址单元开始执行。

特殊功能寄存器复位后的状态是确定的。 $\text{P0}\sim\text{P3}$  为 FFH，SP 为 07H，SBUF 不定，IP、IE 和 PCON 的有效位为 0，其余的特殊功能寄存器的状态为 00H。相应的意义为：

- $\text{P0}\sim\text{P3}=\text{FFH}$ ，相当于各口锁存器已写入 1，此时不但可用于输出，也可以用于输入；
- $\text{SP}=07\text{H}$ ，堆栈指针指向片内 RAM

的 07 单元；

- IP、IE 和 PCON 的有效位为 0，各中断源处于低优先级且均被关断、串行通讯的波特率不加倍；
- PSW=00H，当前工作寄存器为 0 组。

### 7.80C51 的存储器组织

功能：存储信息（程序和数据）

a.程序存储器低端的一些地址被固定地用作特定的入口地址：

- 0000H：单片机复位后的入口地址；
- 0003H：外部中断 0 的中断服务程序入口地址；
- 000BH：定时/计数器 0 溢出中断服务程序入口地址；
- 0013H：外部中断 1 的中断服务程序入口地址；
- 001BH：定时/计数器 1 溢出中断服务程序入口地址；
- 0023H：串行口的中断服务程序入口地址。

### 8.80C51 单片机的数据存储器配置

80C51 单片机数据存储器分为片外 RAM 和片内 RAM 两大部分。

80C51 片内 RAM 的 128B 部分分成工作寄存器区、位寻址区、通用 RAM 三大部分。

基本型单片机片内 RAM 的地址范围是 00H~7FH。

增强型单片机片内除地址范围为 00H~7FH 的 128B RAM 外，又增加了 80H~FFH 的高 128B 的 RAM。增加的这部分 RAM 仅能采用间接寻址方式访问。

片外 RAM 地址空间为 64KB，地址范围是 0000H~FFFFH。

与程序存储器地址空间不同的是，片外 RAM 地址空间与片内 RAM 地址空间在地址的低端 0000H~007FH 是重叠的。访问片外 RAM 时采用指令 MOVX 实现，访问片内 RAM 采用指令 MOV，无读写信号产生。

a.工作寄存器区

80C51 单片机片内 RAM 低端的

00H~1FH 共 32B，分成 4 个工作寄存器组，每组占 8 个单元。

- 寄存器 0 组：地址 00H~07H；
- 寄存器 1 组：地址 08H~0FH；
- 寄存器 2 组：地址 10H~17H；
- 寄存器 3 组：地址 18H~1FH。

程序运行时，只能有一个工作寄存器组作为当前工作寄存器组；当前工作寄存器组的选择由特殊功能寄存器中得程序状态字寄存器 PSW 的 RS1、RS0 来决定。

b.位寻址区

内部 RAM 的 20H~2FH 共 16 个字节是位寻址区。其 128 位的地址范围是 00H~7FH。

c.通用 RAM 区

位寻址区之后的 30H~7FH 共 80 个字节为通用 RAM 区。这些单元可以作为数据缓冲器使用。

实际应用中，常需在 RAM 区设置堆栈。80C51 的堆栈一般设在 30H~7FH 的范围内，栈顶的位置由堆栈指针 SP 指示。复位时 SP 的初值为 07H，在系统初始化时可以重新设置。

### 9.80C51 单片机的特殊功能寄存器

a.程序状态字寄存器 PSW，8 位。含义如下：

- CY：进位、借位标志。有进位、借位时 CY=1，否则 CY=0；
- AC：辅助进位、借位标志
- F0：用户标志位，由用户自己定义
- RS1、RS0：当前工作寄存器组选择位
- OV：溢出标志位。有溢出时 OV=1，否则 OV=0
- P：奇偶标志位。存于 ACC 中的运算结果有奇数个“1”时 P=1，否则 P=0。

### 10.P0 口、P2 口的结构

P0 口作为分时复用的低 8 位地址/数据总线，P2 口作为高 8 位地址总线。

a.P0 口的结构

- P0 用作通用 I/O 口



输出驱动级工作在需外接上拉电阻的漏极开路方式；

P0 口在作为通用 I/O 口，属于准双向口。

- P0 用作地址/数据总线

b.P2 口的结构

P2 口的输出驱动电路与 P0 口不同，内部设有上拉电阻。

- P2 用作通用 I/O 口

P2 口在作为通用 I/O 口，属于准双向口。

- P2 用作地址总线

11.P3 口结构

a.P3 用作第一功能的通用 I/O 口

b.P3 用作第二功能使用

- P3.0: RXD（串行口输入）；
- P3.1: TXD（串行口输出）；
- P3.2:  $\overline{\text{INT0}}$ （外部中断 0 输入）；
- P3.3:  $\overline{\text{INT1}}$ （外部中断 1 输入）；
- P3.4: T0（定时器 0 的外部输入）；
- P3.5: T1（定时器 1 的外部输入）；
- P3.6:  $\overline{\text{WR}}$ （片外数据存储器“写”选通控制输出）；
- P3.7:  $\overline{\text{RD}}$ （片外数据存储器“读”选通控制输出）。

第三章 80C51 的指令系统			
1.80C51 的寻址方式			
表 3.1 寻址方式所对应的寄存器和存储空间			
序号	寻址方式		寄存器或存储空间
1	基本方式	寄存器寻址	寄存器 R0~R7, A、AB、DPTR 和 C（布尔累加器）
2		直接寻址	片内 RAM 低 128 字节、SFR
3		寄存器间接寻址	片内 RAM（@R <sub>0</sub> , @R <sub>1</sub> , SP） 片外 RAM（@R <sub>0</sub> , @R <sub>1</sub> , @DPTR）
4		立即寻址	ROM
5	扩展方式	变址寻址	ROM（@A+DPTR, @A+PC）
6		相对寻址	ROM（PC 当前值的-128~+127 字节）
7		位寻址	可寻址位（内部 RAM20H~2FH 单元的位和部分 SFR 的位）

2.特殊传送指令

表 3.3 特殊传送指令				
编号	指令分类	指令	机器码字节	机器周期数
1	ROM 查表	MOVC A, @A+DPTR	93H	2
2		MOVC A, @A+PC	83H	2
3	读片外 RAM	MOVX A, @DPTR	E0H	2
4		MOVX A, @Ri	E2H(~E3H)	2
5	写片外 RAM	MOVX@DPTR, A	F0H	2
6		MOVX@Ri, A	F2H(~F3H)	2
7	堆栈操作	PUSH direct	C0H	2

			direct	
8		POP direct	D0H	2
			direct	

● 堆栈操作

在 80C51 单片机中，堆栈的生长方向是向上的。入栈操作时，先将 SP+1，然后将指令指定的直接地址单元的内容存入 SP 指向的单元；出栈操作时，将 SP 指向的单元内容传送到指令指定的直接地址单元，然后 SP-1。  
系统复位时，SP 的内容位 07H。SP 的值越小，堆栈深度越深。

PUSH direct: SP ← (SP) + 1, (SP) ← (direct)

POP direct: direct ← ((SP)), SP ← (SP) - 1

一般在中断、子程序调用时发生堆栈操作

● MOV、MOVB、MOVC 区别

MOV: 单片机内部数据传递；MOVB: 单片机与片外 RAM 和 I/O 口电路的数据传递；MOVC: ROM 的查表

3.逻辑运算与循环类指令

逻辑与:

编号	指令分类	指令
1	逻辑与	ANL direct, A
2		ANL direct, #data
3		ANL A, Rn
4		ANL A, direct
5		ANL A, @Ri
6		ANL A, #data

前两条把源操作数与直接地址指示内容相与；  
后四条是把源操作数与累加器 A 的内容相与，送入 A  
逻辑或:

编号	指令分类	指令
1	逻辑或	ORL direct, A
2		ORL direct, #data
3		ORL A, Rn
4		ORL A, direct
5		ORL A, @Ri
6		ORL A, #data

逻辑异或:

编号	指令分类	指令
1	逻辑异或	XRL direct, A
2		XRL direct, #data
3		XRL A, Rn
4		XRL A, direct
5		XRL A, @Ri
6		XRL A, #data

累加器清 0 和取反		
CLR	A	清 0
CPL	A	取反

累加器循环移位	
RR	右移
RRC	
RL	左移
RLC	

4.控制转移类指令  
短跳转：AJMP ；长跳转：LJMP ；相对跳转：SJMP 。由于 LJMP 指令提供的是 16 位地址，因此程序可以转向 64KB 的程序存储器地址空间的任何单元。

累加器判 0 转移：  
JZ rel: (A) = 0, 则 PC ← PC + rel;  
JNZ rel: (A) ≠ 0, 则 PC ← PC + rel

比较不相等转移(目的字节与源字节不相等则转移):  
CJNE A, direct rel  
A  
CJNE Rn, #datarel  
@Ri

DJNZ	Rn, rel
DJNZ	direct rel



将操作数的循环控制单元的内容减 1，并判断是否为 0.若不为 0，一直循环。若为 0，程序往下执行。  
调用：  
ACALL 短调用；LCALL 长调用  
返回：  
RET 子程序返回指令；RETI 中断服务子程序返回指令  
CPL 取反

## 第四章

1.查表程序  
有一变量存放在片内 RAM 的 20H 单元，其取值范围为 00H~05H 。要求编制一段程序，根据变量值求平方值，并存入片内 RAM 的 21H 单元。  
程序如下：  
ORG 1000H  
START : MOV DPTR, #2000H  
MOV A, 20H  
MOVC A, @A+DPTR  
MOV 21H, A



```

        SJMP  $
        ORG  2000H
TABLE:  DB   00,01,04,09,16,25
        END
    
```

## 2.双字节无符号数加法

设被加数存放在内部 RAM 的 51H、50H 单元，加数存放在内部 RAM 的 61H、60H 单元，相加的结果存放在内部 RAM 的 51H、50H 单元，进位存放在位寻址区的 00H 位中，实现此功能的程序段如下：

```

MOV    R0, #50H  ; 被加数的低字节地址
MOV    R1, #60H  ; 加数的低字节地址
MOV    A, @R0    ; 取被加数低字节
ADD    A, @R1    ; 加上加数低字节
MOV    @R0, A    ; 保存低字节相加结果
INC    R0        ; 指向被加数高字节
INC    R1        ; 指向加数高字节
MOV    A, @R0    ; 取被加数高字节
ADDC   A, @R1    ; 加上加数高字节（带进位加）
MOV    @R0, A    ; 保存高字节相加结果
MOV    00H, C    ; 保存进位
    
```

## 3.双分支程序

设变量 x 以补码的形式存放在片内 RAM 的 30H 单元，变量 y 与 x 的关系是：当  $x > 0$  时， $y = x$ ；当  $x = 0$  时， $y = 20H$ ；当  $x < 0$  时， $y = x + 5$ 。编制程序，根据 x 的大小求 y 并返回原单元。程序段如下：

```

START: MOV  A, 30H
        JZ   NEXT
        ANL  A, #80H ; 判断符号位
        JZ   LP
        MOV  A, #05H
        ADD  A, 30H
        MOV  30H, A
        SJMP LP
NEXT:   MOV  30H, #20H
        LP:  SJMP $
    
```

## 4.多分支程序

根据 R7 的内容转向相应的处理程序。

设 R7 的内容为 0~N，对应的处理程序的入口地址分别为 PP0~PPN。程序段如下：

```

START: MOV  DPTR, #TAB ; 置分支入口地址表首址
        MOV  A, R7
        ADD  A, R7
        MOV  R3, A
        MOVC A, @A+DPTR ; 取高位地址
        XCH  A, R3
        INC  A
        MOVC A, @A+DPTR ;取低位地址
    
```

```

MOV    DPL, A ; 处理程序入口地址低 8 位送 DPL
MOV    DPH, R3 ; 处理程序入口地址高 8 位送 DPH
CLR    A
JMP    @A+DPTR
TAB:   DW    PP0
        DW    PP1
        ...
        DW    PPN
    
```

## 5.循环程序

先执行后判断；先判断后执行

### 50ms 延时程序

若晶振频率为 12MHZ，则一个机器周期为  $1\mu s$ ，执行一条 DJNZ 指令需要 2 个机器周期，即  $2\mu s$ 。采用循环计数法延时，循环次数可通过计算获得，并选择先执行后判断的循环结构。程序段如下：

```

DEL:: MOV  R7, #200 ;  $1\mu s$ 
DEL1: MOV  R6, #123 ;  $1\mu s$ 
        NOP ;  $1\mu s$ 
DEL2: DJNZ R6, DEL2 ;  $2\mu s$  共  $(2*123)\mu s$ 
        DJNZ R7, DEL1 ;  $2\mu s$  共  $[(2*123+2+2)*200+1]\mu s$ , 即 50.001ms
        RET
    
```

## 6.子程序及其调用

### a.现场保护与恢复的结构

- 在主程序实现
- 在子程序中实现

### b.子程序调用时，参数的传递方法

- 利用累加器或寄存器
- 利用存储器
- 利用堆栈操作

## 7.常用程序举例

### a.多字节无符号数的加法

设两个 N 字节的无符号数分别存放在内部 RAM 中以 DATA1 和 DATA2 开始的单元中。相加后的结果要求存放在 DATA2 数据区中。程序段如下：

```

MOV    R0, #DATA1
MOV    R1, #DATA2
MOV    R7, #N ; 置字节数
CLR    C
LOOP: MOV  A, @R0
        ADDC A, @R1 ; 求和
        MOV  @R1, A ; 存结果
        INC  R0 ; 修改指针
        INC  R1
        DJNZ R7, LOOP
    
```

### b.多字节无符号数的减法

设两个 N 字节的无符号数分别存放在内部 RAM 中以 DATA1 和 DATA2 开始的单

元中。相减后的结果要求存放在 DATA2 数据区中。程序段如下：

```
MOV    R0 , #DATA1
MOV    R1 , #DATA2
MOV    R7 , #N    ; 置字节数
CLR    C
LOOP:MOV A    , @R0
      SUBB A , @R1    ; 求差
      MOV @R1 , A    ; 存结果
      INC R0 ; 修改指针
      INC R1
      DJNZ R7 , LOOP
```

c.十六进制数与 ASCII 码间的转换

当十六进制数在 0~9 之间时，其对应的 ASCII 码值为该十六进制数加 30H；当十六进制数在 A~F 之间时，其对应的 ASCII 码值为该十六进制数加 37H

d. BCD 码与二进制数之间的转换

双字节二进制数转换成 BCD 码

设 (R2R3)为双字节二进制数，(R4R5R6 )为转换完的压缩型 BCD 码。程序段如下：

```
DCDTH  : CLR    A
          MOV    R4 , A ; R4 清 0
          MOV    R5 , A ; R5 清 0
          MOV    R6 , A ; R6 清 0
          MOV    R7 , #16 ; 计数初值
LOOP   : CLR    C
          MOV    A , R3
          RLC    A
          MOV    R3 , A ; R3 左移一位并送回
          MOV    A , R2
          RLC    A
          MOV    R2 , A ; R2 左移一位并送回
          MOV    A , R6
          ADDC   A , R6
          DA     A
          MOV    R6 , A ; (R6)乘 2 并调整后送回
          MOV    A , R5
          ADDC   A , R5
          DA     A
          MOV    R5 , A ; (R5)乘 2 并调整后送回
          MOV    A , R4
          ADDC   A , R4
          DA     A
          MOV    R4 , A ; (R4)乘 2 并调整后送回
          DJNZ   R7 , LOOP
```



第五章 80C51 的中断系统及定时/计数器

1.80C51 的结构

80C51 系列单片机的中断系统有 5 个中断源，2 个优先级，可实现二级服务中断嵌套。

表 5.1 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 T1	0023H	低

注：R1、T1 必须由软件清除。另外，所有能产生中断的标志位均可由软件置 1 或清 0，由此可以获得与硬件使之置 1 或清 0 同样的效果。

2.中断响应条件

CPU 响应中断的条件是：第一，中断源有中断请求；第二，此中断源的中断允许位为 1；第三是 CPU 开中断(即 EA=1)。同时满足这 3 个条件时，CPU 才可能响应中断。

3.中断返回（使用 RETI 指令）

RETI 指令能使 CPU 结束中断服务程序的执行，返回到曾经被中断过的程序处，继续执行主程序。

功能：

- 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：

- 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。电平触发方式适合于外部中断输入以低电平输入且在中断服务程序中能清除外部中断的情况。
- 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 中将所存一个逻辑 1。边沿触发方式适合于以负脉冲形式输入的外部中断请求。

4.80C51 的定时/计数器

80C51 单片机片内集成有两个可编程的定时/计数器：T0 和 T1，它们既可以工作于定时模式，也可以工作于外部事件计数模式，此外，T1 还可以作为串行口的波特率发生器。

5.定时/计数器的控制

TMOD 用于设置其工作方式；TCON 用于控制其启动和中断申请。

定时模式 C/ $\overline{\text{T}}$ =0；计数模式 C/ $\overline{\text{T}}$ =1

6.定时/计数器的定时方式和计数方式的区别

- 定时方式的计数脉冲源自内部的时钟脉冲，每个机器周期寄存器的值+1，计数频率为震荡频率的 1/12；
- 计数方式的计数脉冲源自相应的外部输入引脚 T0 或 T1，每个下降沿寄存器值+1，最高计数频率为振荡频率的 1/24.

7.定时/计数器的工作方式

80C51 单片机定时/计数器 T0 有 4 种工作方式（方式 0、1、2、3），T1 有 3

种工作方式（方式 0、1、2）。

a. 方式 0（M1M0=00）

13 位计数器，由 T0 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。

计数初值公式： $X=2^{13}-N$

b. 方式 1（M1M0=01）

16 位计数器，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： $X=2^{16}-N$

c. 方式 2（M1M0=10）

方式 2 为自动重装初值的 8 位计数方式。TH0 为 8 位初值寄存器。当 TL0 计数溢出时，由硬件使 TF0 置 1，向

CPU 发出中断请求，并将 TH0 中的计数初值自动送入 TL0。TL0 从初值重新进行加 1 计数。周而复始，直至 TR0=0 才会停止。

计数个数与计数初值的关系： $X=2^8-N$

d. 方式 3（M1M0=11）

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。

方式 3 时，T0 分为两个独立的 8 位计数器 TL0 和 TH0，TL0 使用 T0 的所有

控制位：C/ $\bar{T}$ 、GATE、TR0、TF0 和

$\overline{INT0}$ 。当 TL0 计数溢出时，由硬件使 TF0 置 1，向 CPU 发出中断请求。

## 8.定时/计数器的应用举例

a.利用定时/计数器 T0 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 12MHZ。

解：（1）计算计数初值 X：

由于晶振为 12MHZ，所以机器周期  $T_{cy}$  为  $1\mu s$ 。所以：

$$N=t/T_{cy}=10*10^{-3}/1*10^{-6}=10000$$

$$X=2^{16}-N=65536-10000=55536=D8F0H$$

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/ $\bar{T}$ =0，可取方式控制字为 01H。

（3）程序清单如下

```

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
```

b.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。

解：采用定时 20ms，然后再计数 1、49 次的方法实现。

a. T0 工作在定时方式 1 时，控制字 TMOD 配置：

M1M0=01，GATE=0，C/T=0，可取方式控制字为 01H。

b. 计算计数初值 X：

晶振为 12MHZ，所以机器周期  $T_{cy}$  为  $1\mu s$ 。所以：

$$N=t/T_{cy}=20*10^{-3}/1*10^{-6}=20000$$

$$X=2^{16}-N=65536-20000=45536=4E20H$$

即应将 4EH 送入 TH1 中，20H 送入 TL1 中。

c. 实现程序如下：

```
ORG    0000H
AJMP   MAIN ; 跳转到主程序
ORG    0030H
MAIN:  MOV    TMOD, #01H ; 置 T0 工作于方式 1
        MOV    TH0, #4EH ; 装入循环计数初值
        MOV    TL0, #20H ; 首次计数值
LP0:   SETB    P1.0
        ACALL   NT0
        CLR     P1.0
        MOV     R7, #49 ; 计数 49 次
LP1:   ACALL   NT0
        DJNZ    R7, LP1
        AJMP    LP0
NT0:   MOV     TH0, #4EH
        MOV     TL0, #20H
        SETB    TR0
        JNB     TF0, $
        CLR     TR0
        CLR     TF0
        RET
END
```

c.要求从 P1.1 引脚输出 1000HZ 方波，晶振频率为 12MHZ。试设计程序。

解：采用 T0 实现。

① T0 工作在定时方式 1 时，控制字 TMOD 配置：

M1M0=01，GATE=0，C/T=0，可取方式控制字为 01H。

② 计算计数初值：

晶振为 12MHZ，所以机器周期  $T_{cy}$  为  $1\mu s$ 。  $1/1000=1*10^{-3}$ 。所以：

$$N=t/T_{cy}=0.5*10^{-3}/1*10^{-6}=500$$



$$X=2^{16}-N=65536-500=65036=FE0CH$$

即应将 FEH 送入 TH0 中，0CH 送入 TL0 中。

③ 实现程序如下：

```
ORG      0000H
AJMP     MAIN ; 跳转到主程序
ORG      000BH ; T0 的中断入口地址
LJMP     DVT0
ORG      0030H
MAIN: MOV  TMOD, #01H ; 置 T0 工作于方式 1
MOV      TH0, #0FEH ; 装入循环计数初值
MOV      TL0, #0CH ; 首次计数值
SETB     ET0 ; T0 开中断
SETB     EA ; CPU 开中断
SETB     TR0 ; 启动 T0
SJMP     $ ; 等待中断
DVT0: CPL  P1.1
MOV      TH0, #0FEH
MOV      TL0, #0CH
SETB     TR0
RETI
END
```



第六章

1.80C51 串行口的控制寄存器

80C51 系列单片机有一个可编程的全双工串行通信口。

80C51 串行口的控制寄存器 { 特殊功能寄存器SCON(98H)  
电源控制寄存器PCON(97H)

SM0 和 SM1：串行口工作方式选择位，可选择 4 种工作方式，如表 6.2 所示。

表 6.2 串行口的工作方式

SM0	SM1	方式	说明	波特率
0	0	0	移位寄存器	$f_{osc}/12$
0	1	1	10 位异步收发器(8 位数据)	可变
1	2	2	11 位异步收发器(9 位数据)	$f_{osc}/64$ 或 $f_{osc}/12$
1	3	3	11 位异步收发器(9 位数据)	可变

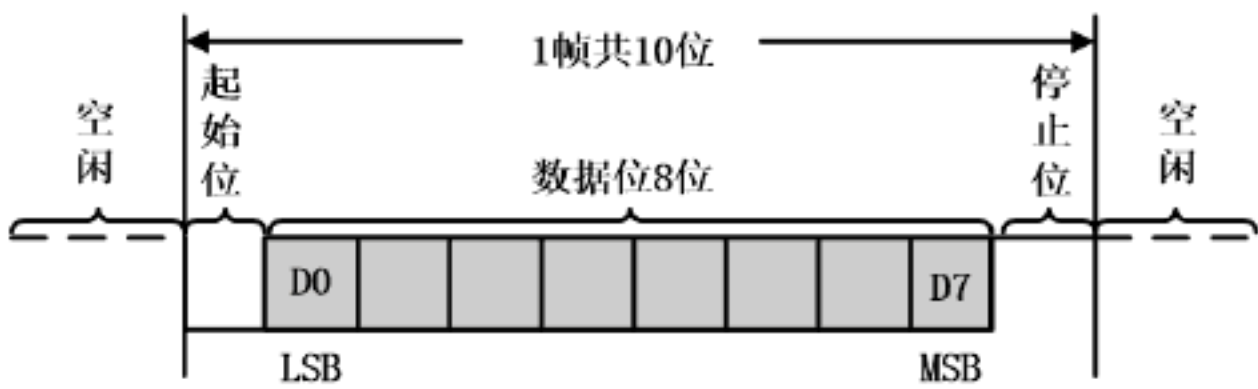
2.80C51 串行口的工作方式

a. 方式 0

方式 0 时，串行口为同步移位寄存器的输入输出方式。主要用于扩展并行输入或输出口。数据由 RXD（P3.0）引脚输入或输出，同步移位脉冲由 TXD（P3.1）引脚输出。发送和接收均为 8 位数据，低位在先，高位在后。波特率固定为  $f_{osc}/12$ 。

b. 方式 1（必须会）

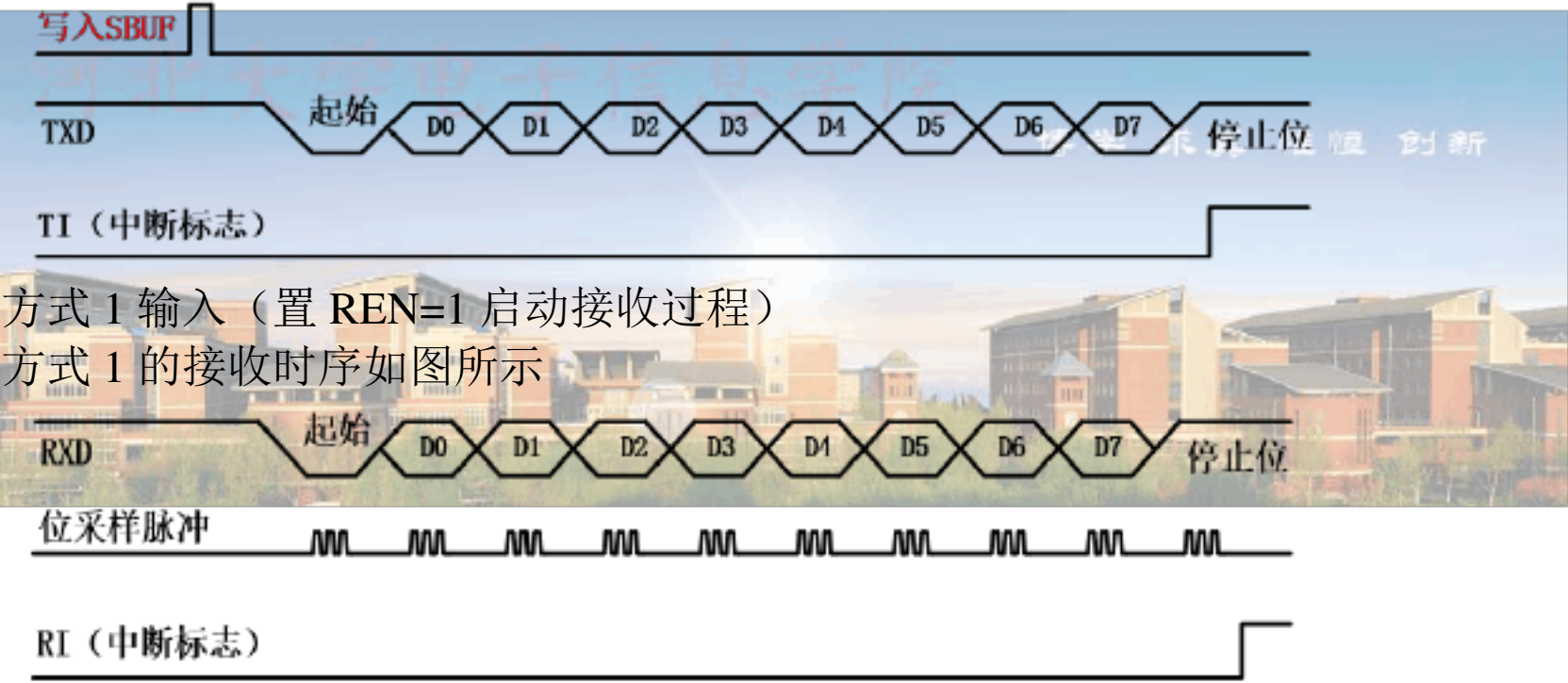
串行口定义为方式 1 时，是 10 位数据的异步通信口。TXD 为数据发送引脚，RXD 为数据接收引脚。其中 1 位起始位，8 位数据位，1 位停止位。



### 方式 1 输出

当执行一条写 SBUF 的指令时，就启动了串行口发送过程。在发送移位时钟（由波特率确定）的同步下，从 TXD 的引脚先送出起始位，然后是 8 位数据位，最后停止位。一帧数据发送完后，中断标志 TI 置 1。方式 1 的波特率由定时器 T1 的溢出率决定。

方式 1 的发送时序如图所示



### 方式 1 输入（置 REN=1 启动接收过程）

方式 1 的接收时序如图所示

