附录三　MCS-51 系统结构与指令系统

第一章 概述与历史

随着半导体技术的发展，能够在一个芯片上制造出上百万个晶体管，于是出现了由大规模集成电路组成的中央处理器，以及大容量的集成电路半导体存储器，通用和专用的输入输出接口电路，由这些大规模集成电路组成各种类型的微型计算机（Microcomputer）。

从七十年代中期开始，出现了以一个大规模集成电路为主组成的微型计算机：单片微型计算机，简称为单片机（single chip microcomputer）。由于单片机面向控制，特别适合于控制领域，因而又名微控制器（Microcontroller）。

从 1976 年起，Intel 公司开始生产 MCS 系列单片机。1976 年推出 MCS－48 系列 8 位单片机，其典型产品为 8048。在 1980 年推出 MCS－51 系列单片机，这是一个高性能的 8 位单片机，在各个方面性能都得到大大增强。MCS－51 系列的典型产品为 8051/8031，其内部资源为：

8 位 CPU
128 字节 RAM 数据存贮器
32 位 I/O 线
两个 16 位的定时器/计数器
一个全双工异步串行口
五个中断源，两个中断优先级
64K 程序存储器空间
64K 外部数据存贮器空间
片内振荡器，频率范围为 1.2MHz 到 12MHz

Intel 公司于 1984 年推出 16 位 MCS－96 系列单片机，它的典型产品为 8397BH。

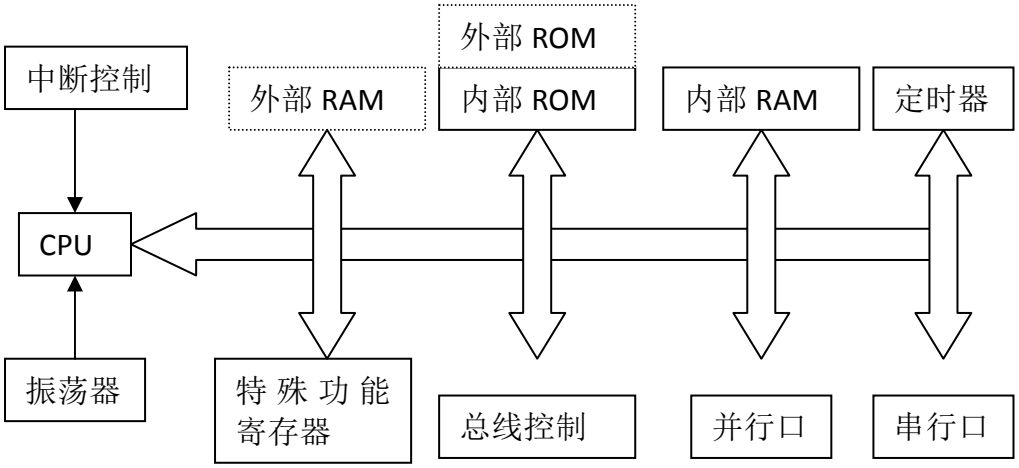
虽然单片机产生已经很长时间，但由于在智能仪表、家用电器等实时控制和分布式控制领域需要这种结构简单，控制灵活，并不需要很快的计算速度和存储容量的计算机，因此在计算机控制等领域单片机仍然具有不可替代的地位。

第二章 MCS－51 系列单片机系统结构

自从 MCS－51 单片机问世以来，该系列已经发展到几十种型号，它们都具有相同的内部核心结构和基本相同的外部引脚，其中以 8031 为代表产品。我们以 MCS－51 表示这些以 8031 为内核的单片机。

2.1 总体结构

MCS－51 是一个集成了 CPU 和常用的外围器件的芯片，结构简图如下。



2.2 中央处理器 CPU

2.2.1 运算器

运算器主要包括 ALU、累加器、B 寄存器、程序状态寄存器 PSW，十进制调整电路和布尔处理器等。8031 可以执行的运算功能包括：

算术运算：加减乘除；

逻辑运算：与、或、异或、非；

布尔运算：置 1、置 0、取反；

增量、减量运算；

十进制调整；

数据传送。

2.2.2 控制器

控制器包括时钟发生器、定时控制逻辑、复位电路、指令译码器、指令寄存器、程序计数器 PC、数据指针 DPTR、堆栈指针等部件。

① 时钟电路

时钟电路是计算机的心脏，它控制着计算机的工作节奏。MCS－51 单片机内有一个由反相器构成的振荡器，引脚 XTAL1 是反相器的输入端，XTAL2 是反相器的输出端。

MCS－51 的时钟可以利用其内部的振荡器产生，只要在引脚 XTAL1 和 XTAL2 上外接定时反馈电路，内部振荡器便自激振荡，产生时钟输出到内部的定时控制逻辑。定时反馈电路一般为石英晶振和电容组成的并联电路。振荡频率主要由晶振的频率决定，一般在 0.5MHz 到 16MHz 之间，典型值为 12MHz 和 11.0592MHz。电容的主要作用是帮助振荡器起振，对振荡频率也有微小的影响。典型值为 30pF。MCS－51 的工作时钟也可以由外部提供。

② 复位电路

计算机在启动时都需要复位，使 CPU 和系统中的其他部件都处于一个确定的初始状态，并从这个状态开始工作。

MCS－51 的 RST 引脚就是复位信号施密特输入端。当 RST 端保持 16 个时钟周期以上的高电平时，就可以使 CPU 复位，内部寄存器处于确定的初始状态，当 RST 为低电平后，退出复位，系统从初始状态开始工作。程序从 0000H 处开始执行，一般须在此处放一条长转移指令转移的真正的程序入口（原因见中断一节）。

复位后的内部寄存器状态			
寄存器	内容	寄存器	内容
PC	0000H	TMOD	00H
ACC	00H	TCON	00H
B	00H	TH0	00H
PSW	00H	TL0	00H
SP	07H	TH1	00H
DPTR	0000H	TL1	00H
P0－P3	0FFH	SCON	00H
IP	00H	SBUF	不定
IE	00H	PC	0000H

在 RST 引脚外只要接一个电阻和电容构成的复位电路就能实现上电自动复位。对应 CHMOS 器件只要接一个电容即可。复位电路在上电时的高电平时间应包括电源电压上升时间（约 10ms）和内部复位时间（12 个时钟周期），为可靠起见，一般为 20ms，这个时间取决于电路的时间常数 RC，当时钟周期为 12MHz 时，典型值为 R=10K，C=10μF。

除上电复位以外，常常需要手工复位，将一个常开按钮并联于上电复位电路，开关闭合时就能实现器件复位。

在单片机应用系统中，单片机的外围电路也需要和单片机同步复位，保证单片机对外围电路有效初始化。由于单片机的 RST 端是施密特输入，而一般接口电路是 TTL 电平输入，需要把 RST 电路信号进行整形后转换为系统复位信号输出到各外围电路。同时应注意有些器件的复位引脚是低电平有效的，需要逻辑转换。

③ CPU定时

CPU 取出一条指令到该指令执行完成的时间为指令周期。指令周期是以机器周期为单位的。MCS－51 的一个机器周期由六个状态构成（S1－S6），每一个状态由两个时钟周期构成（P1，P2）。因此一个机器周期由 12 个时钟周期组成，若晶振频率为 12MHz，则一个机器周期为 1 微秒。

大多数 MCS－51 的指令执行时间为一个机器周期，小部分指令为 2 个机器周期，只有乘除法需要 4 个机器周期。指令需要的指令周期在指令表中有详细说明。通过计算指令的执行周期可以精确控制程序的执行时间。

2.3 存储器组织

MCS－51 有五个独立的寻址空间。

64K 字节程序存储器空间（0－0FFFFH）

64K 外部数据存储器空间（0－0FFFFH）

256 字节内部 RAM 空间（0－0FFH）

256 位寻址空间（0－0FFH）

工作寄存器区

这些寻址空间中，工作寄存器区重合在内部 RAM 的前 128 字节空间中，后 128 字节是内部特殊功能寄存器（SFR）空间，位寻址区的前 128 个地址重合在内部 RAM 中，后 128 个地址重合在 SFR 中的一部分寄存器中。MCS－51 系列中不同型号的单片机，特殊功能寄存器的定义和使用不完全一致。

2.3.1 程序存储器

MCS－51 系列的单片机有些型号在内部有 4K 字节的程序存储器（如 8051 和 8751），有些有 8K 或 16K 内部程序存储器（如 83C51），有些则没有（如 8031）。不论哪种型号的单片机，都可以通过外部存储器扩展到 64K 的程序空间。

对于内部有程序存储器的单片机，如果引脚 EA 接 VCC，则程序计数器 PC 的值在内部寻址空间范围之内时，CPU 从内部的程序存储器读取指令；否则访问外部的程序存储器。如果 EA 接地，则总是从外部程序存储器读取指令。对于内部没有程序存储器的单片机，EA 必须接地，只能从外部读取指令。

单片机外部扩展的程序存储器一般为 EPROM（紫外线可擦除电可编程的只读存储器），引脚 PSEN（低有效）输出外部程序存储器的选通信号。

2.3.2 外部数据存储器

MCS－51 的外部程序存储器和数据存储器是分离的，对于 CPU 来说，程序存储器是只读的，通过 PSEN 信号控制；数据存储器是可读写的，通过 RD 和 WR 信号控制。在程序存储器中也可以固化一些数据（如预先放置的表格），但读取这些数据的指令与读取数据存储器的指令也是不同的。这一点是和一般的微机系统不一样的。这是由于单片机应用的领域，程序都是固定的，为了防止可能的对程序的破坏而采取的一种保护措施。

在一般情况下，单片机都不需要很多的数据存储器，这时内部的 128 字节 RAM 基本是够用的。在有些情况下，需要大量的数据存储器，这时可以外接最大直至 64K 的外部数据存储器。

外部数据存储器的寻址空间也可以直接作为扩展 I/O 口的寻址空间使用，对此 CPU 使用相同的操作指令来读写。在简单的扩展中，常使用位选的方式来简单扩展端口。

2.3.3 内部 RAM 数据存储器

MCS－51 单片机的内部 RAM 的寻址空间为 256 字节，实际提供给用户的通用 RAM 空间一般为前 128 字节。内部 RAM 的不同地址区域从功能和用途可以区分为：内部工作寄存器区，位寻址区，堆栈和通用数据存储区。

内部 RAM 的 0－1FH 为 4 组工作寄存器区，每个区有 8 个工作寄存器（R0－R7）。在同一时刻，只能使用一组工作寄存器，这是通过程序状态字 PSW 的地 3，4 位来控制的。例如当此两位为 00 时，使用第 0 组工作寄存器，对应于 00H 到 07H 的内部 RAM 空间。也就是说，这时指令中使用 R0 与直接使用 00 单元是等价的，不过使用工作寄存器的指令简单，且执行快。

程序通过修改 PSW 的这两位，就可以选择一个工作寄存器区，这个特性提高了 MCS－51 上下文切换的速度，对于提高 CPU 响应中断的速度和现场保护与恢复是很有利的。

内部 RAM 的 20H－2FH 为位寻址区域，这 16 个单元的每一位都对应一个位地址，占据位地址空间的 0－7FH。每一位都可以独立置位、清除、取反等操作，

也可以作为条件转移的条件使用。这个特性对应单片机应用于开关量控制和判断中是很有用处的。在 SFR 中的地址为 8 的整数倍的寄存器也可以位寻址，这样可以方便控制 SFR 中的某一位。

在中断和子程序调用中都需要堆栈。MCS－51 的堆栈理论上可以设置在内部 RAM 的任意区域，但由于 0－1FH 和 20－2FH 区域有上面说的特殊功能，因此一般设置在 30H 以后。堆栈的栈顶位置由堆栈指针寄存器 SP 指出，SP 在复位后为 07H，一般在程序初始化阶段将其调整到需要的位置。堆栈是向上增长的，也就是从当前栈顶位置向地址增加的位置扩展，而这种扩展没有限制，在实际编程中要注意留出足够的区域以免发生冲突。

在内部 RAM 中，所有的单元都可以作为通用的数据存储器使用，存放输入的数据或计算的中间结果等。如果程序不使用其他组的工作寄存器、位寻址单元等，这些地址都可以自由使用。

2.3.4 特殊功能寄存器 SFR

MCS-51 内部的 I/O 口锁存器及定时器、串行口、中断等各种控制寄存器和状态寄存器等都称为特殊功能寄存器。它们控制着各种 CPU 周边设备，保存着它们的状态。SFR 离散地分布在 80H－0FFH 的特殊功能寄存器地址空间中。不同型号的单片机内部 I/O 功能有所不同，实际存在的特殊功能寄存器数量差别较大。特殊功能寄存器空间中有些单元是空着的，在有些控制寄存器中的某些位也没有定义。这些单元是为了以后新型的单片机保留的，为了软件与以后的新型号兼容，用户程序不要对这些空单元进行操作。

特殊功能寄存器中地址为 8 的倍数的寄存器可以进行位寻址，它们的位地址位于 80H 到 0FFH 的地址空间。它们的位地址恰好等于 SFR 的地址加上位的次序（如 PSW 第 7 位 CY 的位地址是 PSW 的地址 0D0H 加上 7，即 0D7H）。

2.4 输入输出接口

MCS－51 单片机有 4 个双向 8 位输入输出口 P0－P3 口。每一个口由口锁存器、输出缓冲器、输入缓冲器组成。这些位的复位后的状态都是 1。其中 P0 口是三态双向 I/O 口，P1、P2、P3 口内部有提升电阻，称为准双向口。

P1 口为准双向口。它的每一位都可以分别定义为输入或输出线。作为输出时，程序直接向 P1 口的该位输出 0 或 1，外部电路就锁存为低或高电平。作为输入端使用时，必须向其输出 1，读取时，此位的状态就由外电路决定。

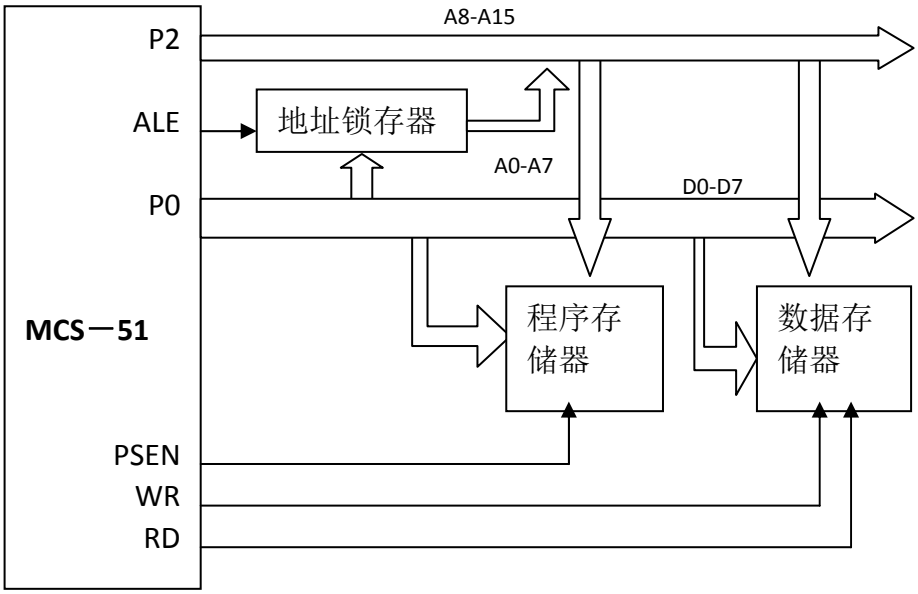
P3 口为多功能口，用户可以使用系统预设的功能，这时必须向其输出 1；也可以作为普通 I/O 口使用，这时使用方法与 P1 口一致。当系统定义的功能如下表所示：

P3 口系统功能定义		
P3.0	RXD	串行输入口
P3.1	TXD	串行输出口
P3.2	INT0	外部中断 0 输入
P3.3	INT1	外部中断 1 输入
P3.4	T0	计数器 0 外部输入
P3.5	T1	计数器 1 外部输入
P3.6	WR	外部数据存储器写

P3.7	RD	外部数据存储器读

P2 口也有两种功能，一种功能是作为外接程序/数据存储器的地址高位线输出，一种是普通的 I/O 口。对于内部没有程序存储器或需要外接的应用，P2 口就不能作为输入输出口使用。

P0 口为三态双向 I/O 口。和 P2 口类似，当外接程序/数据存储器时，必须作为地址/数据口使用，否则也可以作为普通输入输出口使用。当 P0 口作为地址/数据口使用时，在机器周期的前半时刻输出程序地址（或外部数据地址）的地址低 8 位，后半部分输入输出数据。这是通过 ALE 线来控制的。典型的总线扩展方法如图所示。



2.5 定时器/计数器

由于在绝大多数的应用中都需要使用定时器/计数器，在 MCS－51 中设立了两个 16 位定时器/计数器 T0，T1，在某些型号中还有第三个定时器/计数器 T2。

和定时器/计数器相关的特殊功能寄存器有以下：TH0，TL0，TH1，TL1，TMOD，TCON。通过对它们的设置和读写就可以控制使用定时器/计数器。

TH0，TL0 为 T0 的 16 位计数的高 8 位和低 8 位；TH1，TL1 为 T1 的 16 位计数的高 8 位和低 8 位；TMOD 是方式寄存器；TCON 是状态和控制寄存器。

2.5.1 方式寄存器 TMOD

TMOD 的格式如下，高 4 位和低 4 位分别控制 T1 和 T0，它们的含义是完全相同的：

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/T	M1	M0	GATE	C/T	M1	M0
T1 方式字段				T0 方式字段			

工作方式选择位 M1M0 确定了定时器的工作方式，其具体含义稍后详述。

定时计数选择位 C/T＝0 时为定时方式。这时，以振荡器的 12 分频信号作为计数信号，也就是每一个机器周期定时器内的值加 1。若晶振为 12MHz，则定时器的计数频率为 1MHz。定时器从初值开始加 1 到溢出所需的时间是固定的，也

就是（$2^{\text{定时器最大位数}}$－定时器初值）×定时周期，所以称为定时方式。

C/T＝1 为外部计数方式，这种方式采用外部引脚（T0 为 P3.4，T1 为 P3.5）的输入脉冲作为计数脉冲。内部硬件在每一个机器周期的 S5P2 采样外部引脚的状态，当上一个周期为高电平，此周期为低电平，也就是发生负跳变时计数器加 1。外部计数事件的最高计数频率为晶振频率的 1/24，外部输入脉冲的高电平和低电平时间必须持续一个机器周期以上。计数器的用途通常是为了测量外部脉冲的周期、频率等。

门控位 GATE 为 1 时，定时器的计数受外部输入电平的控制（INT0 控制 T0，INT1 控制 T1），如外部控制电平为 0，定时器不计数。GATE 为 0 时定时器不受外部控制。这种方式可以使定时器的启动与停止受外部电路控制。

2.5.2 控制寄存器 TCON

控制寄存器 TCON 的高 4 位存放定时器的运行控制位和溢出标志位，低 4 位存放外部中断控制位（详见中断一节），格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

运行控制位 TR0 和 TR1 分别控制两个定时器是否允许计数。此位由软件置位和清零。当 GATE 为 0 时，TR 为 1 时允许计数；当 GATE 为 1 时，只有 TR 为 1 且外部控制电平为高时，定时器才计数。

溢出标志位 TF0 和 TF1 标志定时器是否已经计数到最大值，发生溢出。这一位可以由程序来查询和清零。如果 CPU 响应定时器中断则此位为 1 时，发生定时器中断，在中断响应时由硬件清零。

2.5.3 定时器的工作方式

定时器有 4 种工作方式，其中方式 3 只适用于 T0。下面详细讨论这 4 种工作方式。

如果方式控制字 M1M0 为 00，定时器处于方式 0。此时定时器为一 13 位的计数器，由 TL 的低 5 位和 TH 的 8 位构成，当 13 位计数溢出时置位 TF。

M1M0 为 01 时为方式 1，方式 1 与方式 0 的区别仅在于它是 16 的定时/计数器，由 TH 和 TL 联合构成。

方式 2，即 M1M0＝10 是自动恢复初值的 8 位计数器。这时 TL 作为 8 位计数器，TH 作为计数初值寄存器。这种方式下，定时器发生溢出后自动恢复为 TH 中的初值，不需 CPU 干预，因此定时更加精确。方式 2 经常作为串行口的波特率发生器使用（见下节）。

当 M1M0＝11 时为方式 3，它只适用于 T0，若 T1 被设置为方式 3 时，它将停止计数。在此种方式下，T0 被分为两个独立的 8 位计数器 TL0 和 TH0。TL0 使用 T0 的所有状态和控制位，可以作为定时器或计数器使用。TH0 被固定为一个 8 位定时器方式，使用 T1 的状态控制位 TR1 和 TF1。TR1 为 1 时允许 TH0 计数，当 TH0 计数溢出时置位标志 TF1。一般来说，当 T1 用作串行口的波特率发生器时（详见串行接口一节），T0 才在需要的情况下使用方式 3，以增加一个计数器，这时 T1 在计数溢出时不置位 TF1。

2.6 串行接口

CPU 与外部的接口方式有并行和串行两种。并行通信中信息传输线的位数和传送的数据位数相等，通信数据快，适合于近距离通信。串行通信仅需要一个数据线，一根地线（全双工需要发送和接收两根数据线），通信成本小，速度相对慢，适合于远距离通信。

串行通信有两种方式：同步通信和异步通信。

异步通信方式是按字符传送的，在字符前有低电平的起始位，在后面有高电平的停止位，字符之间没有固定的间隔长度。这种方式通信效率比较低，但实现比较简单。

同步通信是按数据块传送的，在数据块之前有特定的同步字符，后面有校验字符。同步通信的字符之间没有间隔，通信效率比较高。

在串行通信时，通信的双方必须事先商定每秒传送的数据位数，称之为波特率（Baud Rate），已经其他一些设置，如奇偶校验位、停止位等。

MCS－51 具有内置的全双工异步串行接口，可以同时发送和接收数据。在某些型号中还增加了通信功能更强的串行接口。

2.6.1 串行接口的构成和特性

在 MCS－51 的外部引脚中，由 P3.0（RXD）和 P3.1（TXD）作为串行接口的输入和输出线。在方式 0 时，RXD 作为数据输入输出线，TXD 为时钟输出线。

串行口的内部有数据接收缓冲器和数据发送缓冲器，数据接收缓冲器只能读出不能写入，数据发送缓冲器只能写入不能读出，这两个数据缓冲器都用符号 SBUF 表示，地址是 99H。

特殊功能寄存器 SCON 存放串行口的状态和控制信息，串行口用定时器 T1 作为波特率发生器，特殊功能寄存器 PCON 的最高位 SMOD 为串行口波特率的倍率控制位。

SCON 的地址为 98H，具有位寻址功能。其格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
SM1	SM0	SM2	REN	TB8	RB8	TI	RI

由 SM1 和 SM0 组成串行口的方式控制位，分别对应方式 0 到 3。SM2 位为允许多机通信控制位。REN 为允许串行接收位，由程序置位来允许接收。在方式 2 和 3 中，TB8 为发送的第 9 位数据，RB8 为接收的第 9 位数据。对于方式 1，RB8 是接收的停止位。TI 是发送中断标志，由硬件在方式 0 串行发送第 8 位结束时置位，或在其他方式下发送停止位时置位，必须由软件清零。RI 是接收中断标志，由硬件在方式 0 接收到第 8 位结束时置位，或在其他方式下接收到停止位时置位，必须由软件清零。

当 PCON 的最高位 SMOD 为 1 时使波特率加倍。

2.6.2 串行接口的工作方式

① 方式 0

方式 0 是外接移位寄存器的工作方式，用于扩展 I/O 接口。在这种方式下，数据由 RXD 串行输入或输出，由 TXD 端输出移位脉冲。波特率固定为振荡器的 1/12。

当用作输出时，一般外接串行输入并行输出移位寄存器（如 74LS164 等），

移位寄存器的串行数据端接 RXD，时钟端接 TXD。当 CPU 对 SBUF 写入数据后，就启动了串行口的输出。内部的定时逻辑从下一个机器周期开始，在每个机器周期的 S6P2 阶段使内部的移位寄存器右移 1 位送入 RXD，左边移入 0（也就是说最低位最先移出）；而 TXD 在 S3、S4、S5 期间为 0，S6、S1、S2 期间为 1。当 8 位都移出后，置位发送中断标志 TI，完成了一个字节的发送。

当用作输入时，一般外接并行输入串行输出移位寄存器（如 74LS166 等），移位寄存器的串行数据端接 RXD，时钟端接 TXD。当 REN＝1，RI＝0 时开始启动串行口接收。TXD 引脚在每个机器周期跳变一次，使外部数据通过 RXD 输入到内部移位寄存器中。当 8 次移位完成后，数据写入 SBUF，停止输出移位脉冲，置位接收中断标志 RI，完成一个字节的输入。

②方式 1

方式 1 是一种 8 位异步通信口方式。TXD 为数据输出线，RXD 为数据输入线。传送一个字节的格式为：前面一位低电平的起始位，中间从低位到高位的 8 位数据，后面一位高电平的停止位，共计 10 位。

在使用方式 1 输出时，当 CPU 向 SBUF 写入一个数据，就启动串行口发送。内部的移位寄存器根据波特率将数据按照帧格式不断输出到 TXD，当输出结束后，置位发送中断标志 TI

在 REN＝1 时，就允许串行口接收。接收器以所选波特率的 16 倍的速率采样 RXD 端的电平，当检测到 RXD 端出现负跳变时，就启动接收逻辑。接收器把接收一位的时间（即波特率的倒数）16 等分，在每位时间的第 7、8、9 三个状态检测 RXD 的电平，并取其中的至少两个相同的值作为数据输入，这样可以防止干扰。如果第一位接收到的不是 0，则起始位无效，重新进入等待起始位状态。9 次接收之后，将数据装入 SBUF 和 RB8（停止位），并置位 RI。如接收的最后一位不是 1 或 RI＝1，则本次接收的数据无效，重新等待。

③ 方式 2 和方式 3

在这两种方式下，串行口是一个 9 位的异步通信口。区别在于方式 2 的波特率固定为振荡器频率的 1/64 或 1/32，而方式 3 的波特率由 T1 的溢出速率决定。

在这两种方式下，一帧信息包括 11 位，数据位为 9 位，其余和方式 1 一致。附加的第 9 位数据最后发送或接收，它在发送时为 SCON 的 TB8，接收时存到 SCON 的 RB8。

在这两种方式下，发送的过程与方式 1 完全一致，只是多发送一位而已。接收的过程也大致类似，区别在于接收到数据后，除了判断 RI 标志外还要判断本机 SCON 的 SM2 标志。如果 SM2＝0 则接收数据（将接收的数据送入 SBUF 和 RB8，置位 RI）；当 SM2＝1 时，只有第 9 位数据为 1 时才接收数据，否则就丢失数据，不置位 RI，等待下一次接收。

2.6.3 波特率

在方式 0 下，波特率是振荡器频率除以 12。

在方式 2 下，波特率由振荡器的频率和 SMOD 决定。如果 SMOD 为 0，波特率等于振荡器频率的 1/64；当 SMOD 为 1 时，为振荡器频率的 1/32。

在方式 1 和方式 3 下，波特率由定时器 T1 决定（在某些型号中也可以由第三个定时器 T2 决定，这里不讨论）。由于定时器是可编程的，因此可选的波特率范围也比较大，这两种方式也比较常用。

这时，波特率由下式决定：

波特率＝2^{smod}*(T1 的溢出频率)/32

T1 作为波特率发生器时，通常应禁止 T1 中断，工作在定时方式，选择方式 2。这种方式是自动重装初始值的方式，可以保证定时的精确性。这种工作模式下，波特率的计算公式为：

波特率＝2^{smod}*振荡器频率/[32*12*(256-TH1)]

当振荡器频率为 11.0592MHz 时，对于常用的波特率如 1.2K，9.6K 等都可以计算出精确的定时器初值，因此这个频率常常被使用。

2.7 中断系统

现代的计算机都具有中断功能，能够对外界异步发生的事件进行及时处理。MCS－51 中不同型号的单片机的中断数量是不同的，典型（基本）的 8031 有 5 个中断源，具有两个中断优先级，可以实现二级中断服务程序嵌套。可以分别指定每一个中断源的中断级别和中断允许字。

2.7.1 中断源

8031 的 5 个中断源是 INT0，INT1 上的外部中断源，定时器 T0，T1 的溢出中断和串行口的发送接收中断等三个内部的中断源。

INT0 和 INT1 上输入的两个外部中断源和它们的触发方式锁存在特殊功能寄存器 TCON 的低四位，高四位是 T0 和 T1 的运行控制位和溢出标志位。它的结构见前面的说明。

IEn 是外部中断的请求源标志。当外部中断向 CPU 请求中断时置位，当 CPU 响应此中断由硬件清零。

ITn 是外部中断的触发方式控制位。当 IT＝0 时是电平触发方式，外部引脚出现低电平时产生中断请求。当 IT＝1 时是边沿触发方式。其详细区别后面叙述。

内部中断的中断请求标志分别在不同器件的控制寄存器中。TCON 的 TF0 和 TF1 标志 T0 和 T1 的溢出中断请求，这两个标志都由硬件置位和清零。串行口有两个中断原因：接收中断 RI 和发送中断 TI，它们逻辑或之后作为一个中断源。当串行口发送完一个字符后硬件置位 TI，接收到一个字符后硬件置位 RI。要注意当 CPU 响应串行口中断后，硬件并不将这两个标志清零，必须在中断服务程序中将其清零（这是为了在一个中断程序中能够区分两种中断原因）。

这些标志也可以由软件置位和清零，可以达到由程序模拟外部中断的目的。

2.7.2 中断控制

每一个中断源的开放和屏蔽是由中断允许寄存器 IE 控制的。IE 的字节地址是 0A8H，也是可以位寻址的特殊功能寄存器。其格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
EA	空	空	ES	ET1	EX1	ET0	EX0

EA 是整个 CPU 的中断允许标志。当 EA＝1 时，CPU 可以响应中断；当 EA＝0 时，屏蔽所有中断申请。

ES 是串行口中断允许位；ET1 和 ET0 是 T1 和 T0 的中断允许位；EX1 和 EX0 是外部中断 INT1 和 INT0 的中断允许位。这些位为 1 时允许响应对应的中断申请，为 0 时屏蔽之。

MCS－51 有两级中断优先级。一个正在执行的中断服务程序可以被高优先级

的中断申请所打断，但不能被相同优先级或低优先级的中断申请所中断，这些新的中断申请必须在现在执行的中断处理程序结束后返回主程序，并执行一条指令后才能得到响应。MCS－51 内部有不可存取的优先级状态触发器来标志 CPU 是否处于运行何种优先级的中断服务程序。

MCS－51 内部有一个中断优先级寄存器 IP（地址为 0B8H），定义各个中断源的优先级，格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
空	空	空	PS	PT1	PX1	PT0	PX0

从 D4 开始，按顺序分别为串行中断、T1、INT1、T0，INT0 的中断优先级标志，若为 1，表示此中断的优先级为高，否则为低。

当 CPU 接收到同样优先级的几个中断申请时，按照以下的次序决定中断响应的优先次序：INT0，T0，INT1，T1，串行中断。

CPU 复位后，IE，IP 的内容均为 0，禁止所有中断。由初始化程序对 IE、IP 编程，以根据需要决定中断的允许与优先级。

2.7.3 中断响应过程

CPU 在每一个机器周期的 S6 阶段采样所有允许的中断申请，如果发现有中断申请，并且没有下列条件的阻止，将在下一个机器周期响应激活的最高级中断申请。

- ◆ CPU正在处理相同的或更高级别的中断；
- ◆ 现在的机器周期不是所执行指令的最后一个机器周期（多周期指令）；
- ◆ 正在执行的指令不是中断返回指令RETI，或是对IE，IP的写操作指令；

CPU 响应中断时，先置位响应的优先级状态触发器（防止其他相同级别或低级别中断的执行）；将中断请求标志清零（TI 和 RI 除外）；把当前程序计数器 PC 的内容压入堆栈（但不保护 PSW），然后根据中断源的类型，将程序转移到对应的中断服务程序入口地址。各中断服务程序的入口地址是固定的，如下所示：

中断源	入口地址
INT0	0003H
T0	000BH
INT1	0013H
T1	001BH
串行口中断	0023H
定时器 T2（8052 等型号）	002BH

通常在中断的入口处，放置一条长跳转指令，转移到用户设计的中断处理程序中。

CPU 在执行中断处理程序一直遇到 RETI 指令为止，RETI 表示中断处理程序的结束。CPU 在执行这条指令时，把响应中断时所置的优先级状态触发器清零，从堆栈的顶部弹出两个字节到程序计数器 PC，CPU 重新执行被打断的主程序。

在中断处理过程中，必须由用户程序自己保护和恢复现场。

CPU 响应一个外部中断的时间取决于当前执行的指令，如果当前执行高级别的中断处理程序，就要等待到这个程序结束。如果中断响应没有被延迟，在 3－8 个周期内就可以响应这个中断。

2.7.4 外部中断触发方式选择

若外部中断定义为电平触发方式，外部中断申请触发器的状态随着 CPU 在每个机器周期采样到的外部中断输入线的电平变化而变化。外部要申请中断时，必须保持有效（低电平），直到 CPU 实际响应该中断为止，同时在中断服务程序返回之前，外部中断必须无效（高电平），否则 CPU 将再次响应该中断。所以电平触发方式适合于外部中断以低电平输入的，并且 CPU 能够清除外部中断输入源的状态。

外部中断定义为边沿触发方式，外部中断申请触发器能锁存外部中断输入线上的负跳变，即使 CPU 暂时不能响应，中断申请标志也不会丢失。在这种方式下，只有在相邻的两次采样中外部中断由高转为低，才置位中断申请触发器。因此外部输入的脉冲宽度必须至少保持 12 个时钟周期，才不会丢失中断。这种方式适合于以脉冲方式输入的外部中断请求。

习题和思考题

1. 什么是堆栈，堆栈应该设在什么地方。
2. 在8031的内部RAM中，那些单元适合作数据缓冲区。
3. 若晶振的频率是6MHz，使用T0产生1毫秒的定时，可以使用那些方式，分别写出各种方式的初始化程序。
4. 若晶振的频率是12MHz，如何测量外部20-1KHz之间的方波周期，若频率约为0.5MHz时，又应如何测量。讨论每种方式的误差范围。
5. MCS－51的中断处理程序能否放在64K范围的任意位置，若能，如何实现。
6. 串行口的0方式输出，能否外接多个74LS164，若不可以说明原因，若能，画出实现方式的逻辑框图，说明数据输出方法。

第三章 MCS－51 指令系统

计算机的指令系统是一套控制计算机操作的编码，称为机器语言。计算机只能识别和执行机器语言的指令。为了容易使人们所理解，便于记忆和使用，通常用符号指令来描述计算机的指令系统，称为汇编语言。将汇编语言翻译为机器语言的程序是汇编程序。这一章中将使用 Intel 的标准格式汇编指令来介绍 MCS－51 的指令系统。

3.1 指令格式

3.1.1 汇编指令

MCS－51 汇编指令由标号、操作码助记符字段和操作数字段组成。指令格式如下：

[标号：] 操作码　[操作数 1], [操作数 2], [操作数 3]

第一部分是标号，一般可以省略。标号必须以冒号结束。

第一部分是指令操作码助记符，它由二至五个英文字母组成。

第二部分是操作数，它以一个或几个空格与操作码分开，操作数之间由逗号分隔。根据指令不同，可能有一个、二个、三个或没有操作数。

3.1.2 伪指令

大多数汇编程序都提供许多伪指令供用户使用。大多数伪指令汇编时不产生机器语言指令，仅仅为汇编程序提供信息。最常用的伪指令有以下几条。

一、　定位伪指令

ORG xxxx

xxxx 为十进制或十六进制数。它指出了该伪指令后的指令的汇编地址，即生成的机器指令存放的存储器地址。在一个汇编语言源程序中允许使用多条伪指令，但一般情况下，指令区域不能重叠。

二、　定义字节伪指令

DB X1[, X2…]

X 为单字节数据，它为十进制或十六进制数，X 也可以是由单引号括起来的字符串。该伪指令把后面定义的数据送到目标程序存储器，通常用于定义一个常数表。

三、　字定义伪指令

DW W1[,W2…]

本指令与上一个伪指令作用相类似，用以存放双字节数据。通常可以定义一个地址表。

四、　汇编结束指令

END

该伪指令指出结束汇编，汇编程序不再对后面的内容进行处理。

五、　注释

汇编程序允许用户在源程序中添加注释。注释以分号开始。

3.1.3 常用的缩写符号

在下面描述 MCS－51 的指令系统时，我们使用下面的缩写符号，含义如下：

A　累加器 ACC

AB 累加器 ACC 和辅助寄存器 B 组成的寄存器对，构成一个 16 位 2 进制数。

Direct　直接地址

＃data 立即数，表示一个常数

@ 间接寻址

$　当前指令起始地址

(X) X寄存器内容

((X))　由 X 寄存器寻址的内部 RAM 存储器单元内容

注意本章中所说的寄存器一般指工作寄存器 R0－R7

在 MCS－51 汇编语言中，一般可以直接使用 SFR 的符号寻址这些寄存器，也可以使用类似 IE.2 的形式寻址 SFR 中的位地址。

3.2 寻址方式

寻址方式决定一条指令中存取参与运算的数据的方式。MCS－51 有五种寻址方式：寄存器寻址、直接寻址、寄存器间接寻址、立即寻址、和基址变址寻址。

3.2.1 寄存器寻址

由指令指出某一个寄存器的内容作为操作数称为寄存器寻址。一般在指令中有 3 位来表示所用的寄存器（R0－R7）。累加器 ACC、B、DPTR 和 C（布尔处理中的累加器）也可以使用寄存器寻址方式访问，只是对它们的寻址都隐含在相应的指令中。例如指令：

INC R0　；表示将 R0 的内容增一。

3.2.2 直接寻址

在指令中含有操作数的直接地址，该地址指出了参与运算的数据所在的字节单元地址（内部 RAM）或位地址。

直接寻址方式访问以下三种存储空间：

特殊功能寄存器（只能用直接寻址方式访问）；

内部数据存储器；

位地址空间；

例如指令：

ANL A, 41H；将 ACC 的内容和内部 RAM 中 41H 单元的内容相逻辑与，结果写回 ACC 中

本指令使用了两个操作数，A 是寄存器寻址，隐含在指令中；操作数 2 是直接寻址，出现在指令码中。

3.2.3 寄存器间接寻址

由指令指出某一个寄存器的内容作为操作数的地址，这种寻址方式叫做寄存器间接寻址。

寄存器间接寻址使用 R0，R1 作为内部 RAM 的寻址指针；使用 DPTR 作为外

部扩展数据存储器的寻址指针。使用符号@表示。

例如指令：

MOV @R0, 44H

若此时 R0 的内容为 32H，则将内部 RAM 中 44H 单元的内存传送到内部 RAM 中 32H 单元中。第一个操作数使用寄存器间接寻址。

3.2.4 立即寻址

在指令中包含操作数本身。用符号#表示，例如指令：

MOV A, #75H

将常数 75H 传送到累加器 A 中。

3.2.5 基址变址寻址

这种方式以 16 位的程序计数器 PC 或数据指针 DPTR 作为基寄存器，以 8 位的累加器 A 作为变址寄存器，基址寄存器和变址寄存器的内容相加作为 16 位的地址访问程序存储器。如：

MOVC A，@A+PC

MOVC A, @A+DPTR

3.2.6 转移指令寻址

在转移指令中，欲转移到的程序地址的寻址方式有 4 种。即长转移寻址、短转移寻址、相对转移寻址和基址变址寻址。

长转移寻址直接在指令中指出 16 位的目标地址，可以转移到程序存储器的任意位置，而不管当前指令的位置。类似于 8086 的段间转移指令。

短转移寻址可以转移到与下条指令存放在同一个 2K 区域内的一个地址。在指令执行时，首先将 PC 加 2，然后将 PC 的高 5 位和指令中指出的 11 位低位地址相连作为目标地址。类似于 8086 的段内转移指令。

相对转移寻址在指令中给出目标地址与当前 PC 的单字节有符号相对位移。执行时，首先将 PC 加 2，然后将指令中的有符号相对地址与 PC 相加计算出目标地址。因此，目标地址可以在本指令的前 128 个字节到后 127 个字节之间。

转移指令的基址变址寻址使用 DPTR 和 A，只有一条指令：

JMP @A+DPTR

这条指令将 DPTR 中的 16 位数与 A 中的 8 位无符号数相加得到目标地址，通常用于查表式的多地址跳转。

3.3 指令类型

MCS－51 汇编语言有 5 大类，33 种操作功能，42 种操作码助记符来描述。每一种功能可以有多种寻址方式，这样共有 111 种指令。按字节数分有 49 条单字节指令，45 条双字节指令和 17 条 3 字节指令。按指令执行时间分，有 64 条单周期指令、45 条双周期指令和 2 条 4 周期指令。

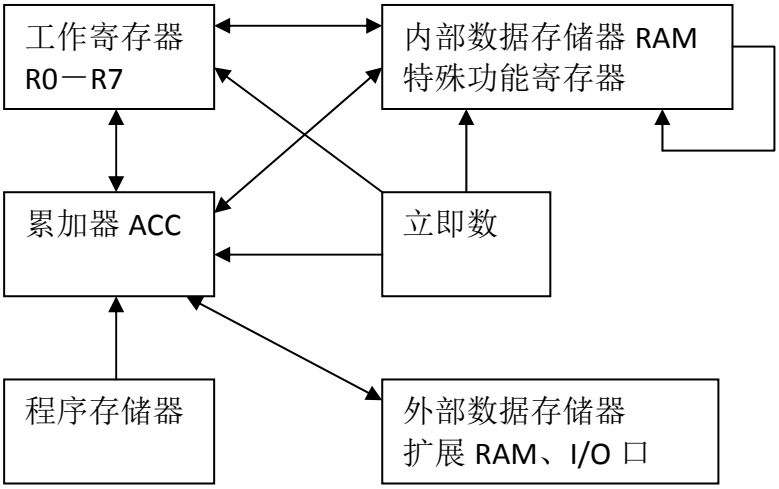
按功能分类，MCS－51 的指令系统可分为：

- ◆ 数据传送指令
- ◆ 算术运算指令
- ◆ 逻辑运算指令

◆ 位操作指令
◆ 控制转移指令
下面我们分别介绍各类指令

3.4 数据传送指令

数据传送指令用于在存贮器空间的各个地址之间传送数据。
数据传送的方向如图所示，对 A 和工作寄存器的操作最多。另外不是所有操



作方向的所有寻址方式都是有效的。

3.4.1 内部数据传送指令

1) 以累加器A为目的操作数的指令

MOV A, Rn
MOV A, direct
MOV A, @Rn
MOV A, #data

2) 以Rn为目的操作数的指令

MOV Rn, A
MOV Rn, direct
MOV Rn, #data

3) 以直接寻址单元为目的操作数的指令

MOV direct, A
MOV direct, Rn
MOV direct, direct
MOV direct, @Ri
MOV direct, #data

4) 以寄存器间接寻址为目的操作数的指令

MOV @Ri, A
MOV @Ri, direct
MOV @Ri, #data

5) 栈操作指令

PUSH direct
POP direct

在堆栈指令中，SP 指向栈顶的位置。执行压栈指令时，首先将 SP 加 1，然后将直接地址指出的内容传送到 SP 寻址的内部 RAM 中。退栈指令的过程与此相反。

6) 交换指令

XCH A, Rn

XCH A, direct

XCH A, @Ri

XCHD A, @Ri

前三条交换指令将 A 的内容与对应寻址单元中的内容相交换。最后一条指令将 A 的低四位与 R0 或 R1 指出的间接寻址单元中的内容的低四位相交换，各自的高四位不变。

3.4.2 累加器 A 与外部数据存储器传送指令

MOV DPTR, #data16

MOVX A, @DPTR

MOVX @DPTR, A

MOVX A, @Ri

MOVX @Ri, A

外部数据存储器的存取指令是通过 DPTR 或@Ri 寻址的。上面的第一条指令可以将一个 16 位立即数赋予 DPTR。下两条指令使用 DPTR 指出的地址存取外部数据存储器。也可以使用@Ri 来寻址，这时地址的高位等于当前程序地址的高位，低位是 Ri 的内容。

3.4.3 查表指令

MOVC A, @A+PC

这条指令以 PC 作为基址寄存器，A 的内容作为无符号数和 PC 内容（下一条指令的起始地址）相加得到的 16 位地址，由该地址指出的程序存储器单元内容送到累加器。此指令常用于查表，要求表格整个存放在查表指令以下的 256 字节之内。

MOVC A, @A+DPTR

此指令也用于查表，以 DPTR 作为基址寄存器，A 的内容和 DPTR 相加作为程序存储器的地址，此地址内容送到 A。这种查表指令比较方便使用，表格可以存放到任意地址，可在多处使用。但表格的大小仍不能超过 256 字节。

3.5 算术运算指令

MCS－51 的算术运算指令有加、减、乘、除和增量、减量指令。算术运算的结果不仅反映在结果寄存器中，还反映在程序状态寄存器 PSW 中。PSW 是一个特殊功能寄存器，字节地址是 0D0H，可位寻址，位寻址地址是 0D0H－0D7H。它反映了算术运算指令的结果状态，可以作为条件转移指令的条件。其格式和各位功能如下：

D7	D6	D5	D4	D3	D2	D1	D0
CY	AC	F0	RS1	RS0	OV	F1	P

CY：进位标志。同时也是布尔处理机的累加器 C。如果运算结果最高位有进位或借位时，该位置 1，否则清零。

AC：辅助进位标志。如果运算结果的低四位有进位或借位时置位，否则清零。此标志一般用于二进制运算的十进制调整。

OV：溢出标志。如果操作结果有进位进入最高位但最高位没有产生进位或

者最高位产生进位而低位没有向最高位进位，这时 OV 标志置位，否则清零。OV 标志是有符号运算（补码）的溢出标志，而 CY 作为无符号数运算的溢出标志。

P：奇偶标志。这时 ACC 的奇偶标志。如果 ACC 的 8 位模 2 和为 1（即有奇数个 1），则 P 标志置位，否则清零。由于 P 总是表示 A 的奇偶性，只随 ACC 的内容而变化，所以一个数写入 PSW，P 位的值不变。

RS1，RS0：工作寄存器区选择位。前面已经说明。

F0：用户标志位。用户可以作为软件标志自由使用。

F1：保留，用户也可自己使用。

3.5.1 加法指令

ADD A， Rn

ADD A, direct

ADD A, @Ri

ADD A, #data

这些是将寻址方式指出的内容与 A 的内容相加，结果仍送入 A。PSW 各位标志根据运算结果改变。

ADDC A， Rn

ADDC A, direct

ADDC A, @Ri

ADDC A, #data

这些指令在作加法时把寻址方式指出的内容、A 的内容和进位标志 CY 相加送入 A。通常用于多字节加法。

3.5.2 减法指令

SUBB A, Rn

SUBB A, direct

SUBB A, @Ri

SUBB A, #data

减法指令只有带借位的减法一种。它从累加器中减去指定的变量和进位标志 CY，结果在累加器中。PSW 各位标志根据运算结果而定。如果需要进行普通减法，可以先将 CY 清零。

3.5.3 十进制调整指令

DA A

这条指令将累加器中的数据根据 AC 标志将其调整为压缩 BCD 码。可以对两个压缩 BCD 运算后的结果调整为正确的十进制数据。

3.5.4 增量、减量指令

INC A

INC Rn

INC direct

INC @Ri

INC DPTR

DEC A

DEC Rn

DEC direct

DEC @Ri

这些指令将指定的变量加 1（或减 1）存放到原位置。不影响任何标志。当

使用这些指令改变输出口的内容时，原始数据从口的锁存器中读入，不从引脚读入。

3.5.5 乘除法指令

MUL AB

这条指令把 A 和 B 中的无符号 8 位数相乘，其 16 位积的低位存放在 A 中，高位字节在 B 中。如果积大于 255，则置位溢出标志 OV，否则清零。进位标志 CY 总是清零。

DIV AB

这条指令的功能是把 A 的 8 位无符号整数除以 B 中的 8 位无符号整数，所得商的整数部分存放在 A 中，余数存放在 B 中。

如果除数（B）为零，则结果 A 和 B 中的内容不定，并置位溢出标志 OV。进位标志 CY 总是清零。

3.6 逻辑运算指令

3.6.1 累加器 A 的逻辑操作指令

CLR A

这条指令的功能是将 A 清零，不影响 CY，AC，OV 等标志。

CPL A

将 A 的每一位逻辑取反。不影响标志。

RL A

将 A 的内容向左循环移位 1 位，位 7 移入位 0。不影响标志。

RLC A

将 A 和进位标志一起向左循环移位 1 位。位 7 移入 CY，CY 移入位 0。不影响标志。

RR A

类似 RL 指令，方向是向右移位。

RRC A

类似 RLC 指令，方向是向右移位。

SWAP A

这条指令将累加器 A 的高四位和低四位相互交换。

3.6.2 两个操作数的逻辑操作指令

ANL A, R0

ANL A, direct

ANL A, @Ri

ANL A, #data

ANL direct, A

ANL direct, #data

上述指令在指定的变量之间基于位的逻辑与操作。结果存放在前一个变量中。不影响标志。

ORL A, R0

ORL A, direct

ORL A, @Ri

ORL A, #data

ORL direct, A

ORL direct, #data

上述指令在指定的变量之间基于位的逻辑或操作。结果存放在前一个变量中。不影响标志。

XRL A, R0

XRL A, direct

XRL A, @Ri

XRL A, #data

XRL direct, A

XRL direct, #data

上述指令在指定的变量之间基于位的逻辑异或操作。结果存放在前一个变量中。不影响标志。

这些指令在修改输出口（P0－P3）时，原始数据从锁存器读入，不读引脚状态。

3.7 位操作指令

MCS－51 系列单片机内有一个布尔处理机，它以 CY（PSW.7）作为累加器 C，以 RAM 和 SFR 内的位寻址区的位单元作为操作数，进行位变量的传送，修改和逻辑运算指令。

3.7.1 位传送指令

MOV C, bit

MOV bit, C

使用位传送指令中一个操作数必须是 C，另一个是直接寻址位单元。

3.7.2 位修改指令

CLR C

CLR bit

SETB C

SETB bit

CPL C

CPL bit

这些指令将 C 或位单元的内容清零、置 1 和取反。

3.7.3 位变量逻辑操作指令

ANL C, bit

ANL C, /bit

逻辑与（AND）指令将 C 与位地址 bit 中内容相与送入 C 中。后面指令中在位地址前的“/”符号是将此位地址内容取反后（但不改变其本身）参与运算。

ORL C, bit

ORL C, /bit

进行逻辑或。其他类似 ANL 指令。

3.8 控制转移指令

3.8.1 无条件转移指令

转移指令使用的寻址方式在前面已经描述。

长跳转指令 LJMP addr16 使用 16 位绝对地址寻址，可以跳转到程序的任意地点。

短跳转指令 AJMP addr11 使用 11 位短转移寻址，可以跳转到和下条指令处于相同的 2K 页面的其他位置。

相对转移 SJMP rel 使用相对转移寻址，可以跳转到本条指令的前 128 个字节

到后 127 个字节之内。

基址变址寻址 JMP @A+DPTR 可以实现查表转移。

3.8.2 条件转移指令

条件转移指令是依照某种特定条件实现转移的指令。条件满足时跳转到指令指出的目的地址，不满足时执行下面的指令。条件转移指令都是使用相对转移指令。

1) 测试条件符号转移指令

JZ rel

如果累加器为零则转移；

JNZ rel

如果累加器不为零则转移；

JC rel

如果 CY=1 则转移；

JNC rel

如果 CY=0 则转移；

JB bit, rel

如果直接寻址单元的值为 1，则转移；

JNB bit, rel

如果直接寻址单元的值为 0，则转移；

JBC bit, rel

如果直接寻址单元的值为 1，则将此位清零，然后转移；

2) 比较不相等转移指令

CJNE A, direct, rel

CJNE A, #data, rel

CJNE Rn, #data, rel

CJNE @Ri, #data, rel

这些指令比较两个操作数是否相等，如果不相等则转移。如果第一操作数小于第二个操作数，则置位 CY，否则 CY 清零。

3) 减量转移指令

DJNZ Rn, rel

DJNZ direct, rel

这组指令把源操作数减 1，结果回送到源操作数中；如果结果不为 0 则转移。不改变标志位。

3.8.3 调用和返回指令

调用和返回指令是用于子程序设计的。在调用时地址是靠堆栈来保存返回地址的，这样可以实现子程序的嵌套。

短调用指令 ACALL addr11 使用 11 位短转移寻址，长调用指令 LCALL addr16 使用 16 位长转移寻址。它们在执行时，都将当前地址压入堆栈，并将栈指针加 2。接着将目标地址装入 PC 中。

子程序返回指令 RET 从堆栈中弹出 PC 的高位和低位字节，把栈指针减 2，返回到子程序调用的位置继续执行。不影响任何标志。

中断返回指令 RETI 除了执行 RET 的功能外，还清除内部相应的中断状态寄存器（该寄存器由 CPU 在中断响应时由硬件置位）。因此，中断服务程序必须由 RETI 结束。CPU 执行 RETI 指令后至少再执行一条指令，才能响应新的中断请求。

空操作指令 NOP 不执行任何操作。一般在延迟等程序中用于调整 CPU 的执行时间。

习题和思考题

1．有多少种指令可以将A的内容清零，写出每种指令，分析其对各种标志的影响。

2．如果条件转移指令的目标地址距离当前地址的距离大于128字节，如何转移。

3．实现循环10次的方法有哪些。

4．编制一段程序，将存放于30H，31H中的4位压缩BCD码作为减数，减去存放于32H和33H的4位压缩BCD码（十进制减法），结果存放在35H和36H中，如果结果为负数，置位CY。

5．编制一段程序，将存放在30H－33H中的四位ASCII码数字转换为二进制数，存放于34H－35H中。

6．编制一段程序，读取P1口的低四位，取反后输出到P1的高四位，保持P1的低四位不变。

7．编制一段程序，计数P1.0位上的负跳变（从1到0），要求对于小于8个机器周期的干扰性跳变不予计数。

第四章　新型单片机增加功能

STC89C516RD+和 STC12C5A16AD 是两种扩展的单片机。

STC89C516RD+ 增加以下功能：

1） 增强型 6 时钟/机器周期，12 时钟/机器周期 8051 CPU

2） 用户应用程序空间 64K 字节

3） 片上集成 1280 字节 RAM

4） 增加 P4 端口

5） 增加看门狗定时器

6） 支持 ISP 和 IAP

STC12C5A16AD 增加以下功能：

1） 增强型 8051 CPU，单时钟/机器周期，指令代码完全兼容传统 8051，速度比普通 8051 快 8~12 倍

2） 用户应用程序空间 16K 字节

3） 片上集成 1280 字节 SRAM

4） 内有 A/D 转换，10 位 ADC，共 8 路，转换速度可达 250K/S(每秒钟 25 万次)

5） PWM（2 路）/PCA（可编程计数器阵列，2 路）

6） 增加 P4 端口

7） 增加看门狗定时器

8） 支持 ISP 和 IAP

9） 有 EEPROM 功能（其实是 Flash）