微型机开发与应用

1984年增刊

复旦大学计算机科学研究所
微机开发应用研究室

内　容　提　要

这书内容取材于实践，比较详细地介绍了MCS－48单片机汇编语言程序设计方法和技巧，每个程序都附有详细流程图，还列举了实用的键盘、显示、A／D、D／A等详细而正确硬件接线图。书中所列举的程序设计方法具有普遍意义。

本书是对从事微机应用、特别对智能仪器仪表、工业自动化控制中应用的工程技术人员是一本实用的参考书，也可作为微机初学者自学入门指导书。

本书还附上TD—35在线仿真器的监控仿真程序，对研制和开发新的微型机的工程技术人员有一定的参考价值。

前　言

近几年来，微型电脑的应用发展很快，Intel公司的MCS—48单片机系列，在一块芯片上就集成了CPU，RAM，ROM，定时／计数器和多种功能I／O，具有体积小，价格低，控制功能强，稳定可靠等优点，广泛地用于生产过程的自动检测、实时控制和智能仪器仪表中，是目前微机应用产品化最合适和应用最广泛的一种机种，受到广大用户的欢迎。

由于单片机内存容量小，不可能用高级语言来编制各种应用程序，同时在有些实时控制中高级语言难于完成快速的实时控制，这就要求用户使用汇编语言来编制程序。这种语言编制程序难度较大，给单片机的推广普及带来困难，我们在多次举办单片机应用短训班的基础上，根据广大用户要求，利用短短几个月的时间，编制了一些应用性的实用子程序，其中大部分程序是实际应用中的例子。本书为初学者介绍了一些汇编语言程序设计的基础和技巧，每个程序都画出详细的流程图，所提供的设计方法基本适用于8080／8085，z80，6800等各种微型机的汇编语言程序设计。

本书由姚志石、涂时亮同志编写。参加编写的还有张友德同志。高哲民也参加了部分编写工作。

在编写过程中，受到徐君毅、陈章龙同志的热情帮助和指导，薛剑虹同志为本书出版做了大量工作，在此表示感谢。

由于编者水平有限，时间仓促，书中内容定有不当之处，敬请读者批评指正。

编　者

1984年10月

第一部分

MCS—48 单片微计算机概论

目　　录

1.2 程序存贮器

内部程序存贮器为1024(8048)或2048(8049)字节8035/8039内部无程序存贮器，必须使用外部器件。程序存贮器中，有三个单元特别重要。

1. 0号单元

处理机复位(RESET)时，计算机总是从0号单元开始执行指令。

2. 3号单元

在允许中断时，当中断输入线上有中断信号时计算机总是转到3号单元执行中断服务程序。

3. 7号单元

当允许定时/计数器中断时，若定时器计数器溢出时，则产生定时/计数中断，计算机转到7号单元执行定时/计数器中断服务程序。

在使用时，0号单元应该放一条转移指令，转到用户主程序；3号单元放转移指令，使用中断时，转到外部中断服务子程序上；7号单元的转移指令转到定时/计数器中断服务子程序上。

1.3 数据存贮器

内部数据存贮器有64/128字节

64×8 RAM有三个地址范围(0～7，8～23，24～31)，有专门的使用方法。

当初始通电时，状态寄存器中的寄存器开关BS为0。BS＝0，则0～7就作为直接可寻址的高速暂存寄存器R_0～R_7，这样指令MOV A，R7即把RAM中的第7号单元内容送到累加器A。

可以使用寄存器区开关指令使BS＝1，当BS＝1时，则把另8个单元24～31定义为R_0～R_7，此时MOV A，R7即把内存31号单元内容送到累加器A。8～23号单元既可作一般的存贮器，也可作堆栈使用，在调用子程序时用作记述。可容纳8个返回地址，故子程序的嵌套最大可以8级。除了0～7，24～31可以直接寻

址外其它的内存单元的寻址只能通过R_0或R_1间接寻址。

1.4 输入／输出

有27根可用作输入或输出的线，它们是三个8位口，P_1口、P_2口、BUS口，P_1 P_2具有相同的特性，写到这些口的数据被锁存起来，输入时无锁存。

P_1，P_2为准双向口，允许每一根线作为输入，输出或同时作输入 输出，在这种用法时，由于它的内部特殊结构，在输入时必须先写一个“1”，否则输入要出错，当不同时作输入／输出时，无此要求。

P_1，P_2，BUS口还可以和立即数进行与／或操作。BUS口是真正的双向口，输入／输出不能混用。

另有三个输入／输出线是：T_0、T_1，$\overline{INT}$。能用条件转移指令来测试。

此外，T_0 可作时钟输出

T_1 作为计数输入

$\overline{INT}$ 作中断输入

1.5 程序状态字

程序状态字(PSW)是一个8位状态字，发生子程序调用或中断时，PSW高四位和程序计数器PC一起被存入堆栈中(内存8～23号单元)，并能用RETR指令来恢复。

PSW各位定义如下：

位0～2，栈指针，初态为000，指向RAM8和RAM9号单元，第一个子程序调用或中断时把程序计数器的内容和PSW的高四位存入栈中。然后栈指针加1，指向10和11号单元。

子程序末尾应为返回指令，可使栈指针减1。

位3　不使用。

位4　工作寄存器区开关(BS)，0＝区0，1＝区1。

位5　　标志F。，它是用户控制的标志位

位6　　辅助进位（AC），由ADD指令产生，在十进调正指令DAA时使用。

位7　　进位位（CY），指示前面操作时累加器A溢出情况。

1.6　中断

可有二种中断，在INT上加一个低电平输入，可产生中断（在允许中断时），程序转到003号单元。当定时／计数器发生溢出时，也可产生中断，程序转到007号单元。

中断系统为单级中断，当检测到一个中断后，将不再理睬其他中断，只有执行RETR指令后，才重新允许中断输入，当中断系统检测到外部中断和定时／计数器中断同时存在时，则先响应外部中断。

1.7　复位与单步操作

复位时具有以下功能

1．清0程序计数器，程序从0号单元开始执行。

2．清0栈指针。

3．清0寄存器区开关BS＝0

4．清0存贮器区开关DBF＝0

5．关中断（定时和外部中断）

6．P_1口，P_2口为输入方式。

7．清0 F0，F1标志位。

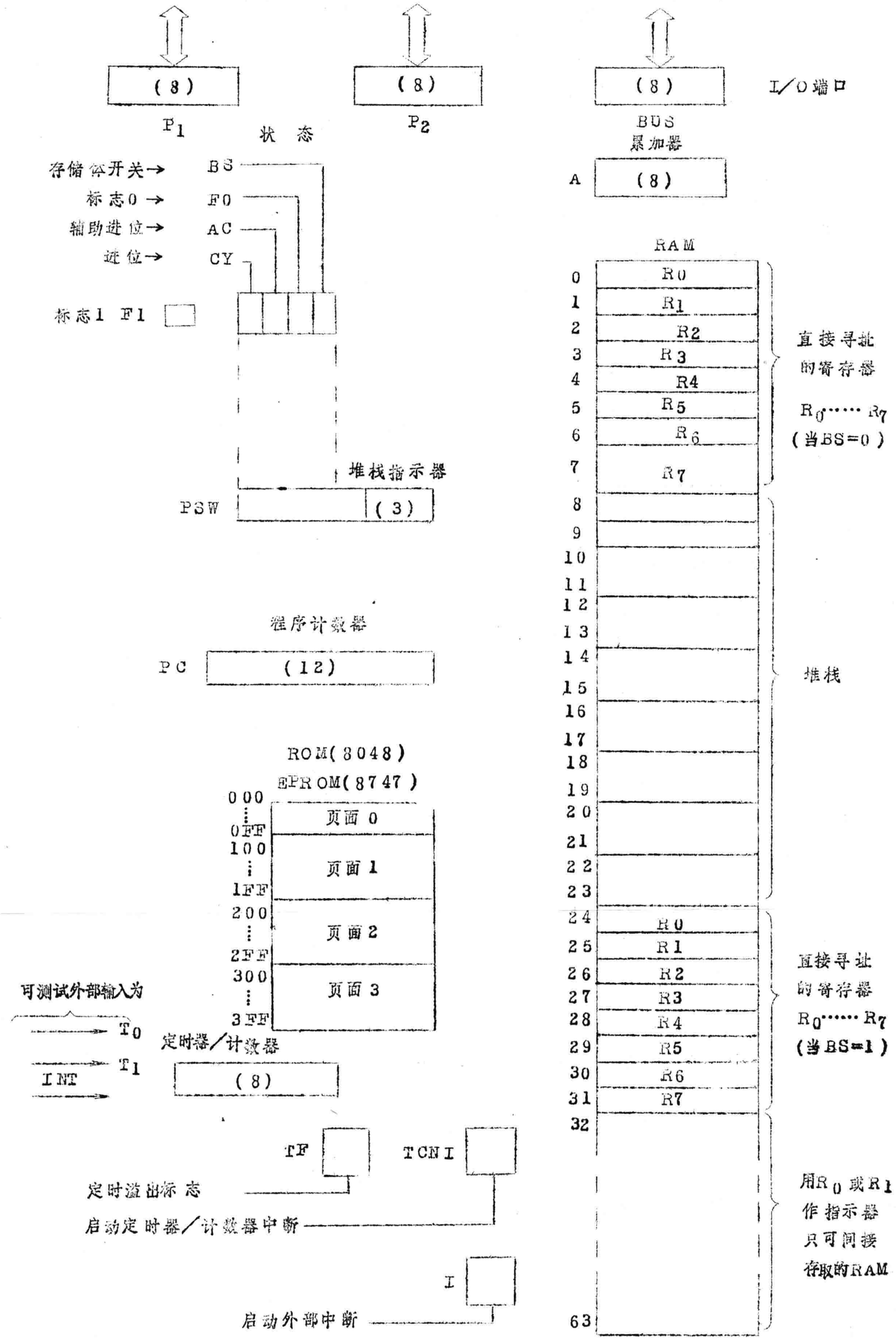
8．不允许CLK从T_0输出。

9．BUS口为高阻状态。

10．停止计时。

11．清0定时器标志

单步功能给用户调试手段，$\overline{SS}$上低电平时，要求停止处理机运行，当SS升为高电平时，退出停止方式取下一个指令。



附图1—2 Intel 8048/8748程序设计模型

第二章　MCS—48 指令系统摘要

	助记忆符		说　　明	字节	周期
	* ADD	A,Ri	寄存器加A	1	1
	* ADD	A,@Rj	数据存贮器加A	1	1
	* ADD	A,#data	立即数加A	2	2
	* ADDC	A,Ri	寄存器带进位加A	1	1
	* ADDC	A,@Rj	数据存贮器带进位加A	1	1
	* ADDC	A,#data	立即数带进位加A	2	2
	ANL	A,Ri	寄存器“与”A	1	1
	ANL	A,@Rj	数据存贮器“与”A	1	1
	ANL	A,#data	立即数“与”A	2	2
	ORL	A,Ri	寄存器“或”A	1	1
	ORL	A,@Rj	数据存贮器“或”A	1	1
累	ORL	A,#data	立即数“或”A	2	2
加	XRL	A,Ri	寄存器“异或”A	1	1
器	XRL	A,@Rj	数据存贮器“异或”A	1	1
类	XRL	A,#data	立即数“异或”A	2	2
	INC	A	A加1	1	1
	DEC	A	A减1	1	1
	CLR	A	A清0	1	1
	CPL	A	A取反	1	1
	DA	A	A十进制调整	1	1
	SWAP	A	A半字节交换	1	1
	RL	A	A左环移	1	1
	* RLC	A	A左环移带进位	1	1
	RR	A	A右环移	1	1
	* RRC	A	A右环移进位	1	1

以上指令中 R_i 为 R_0 R_1 ……R_7　　打 * 者指令影响 CY

R_j 为 R_0，R_1，运算结果均在 A

	助记忆符		说　　明	字节	周期
输入/输出	IN	A,PK	P口输入到A	1	2
	OUTL	PK,A	A输出到P口	1	2
	ANL	PK,#data	P口“与”立即数	2	2
	ORL	PK,#data	P口“或”立即数	2	2
	INS	A,BUS	BUS输入到A	1	2
	OUTL	BUS,A	A输出到BUS	1	2
	ANL	BUS,#data	BUS“与”立即数	2	2
	ORL	BUS,#data	BUS“或”立即数	2	2
	MOVD	A,PH	扩展口输入到A	1	2
	MOVD	PH,A	A输出到扩展口	1	2
	ANLD	PH,A	扩展口“与”A	1	2
	ORLD	PH,A	扩展口“或”A	1	2
	以上指令中Pk为P_1、P_2口 P_H为扩展的P_4、P_5、P_6、P_7口				
寄存器类	INC	Ri	寄存器加1	1	1
	INC	@Rj	数据存贮器加1	1	1
	DEC	R	寄存器减1	1	1
转移类	JMP	addr	无条件转跳	2	2
	JMPP	@A	间接跳转	1	2
	DJNZ	Ri,addr	寄存器减1非零跳	2	2
	JC	addr	当Carry=1时跳	2	2
	JNC	addr	当Carry＝0时跳	2	2
	JZ	addr	当累加器为零时跳	2	2
	JNZ	addr	当累加器不等于零时跳	2	2
	JTO	addr	当T0＝1时跳	2	2
	JNTO	addr	当T0＝0时跳	2	2
	JT1	addr	当T1＝1时跳	2	2
	JNT1	addr	当T1＝0时跳	2	2
	JF0	addr	当F0＝1时跳	2	2

	助记忆符		说　明	字节	周期
	JF1	addr	当F1=1 时跳	2	2
	JTF	addr	定时器标志为1 时跳	2	2
	JNI	addr	当 $\overline{INT}$=0 时跳	2	2
	JBb	addr	累加器第b 位为1 时跳	2	2

以上转移指令除JMP addr 外，均在指令的第2 字节所在页内转，JMP指令在2K内转，若从前2K转到后2K去，则先用存贮器区开关指令SEL MB1，反之用SEL MB0 。

子程序类	CALL	addr	转子程序	2	2
	RET		返回	1	2
	RETR		返回並恢复状态	1	2

CALL 指令的转移同JMP 指令，但该指令不要出现在内存的2046，2047，2048 单元处，同样也不要出现在4093，4094，4095 处，因为返回时容易出错。

标志类	CLR	C	清除进位位	2	1
	CPL	C	进位位取反	1	1
	CLR	F0	清除标志F0	1	1
	CPL	F0	标志F0 相反	1	1
	CLR	F1	清除标志F1	1	1
	CPL	F1	标志F1 取反	1	1
数据传送类	MOV	A,Ri	寄存器送A	1	1
	MOV	A,@Rj	数据存贮器送A	1	1
	MOV	A,#data	立即数送A	2	2
	MOV	Ri,A	A送寄存器	1	1
	MOV	@Rj,A	A送数据存贮器	1	1
	MOV	R_i,#data	立即数送寄存器	2	2
	MOV	@R_j,#data	立即数送数据存贮器	2	2
	MOV	A,PSW	PSW 送A	1	1
	MOV	PSW,A	A送PSW	1	1

	助记忆符		说　　明	字节	周期
	XCH	A,R_i	A和寄存器交换	1	1
	XCH	A,@R_j	A和数据存贮器交换	1	1
	XCHD	A,@R_j	A和寄存器半字节交换	1	1
	MOVX	A,@R_j	外部数据存贮器送A	1	2
	MOVX	@R_j,A	A送外部数据存贮器	1	2
	MOVP	A,@A	现行页面送A	1	2
	MOVP3	A,@A	页面3送A	1	2

以上指令中R_i表示R_0～R_7，R_j表示R_0，R_1

定时器/计数器类	MOV	A,T	读定时器/计数器	1	1
	MOV	T,A	写定时器/计数器	1	1
	STRT	T	启动定时器	1	1
	STRT	CNT	启动计数器	1	1
	STOP	TCNT	停止定时器/计数器	1	1
	EN	TCNTI	允许定时器/计数器中断	1	1
	DIS	TCNTI	禁止定时器/计数器中断	1	1
控制类	EN	I	允许外部中断	1	1
	DIS	I	禁止外部中断	1	1
	SEL	RB0	选寄存器堆0	1	1
	SEL	RB1	选寄存器堆1	1	1
	SEL	MB1	选存贮区1	1	1
	ENTO	CLK	允许TO作时钟输出	1	1
	NOP		空操作	1	1

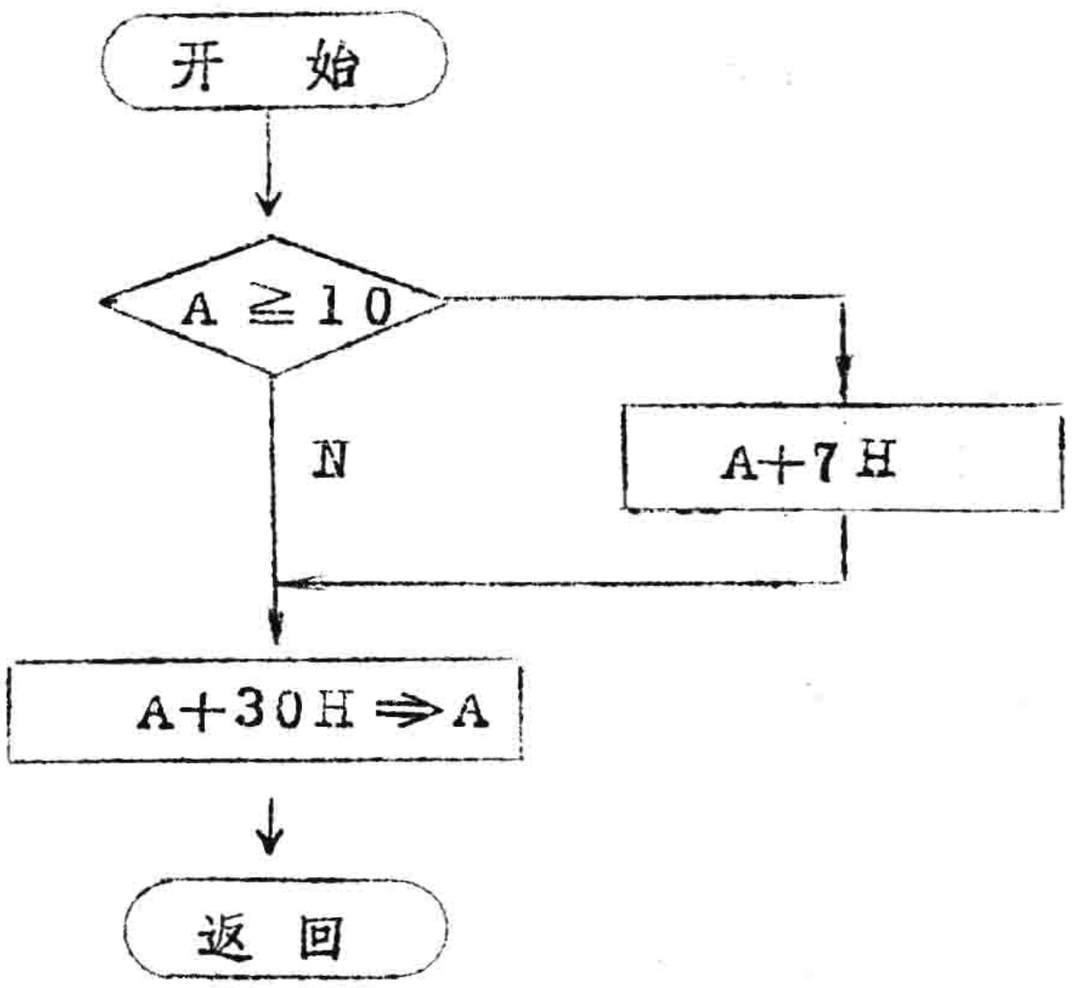
第一章　代码转换

1--1　十六进制→ASCII码转换程序

入口参量：十六进制数在累加器中

出口参量：该数的ASCII码放在累加器A中

框图：



程序：

```
0/0   AA          HEASC:  MOV  R2,A
  1   03  F6              ADD  A,#F6H
  3   FA                  MOV  A,R2
  4   E6  18              JNC  AD30
  6   03  07              ADD  A,#07H
  8   03  30    AD30:     ADD  A,#30H
  A   83                  RET
```

说明：由于MCS—48中无比较指令也无减法指令，使用A－B＝A＋〔B〕补，然后根据CY来判断：CY＝1时，A≧B；CY＝0

时，A＜B，第二条指令中的F6H 即为〔10〕补。

1－2　　ASCII 码→十六进制

入口参量：ABCII 码在累加器A 中

出口参量：转移后的十六进制数在A中

入口：ABCHE

方法：若A中放的是0～9的ASCII 码，则减去30H后即得0～9之间的数。若A中是A～F的ASCII 码，则减去37H后得A～F之间的数。

框图：