微机原理与应用实验

■无线电实验丛书

微机原理与应用实验

俞承芳 编

复旦大学出版社

无线电实验丛书

微机原理与应用实验

俞 承 芳 编

复旦大学出版社

内 容 提 要

本书系统地介绍了微型计算机的程序设计方法、存贮器和外围电路的扩展以及微机应用系统的设计方法。

全书分为九个单元，其中包括二十八个实验。第一、二单元分别介绍Z80及INTEL8031单板机的使用和程序设计方法；第三单元讲述了各类存贮器及其扩展技术；第四单元阐述并行接口；第五单元介绍计数定时电路；第六单元讲述键盘、拨盘及显示器；第七单元叙述数模和模数转换电路；第八单元介绍串行接口电路；第九单元论述微机应用系统的设计过程。

元线电实验丛书

1. 脉冲与数字电路实验教程
2. 模拟电子线路实验
3. 微机原理与应用实验
4. 近代无线电实验
5. 常用无线电仪器和器件手册

责任编辑 林溪波

微机原理与应用实验

俞承芳 编

复旦大学出版社出版

（上海国权路579号）

新华书店上海发行所发行 复旦大学印刷厂印刷

开本 787×1092 1/16 印张 13 字数 321,000

1991年7月第1版 1991年7月第1次印刷

印数1—5,000

ISBN7-309-00632-1/T·25

定价：7.50元

前　言

近几年来，微型计算机技术飞速发展，并迅速渗透到各个领域。微机原理与应用实验日益受到人们的普遍重视。通过这些实验，促使学生学好“微机原理与应用”课程，有助于学生使用微机乃至为设计微机应用系统打下扎实的基础。

本书以Z80和INTEL8031为例，系统介绍微机的程序设计方法、存贮器和外围电路的扩展以及微机应用系统的设计方法。在编写过程中，我们尽可能地考虑教材的普及性和实用性。本书的第一、二单元分别介绍了Z80及INTEL 8031单板机的使用和程序设计方法；第三单元讲述了各类存贮器以及微机系统存贮器的扩展技术；第四至八单元阐述外围电路的工作原理及其扩展技术：其中第四单元讲述并行接口；第五单元介绍计数定时电路；第六单元阐述键盘、拨盘及显示器；第七单元叙述数模和模数转换电路；第八单元介绍串行接口电路；第九单元以应用课题为例，阐述应用系统的设计方法。在本书的各个单元中，我们都安排一些实验，其中第一、二单元为软件实验；第四至八单元为硬件实验，在完成上述实验的基础上，可选做第九单元的综合实验。第九单元的综合实验分为两类：一类在提出实验要求的同时，还提供实施的参考方案；另一类仅提出实验要求，由学生自己完成实验。本书各单元之间既有一定的联系，又有相对的独立性，它为学生今后设计专用微机系统打下良好的基础。

在本书的编写过程中，编者得到了许多同志的热情帮助。陈瑞涛副教授审阅了全部书稿，并提出了许多宝贵意见；陈训亮高级工程师、马杰副教授给予热情的支持和帮助，刘祖望、黄群满、夏永寿、钱培专、祁兵等同志在实验的编排工作中做了大量的工作，在此向他们表示感谢。

由于本人水平有限，书中难免存在错误和不妥之处，敬请读者批评指正。

编者于复旦大学

1990.5.

目　录

第六单元 键盘、拨盘与显示器接口

第七单元 数模(D/A)转换器和模数(A/D)转换器及其接口

第八单元 串行通信及接口

第九单元 综合应用实验

第一单元　TP-801单板机的使用及程序设计

一、TP-801 单板机的使用

（一）TP-801 单板机的构成

TP-801 单板机由以下部件构成：Z80-CPU，装有监控程序的只读存贮器(ROM)，随机存取存贮器(RAM)，计数定时电路 CTC，并行接口电路 PIO，键盘，LED 显示器以及控制电路。采用的时钟频率为 2M。

Z80-CPU 的寻址范围为 64K，在单板机中装了 2K ROM 及 4K RAM。2K ROM 的地址为 0000H～07FFH，ROM 内装有监控程序。4K RAM 的地址为 2000H～2FFFH，其中可供用户使用的为 2000H～2F87H。2F88H～2FFFH 为监控程序工作区及堆栈工作区。在供用户使用的 RAM 中，2000H～27FFH 的 2K 装有写保护开关。当开关置于正常位置时，可对这 2K RAM 进行正常的读写操作，当开关置于写保护位置时，只能执行读操作，而无法写入数据。因此通常可将用户程序安排在 2000H～27FFH 的 2K 内，以免在执行程序时因误操作而破坏程序。数据通常安排在 2800H 以后的单元中。

在单板机上还装有 4K EPROM 的空插座 PROM1 及 PROM2，供用户插入装有成熟的应用程序 EPROM，其地址分别为 0800H～0FFFH 及 1000H～17FFH。其中 PROM2 插座还可供用户向 EPROM 写入程序。

单板机上还装有 6 位 LED 显示器以及 28 个按键的键盘。

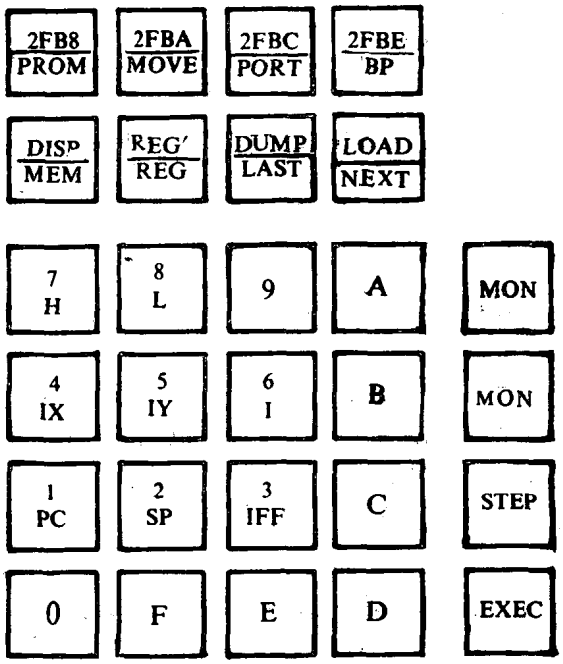

图 1-1 TP-801 的键盘

6 位显示器通常以左面四位显示地址，以右面的两位显示数据。

28 个按键的键盘如图 1-1 所示。其中 16 个键为数字键，12 个为命令键。

16 个数字键为十六进制数 0～F，它们被用来向计算机输入十六进制数码。在检查寄存器的内容时，则用来代表不同的寄存器。

12 个命令键用来产生 20 种命令，它们是：

MON 键　进入监控程序，换下档。

MON′ 键　进入监控程序，换上档。

STEP 键　单步执行程序。

EXEC 键　连续执行程序。

DISP/MEM 键　上档计算偏移量，下档检查存贮单元。

REG′/REG 键　上档检查辅助寄存器，下档检查寄存器。

DUMP/LAST 键　上档输出磁带机信息，下档检查上一个存贮单元。

LOAD/NEXT 键　上档输入磁带机信息，下档检查下一个存贮单元。

2FB8/PROM 键　上档定义用户功能，下档为写 EPROM。

2FBA/MOVE 键　上档定义用户功能，下档移动存贮器内容。

2FBC/PORT 键　上档定义用户功能，下档检查外部口。

2FBE/BP 键　上档定义用户功能，下档设置断点。

单板机是在监控程序的控制下工作的。它主要有以下四个部分：

1．初始化、显示和键盘分析程序

初始化程序设定堆栈指针、清除键盘状态标志，并在最左端显示器所对应的显示存贮单元中填入“P”，显示程序将显示存贮单元的内容取出送往 LED 显示器显示。键盘分析程序搜索键盘的输入，若无按键按下则返回显示程序，当有按键按下时，如按下的是数字键则进行数字键处理，如果是命令键则转入相应的键命令程序段执行此命令。

2．键命令程序

对应二十种命令的程序段。

3．实用子程序

包括延时程序、ASCII 码与十六进制码的相互转换等，这些程序供监控程序使用，也可供用户调用。

4．表格程序

供监控程序工作所用的数据表格，如键值编码、显示字型等。

在单板机上装有复位按钮开关 S_1，当此开关按下时产生复位信号 $\overline{\text{RESET}}$。单板机在接通电源时也产生此信号。在复位信号有效期间，地址总线、数据总线处于高阻状态，所有的控制信号无效。在此信号的作用下 Z80-CPU 初始化，程序计数器为 0。因此，按下复位按钮并释放可使单板机从 0000H 开始执行监控程序。

根据单板机上装的选择开关 S_2 的位置，监控程序的执行有所不同。当 S_2 置于 MON RST 位置时，复位后从 0000H 开始执行监控程序，然后在初始化之后即执行显示程序和键盘分析程序，显示待命符“P”。当有键按下时则执行相应的动作。当 S_2 置于 PROM1 RST 的位置时，复位后从 0000H 执行监控程序中的初始化程序，然后自动转到地址 0800H 去执行 PROM1 中的用户程序。

用户除了通过复位按钮进入监控程序外，还可按 MON 键和 MON′ 键进入监控程序。

MON 键的作用是返回监控程序。与复位按钮不同的是 MON 键能保护 CPU 寄存器的状态及监控程序所用的某些单元的内容。当用户输入数据或执行命令时，可用此键中止输入数据或退出命令状态，回到监控程序的初始化部分，以便接受新的命令或数字。当用户正在执行一程序时，则由此键中止此程序的执行，回到监控程序的初始化部分，这时由于 CPU 寄存器的内容未被破坏，因而对分析程序的执行情况十分有用。

通常在进行一种新的操作方式之前，都必须按下 MON 键，使显示器显示 "P" 之后再输入新的数字或命令。在按下 MON 键后再输入的命令键均为下档键。

MON′ 键除了具备 MON 键的功能外，在按下它之后再输入的命令键均为上档键。例如 REG′/REG 键为寄存器检查键，如按 MON 之后，再按此命令键为检查寄存器内容。如按 MON′ 之后再按此命令键则为上档的功能，即检查辅助寄存器的内容。

（二）存贮器和寄存器内容的检查与修改

存贮器的检查和修改由 MEM 键、LAST 键和 NEXT 键实现。其方法如下：在按 MON 键并显示"P"之后，先通过键盘输入要求检查的存贮单元的地址，此地址由四位十六进制数构成，此时在显示器的左面四位显示该地址。然后按下 MEM 键，此时显示器右面的两位即显示此存贮单元的内容。如要修改此存贮单元的内容，只需输入两位十六进制数，在送完两位数后，该单元的内容即被修改。如果只输入一个数字，此单元的内容不会被修改。如在输入两个数字后再输入两个数字，则后输入的两个数字替代原先的内容。在输入地址的过程中，如未送满四个数字即按下 MEM 键，则 MEM键无效，而原先送入的数字仍有效。

为方便存贮器内容的检查和修改，可利用 LAST 键和 NEXT 键对现行存贮单元的上一个或下一个进行检查和修改。

对于只读存贮器及随机存取存贮器中处于写保护状态的那部分，只能进行检查而不能修改。

存贮器内容的检查和修改的具体过程如下例所示。

按 键	显 示		说 明
MON	P		
2000	2000		送入四位地址
MEM	2000	××	该单元内为随机数
0	2000	××	送入一个数字不会改变内容
0	2000	00	送入两个数字改变内容
11	2000	11	送入的 11H 替代了 00H
NEXT	2001	××	下一个单元为随机数
22	2001	22	送入 22H
NEXT	2002	××	
33	2002	33	
LAST	2001	22	上一个单元内为 22H
LAST	2000	11	

按 键	显 示		说 明
MON	P		
0000	0000		送入四位地址
MEM	0000	AF	该单元内为 AFH
NEXT	0001	00	下一个单元内为 00H
NEXT	0002	31	下一个单元内为 31H
55	0002	31	ROM 内的程序无法修改
MON	P		
20	20		
MEM	20		送入两位数字时 MEM 键无效
00	200C		继续送入数字
MEM	2000	11	以前送入的两位数字仍有效
NEXT	2001	22	

寄存器和辅助寄存器内容的检查和修改由 REG键以及 REG′ 键实现。

利用 REG 可以检查或修改寄存器的内容。它 包 括 A、B、C、D、E、F、H、L、I、IFF、PC、IX、IY。SP 寄存器的内容只能检查而不能更改。

利用 REG′ 可以检查或修改辅助寄存器的内容。它包括 A′、B′、C′、D′、E′、F′。H′、L′。

修改和检查寄存器内容的方法如下：在按 MON 键并显示"P"之后，先按下代 表 寄 存器号的数字键，对于 A～F 寄存器即为数字键，其余的用其他数字表示：1 代表 PC，2代表 SP，3 代表 IFF，4 代表 IX，5 代表 IY，6 代表 I，7 代表 H，8 代表 L。此时显示器最 左面显示该数 字。然后按 REG 键，显示器的右边两位或四位显示该寄存器的内容。如要修改寄存器的内容则应输入两位或四位数字。在检查完一个寄存器的内容之后，如要检查另一个寄存器的内容或要送入新的命令时，应先按下 MON 键。在寄存器检查 的工作方式时 NEXT 键及 LAST 键无效。

如果要求检查和修改辅助寄存器的内容，应先按 MON′ 键，此时显示器 最左 面显示"′"，然后按代表寄存器号的数字键，此时显示器显示该 数字，再 按 REG′/REG 键，此时显示器左面显示数字及"′"，右面显示该辅助寄存器的内容。修改内容的方法和修改寄存器内容的方法一样。在检查辅助寄存器的内容时，NEXT键和 LAST 键同样无效。

寄存器和辅助寄存器内容的检查和修改的具体过程如下例所示。

按 键	显 示		说 明
MON	P		
A	A		
REG	A	××	××为累加器 A 中原来的数据
00	A	00	将累加器 A 清零
11	A	11	累加器 A 被送入11H
MON	P		
7	7		检查并修改寄存器H
REG	7	××	

按 键	显 示	说 明
22	7 22	
MON	P	
1	1	检查并修改程序计数器 PC
REG	1 ×× ××	
2000	1 20 00	
MON	P	
2	2	检查堆栈指针 SP
REG	2 2F B8	
2800	2 2F B8	SP 不能修改
MON′	′	检查并修改辅助寄存器
A	A	
REG	A′ ××	
00	A′ 00	
MON′	′	
C	C	
REG	C′ ××	
11	C′ 11	
MON	P	

（三）程序的执行

对于已编好的程序，我们应先把它译为机器码，然后用修改存贮器内容的方法将其逐条输入。当此程序正确无误地送入之后，我们即可执行此程序。

程序的执行键为 EXEC 键，执行程序的过程如下：在送完程序之后，先按 MON 键，此时显示器上显示“P”。然后输入四位数字的程序起始地址，此时显示器左面四位显示该地址，然后按 EXEC 键。此程序即从此地址开始执行。

如果未送入程序的起始地址，在按 MON 键显示“P”后直接按 EXEC 键时，程序从保存在“用户寄存器存放区”中的 PC 的地址开始执行程序。

按 EXEC 键执行用户程序后，可按 MON 键中止该程序的执行并返回监控程序，此时 CPU 寄存器的内容都未破坏，我们可以检查程序的执行情况。

下例为执行一程序的具体过程：

假定要求用传送指令将累加器 A 清零，将 01H 送到 B 寄存器。其汇编程序如下：

标 号	操作码	操作数
	ORG	2000H
	LD	A,00H
	LD	B,01H
	HALT	

译为机器码如下：

地　址	机器码	标　号	操作码	操作数
			ORG	2000H
2000	3E00		LD	A,00H
2002	0601		LD	B,01H
2004	76		HALT	

先将机器码送入存贮器

按 键	显 示		说　明
MON	P		
2000	2000		送入机器码
MEM	2000	××	
3E	2000	3E	
NEXT	2001	××	
00	2001	00	
NEXT	2002	××	
06	2002	06	
NEXT	2003	××	
01	2003	01	
NEXT	2004	××	
76	2004	76	机器码已全部送入
MON	P		
2000	2000		送入程序的起始地址
EXEC			执行程序

此时程序已执行，由于第三条指令为暂停指令，程序将在执行完第一第二条指令后进入暂停状态。

我们可按 MON 键返回监控程序。在显示"P"之后，通过检查寄存器内容知道寄存器 A 被清零，寄存器 B 被送入 01H，程序计数器 PC 的值为 2005H，这是下一条要执行的指令地址。

二、程序设计方法

程序设计是用计算机解决实际问题过程中的一个环节。在用计算机解决实际问题的过程中包括了建立数学模型，选择计算方法，设计程序及上机调试，运行程序并分析计算结果几部分。对一个具体的实际问题经分析后我们可以得出各个量之间的关系，并用数学式进行描述，称之为建立数学模型。有了数学模型还必须选择适合于计算机实现的具体计算方法，然后再设计程序，为保证解题的正确性，应先对程序进行调试。

（一）程序设计的步骤

程序的设计通常分为下面四个步骤：

1．作程序流程图

根据题意，对提出的要求找出最合理、最简便的解决方法并作程序流程图。程序流程图表示了人们解决问题的思路。流程图有粗略的和详细的两种。粗略的流程图可以给出解题的大致步骤，而详细的流程图则给出每一步骤的细节。对一些大的问题，应先给出粗略的流程图以得出总体概念，然后作详细的流程图对每一步骤作出具体的描述。

2．存贮单元分配

根据题意合理地分配程序和数据在存贮器中的地址。存贮单元分配得合理，将会给编制程序带来方便。不然可能会增加麻烦，甚至使程序产生错误。一般应避免程序区与数据区相互混合、交叉。

3．源程序编制

将流程图中每步所规定的操作用程序语言来实现，为此可选用不同的指令，因而各人所编的程序可能是不同的。应选用最合理的程序来实现。程序质量的衡量标准通常为：程序比较短，占用的存贮单元少，运行的时间短，而且程序的结构合理。

4．译为机器码

将源程序逐条译为机器码以便送入计算机。

在上述步骤中，作程序流程图是程序设计的重要步骤。在复杂的问题中，它可以帮助设计者寻找最佳方案，减少源程序编写过程中的错误。在调试过程中它又可帮助寻找错误。而且它也是对源程序的简明解释，便于保留与交流。通常我们应先作程序流程图再编源程序。

【例 1-1】 在存贮器的 2800H 单元中存有一个两位的 BCD 码(0～9)，试将这两位 BCD 码分别转为 ASCII 码，并将其低字节对应的 ASCII 码存于 2900H 单元中，将高字节对应的 ASCII 码存于 2901H 单元中。

根据题意，应分别取出这两个 BCD 码并转为对应的 ASCII 码。由于 BCD 码(0～9)对应的 ASCII 码为十六进制数表示的 30H～39H，因此在取出 BCD 码之后只要加上 30H 即为对应的 ASCII 码了。

此题的程序流程图如图 1-2 所示。

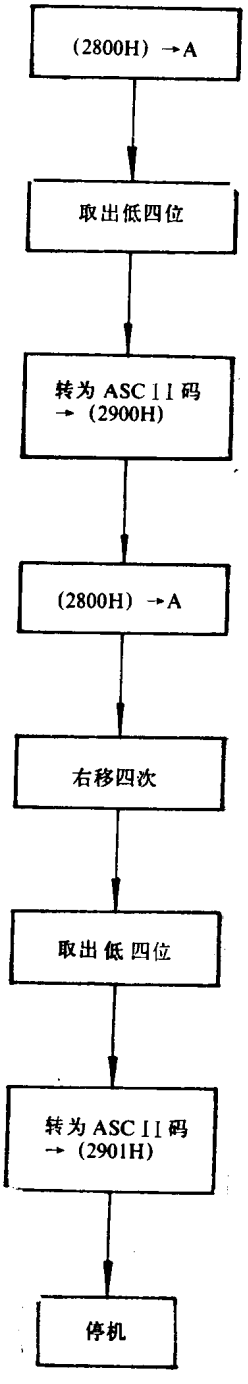

图 1-2　例1-1的程序流程图

在此题中已经规定了数据的存放单元，因此只要注意将程序区避开数据存放单元即可。现将程序的起始地址安排在 2000H。

我们以 HL、DE 寄存器对作为原始数据及变换后数据的地址寄存器，根据程序流程图可得到源程序如下：

标 号	操作码	操作数
	ORG	2000H
	LD	HL,2800H
	LD	DE,2900H
	LD	A,(HL)
	AND	0FH
	ADD	A,30H
	LD	(DE),A
	INC	DE
	LD	A,(HL)
	RRA	
	RRA	
	RRA	
	RRA	
	AND	0FH
	ADD	A,30H
	LD	(DE),A
	HALT	

将上述源程序译为如下所示的机器码：

地 址	机器码	标号	操作码	操作数
			ORG	2000H
2000	210028		LD	HL,2800H
2003	110029		LD	DE,2900H
2006	7E		LD	A,(HL)
2007	E60F		AND	0FH
2009	C630		ADD	A,30H
200B	12		LD	(DE),A
200C	13		INC	DE
200D	7E		LD	A,(HL)
200E	1F		RRA	
200F	1F		RRA	
2010	1F		RRA	
2011	1F		RRA	
2012	E60F		AND	0FH
2014	C630		ADD	A,30H
2016	12		LD	(DE),A
2017	76		HALT	

（二）分支程序的设计

在例 1-1 中我们将一个 BCD 码转换为对应的 ASCII 码，此时只要把这个 BCD 码加上 30H 即可。但如果要把一个十六进制数 0～F 转为 ASCII 码，情况就不同了。由于A～F 的 ASCII 码为 41H～46H，因此 A～F 的 ASCII 码应为其值加上 37H。这样，把一个十六进制数转换为 ASCII 码时，应根据其数值作不同的处理。

对一个问题的处理过程中，如果要求根据不同的情况和条件作不同的处理时，应采用分支程序来实现。

分支程序的程序流程图如图 1-3 所示。

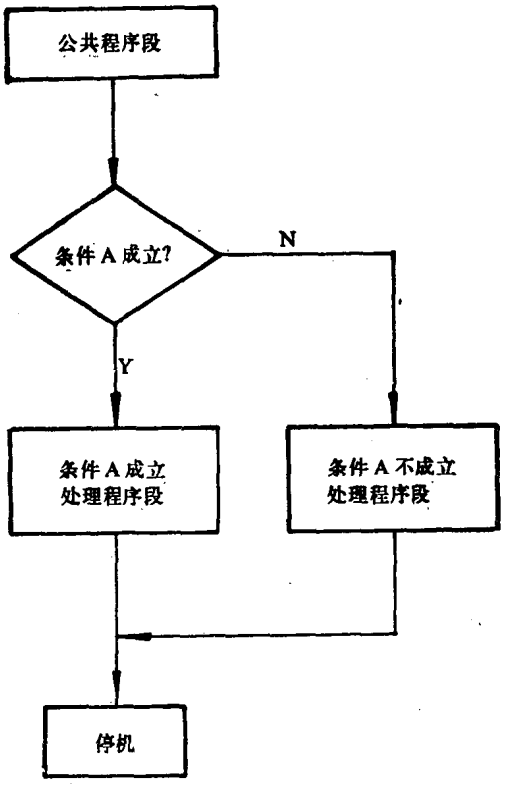

图 1-3　分支程序的程序流程图

表 1-1　条件转移指令表

指　　令	机　器　码	功　　能
JP　C,nn	DAnn	进位转移
JP　NC,nn	D2nn	无进位转移
JP　Z,nn	CAnn	零转移
JP　NZ,nn	C2nn	非零转移
JP　PE,nn	EAnn	偶数转移

（续表）

指　令	机　器　码	功　能
JP PO,nn	E2nn	奇数转移
JP M,nn	FAnn	符号为负转移
JP P,nn	F2nn	符号为正转移
JR C,e	38e	进位转移
JR NC,e	30e	无进位转移
JR Z,e	28e	零转移
JR NE,e	20e	非零转移

在Z80指令系统中，可利用条件转移指令来构成分支程序。这些指令如表1-1所示。

【例1-2】 在累加器A中，有一个十六进制数0～F，现要求将其转换为对应的ASCII码，仍放于累加器A中。

在转换的过程中应根据累加器A中的数值作不同的处理，当累加器A中的值小于十六进制数A时，应将其加上30H，如累加器A中的值大于或等于十六进制数A时，则应加上37H。此题的程序流程图如图1-4所示。

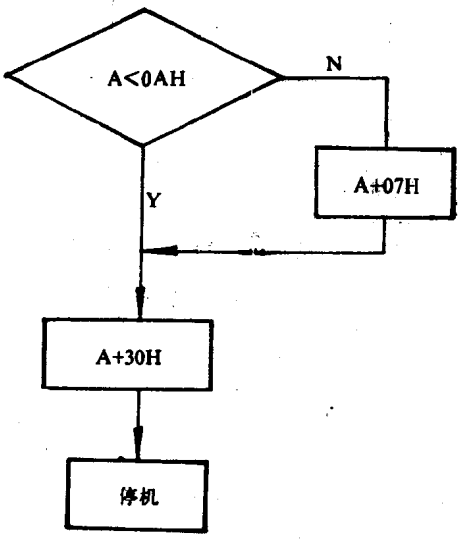

图1-4　例1-2的程序流程图

例1-2的源程序及相应的机器码如下所示：