

苏州大学实验报告

院、系	计算机学院	年级专业	21 计科	姓名	方浩楠	学号	2127405048
课程名称	计算机组成原理					成绩	
指导教师	张春生	同组实验者	无	实验日期	2023.5.10		

实验名称 微程序控制器实验

一. 实验目的

掌握微程序控制器的组成原理。

二. 实验设备

PC 机一台, TD-CMA 实验系统一套。

排线: 8 芯 7 根、6 芯 1 根、4 芯 3 根、2 芯 10 根。

三. 实验内容

微程序的编制、写入, 观察微程序的运行过程。

四. 实验原理

微程序控制器的基本任务是完成当前指令的翻译和执行, 即将当前指令的功能转换成可以控制的硬件逻辑部件工作的微命令序列, 完成数据传送和各种处理操作。它的执行方法就是将控制各部件动作的微命令的集合进行编码, 即将微命令的集合仿照机器指令一样, 用数字代码的形式表示, 这种表示称为微指令。这样就可以用一个微指令序列表示一条机器指令, 这种微指令序列称为微程序。微程序存储在一种专用的存储器中, 称为控制存储器, 微程序控制器原理框图如图 3-2-1 所示。

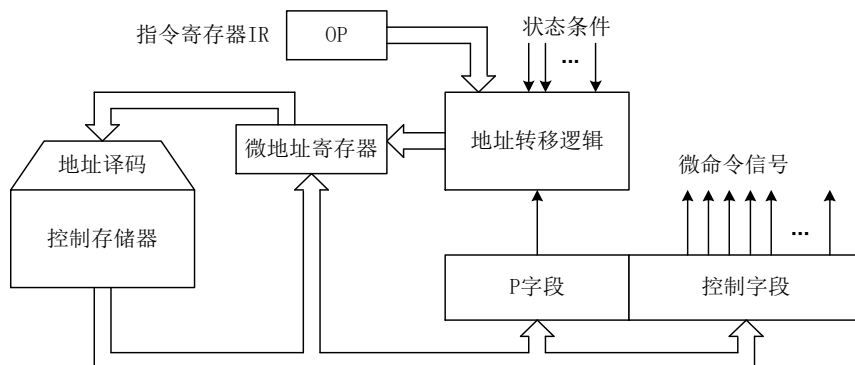
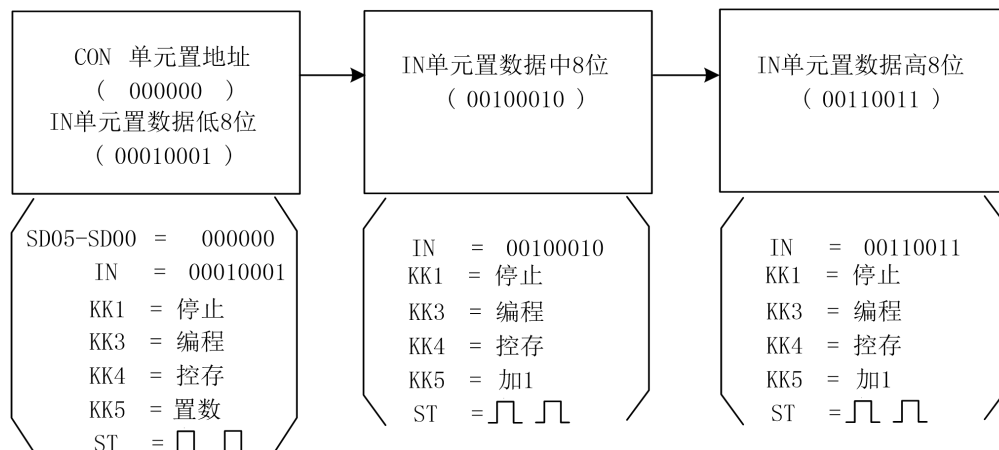


图 3-2-1 微程序控制器组成原理框图

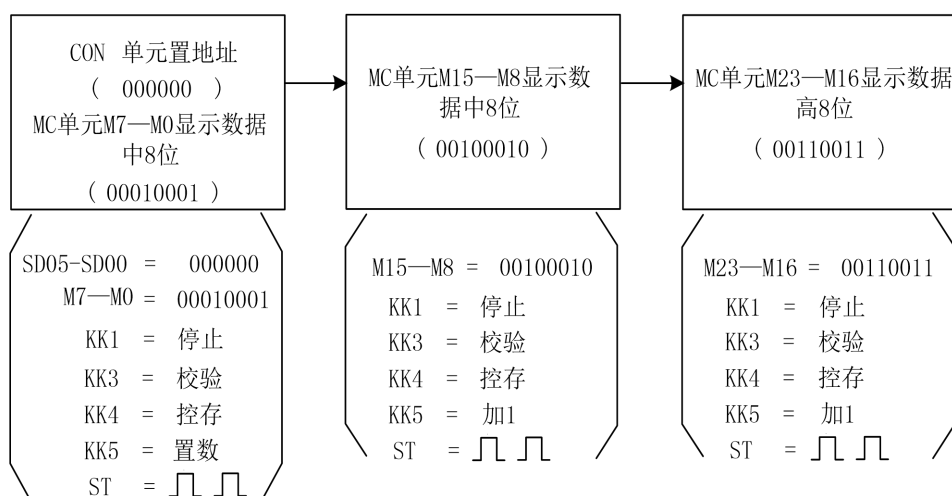
控制器是严格按照系统时序来工作的, 因而时序控制对于控制器的设计是非常重要的, 从前面的实验可以很清楚地了解时序电路的工作原理, 本实验所用的时序由时序单元来提供, 分为四拍 TS1、TS2、TS3、TS4, 时序单元的介绍见附录 2。

微程序控制器的组成见图 3-2-2, 其中控制存储器采用 3 片 2816 的 E²PROM, 具有掉电保护功能, 微命令寄存器 18 位, 用两片 8D 触发器 (273) 和一片 4D (175) 触发器组成。微地址寄存器 6 位, 用三片正沿触发的双 D 触发器 (74) 组成, 它们带有清“0”端和预置端。在不判别测试的情况下, T2 时刻打入微地址寄存器的内容即为下一条微指令地址。当 T4 时刻进行测试判别时, 转移逻辑满足条件后输出的负脉冲通过强置端将某一触发器置为“1”状态, 完成地址修改。

前地址 (000000), M7——M0 显示当前数据 (00010001)。然后将 KK5 拨至‘加 1’档, IN 单元开关给出该控存单元数据的中 8 位 (00100010), 连续两次按动开关 ST, 完成对该控存单元中 8 位数据的修改, 此时 MC 单元的指示灯 MA5——MA0 显示当前地址 (000000), M15——M8 显示当前数据 (00100010); 再由 IN 单元开关给出该控存单元数据的高 8 位 (00110011), 连续两次按动开关 ST, 完成对该控存单元高 8 位数据的修改此时 MC 单元的指示灯 MA5——MA0 显示当前地址 (000000), M23——M16 显示当前数据 (00110011)。此时被编辑的控存单元地址会自动加 1 (01H), 由 IN 单元开关依次给出该控存单元数据的低 8 位、中 8 位和高 8 位配合每次开关 ST 的两次按动, 即可完成对后续单元的编辑。



编辑完成后需进行校验, 以确保编辑的正确。以校验 00H 单元为例, 对于控制存储器进行校验的具体操作步骤如下: 首先将 KK1 拨至‘停止’档、KK3 拨至‘校验’档、KK4 拨至‘控存’档、KK5 拨至‘置数’档。由 CON 单元的 SD05——SD00 开关给出需要校验的控存单元地址 (000000), 连续两次按动开关 ST, MC 单元指示灯 M7——M0 显示该单元低 8 位数据 (00010001); KK5 拨至‘加 1’档, 再连续两次按动开关 ST, MC 单元指示灯 M15——M8 显示该单元中 8 位数据 (00100010); 再连续两次按动开关 ST, MC 单元指示灯 M23——M16 显示该单元高 8 位数据 (00110011)。再连续两次按动开关 ST, 地址加 1, MC 单元指示灯 M7——M0 显示 01H 单元低 8 位数据。如校验的微指令出错, 则返回输入操作, 修改该单元的数据后再进行校验, 直至确认输入的微代码全部准确无误为止, 完成对微指令的输入。



位于实验平台 MC 单元左上角一列三个指示灯 MC2、MC1、MC0 用来指示当前操作的微程序字, 分别对应 M23——M16、M15——M8、M7——M0。实验平台提供了比较灵活的手动操作方式,

比如在上述操作中在对地址置数后将开关 KK4 拨至‘减 1’档，则每次随着开关 ST 的两次拨动操作，字节数依次从高 8 位到低 8 位递减，减至低 8 位后，再按动两次开关 ST，微地址会自动减一，继续对下一个单元的操作。

微指令字长共 24 位，控制位顺序如表 3-2-1：\

23	22	21	20	19	18-15	14-12	11-9	8-6	5-0
M23	M22	WR	RD	IOM	S3-S0	A字段	B字段	C字段	MA5-MA0

A字段

14	13	12	选择
0	0	0	NOP
0	0	1	LDA
0	1	0	LDB
0	1	1	LDR0
1	0	0	保留
1	0	1	保留
1	1	0	保留
1	1	1	LDIR

B字段

11	10	9	选择
0	0	0	NOP
0	0	1	ALU_B
0	1	0	R0_B
0	1	1	保留
1	0	0	保留
1	0	1	保留
1	1	0	保留
1	1	1	保留

C字段

8	7	6	选择
0	0	0	NOP
0	0	1	P<1>
0	1	0	保留
0	1	1	保留
1	0	0	保留
1	0	1	保留
1	1	0	保留
1	1	1	保留

表 3-2-1 微指令格式

其中 MA5...MA0 为 6 位的后续微地址，A、B、C 为三个译码字段，分别由三个控制位译码出多位。C 字段中的 P<1>为测试字位。其功能是根据机器指令及相应微代码进行译码，使微程序转入相应的微地址入口，从而实现完成对指令的识别，并实现微程序的分支，本系统上的指令译码原理如图 3-2-3 所示，图中 I7...I2 为指令寄存器的第 7...2 位输出，SE5...SE0 为微控器单元微地址锁存器的强置端输出，指令译码逻辑在 IR 单元的 INS_DEC (GAL20V8) 中实现。

从图 3-2-2 中也可以看出，微控器产生的控制信号比表 3-2-1 中的要多，这是因为实验的不同，所需的控制信号也不一样，本实验只用了部分的控制信号。

本实验除了用到指令寄存器（IR）和通用寄存器 R0 外，还要用到 IN 和 OUT 单元，从微控器出来的信号中只有 IOM、WR 和 RD 三个信号，所以对这两个单元的读写信号还应先经过译码，其译码原理如图 3-2-4 所示。IR 单元的原理图如图 3-2-5 所示，R0 单元原理如图 3-2-7 所示，IN 单元的原理图见图 2-1-3 所示，OUT 单元的原理图见图 3-2-6 所示。

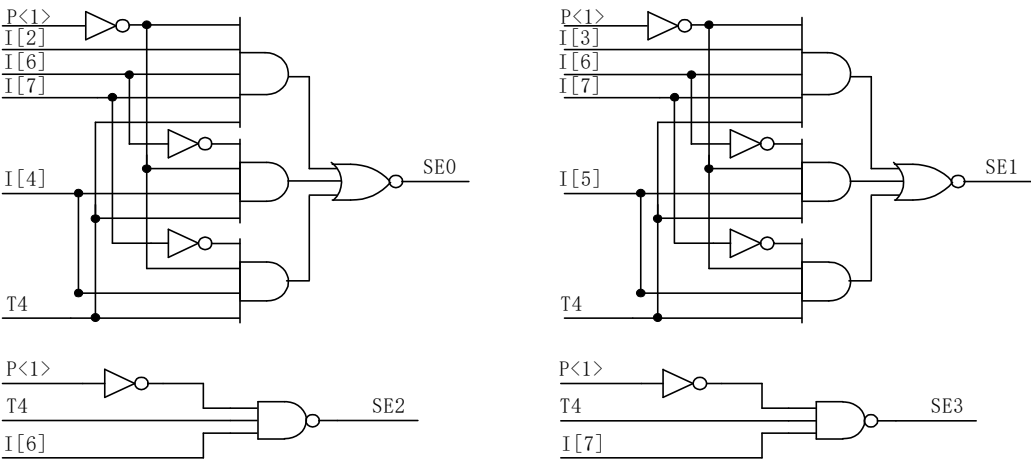


图 3-2-3 指令译码原理图

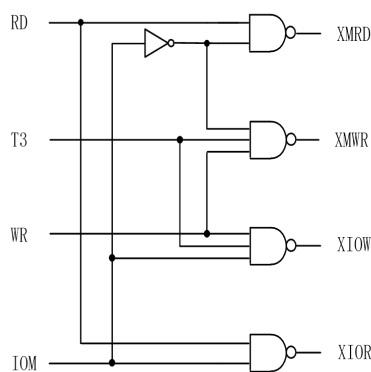


图 3-2-4 读写控制逻辑

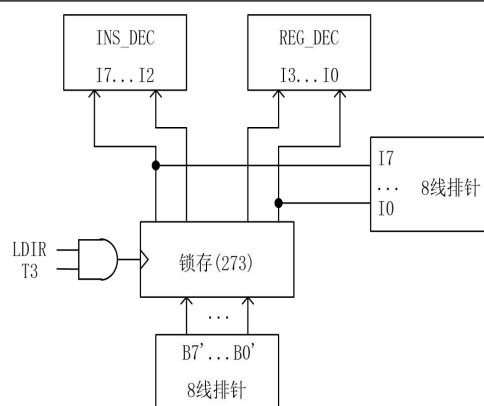


图 3-2-5 IR 单元原理图

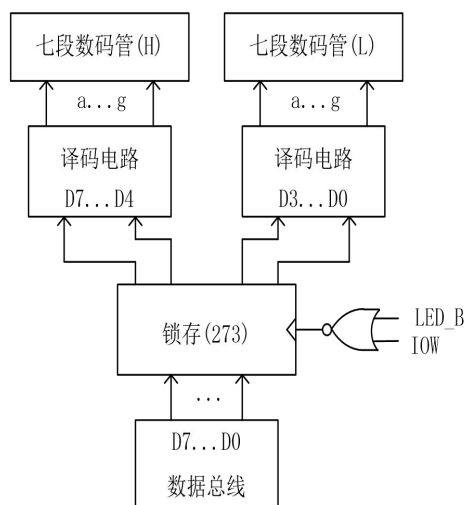


图 3-2-6 OUT 单元原理图

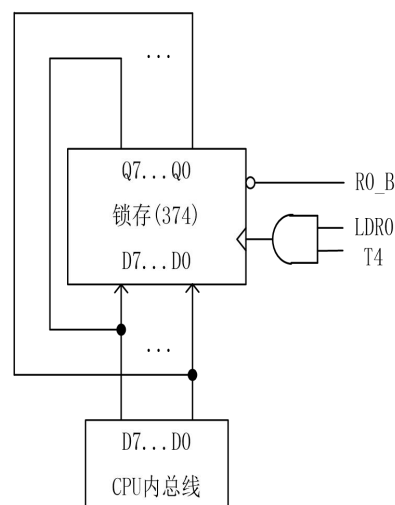


图 3-2-7 R0 原理图

本实验安排了四条机器指令，分别为 ADD (0000 0000)、IN (0010 0000)、OUT (0011 0000) 和 HLT (0101 0000)，括号中为各指令的二进制代码，指令格式如下：

助记符	机器指令码	说明
IN	0010 0000	IN → R0
ADD	0000 0000	R0 + R0 → R0
OUT	0011 0000	R0 → OUT
HLT	0101 0000	停机

实验中机器指令由 CON 单元的二进制开关手动给出，其余单元的控制信号均由微程序控制器自动产生，为此可以设计出相应的数据通路图，见图 3-2-8 所示。

几条机器指令对应的参考微程序流程图如图 3-2-9 所示。图中一个矩形方框表示一条微指令，方框中的内容为该指令执行的微操作，右上角的数字是该条指令的微地址，右下角的数字是该条指令的后续微地址，所有微地址均用 16 进制表示。向下的箭头指出了下一条要执行的指令。P<1>为测试字，根据条件使微程序产生分支。

33	28 04 01	00101	0000	000	010	000	000001
35	00 00 35	00000	0000	000	000	000	110101

表 3-2-2 二进制微代码表

五. 实验步骤与结果

1. 按图 3-2-10 所示连接实验线路，仔细查线无误后接通电源。如果有‘滴’报警声，说明总线有竞争现象，应关闭电源，检查接线，直到错误排除。

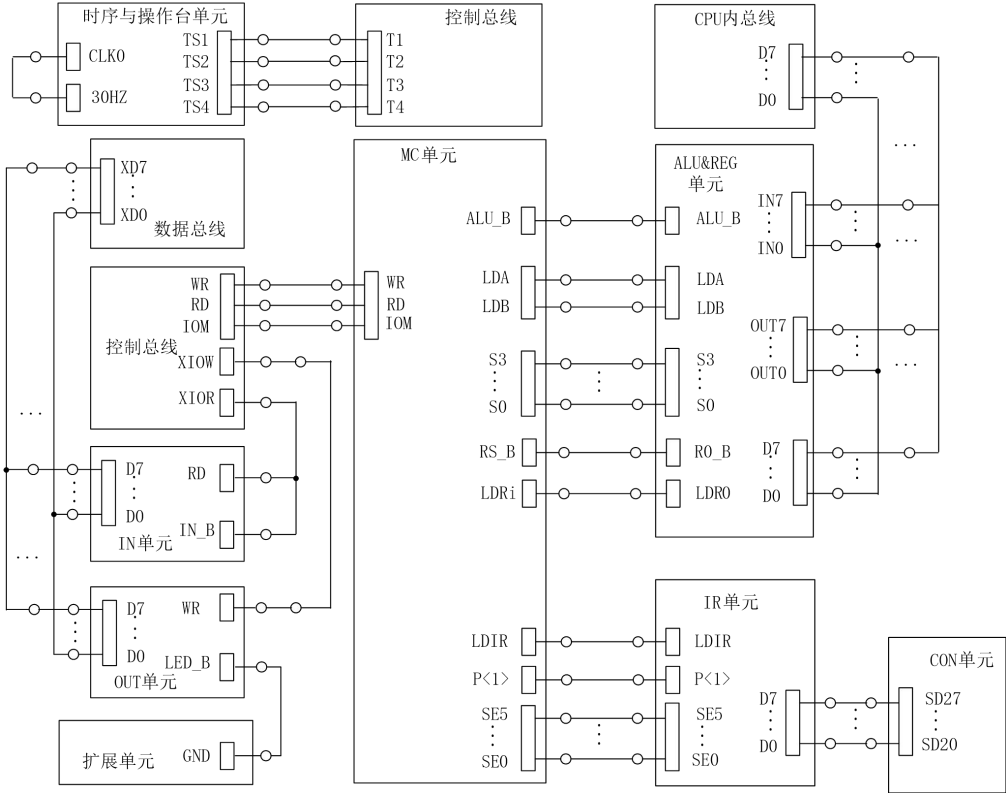


图 3-2-10-A 实验接线图

排线: 8 芯 7 根、6 芯 1 根、4 芯 3 根、2 芯 8 根

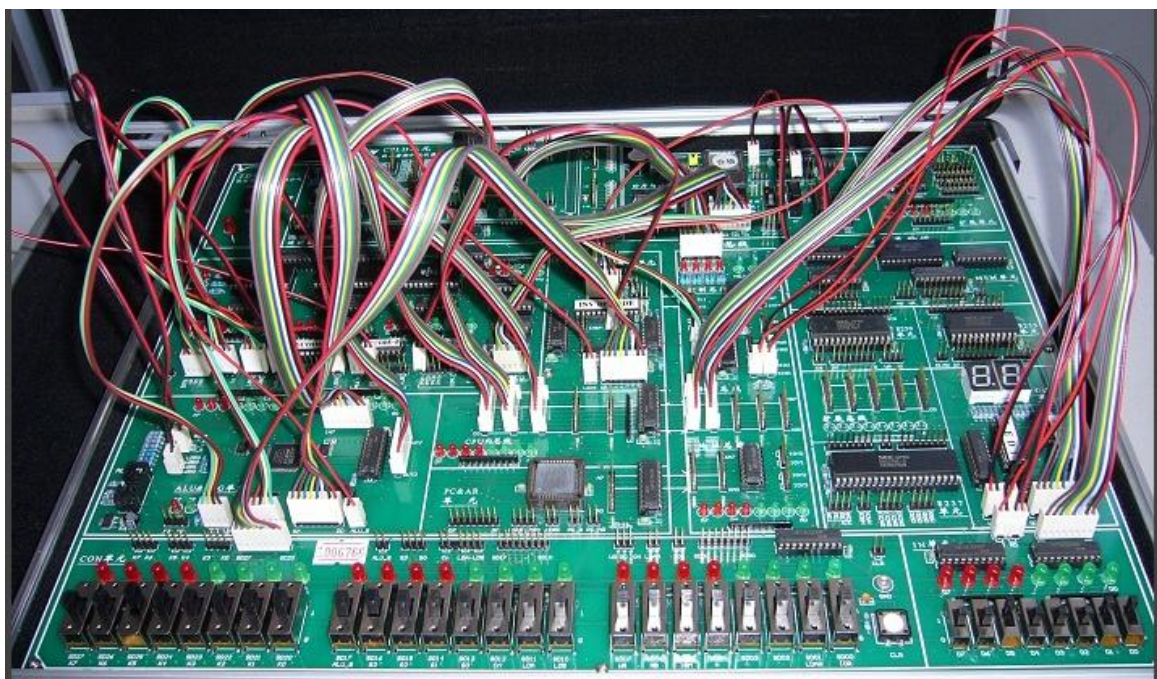


图 3-2-10-B 实验接线图

排线: 8 芯 7 根、6 芯 1 根、4 芯 3 根、2 芯 10 根

2. 对微控器进行读写操作, 分两种情况: 手动读写和联机读写。

1) 手动读写

(1) 手动对微控器进行编程 (写), 例如: 在微地址 00H 中写入微代码 000001H。

① 将时序与操作台单元的开关 KK1 置为 ‘停止’ 档, KK3 置为 ‘编程’ 档, KK4 置为 ‘控存’ 档, KK5 置为 ‘置数’ 档。

② 使用 CON 单元的 SD05——SD00 给出微地址 (00),

IN 单元给出低 8 位应写入的数据 (00000001), 连续两次按动时序与操作台的开关 ST, 将 IN 单元的数据写到该单元的低 8 位。

观察 MC 单元的 M7——M0 与 MC0 (低位)。

③ 将时序与操作台单元的开关 KK5 置为 ‘加 1’ 档。

④ IN 单元给出中 8 位应写入的数据 (00000000), 连续两次按动时序与操作台的开关 ST, 将 IN 单元的数据写到该单元的中 8 位。

观察 MC 单元的 M15——M8 与 MC1(中位)。

IN 单元给出高 8 位应写入的数据 (00000000), 连续两次按动时序与操作台的开关 ST, 将 IN 单元的数据写到该单元的高 8 位。

观察 MC 单元的 M23——M16 与 MC2 (高位)。

⑤ 重复①、②、③、④四步, 将表 3-2-2 的微代码写入 2816 芯片中。

(2) 手动对微控器进行校验 (读), 例如: 在微地址 03H 中读出微代码 107070H。

① 将时序与操作台单元的开关 KK1 置为 ‘停止’ 档, KK3 置为 ‘校验’ 档, KK4 置为 ‘控存’ 档, KK5 置为 ‘置数’ 档。

② 使用 CON 单元的 SD05——SD00 给出微地址 (03),

连续两次按动时序与操作台的开关 ST,

MC 单元的指数数据指示灯 M7——M0 显示该单元的低 8 位 (01110000)。

③ 将时序与操作台单元的开关 KK5 置为 ‘加 1’ 档。

④ 连续两次按动时序与操作台的开关 ST,

MC 单元的指数数据指示灯 M15——M8 显示该单元的中 8 位 (01110000)。

连续两次按动时序与操作台的开关 ST,

MC 单元的指数数据指示灯 M23——M16 显示该单元的高 8 位 (00010000)。

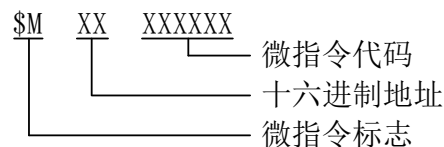
⑤ 重复①、②、③、④四步,完成对微代码的校验。如果校验出微代码写入错误,重新写入、校验,直至确认微指令的输入无误为止。

2) 联机读写

(1) 将微程序写入文件

联机软件提供了微程序下载功能,以代替手动读写微控制器,但微程序得以指定的格式写入到以 TXT 为后缀的文件中,微程序的格式如下:

微指令格式说明:



如\$M 1F 112233,表示微指令的地址为 1FH,微指令值为 11H (高)、22H (中)、33H (低),本次实验的微程序如下,其中分号‘;’为注释符,分号后面的内容在下载时将被忽略掉。

(2) 写入微程序

单击【开始】/【程序】/TangDu/CMA/CMA 的程序如图 1-1-6 所示。

用联机软件的“【转储】—【装载 TangDu/CMA/CMA/Sample/微程序控制实验.Txt】”功能将该格式 (*.TXT) 文件装载入实验系统。装入过程中,在软件的输出区的‘结果’栏会显示装载信息,如当前正在装载的是机器指令还是微指令,还剩多少条指令等。

(3) 校验微程序

选择联机软件的“【转储】—【刷新指令区】”可以读出下位机所有的机器指令和微指令,并在指令区显示。检查微控制器相应地址单元的数据是否和表 3-2-2 中的十六进制数据相同,如果不同,则说明写入操作失败,应重新写入,可以通过联机软件单独修改某个单元的微指令,先用鼠标左键单击指令区的‘微存’,然后再单击需修改单元的数据,此时该单元变为编辑框,输入 6 位修改数据并回车,编辑框消失,并以红色显示写入的数据。

```
; //***** //
; // //
; //      微控制器实验指令文件      //
; // //
; //      By TangDu CO.,LTD           //
; // //
; //***** //
; //**** Start Of MicroController Data **** //
$M 00 000001 ; NOP
$M 01 007070 ; CON(INS)->IR, P<1>
$M 04 002405 ; R0->B
$M 05 04B201 ; A 加 B->R0
$M 30 001404 ; R0->A
$M 32 183001 ; IN->R0
$M 33 280401 ; R0->OUT
$M 35 000035 ; NOP
; //***** End Of MicroController Data ***** //
```

3. 运行微程序

运行时也分两种情况:本机运行和联机运行。

1) 本机运行

① 将时序与操作台单元的开关 KK1、KK3 置为‘运行’档，按动 CON 单元的 CLR 按钮，将微地址寄存器（MAR）清零，同时也将指令寄存器（IR）、ALU 单元的暂存器 A 和暂存器 B 清零。

② 将时序与操作台单元的开关 KK2 置为‘单拍’档，然后按动 ST 按钮，体会系统在 T1、T2、T3、T4 节拍中各做的工作。T2 节拍微控制器将后续微地址（下条执行的微指令的地址）打入微地址寄存器，当前微指令打入微指令寄存器，并产生执行部件相应的控制信号；T3、T4 节拍根据 T2 节拍产生的控制信号做出相应的执行动作，如果测试位有效，还要根据机器指令及当前微地址寄存器中的内容进行译码，使微程序转入相应的微地址入口，实现微程序的分支。

③ 按动 CON 单元的 CLR 按钮，清微地址寄存器（MAR）等，并将时序与单元的开关 KK2 置为‘单步’档。

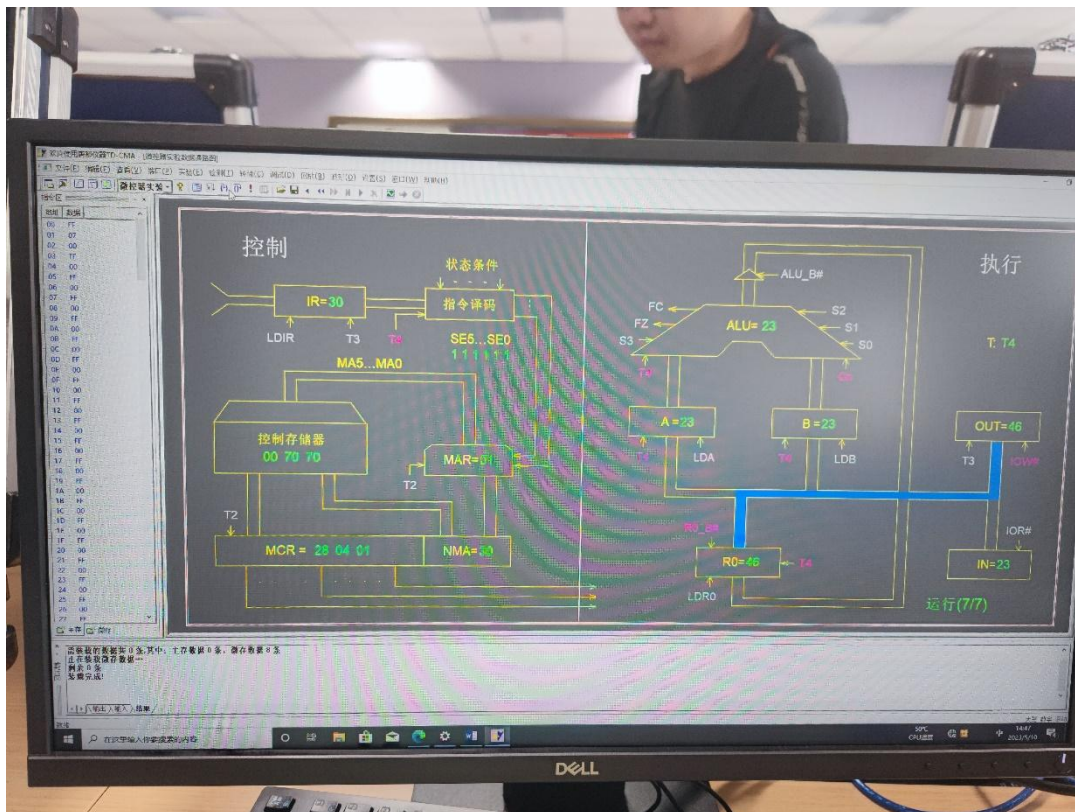
④ 置 IN 单元数据为 00100011，按动 ST 按钮，当 MC 单元后续微地址显示为 000001 时，在 CON 单元的 SD27...SD20 模拟给出 IN 指令 00100000 并继续单步执行，当 MC 单元后续微地址显示为 000001 时，说明当前指令已执行完；在 CON 单元的 SD27...SD20 给出 ADD 指令 00000000，该指令将会在下个 T3 被打入指令寄存器（IR），它将 R0 中的数据和其自身相加后送 R0；接下来在 CON 单元的 SD27...SD20 给出 OUT 指令 00110000 并继续单步执行，在 MC 单元后续微地址显示为 000001 时，观察 OUT 单元的显示值是否为 01000110。

2) 联机运行

联机运行时，进入软件界面，在菜单上选择【实验】—【微控器实验】，打开本实验的数据通路图，也可以通过工具栏上的下拉框打开数据通路图，数据通路图如图 3-2-8 所示。

将时序与操作台单元的开关 KK1、KK3 置为‘运行’档，按动 CON 单元的总清开关后，按动软件中单节拍按钮，当后续微地址（通路图中的 MAR）为 000001 时，置 CON 单元 SD27...SD20，产生相应的机器指令，该指令将会在下个 T3 被打入指令寄存器（IR），在后面的节拍中将执行这条机器指令。仔细观察每条机器指令的执行过程，体会后续微地址被强制置转换的过程，这是计算机识别和执行指令的根基。也可以打开微程序流程图，跟踪显示每条机器指令的执行过程。

按本机运行的顺序给出数据和指令，观察最后的运算结果是否正确。



六. 实验总结

本次实验让我了解到了微程序控制器的实验原理