计算机组成原理实验指导书

Principles of Computer Organization Experiment Instruction Book

实验 4 微程序控制器实验

燕山大学软件工程系

实验 4 微程序控制器实验

4.1 实验目的

（1）掌握微程序控制器的组成原理和工作过程。

（2）理解微指令和微程序的概念，理解微指令与指令的区别和联系。

（3）掌握指令操作码与控制存储器中微程序的对应方法，熟悉根据指令操作码从控制存储器中读出微程序的过程。

4.2 实验要求

（1）做好实验预习，读懂实验电路图，熟悉实验元器件的功能特性和使用方法。

（2）按照实验内容与步骤的要求，独立思考，认真仔细地完成实验。

（3）写出实验报告。

4.3 实验原理

图 4.1 为实验电路图，其中 3 片 EPROM2716 构成控制存储器，1 片 74LS175 为微地址寄存器，与 74LS175 数据输入引脚相连的输入信号线及 6 个门电路构成了地址转移逻辑。注意，2716 输出信号中带后“#”的信号为低电平有效信号，不带后缀“#”的信号为高电平有效信号。为简化电路结构，本实验没有使用微命令寄存器，并且在虚拟实验系统中，将 3 片 EPROM 组合为一个虚拟 EPROM 组件。本实验使用的 EPROM 和时序发生器一样，均为虚拟实验系统提供的虚拟组件。

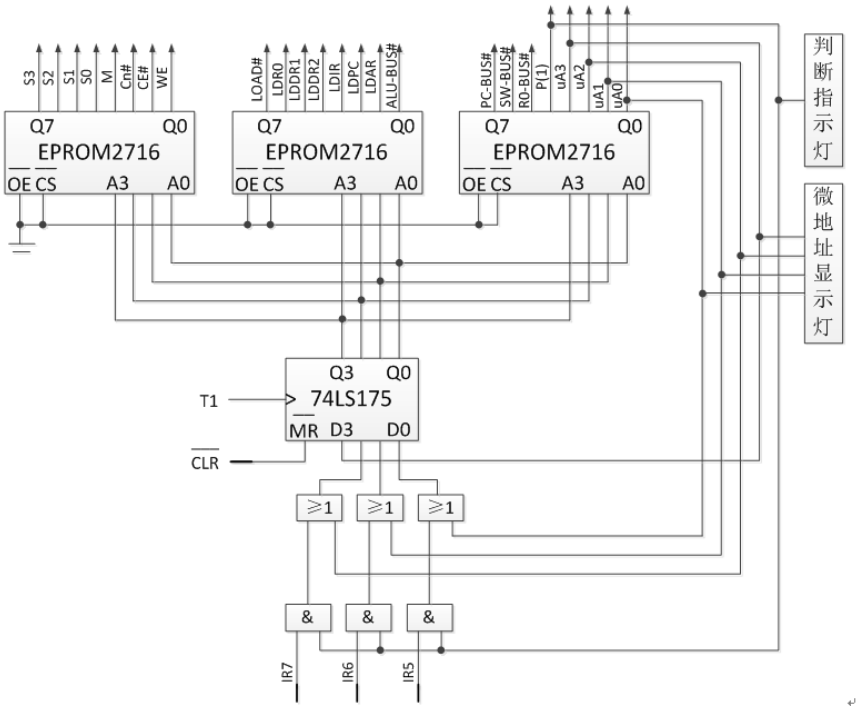

图 4.1 微程序控制器电路

实验电路中涉及的主要控制信号如下：

（1）$\overline{CE}$：2716 芯片的片选信号。为 0 时 2716 正常工作，实验中将其接地，恒置为 0。

（2）$\overline{OE}$：2716 读信号。$\overline{CE}$=0，$\overline{OE}$=0 时为读操作，实验中将其接地，恒置为 0。

（3）$\overline{CLR}$：芯片 74LS175 的清零信号，低电平有效。

（4）T1：微地址加载信号，在 T1 的上升沿将微地址锁存到 74LS175。

（5）IR5~IR7：指令操作码的输入信号，这几条信号线本应与指令寄存器的输出引脚相连，但在本实验中，与数据开关相连，指令操作码通过数据开关手动设置。

（6）$\overline{\text{LOAD}}$：PC 的置数信号，为 0 时 PC 工作在置数模式，可在此模式下为 PC 设置初值。

（7）LDR0：R0 的数据载入信号，为 1 时将数据存入 R0。

（8）LDIR：IR 的加载信号，为 1 时将指令锁存到 IR。

（9）LDPC：PC 的加载信号，为 1 时执行清零、置数或计数操作。

（10）$\overline{\text{PC-B}}$：PC 输出三态门使能信号，为 0 时将 PC 的值输出到总线。

（11）$\overline{\text{R0-B}}$：R0 芯片的输出控制信号，为 0 时将 R0 中的数据输出到总线。

在存储逻辑型计算机中，一条机器指令对应了一个微程序，不同的机器指令对应了不同的微程序，执行一条指令其实就是运行其对应的一个微程序，微程序由微指令组成，是微指令的有序集合。微程序是在设计一台计算机时就预先设计好并且固化在只读存储器中的，以后每当要执行某条指令时，只需找到并运行其对应的微程序。

控制存储器专门用于存放微程序，在本实验中，控制存储器由 3 片 EPROM2716 组成，为了减少连线的复杂度，虚拟实验系统把三片 EPROM2716 集成到一片芯片上，因此，本实验所用到的是 EPROM2716×3(2K×24 位)，其中地址输入引脚为 A10~A0，实验中仅用到 A3~A0，高 7 位地线 A4~A10 接地，实际存储容量为 16×3 字节。Q0~Q23 这 24 个输出引脚与 24 位的微指令相对应。

微指令格式如表 4-1 所示，采用全水平型，字长 24 位，其中操作控制字段 19 位，全部采用直接表示法，不使用译码器，每一位表示一个微命令，用于发出全机的操作控制信号；顺序控制字段 5 位，包括后续微地址 μA3~μA0 和判别位 P1，用于决定下一条微指令的地址。

表 4-1 微指令格式

位	23	22	21	20	19	18	17	16	15	14	13	12
控制信号	S3	S2	S1	S0	M	$\overline{\text{Cn}}$	$\overline{\text{CE}}$	WE	$\overline{\text{LOAD}}$	LDR0	LDDR1	LDDR2

位	11	10	9	8	7	6	5	4	3	2	1	0
控制信号	LDIR	LDPC	LDAR	$\overline{\text{ALU-B}}$	$\overline{\text{PC-B}}$	$\overline{\text{SW-B}}$	$\overline{\text{R0-B}}$	P(1)	μA3	μA2	μA1	μA0

地址转移逻辑电路用于产生下一条微指令的地址，主要由两级与门、或门构成。地址转移逻辑需要用到的数据信号有：后续微地址 μA3~μA0、判别位 P1、指令操作码 IR7~IR5。当判别位 P1=0 时，下一条微指的地址即为后续微地址 μA3~μA0；当判别位 P1=1 时，下一条微指令的地址由指令操作码 IR7~IR5 决定，一般是将操作码进行简单变换，把变换后的值作为下一条微指令的地址，此地址就是该操作码对应的微程序的入口地址。

微地址寄存器 74LS175 为控制存储器提供微指令地址，当 $\overline{\text{CLR}}$ =0 时，微地址寄存器清零，从控制存储器 00H 地址开始执行微程序，地址转移逻辑生成下一条微指令的地址。此后，每当 T1 上升沿到来时，新的微指令地址会打入微地址寄存器，控制存储器随即输出这条微指令，地址转移逻辑继而生成下一条微指令的地址。如果时序信号连续发生，微指令也会按一定的顺序接连输出。

为了教学简单明了，本实验仅用到四条机器指令：IN(输入)、ADD(加法)、STA(存数)、JMP(无条件转移)，操作码分别为 000、001、010、011，指令格式如表 4-1 所示。

表 4-2 机器指令格式

助记符	机器码（A 为内存地址 8bit）	长度	功能
IN	000XXXXXX	8bit	SW→R0
ADD	001XXXXXX A	16bit	R0+(A)→R0
STA	010XXXXXX A	16bit	R0→(A)
JMP	011XXXXXX A	16bit	A→PC(程序跳转到 A 地址执行)

上述四条指的微程序流程设计如图 4.2 所示，其中一个方框就对应一条微指令，方框右上角的数字为八进制表示的微地址。一个方框也表示一个 CPU 周期，执行一条微指令需要一个 CPU 周期。四条指令对应四个微程序，每个微程序包括 N 条微指令，需要执行 N 个 CPU 周期。

图 4.2 中的每条微指令都按照表 4-1 的格式编写了二进制代码,并预存在控制存储器芯片 EPROM22716×3 中。其中部分微指令二进制代码如表 4-3 所示，注意：微地址用八进制表示。

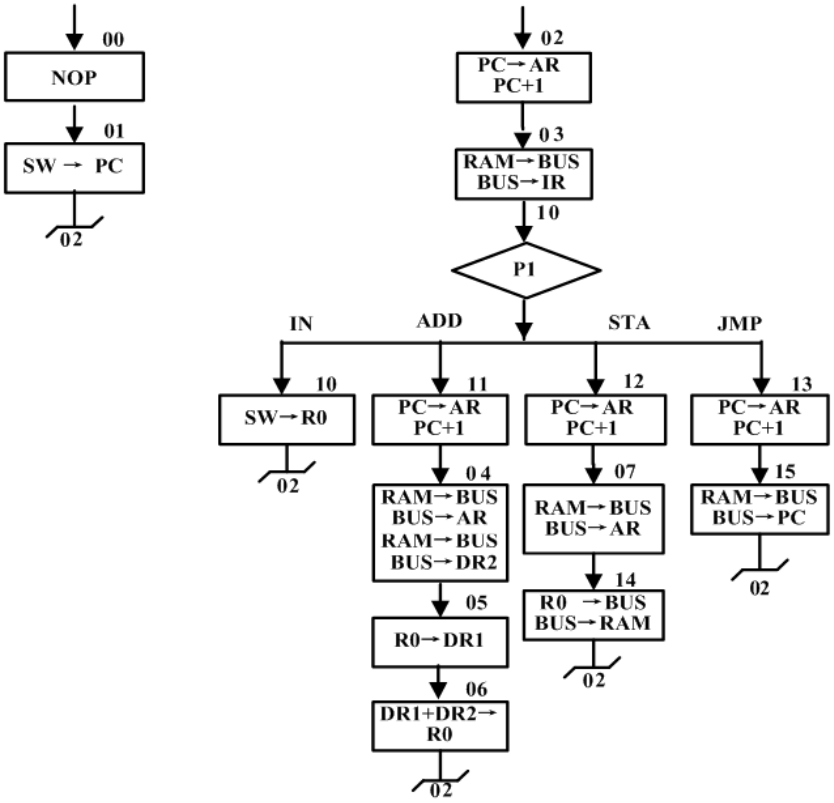

图 4.2 微程序流程图

4.4 实验内容与步骤

1. 运行虚拟实验系统，按照图 4.1 绘制实验电路，生成如图 4.3 所示电路。
2. 电路预设置：将 EPROM2716 芯片 $\overline{CE}$ 、$\overline{OE}$ 、A4、A5 引脚置 0，微地址寄存器 74LS175 的 $\overline{CLR}$ 置 0，时序发生器的 Step 置 1。
3. 打开电源。此时由于 $\overline{CLR}$ =0，微地址寄存器清零，给出微程序入口地址 00H，控制存储器随之输出第 00 号微指令。
4. 将 $\overline{CLR}$ 设置为 1，否则微地址寄存器会一直处于清零状态。
5. 将 IR7~IR5 均设置为 0，思考并回答问题：若此时连续不断地发出时序信号，微程序的

执行流程是怎样的？请按顺序写出前 10 条微指令的地址。

表 4-3 微程序二进制代码表

位	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
地址	S3	S2	S1	S0	M	Cn#	CF#	WF	LOAD#	LDR0	LDR1	LDR2	LDIR	LDPC	LDAR	ALU-B#	PC-B#	SW-B#	R0-B#	P(1)	μA3	μA2	μA1	μA0
00	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1
01	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0	1	0
02	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	0	0	0	1	1
03	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	1	1	1	1	1	1	0	0	0
04																								
05																								
06																								
07																								
10	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	1	0
11	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0
12	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	0	0	1	1	1
13	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	0	1	1	0	1
14																								
15																								

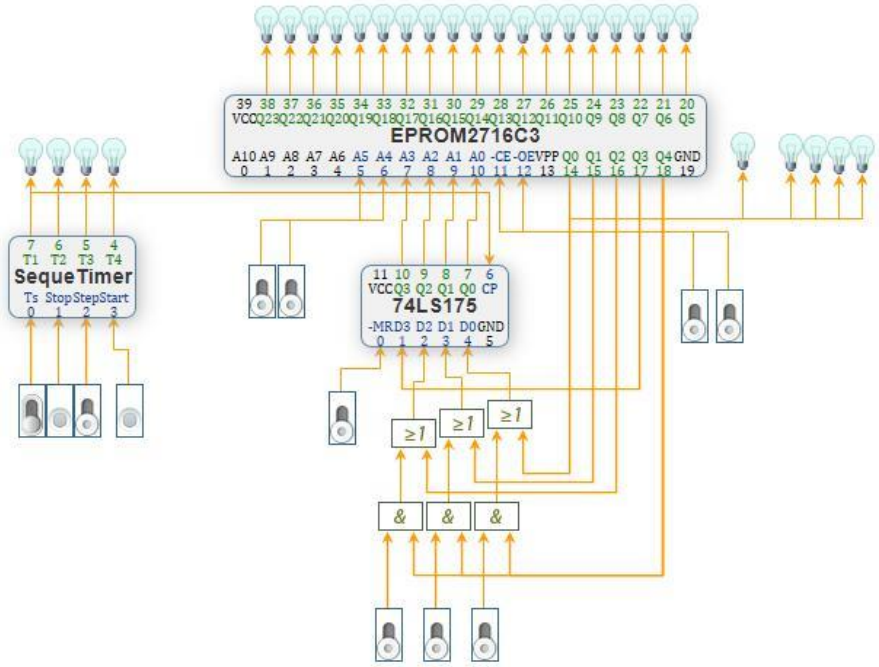

图 4.3 控制器虚拟实验电路

6. 连续单击 Start 按钮，观察微指令的输出顺序，检验控制存储器输出的微指令是否与表 4-3 中的相符，验证上一步预测的顺序是否正确。

7. 设置 IR7~IR5 的不同组合，用单步方式分别读出 ADD、STA 和 JMP 三条指令的微程序，用后续微地址和判别指示灯跟踪微程序执行及转移情况，将表 4-3 中缺少的微程序代码补充完整。

8. 思考并回答问题：若不改变控制器实验电路，IN、ADD、STA 和 JMP 四条指令的微程序在控制存储器中的存放位置是否可以随意安排？有什么限制？为什么？

4.5 实验结果

本实验需要记录的结果是回答 4.4 节实验内容与步骤中，第 5、7、8 步提出的问题：

5. 答：

7. 将表 4-3 补充完整。

8. 答：

4.6 思考与分析

1. 微程序控制器主要由哪些部件组成？各部件的功能是什么？

2. 本实验中，地址转移逻辑电路是怎样利用判别测试字段（P 字段）实现微程序分支的？

3. 如果把微程序控制器看作一个黑盒子，那么它的输入信号有哪些？这些信号是哪些部件提供给它的？它的输出信号有哪些？这些信号是发送给哪些部件的？