

# 上海大学 计算机学院

## 《计算机组成原理实验》报告五

姓名 XXXXXXXX 学号 XXXXXXXX

时间 周二 9-11 机位 14 指导教师 刘跃军

---

实验名称: 指令系统实验

### 一、实验目的

1. 读出系统已有的指令，并理解其含义。
2. 设计并实现一条新指令。

### 二、实验原理

#### 1. 微程序和机器指令：

通过前四次实验我们知道：一个计算机系统的硬件制作好以后，它的最基本操作就确定了，即它的微指令系统就确定了。

我们必须有序地使用这一系列微指令才能完成一个实际需要的操作。在这些“需要”的操作中，有一部分是使用频度很高，而且数条微指令即可完成的简单操作。为方便使用，系统就把这部分简单操作的微指令序列固定下来，存放在一个快速存储器中，这些微指令序列称为“微程序”，而这个快速存储器就是微程序存储器  $\mu\text{EM}$ 。显然，仅有一条微指令组成的微程序是最简单的微程序。

用户在使用计算机时就以“微程序”为单位来排序简单操作，达到解决实际复杂操作的目的。我们可以用不同的“机器指令”或“指令”来调用不同的微程序，以达到完成不同指令的功能。我们将采用这种“硬件使用方式”的计算机系统称为微程序结构系统。

#### 2. 实验箱的机器指令系统：

在实验箱中，每条（机器）指令均由 4 条微指令（等长结构）组成。当实际需要的微指令数量不足 4 条时，我们将用无效微指令（FFFFFH）补足，但最后一条有效微指令一定是 CBFFFFH 微指令，表示本条指令执行完毕，取下一条指令。

由于所有指令的微程序均由 4 条微指令组成，且头尾相接地存放在微程序存储器  $\mu\text{EM}$  中。所以，每个微程序的起始地址（也叫入口地址）的最后两位一定为 00。

同时，这个微程序的起始地址就自然成为这条（机器）指令的二进制数表

达形式。如下图所示。

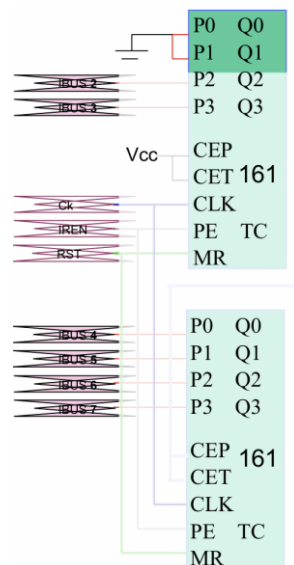


图 1. IBUS 不需要 0 号线和 1 号线

### 3. 实验箱机器指令系统的布线:

#### (1) 存储器 (EM) 布线和 IBUS 信号的生成:

RAM6116 送出的数据总上 IBUS 去寻址  $\mu$ M 微程序存储器。低两位去寻址寄存器 R0 ~ R3, 形成 101 页表中的“机器码 1 的最后 2 位”。

ABUS 的数据来自 PC (取指令) 或 MAR (读写数据) 当数据来自 MAR, 且 EMEN = 0 时, 则数据通过上边的 245 送上数据总线, 此时系统使  $\mu$ PC 的 IREN = 1, 所以 IBUS 上的值不影响  $\mu$ PC。

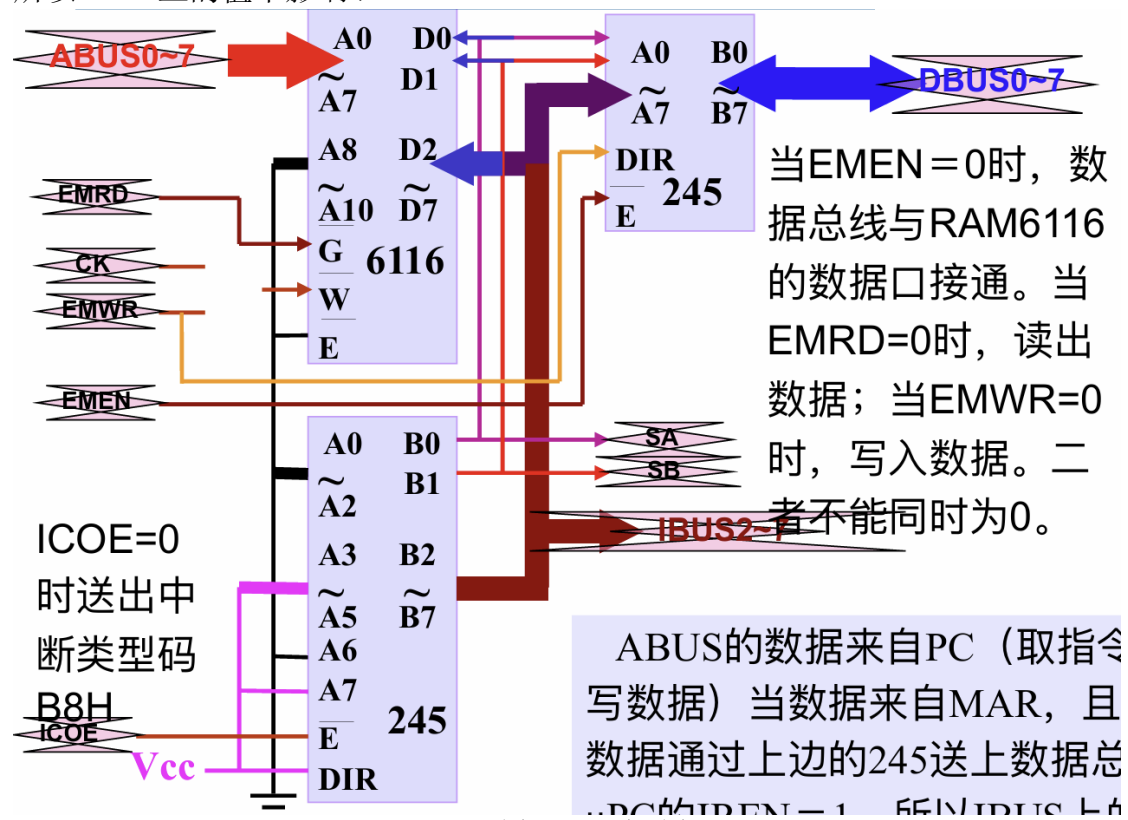


图 2. EM 原理图

## (2) PC 布线和 ABUS 信号的生成:

(1) PCOE=0 时, CK 上升沿使 PC 寄存器的值加 1, 并立即通过下部的 245 送上地址总线(ABUS), 去寻址存放指令的存储器 EM。这是 PC 的主功能。

(2) LDPC=0 时, CK 上升沿将数据总线(DBUS)的值打入 PC。

(3) PCOE-D=0 时, PC 值立刻送上数据总线。

(4) RST=0 时, 使 PC 清 0。

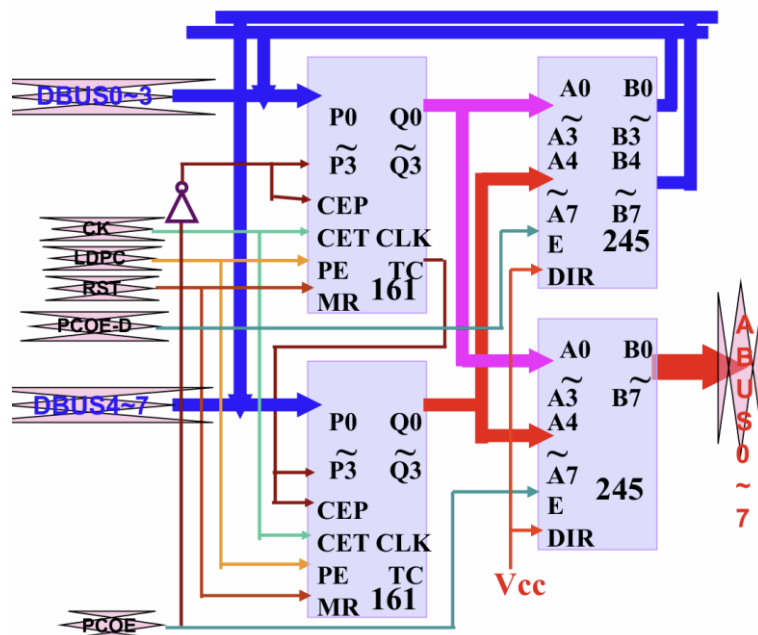


图 3. PC 原理图

## 4. 实验箱机器指令系统的工作原理:

一条指令的执行是从 PC 开始, PC 通过地址总线(ABUS)发送一条指令在存储器中的地址给存储器, 存储器依据该地址单元的值(即: 指令的二进制数形式)的高 6 位通过 IBUS 送到  $\mu$ PC, 低两位送 SA 和 SB,  $\mu$ PC 得到的这 6 位值就是这条指令的微程序入口地址的高 6 位,  $\mu$ PC 据此生成微程序存储器( $\mu$ EM)的地址, 并通过  $\mu$ PC 总线送到  $\mu$ EM,  $\mu$ EM 依据  $\mu$ PC 值(即: 微地址)将选定单元的 24 位控制信号送上控制总线 CBUS, 执行第一条微指令的功能; 同时  $\mu$ PC 加 1, 输出这条指令的第二条微指令, 直到执行一条“取下一条指令”的微指令(即: CBFFFFH), 此时 PC 加 1, 开始执行下一条指令。

## 5. PC 寄存器的打入原理:

一条指令执行完毕后 PC 会自动加 1, 系统顺序执行下一条指令, 但当系统需要进入一个新的指令序列时(如: 跳转、转子程序等), 必须给 PC 打入新的起始值——新指令序列的入口地址。实验箱用图 3 的电路实现把数据总线的值(即: 新指令序列的入口地址)打入 PC 的操作, 以更新 PC 值。图中 151 是 8 选 1 输出器, 选择功能如表 1。

表 1. 151 选择器

表1 151选择表

C	B	A	Z
ELP	IR3	IR2	LDPC
0	0	0	Cy
0	0	1	Z
0	1	0	0
0	1	1	0
1	X	X	1

由表 1 可知：当 ELP=1 时（即：LDPC=1），8 选 1 输出器 151 无效，PC 的值不会被修改。当 ELP=0 时，PC 值将分以下三种情况被修改：

（1）当 IR3、IR2=00，Cy(进位标志)有效时，则修改 PC；（2）当 IR3、IR2=01，Z(0 标志)有效时，则修改 PC；（3）当 IR3、IR2=10 或 11 时，则无条件修改 PC。

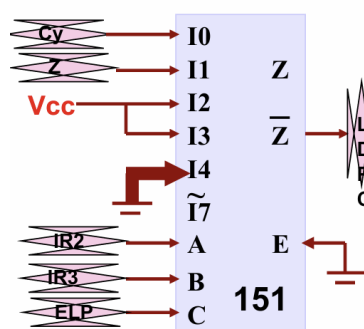


图 4. PC 打入电路

## 6. 程序存储器模式下的操作：

按 TV 键切换到程序存储器模式，在此模式下可以观察和修改程序存储器的内容，实现把一个指令放在指定地址的存储单元中，然后当 PC 输出某个地址时，存储器将依据该地址取出相应地址单元的指令，并执行该指令。操作方法如下图所示。

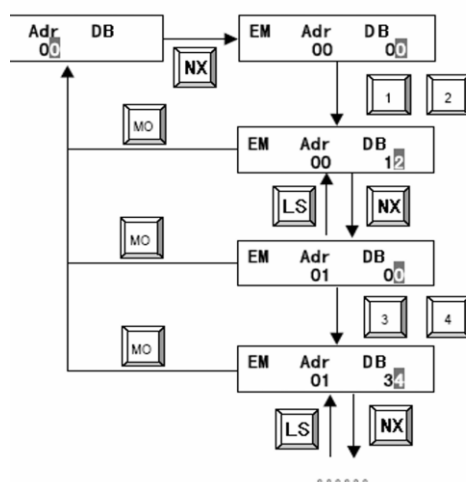


图 5. 程序存储器模式的操作方法

### 三、实验内容

#### 1. 实验任务一

(考察机器指令码为 64H 的各微指令信号, 验证该指令的功能。 修改机器指令码为 E8H 的功能, 使其完成“输出 A+W 的结果左移一位后的值到 OUT”操作。(假设 A=03H, R0=77H, 77 地址单元存放 06H 数据, 两题连起来做, 故写在一个报告的任务中)

##### (1) 实验步骤

因为本次实验不需要用到导线, 所以不需要对导线能否正常工作正常检验。

对于实验一的部分, 先按两下 TV/ME 键进入微程序存储器模式, 可以看到目前显示屏上“Adr”一栏数值为 00 并有光标跳动, 在小键盘上输入 64, 再按下 NEXT 键, 可以看到 MH、MM、ML 对应变成了 FF、77、FF, 将其转换为 24 位二进制数, 即为: C23~C0=1111 1111 0111 0111 1111 1111, 说明了 RRD=0, MAREN=0, 故可以分析出其作用是将 R0~R3 中的某个寄存器内容读出并写入 MAR 寄存器中。

(实验后测试发现, 在内部寄存器模式中的  $\mu$ PC 打入的初始值分别为 64、65、66、67 时, 对应从 R0、R1、R2、R3 寄存器中读数)

接着按下 F2/NX 键, 此时跳转至 65 单元, 可以看到 MH、MM、ML 对应变成了 D7、BF、EF, 将其转换为 24 位二进制数, 即为: C23~C0=1101 0111 1011 1111 1110 1111, 说明了 EMEN=0, MAROE=0, WEN=0, FEN=0, 故可以分析出其作用是将 MAR 寄存器中的内容作为地址, 读出该地址中的数据并将其送到 W 寄存器中。

再次按下 F2/NX 键, 此时跳转至 66 单元, 可以看到 MH、MM、ML 对应变成了 FF、FE、92, 将其转换为 24 位二进制数, 即为: C23~C0=1111 1111 1111 1110 1001 0010, 说明了 X2 X1 X0=1 0 0, AEN=0, S2 S1 S0=0 1 0, FEN=0, 故可以分析出其作用是将 A 寄存器的数据与 W 寄存器的数据进行或运算 (A|W), 并将运算结果送入 A 寄存器。

再按下一次 F2/NX 键, 此时跳转至 67 单元, 可以看到 MH、MM、ML 对应变成了 CB、FF、FF, 将其转换为 24 位二进制数, 即为: C23~C0=1100 1011 1111 1111 1111 1111, 说明了 EMRD=0, PCOE=0, IREN=0, 故可以分析出其作用是执行下一条指令。

由以上四个地址单元各自的作用分析, 整合后可以得出, 将 R0~R3 中的某个寄存器内容读出并写入 MAR 寄存器中, 将 MAR 寄存器中的内容作为地址, 读出该地址中的数据并将其送到 W 寄存器中, 并将 W 寄存器与 A 寄存器进行逻辑或运算, 最终将运算结果送入 A 寄存器。

至此, 分析阶段结束, 因为实验要求中有一条为: 77 地址单元存放 06H 数据, 所以从微程序存储器模式再按 3 次 TV/ME 键, 调节到程序存储器模式, 输出 77, 按下 F2/NX 键, 输入 06, 至此, 实验一的预处理部分结束。

接着进入实验二的预处理部分。

---

实验二的要求为: 编制一条微指令实现“A 非”运算后右移一位的值送 OUT;

把这条微指令放入微程序存储器的 E8H 单元。为了避免在执行完这条微指令后，再次按 STEP 键导致执行了同一微指令下的其他操作，一共需要编写两条微指令，第一条用于实现该功能，第二条用于实现取下一条指令。

第一条微指令的功能是完成输出  $A+W$  的结果左移一位后的值到 OUT，S2 S1 所以需要满足以下条件：S0=0 0 0，X2 X1 X0=1 1 0，FEN=0，OUTEN=0。根据以上条件可写出对应的 24 位二进制数：C23~C0=1111 1111 1101 1110 1101 1000。将这个 24 位二进制数将换算成 16 进制可得：FF DE D8，故分析得出第一条微指令码为：FF DE D8。

第二条微指令的功能为取下一条指令，只需要满足以下条件：EMRD=0，PCOE=0，IREN=0。根据以上条件可写出对应的 24 位二进制数：C23~C0=1100 1011 1111 1111 1111 1111。将这个 24 位二进制数将换算成 16 进制可得：CB FF FF，故分析得出第二条微指令码为：CB FF FF。

至此，分析环节结束，进入实验二的操作部分。

先按数次 TV/ME 键进入微程序存储器模式，可以看到目前显示屏上“Adr”一栏数值为 00 并有光标跳动，在小键盘上输入 E8，按下 NEXT 键，将 FF DE D8 对应填入 MH、MM、ML 处，按下 F2/NX 键，显示 E9，将 CB FF FF 对应填入 MH、MM、ML 处。

因为两个实验要连起来操作，但是 64 单元与 E8 单元并非连起来的，所以再按数次 TV/ME 键进入程序存储器模式，在 Adr 输入 10，按下 F2/NX，再输入 E8。

---

两个实验预操作完成，现在进入收尾操作部分。

再按三次 TV/ME 键，进入内部寄存器模式，在  $\mu$ PC、PC、A、W 栏分别输入 64、10、03、00，再按两次/三次（有点记不清楚了）F2/NX 键，在 R0 一栏输入 77，接着慢慢按下 4 次 STEP 键，每两次之间停留一段时间观察现象。再按两次 STEP 键，同样停留一段时间观察现象。

## (2) 实验现象

在前四次按下 STEP 键中，可以依次观察到 MAR 变为 77，W 寄存器变为 06，D 寄存器变为 07，A 寄存器变为 07。

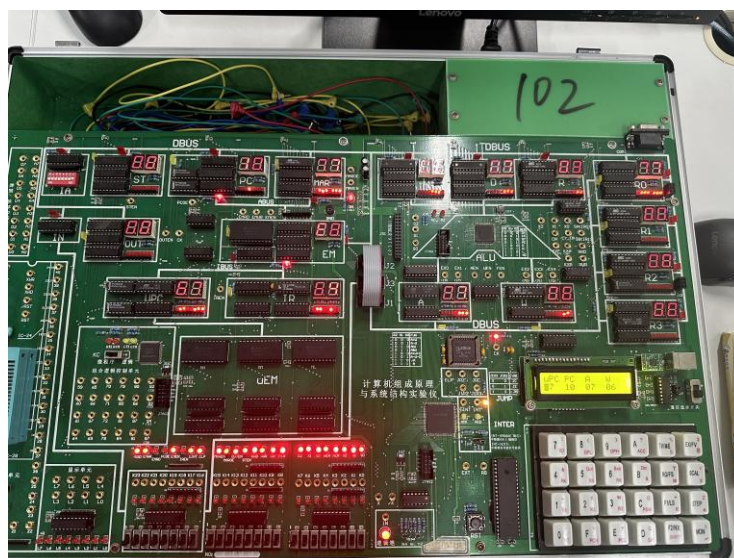


图 6. 实验一现象



再按下两次 STEP 键，在此过程中可以依次看到 D 寄存器变为 1A，OUT 寄存器变为 1A。

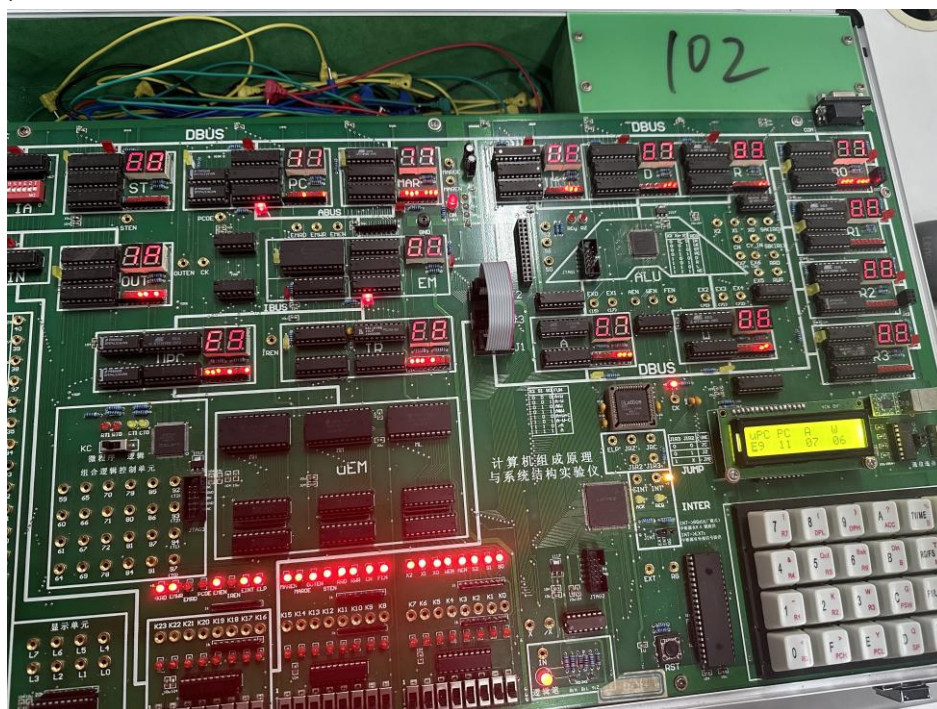


图 7. 实验二现象

### (3) 数据记录、分析与处理

实验步骤及现象中已经分析、处理、记录过了数据，在此不进行过多赘述。

### (4) 实验结论

微指令存储器地址为 64H 单元的控制功能为：“将 R0~R3 中的某个寄存器内容读出并写入 MAR 寄存器中，将 MAR 寄存器中的内容作为地址，读出该地址中的数据并将其送到 W 寄存器中，并将 W 寄存器与 A 寄存器进行逻辑或运算，最终将运算结果送入 A 寄存器”。

正确推出了用于实现用微指令实现“输出 A+W 的结果左移一位后的值到 OUT”的操作，并将其存入 E8 地址单元中，最终计算得出了 1A 的结果并送到了 OUT 寄存器中。

## 四、建议和体会

1、用小键盘操作感觉要比连线方便不少，至少不会因为某根导线无法正常工作或者某一根导线错连而导致需要重新来过==。

2、吸取了前一次的教训，提前预习，即在进行实验课前，需要对该节课实验的内容对应的理论知识（虽然好像与机组理论课关系不大）进行温故与复习并且对实验步骤进行大框架上的构想与思考，最好能够提前学会要用到的知识点。

3、TV/ME 真的非常非常重要！

## 五、思考题

Q：在微指令结构的计算机中，一条指令从启动到产生功能经过那些环节？

A：1、PC 通过 ABUS 把地址发送给存储器

2、存储器根据地址单元的值将高 6 位通过 IBUS 送到  $\mu$ PC，最低的两位送

SA 和 SB。

3、 $\mu$ PC 生成微程序存储器的地址，并送到  $\mu$ EM。

4、 $\mu$ EM 依据  $\mu$ PC 值将选定单元的信号送到 CBUS，执行第一条微指令的功能， $\mu$ PC 加 1。

5、输出这条指令的第二条微指令，再返回到第三步，直到执行一条“取下一条指令”的微指令，本指令完成执行。

(实验报告上传：超星教学平台作业, 请注意上传时间截止日期, 一般为一个星期。请及时上传，晚交会影响报告成绩。)