

当读内存数据时:6116 输出值通过 245 上数据总线,同时上 IBUS,为避免影

4、实验箱集成开发环境介绍

实验六中我们学习了汇编语言程序的编写和运行，体会到人工编译汇编源程序到目标程序的过程就是:通过查表把汇编指令一对一的翻译成机器指令的简单过程。显然，只要把指令对照表存放在 PC 中，计算机做这种简单工作比人更快、更准确。鉴于 PC 机的强大功能，围绕着程序编译工作，PC 还提供了一系列辅助功能，如:源程序的输入和编辑、目标程序的下载和运行控制、运行过程跟踪等等，这些辅助功能使汇编语言程序的编制、运行和调制工作变得非常简单和人性化。所有这些功能被打包成一个软件，称为汇编语言程序集成开发环境。由于机器语言是针对 CPU 的，所以这个开发环境也是针对 CPU 的。

本实验箱的汇编语言程序集成开发环境是 CP226 软件，它是一个 windows 应用程序，所以其工作界面上包含有下拉菜单、操作快捷键等“视窗”基本元素，使用方法也与其他 windows 程序相似。

5、实验箱集成开发环境电脑上介绍

在 windows 环境运行 CP226 软件(如双击图标)，进入它的主界面

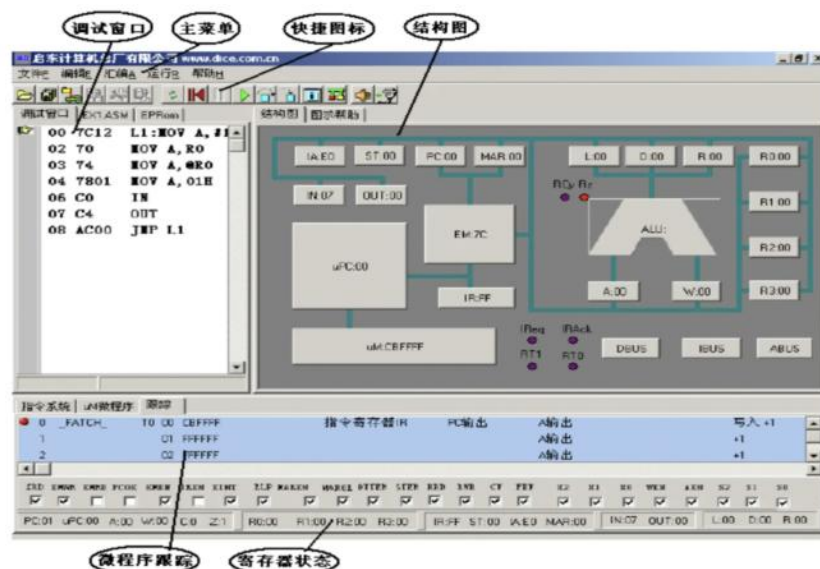


图 4. CP226 软件介绍

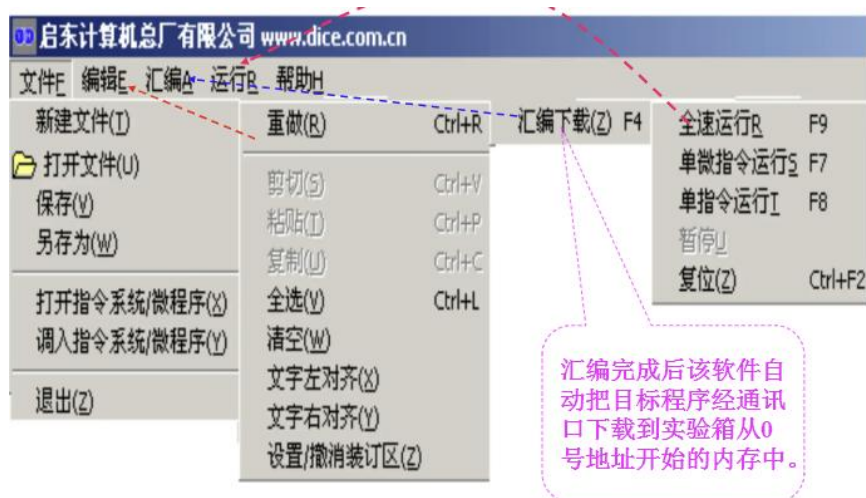


图 5. 下拉菜单介绍

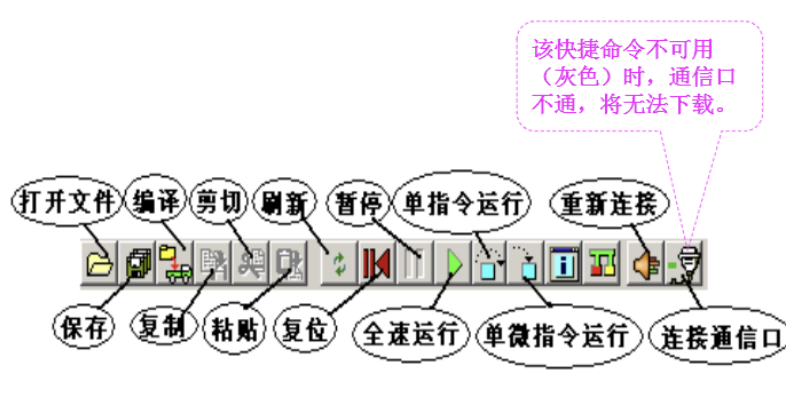


图 6. 快捷命令介绍

6、结构图—显示模型机的内部结构. 在程序单微指令(跟踪)运行时, 可以在结构图上看到数据的走向及寄存器的输入输出状态



图 7. 数据走向

7、伪指令: 不在目标程序中生成机器指令的汇编助记符(汇编命令)其作用是通知汇编软件如何进行“汇编操作”。不同的汇编系统伪指令不一样。本实验系统有伪指令: END 和 ORG。

END 的功能: 通知汇编软件程序段“到此结束”。所以, 当汇编软件遇到这条伪指令时, 就不再理睬后边的任何“汇编指令”。所以任何交给汇编软件的源程序最后一句必须是“END”, 如果忘记了它, 汇编软件就会无止境的汇编下去……, 永远不会最后生成完整的目标程序。

再强调一次: END 不是机器指令, 没有对应的机器指令码!

ORG XX 功能: 通知汇编软件“下面的程序从 xx 地址处开始存放”。这是程序员分段放置程序的途径。

三、实验内容

1. 实验任务一 (手动方式把立即数 33H 写入内存 D1H 单元, 手动方式把 D1H 单元的内容读出, 再送入 E1H 单元。)(由于任务一与任务二可以一起进行实现并在验收环节一起验收, 故放在一起来说)

(1) 实验步骤

先对两个实验分任务进行分析：

1、手动方式把立即数 33H 写入内存 D1H 单元。

在实验箱上，用来表示内存单元的区域是“MAR”所在的显示屏，而该内存单元上存储的内容在“EM”上有所体现，所以需要先将“D1(H)”打入“MAR”内，在 MAR 内已经为 D1 之后，再将“33(H)”打入“EM”内，至此，既可以实现将立即数 33H 写入内存 D1H 单元的操作。而怎么样将“D1(H)”打入“MAR”内呢？只需要将 MAREN 连接的导线置低电平 0，在左下角 K16-K23 的拨电平装置置为八位二进制的 D1H 后，在 HAND 状态下按下小键盘的 STEP，即可将 D1H 打入“MAR”内，即调到内存 D1H 单元。将 33H 打入 EM 同理，将 EMWR 连接的导线置低电平 0，在左下角 K16-K23 的拨电平装置置为八位二进制 33H 后，在 HAND 状态下按下小键盘的 STEP，即可将 33H 打入“EM”内，至此就可以实现将立即数 33H 写入内存 D1H 单元中。

2、手动方式把 D1H 单元的内容读出，再送入 E1H 单元。

将 D1H 单元内容读出需要暂时放到一个寄存器中进行临时存储，再后续从该寄存器读出，写入到 E1H 单元中，一般选择将其存入到堆栈寄存器中，即实验箱中“ST”区域。

送入 E1H 单元的操作与 1 中的类似，故再次不再过多赘述，但是因为读入方式是从堆栈寄存器 ST 中写入，所以需要将 X2 X1 X0 连接的导线置为 010。

下面是具体的连线及操作过程：

将 X2、X1、X0 与 K2、K1、K0 相连，用于控制输入方式，将 EMRD、EMWR、EMEN 与 K10、K9、K8 相连，用于控制某一内存单元的写入与读出，将 MAROE、MAREN 与 K14、K13 相连，将 STEN 与 K15 相连。

初始状态，先将所有的连线置为高电平 1，按下 3 次“TV/ME”按键，进入 HAND 状态。将 K2、K1、K0 置为低电平 0，换为手动方式打入输入，将 K13 置为低电平 0，进入 MAR 的写入状态，K23-K16 分别置为 11010001，即为 D1H 的八位二进制表示，按下 STEP 键，观察到 MAR 显示屏上显示 D1。将 K13 置为高电平 1，将 K9、K8 置为低电平 0，进入 EM 的写入模式，K23-K16 分别置为 00110011，代表 33H 的八位二进制表示，按下 STEP 键，可以观察到 EM 上显示 33。

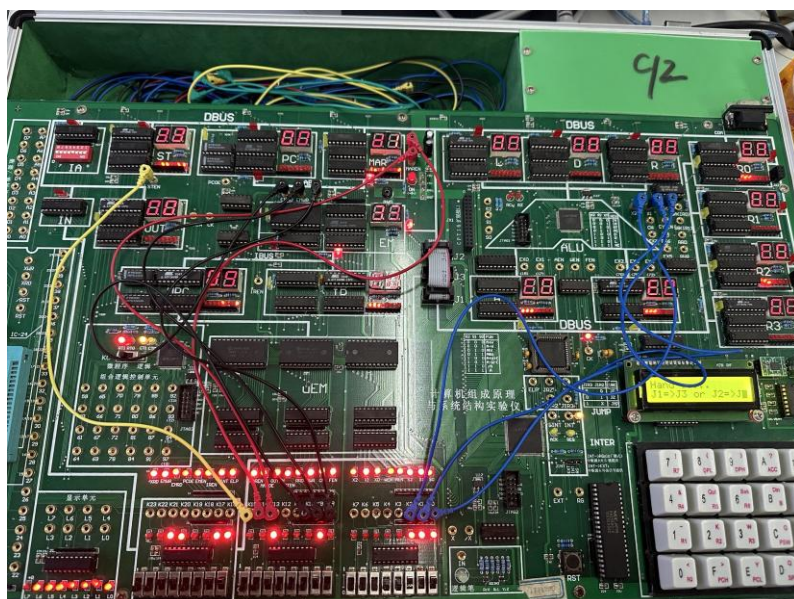


图 8. 实验分任务一操作完后实验箱状态

至此，实验分任务一完成，下面进入实验分任务二：

将 K10 K9 置为低电平 0、高电平 1，进入 EM 的读出模式，将 K15 置为低电平 0，进入 ST 的读入模式，按下 STEP 键，观察到 ST 显示屏上显示 33。将 K10 K9 K15 置为高电平 1。将 K13 置为低电平 0，进入 MAR 的写入状态，K23-K16 分别置为 11100001，即为 E1H 的八位二进制表示，按下 STEP 键，观察到 MAR 显示屏上显示 E1。将 K13 置为高电平 1，将 K9、K8 置为低电平 0，进入 EM 的写入模式，X2、X1、X0 置为 010，改为从 ST 寄存器写入，按下 STEP 键，可以观察到 EM 上显示 33。

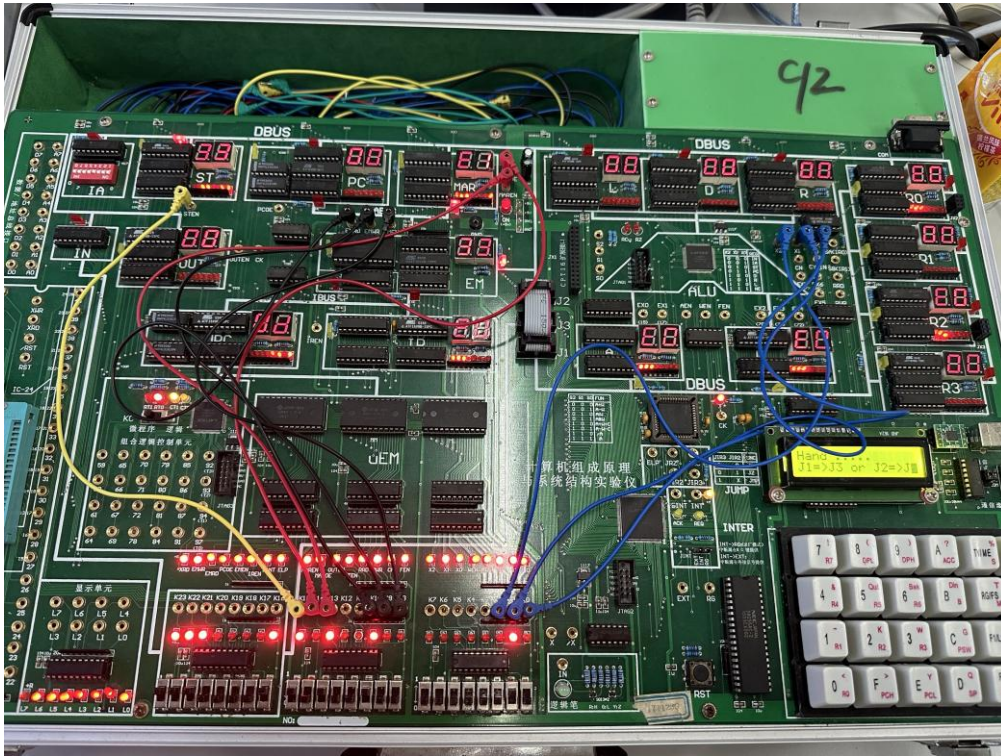


图 9. 实验分任务二操作完后实验箱状态

(2) 实验现象

在完成实验分任务一后，MAR 上显示 D1，EM 上显示 33；在完成实验分任务二后，MAR 上显示 E1，EM 上显示 33，ST 上显示 33。

(3) 数据记录、分析与处理

本实验无数据分析与处理，数据记录同实验现象。

(4) 实验结论

可以认为，本次实验正确地完成了用手动方式把立即数 33H 写入内存 D1H 单元；用手动方式把 D1H 单元的内容读出，再送入 E1H 单元两个操作。

2. 实验任务二（在 CP226 汇编语言程序集成开发环境下编写程序，调试和单微步(跟踪)运行，完成下面任务，观察数据走向及寄存器的输入输出状态。将初始值 05H 赋予累加器 A，每次减 1，到 0 时，OUT 寄存器输出 EFH。）

先对任务进行分析，可以把任务分解为：将 A 寄存器中的数值+5（初始值为 0），A 自减一，判断 A 是否为 0，如果不为 0，再次减一，再判断的循环；如果为 0，则跳出循环，给 A 赋值为 FF，将 A 送入 OUT 中，结束。根据以上分析，汇

编程语言代码如下：

```
MOV A, #05H
LOOP: SUB A, #01H
JZ LOOP1
JMP LOOP
LOOP1: MOV A, #FFH
OUT
END
```

图 10. 实验任务二汇编语言代码

用 CP226 文件打开含有上述代码的 .asm 文件，确定正确打开后，点击快捷操作中的连接实验箱，确保与实验箱相连。点击编译写入，发现实验箱闪烁。按下分步操作的快捷操作，观察实验箱上 A 寄存器与 EM 寄存器中的数值变化，可以发现 A 寄存器中的数值变化为：05、04、03、02、01、00、FF，EM 寄存器中的数值变化为：05、04、03、02、01、00，最后 OUT 寄存器中的数据为 FF，说明正确的编写了程序，实现了任务要求。

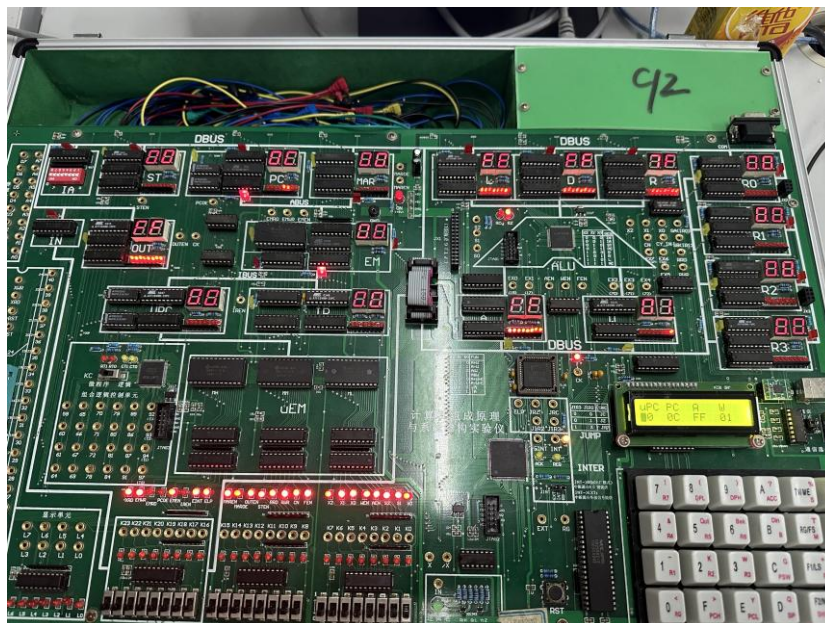


图 10. 实验任务二最终状态截图

四、建议和体会

- 1、实验开始前，要先检查一下导线是否功能正常。
- 2、通过自己的尝试与摸索，自己学会了如何进行 CP226 软件上编写汇编语言与实验箱操作的结合。
- 3、依然还是老样子，提前预习的重要性，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。

（如果不复习、提前构想的话，仅靠课堂上的那两个小时可能不足以完成所有任务，或者说进展的比较慢，没有什么头绪）。

五、思考题

1、既然有 ORG 伪指令，为什么集成开发环境下载到实验箱的目标程序的第一条执行语句最好存放在 0 号地址？

答：为了确保在进行一些复位操作或者启动操作后，处理器能从正确的位置开始执行程序，比较符合大多数硬件的要求，有助于与旧有标准的兼容。