四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机原理与接口技术 | 实验课时 | 4 |
| 实验项目 | 8255并行接口实验 | 实验时间 | 2024.5.20 |
| 实验目的 | 1. 学习并掌握 8255 的工作方式及其应用。 2. 掌握 8255 典型应用电路的接法。 | | |
| 实验环境 | Wdm86 | | |
| 实验内容（算法、程序、步骤和方法） | 【基础实验要求】  1.基本输入输出。编写程序，使8255的A口为输入，B口为输出，完成拨动开关到数据灯的数据传输。要求只要开关拨动，数据灯的显示就发生相应改变。  2.流水灯显示实验。编写程序，使8255的A口和B口均为输出，数据灯D7-D0由左向右，每次仅亮要一个灯，循环显示，D15-D8与D7-D0正相反，由右向左，每次仅点亮一个灯，循环显示。  【扩展实验要求】  1.将16个发光二极管（D0-D15）从低位到高位依次全部点亮，一开始所有灯都是熄灭状态，然后亮1盏、2盏、3盏、4盏灯……，直到16个灯全部点亮，然后所有灯一起闪烁三次；最后回到初始状态一直循环刚才的过程。  2.使用单脉冲开关KK1+作为输入，每次按下发生一次输入，则更新输入次数的计数值，并将二进制计数值显示在D0-D3，若满15次，则重新开始计数。  【基础实验步骤】  **基本输入输出**  本实验使 8255 端口 A 工作在方式 0 并作为输入口，端口 B 工作在方式 0 并作为输出口。用一组开关信号接入端口 A，端口 B 输出线接至一组数据灯上，然后通过对 8255 芯片编程来  实现输入输出功能。具体实验步骤如下述：  （1）实验接线图如下图所示，按图连接实验线路图。  C:\Users\刘桂君\AppData\Local\Temp\ksohtml17060\wps19.jpg   1. 编写实验程序，经编译、连接无误后装入系统，代码如下：   SSTACK SEGMENT STACK             DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  *ASSUME* CS:CODE      START:*MOV*    DX, 0646H  *MOV*    AL, 90H  *OUT*    DX, AL      AA1:  *MOV*    DX, 0640H  *IN*     AL, DX  *CALL*   DELAY  *MOV*    DX, 0642H  *OUT*    DX, AL  *JMP*    AA1      DELAY:*PUSH*   CX  *MOV*    CX, 0F00H      AA2:  *PUSH*   AX  *POP*    AX  *LOOP*   AA2  *POP*    CX  *RET*  CODE ENDS  END START     1. 运行程序，改变拨动开关，同时观察 LED 显示，验证程序功能。   **流水灯显示**  使 8255 的 A 口和 B 口均为输出，数据灯 D7～D0 由左向右，每次仅亮一个灯，循环显示，D15～D8 与 D7～D0 正相反，由右向左，每次仅点亮一个灯，循环显示。实验接线图如图 4.35 所示。实验步骤如下所述：  （1）按下图连接实验线路图。  C:\Users\刘桂君\AppData\Local\Temp\ksohtml17060\wps20.jpg  （2）编写实验程序，经编译、链接无误后装入系统，代码如下：  SSTACK SEGMENT STACK             DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  *ASSUME* CS:CODE      START:*MOV*    DX, 0646H  *MOV*    AL, 80H  *OUT*    DX, AL  *MOV*    BX, 8001H      AA1:  *MOV*    DX, 0640H  *MOV*    AL, BH  *OUT*    DX, AL  *ROR*    BH, 1  *MOV*    DX, 0642H  *MOV*    AL, BL  *OUT*    DX, AL  *ROL*    BL, 1  *CALL*   DELAY  *CALL*   DELAY  *JMP*    AA1      DELAY:*PUSH*   CX  *MOV*    CX, 0F000H      AA2:  *PUSH*   AX  *POP*    AX  *LOOP*   AA2  *POP*    CX  *RET*  CODE ENDS  END START     1. 运行程序，观察 LED 灯的显示，验证程序功能。   【扩展实验】  **点亮流水灯并闪烁**  连线如下图：    代码如下：  START: *MOV* DX, 0646H  *MOV* AL, 80H  *OUT* DX, AL  L3:  *MOV* BX, 0000H  *MOV* CX, 0010H  AA1: *MOV* DX, 0640H  *MOV* AL, BL  *OUT* DX, AL  *MOV* DX, 0642H  *MOV* AL, BH  *OUT* DX, AL  *ROL* BX, 1  *INC* BX  *CALL* DELAY  *CALL* DELAY  *LOOP* AA1  *MOV* CX, 0008H  AA3: *MOV* DX, 0642H  *MOV* AL, BL  *OUT* DX, AL  *MOV* DX, 0640H  *MOV* AL, BH  *OUT* DX, AL  *NOT* BX  *CALL* DELAY  *CALL* DELAY  *LOOP* AA3  *JMP* L3  DELAY: *PUSH* CX  *MOV* CX, 0F000H  AA2:   *PUSH* AX  *POP* AX  *LOOP* AA2  *POP* CX  *RET*  CODE   ENDS         END START  代码思路：   1. 初始化：设置DX寄存器为控制端口地址0646H，并将AL寄存器设置为80H，然后输出到端口。 2. 进入主循环（标签L3）：    * 将BX寄存器清零，CX寄存器设置为16，准备进行16次循环（标签AA1）。 3. 在AA1循环中：    * 设置端口0640H和0642H，将BX寄存器的低8位和高8位分别输出。    * ROL BX, 1将BX寄存器的位向左循环移动，实现位的滚动效果。    * INC BX将BX寄存器的值加1，用于更新要输出的位模式。    * 调用DELAY子程序来实现延时，使变化有一定的时间间隔。 4. 循环16次后，CX寄存器减到0，退出AA1循环。 5. 然后CX寄存器重新设置为8，进入另一个循环（标签AA3）：    * 类似于AA1，但是这次是将BX寄存器取反后输出，实现另一种显示模式。 6. 每次AA3循环结束后，调用DELAY子程序，然后继续循环，直到CX寄存器减到0。 7. 退出AA3循环后，跳回L3，形成无限循环。 8. DELAY子程序通过一个空操作循环来实现延时，防止程序运行得太快。   **开关作为输入并计数**  连线如图：  代码如下：  CODE SEGMENT  *ASSUME* CS:CODE      START:*MOV*    DX, 0646H  *MOV*    AL, 90H  *OUT*    DX, AL      L1:   *MOV*    BL, 00H      AA1:  *MOV*    DX, 0640H  *IN*     AL, DX  *CMP*    AL, 00H  *JNZ*    AA3  *JMP*    AA1      AA3:  *INC*    BL  *CMP*    BL, 10H  *JZ*     L1      AA4:  *MOV*    DX, 0642H  *MOV*    AL, BL  *OUT*    DX, AL  *CALL*   DELAY  *CALL*   DELAY  *JMP*    AA1      DELAY:*PUSH*   CX  *MOV*    CX, 0F000H      AA2:  *PUSH*   AX  *POP*    AX  *LOOP*   AA2  *POP*    CX  *RET*  CODE ENDS          END START  代码思路：   1. **初始化**：首先，代码通过ASSUME指令设置代码段寄存器CS的默认值。 2. **开始标签**（START）：定义了程序的起始点。 3. **设置端口**：MOV DX, 0646H将端口地址0646H加载到DX寄存器，这个端口地址可能用于控制某些硬件设备。 4. **发送控制命令**：MOV AL, 90H将90H（即十进制的144）加载到AL寄存器，然后OUT DX, AL将AL寄存器的值输出到端口DX，这可能是发送一个初始化命令。 5. **主循环**（L1）：定义了一个循环的开始。 6. **初始化计数器**：MOV BL, 00H将BL寄存器设置为0，用作计数器。 7. **检测端口**（AA1）：MOV DX, 0640H将端口地址0640H加载到DX寄存器，然后IN AL, DX从端口DX读取值到AL寄存器。CMP AL, 00H比较AL寄存器的值是否为0，如果是，则跳转到AA3。 8. **非零检测**：如果AL不为0，JNZ AA3跳转到AA3，表示有信号或数据。 9. **计数递增**：在AA3，INC BL递增计数器BL的值。 10. **计数比较**：CMP BL, 10H比较计数器的值是否达到10H（即十进制的16），如果是，则跳回L1重新开始计数。 11. **发送计数结果**：如果计数未达到16，则在AA4将BL的值通过端口0642H发送出去。 12. **延时**：调用DELAY子程序进行延时。 13. **延时子程序**（DELAY）：通过循环来实现延时，PUSH CX和POP CX保存和恢复CX寄存器的值，MOV CX, 0F000H设置循环计数，LOOP AA2进行循环直到CX为0，然后返回。 14. **程序结束**：CODE ENDS和END START分别表示代码段的结束和程序的入口点。 | | |
| 数据记录  和计算 | 【基础实验】  **基本输入输出：**    **流水灯显示**    **【扩展实验】**  **流水灯改进版**    **开关作为输入** | | |
| 结 论  （结 果） | 成功实现了基础实验以及扩展实验的要求，观察到了相应的LED灯的闪烁结果。 | | |
| 小 结 | 实验器材的KK1开关是坏的，我们一直得不到期望的结果，在尝试用KK2来代替KK1后，我们完成了实验。  通过本次实验，我掌握了8255的工作方式，同时也知道了在我们编写相关代码时，如果想要更加明显的看见灯的明灭变化我们可以增加CALL DELAY从而延迟CPU的反应时间，让我们更好观察。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |