四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机系统与接口技术 | 实验课时 | 4课时 |
| 实验项目 | 静态存储器扩展实验 | 实验时间 | 2024年5月6日 |
| 实验目的 | 1. 了解试验系统硬件结构 2. 学习使用Wmd86实验环境 3. 了解存储器扩展的方法和存储器的读/写。 4. 掌握 CPU 对 16 位存储器的访问方法。 | | |
| 实验环境 | 1. PC 机一台 2. TD-PITE 实验装置一套 | | |
| 实验内容（算法、程序、步骤和方法） | **一、实验内容**  了解微机系统与技术实验系统的硬件结构以及实验环境，编写实验程序，将 0000H～000FH 共 16 个数值写入了 SRAM 从 0000H 开始的一段空间中。随后，通过系统命令查看该存储空间，以检测写入的数据是否正确。  【基本实验要求】  1.编写实验程序，将0000H ～ 000FH共16个数写入SRAM的从0000H起始的一段空间中，然后通过系统命令查看该存储空间，检测写入的数据是否正确。  2.改变实验程序，按非规则字写存储器，观察实验结果。  3.改变实验程序，按字节方式写存储器，观察实验结果。  【扩展实验要求】  完成：将学号写入静态存储器中，然后通过D命令查看写入的数据。（注意高位和地位的顺序。）  **二、实验平台**  **2.1 试验系统的硬件结构**  实验系统中的微处理器（80X86）通过系统总线连接主存储器、I/O接口和扩展接口芯片，包括典型的接口芯片如 8255、8254、8259 等。I386EX 微处理器内部为 32 位总线，外部为 16 位数据总线，具备 64MB 的寻址能力。系统还具有 PC104 总线接口，以及 128KB 的 Flash ROM 和 128KB 的 SRAM。上位机运行联机软件（集成开发调试软件）来支持系统的开发与调试。      **2.2 系统信号线说明**  该微机系统（80X86）的信号线和部分寄存器如下：      **2.3 系统编址**  存储器编址如下表：    输入输出接口编址如下表：    **三、实验环境**  **3.1 Wmd86实验环境**  Wmd86联机软件是为TD系列微机原理及接口技术教学实验系统设计的集成开发调试软件。它具备以下特点：  （1）支持汇编语言和C语言两种编译环境。  （2）提供高度可视化的源语言级调试跟踪界面。  （3）实时监控寄存器，允许即时修改寄存器中的值。  （4）支持选择性监控全局变量，提供实时监视和即时修改功能。  （5）提供选择性实时监视堆栈寄存器的功能。  集成虚拟仪器，包括专用图形显示和示波器窗口。  **3.2联机工作**  1.在计算机的设备管理器中查看COM号  2.在“端口”处选择相应的端口号  **四、实验原理**  存储器是用于存储信息的重要组件，也是计算机的关键组成部分。静态 RAM（SRAM）由 MOS 管组成的触发器电路构成，每个触发器可以存储 1 位信息。只要不断电，所存储的信息就不会丢失。因此，SRAM 工作稳定，不需要外加刷新电路，使用方便。  然而，SRAM 的每一个触发器通常由 6 个晶体管组成，这使得 SRAM 芯片的集成度相对较低。当前常用的 SRAM 芯片包括 6116（2K×8 位）、6264（8K×8 位）和 62256（32K×8 位）。本实验平台上使用的是 62256 芯片，采用两片组成 32K×16 位的配置，共 64K 字节。62256 的外部引脚图如图所示。    本系统采用准 32 位 CPU，具有 16 位外部数据总线，即 D0、D1、...、D15，地址总线为 BHE#（# 表示该信号低电平有效）、BLE#、A1、A2、...、A20。存储器被划分为奇体和偶体，分别由字节允许线 BHE# 和 BLE# 进行选通。  在存储器中，从偶地址开始存放的字称为规则字，而从奇地址开始存放的字称为非规则字。处理器访问规则字只需要一个时钟周期，此时 BHE# 和 BLE# 同时有效，从而同时选通存储器的奇体和偶体。处理器访问非规则字则需要两个时钟周期：第一个时钟周期 BHE# 有效，访问奇字节；第二个时钟周期 BLE# 有效，访问偶字节。处理器访问字节只需要一个时钟周期，具体取决于字节所处的存储单元为奇体还是偶体，此时 BHE# 或 BLE# 有效，从而分别选通奇体或偶体。  写入规则字和非规则字的简单时序图如图所示。    实验单元电路图 | | |
| （接上）  实验内容（算法、程序、步骤和方法） | **五、实验步骤**   1. 实验接线图如下图所示，按图接线。       这里接线我们需要注意：  **对应接线颜色排布一致保证接口线相互对应**    **地址线 XA1 到 XA15 被连接到 A0 到 A14**    我们可以注意到在接线过程中：地址线 XA1 到 XA15 被连接到 A0 到 A14，这是因为这个设计需要实现字（word）地址寻址。  具体来说：A0 是用于字节地址的最低位，它决定了一个字中的奇数位还是偶数位。在字地址的情况下，A0 不被使用，而字的实际地址是由 A1 到 A14 指定的。因为字的地址不涉及 A0，直接从 A1 开始即可。所以，XA0 在这种情况下是不必要的。因此，通过连接 XA1 到 XA15 到 A0 到 A14，系统在实际操作中实现了字级寻址，而忽略了最低有效位（A0），从而支持更大的地址空间和适合字（word）操作的布局。   1. 编写实验程序，经编译、链接无误后装入系统。   SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  START PROC FAR  ASSUME CS:CODE  MOV AX, 8000H  MOV DS, AX  AA0: MOV SI, 0000H  MOV CX, 0010H  MOV AX, 0000H  AA1: MOV [SI], AX  INC AL  INC SI  INC SI  LOOP AA1  MOV AX,4C00H  INT 21H  START ENDP  CODE ENDS  END START   1. 先运行程序，待程序运行停止。     4. 通过 D 命令查看写入存储器中的数据：  D8000：0000 回车，即可看到存储器中的数据，应为 0000、0001、0002、…、000F共 16 个字。    3.改变实验程序，按非规则字写存储器，观察实验结果。  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  START PROC FAR  ASSUME CS:CODE  MOV AX, @DATA ; Load data segment address  MOV DS, AX  MOV SI, 0000H ; Initialize source index  MOV CX, 0010H ; Set loop counter to 16 iterations  MOV AX, 0000H ; Initialize AX register  WRITE\_LOOP:  MOV [SI], AX ; Write data to memory at address pointed by SI  ADD SI, 5 ; Increment SI by an irregular amount (5 in this example)  INC AX ; Increment AX  LOOP WRITE\_LOOP ; Continue loop until CX becomes zero  MOV AX, 4C00H ; Set up DOS function to terminate program  INT 21H ; Call DOS interrupt  START ENDP  CODE ENDS  END START  6. 改变实验程序，按字节方式写存储器，观察实验现象。  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  START PROC FAR  ASSUME CS:CODE  MOV AX, @DATA ; Load data segment address  MOV DS, AX  MOV SI, 0000H ; Initialize source index  MOV CX, 0020H ; Set loop counter to 32 iterations (since we're writing byte by byte)  MOV AL, 00H ; Initialize AL register  WRITE\_LOOP:  MOV [SI], AL ; Write data to memory at address pointed by SI  INC SI ; Increment SI by 1 to move to the next byte  INC AL ; Increment AL to write the next value  LOOP WRITE\_LOOP ; Continue loop until CX becomes zero  MOV AH, 4CH ; Set up DOS function to terminate program  INT 21H ; Call DOS interrupt  START ENDP  CODE ENDS  END START  7. 将学号写入静态存储器中，然后通过D命令查看写入的数据。  **方式1：直接写入**  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  DATA SEGMENT  STUDENT\_ID DB 13 DUP(?) ; 为学号分配13个字节的空间  DATA ENDS  CODE SEGMENT  START PROC FAR  ASSUME CS:CODE, DS:DATA, ES:SSTACK  MOV AX, DATA ; 加载数据段地址  MOV DS, AX  MOV ES, AX ; 设置堆栈段地址  MOV SI, OFFSET STUDENT\_ID ; 初始化源索引  MOV DI, 0000H ; 设置目标地址偏移量为 0  MOV CX, 000D ; 设置循环计数器为 13，因为学号有 13 个字符  MOV AL, '2' ; 学号的第一个字符  MOV [SI], AL  INC SI  MOV AL, '0' ; 学号的第二个字符  MOV [SI], AL  INC SI  MOV AL, '2' ; 学号的第三个字符  MOV [SI], AL  INC SI  MOV AL, '1' ; 学号的第四个字符  MOV [SI], AL  INC SI  MOV AL, '1' ; 学号的第五个字符  MOV [SI], AL  INC SI  MOV AL, '4' ; 学号的第六个字符  MOV [SI], AL  INC SI  MOV AL, '1' ; 学号的第七个字符  MOV [SI], AL  INC SI  MOV AL, '4' ; 学号的第八个字符  MOV [SI], AL  INC SI  MOV AL, '6' ; 学号的第九个字符  MOV [SI], AL  INC SI  MOV AL, '0' ; 学号的第十个字符  MOV [SI], AL  INC SI  MOV AL, '1' ; 学号的第十一个字符  MOV [SI], AL  INC SI  MOV AL, '5' ; 学号的第十二个字符  MOV [SI], AL  INC SI  MOV AL, '9' ; 学号的第十三个字符  MOV [SI], AL  MOV AX, 4C00H ; 程序终止  INT 21H  START ENDP  CODE ENDS  END START  **方式二：使用预定义的数据段：**  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  DATA SEGMENT  STUDENT\_ID DB '2021141460159'  DATA ENDS  CODE SEGMENT  START PROC FAR  ASSUME CS:CODE, DS:DATA, ES:SSTACK  MOV AX, DATA ; 加载数据段地址  MOV DS, AX  MOV ES, AX ; 设置堆栈段地址  ; 学号已经在数据段中初始化，无需额外赋值指令  MOV AX, 4C00H ; 程序终止  INT 21H  START ENDP  CODE ENDS  END START  得到对应实验结果： | | |
| 数据记录  和计算 | 实验结果与理论结果相符。根据实验手册，通过 D 命令查看写入存储器中的数据。输入 D8000:0000 并按下回车键，可以看到存储器中的数据应为 0000, 0001, 0002, ... , 000F，共 16 个字。对比实验结果与理论预期，可以发现此次实验的结果与预期完全一致，实验取得了圆满成功。在扩展实验中，我也成功打印出了自己的学号。 | | |
| 结 论  （结 果） | 通过使用D命令查看存储空间后发现，基础实验结果和理论结果相符，扩展实验的输出为自己的学号。 | | |
| 小 结 | 通过这次实验，我了解了存储器扩展的方法和静态存储器的读写操作，掌握了CPU对16位存储器的访问方法，并理解了地址总线的连接方式，从而成功实现了静态存储器的扩展。实验的难点主要在于熟悉所使用的机器，尤其是机器与主机的连接。最好在实验机开机的情况下进行连接，先安装驱动程序，再按动复位按钮，以实现最佳配合。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |

四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机系统与接口技术 | 实验课时 | 4课时 |
| 实验项目 | A/D和D/A转换实验 | 实验时间 | 2024年5月13日 |
| 实验目的 | 1. 学习数/模转换的基本原理。  2. 掌握模/数转换芯片 ADC0809 和数/模信号转换芯片DAC0832 的的使用方法 | | |
| 实验环境 | 1. PC 机一台 2. TD-PITE 实验装置一套 | | |
| 实验内容（算法、程序、步骤和方法） | 1. **实验内容：**   【基础实验要求】 1.A/D转换实验：编写实验程序，将ADC单元中提供的0V-5V信号源作为ADC0809的模拟输入量，进行A\D转换，转换结果通过变量进行显示。  2. D/A转换实验：设计实验电路图实验线路并编写程序，实现D/A转换，要求产生锯齿波、脉冲波（方波），并用示波器观察电压波形  【扩展实验要求】针对D/A转换实验编写程序，将锯齿波修改为三角波和梯形波，并通过虚拟示波仪观察并记录。   1. **实验原理：**   模拟信号与数字信号在表示信息的方式上有根本的区别。模拟信号代表连续变化的量，如温度、压力、流量、声音和位移等，这些信号在自然界中普遍存在，并能表示一个范围内的所有可能值。相对地，数字信号则由离散的有限值组成，这种格式更适合于计算机处理，因为计算机在处理信息时依赖于二进制形式，即信息的表达仅依赖于一系列的0和1。    在这种背景下，模数转换器（ADC）起着至关重要的桥梁作用，它将现实世界中的模拟信号转换成计算机能够理解和处理的数字信号。以ADC0809为例，这款设备采用8位逐次逼近方法进行模数转换，能有效地将模拟信号转换为数字输出。该芯片还集成了一个8通道的模拟多路开关和联合寻址逻辑，这使得它可以在多个信号源之间快速切换，提高了系统的灵活性和效率。    ADC0809内部包含逐次逼近寄存器（SAR）、比较器、数字模拟（D/A）转换器、缓冲器等关键组件。这些组件协同工作，确保了信号转换的准确性和可靠性。特别是它的转换结束指示脚，这一设计为使用者提供了转换状态的直观反馈：在转换过程中输出低电平，而在非转换状态时保持高电平，一旦转换完成，指示脚将恢复到高电平，向系统表明数据已准备好被进一步处理。  DAC0832是8位芯片，采用CMOS工艺和R-2RT形电阻解码网络，转换结果为一对差动电流IOUT1和IOUT2输出。    转换时先把每一位代码按其权的大小转换成相应的模拟分量，然后将各模拟分量相加。   1. **实验步骤：**   **3.1 A/D转换实验**   1. 实验接线图如下图所示，按图接线。        1. 编写实验程序，经编译、链接无误后装入系统。   代码如下：  STACK SEGMENT STACK  DW 32 DUP(?)  STACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:STACK  START: MOV AX, 00H  MOV DX, 600H  MOV AL, 00H  AA1: OUT DX, AL  CALL DELAY  INC AL  MOV CL, AL  CMP AL, 0FFH  JZ AA3  JMP AA1  DELAY: PUSH CX  MOV CX, 03FFH  AA2: PUSH AX  POP AX  LOOP AA2  POP CX  RET  AA3: OUT DX,AL  CALL DELAY  DEC AL  CMP AL,00H  JZ AA1  JMP AA3  CODE ENDS  END START  监测变量VALUE    4. 设置断点并运行，在变量窗口中查看VALUE的值，对比其与理论值是否存在差异  **3.2 D/A转换实验**  设计实验电路图实验线路并编写程序，实现D/A转换，要求产生锯齿波、脉冲波，并用示波器观察电压波形。   1. 按照实验原理图进行接线        1. 编写代码如下：   锯齿波代码：  STACK SEGMENT STACK  DW 32 DUP(?)  STACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:STACK  START: MOV AX, 00H  MOV DX, 600H  MOV AL, 00H  AA1: OUT DX, AL  CALL DELAY  INC AL  JMP AA1  DELAY: PUSH CX  MOV CX, 03FFH  AA2: PUSH AX  POP AX  LOOP AA2  POP CX  RET  CODE ENDS  END START  方波代码：  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE  START: MOV AX, 00H | | |
| （接上）  实验内容（算法、程序、步骤和方法） | MOV DX, 600H  AA1: MOV AL, 00H  OUT DX, AL  CALL DELAY  MOV AL, 07FH  OUT DX, AL  CALL DELAY  JMP AA1  DELAY: PUSH CX  MOV CX, 0FF00H  AA2: PUSH AX  POP AX  LOOP AA2  POP CX  RET  CODE ENDS  END START  3.将代码进行编译，链接，之后装入系统使用示波器测量DA的输出，并观察实验现象  **3.3 扩展实验**  在3.2 D/A转换实验的接线基础上三角波代码：  STACK SEGMENT STACK  DW 32 DUP(?)  STACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:STACK  START:  MOV AX, 00H ; Initialize AX to 0  MOV DX, 0600H ; Set DX to port address 600H  MOV AL, 00H ; Initialize AL to 0  AA1:  OUT DX, AL ; Output AL to port DX  CALL DELAY ; Call delay subroutine  INC AL ; Increment AL  MOV CL, AL ; Move AL to CL  CMP AL, 0FFH ; Compare AL with FFH  JZ AA3 ; If AL == FFH, jump to AA3  JMP AA1 ; Otherwise, jump back to AA1  DELAY:  PUSH CX ; Save CX  MOV CX, 03FFH ; Set CX for delay loop  AA2:  PUSH AX ; Save AX (dummy operation for delay)  POP AX ; Restore AX (dummy operation for delay)  LOOP AA2 ; Loop until CX is 0  POP CX ; Restore CX  RET ; Return from subroutine  AA3:  OUT DX, AL ; Output AL to port DX  CALL DELAY ; Call delay subroutine  DEC AL ; Decrement AL  CMP AL, 00H ; Compare AL with 0  JZ AA1 ; If AL == 0, jump to AA1  JMP AA3 ; Otherwise, jump back to AA3  CODE ENDS  END START  **梯形波代码：**  STACK SEGMENT STACK  DW 32 DUP(?)  STACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:STACK  START: MOV AX, 00H  MOV DX, 600H  MOV AL, 00H  AA1: OUT DX, AL ; 输出当前值  CALL DELAY ; 延迟  INC AL ; 增加AL的值  MOV CL, AL  CMP AL, 0C8H ; 检查AL是否到达顶部前的最大值  JZ AA2 ; 如果到达，跳转到顶部保持  JMP AA1  DELAY: PUSH CX  MOV CX, 03FFH ; 设定通用延迟  DELAY\_LOOP:  PUSH AX  POP AX  LOOP DELAY\_LOOP  POP CX  RET  AA2: OUT DX, AL ; 输出顶部的值  CALL HOLD\_DELAY ; 顶部保持更长时间  CMP AL, 0FFH ; 检查是否达到顶部极限  JZ AA3 ; 如果到达顶部极限，开始下降  INC AL ; 否则继续增加AL的值，增长到最大  JMP AA2  HOLD\_DELAY: PUSH CX  MOV CX, 07FFH ; 设定顶部保持的延迟  HOLD\_LOOP:  PUSH AX  POP AX  LOOP HOLD\_LOOP  POP CX  RET  AA3: OUT DX, AL ; 输出当前值  CALL DELAY ; 延迟  DEC AL ; 减少AL的值  CMP AL, 38H ; 检查AL是否达到底部后的最小值  JZ AA4 ; 如果到达，跳转到底部保持  JMP AA3  AA4: OUT DX, AL ; 输出底部的值  CALL HOLD\_DELAY ; 底部保持更长时间  CMP AL, 00H ; 检查是否达到底部极限  JZ AA1 ; 如果到达底部极限，开始上升  DEC AL ; 否则继续减少AL的值，减少到最小  JMP AA4  CODE ENDS  END START | | |
| 数据记录  和计算 | A/D转换实验：  2.5V:    5V:    D/A转换实验：  **锯齿波：**    **方波：**    **三角波：**    **梯形波：** | | |
| 结 论  （结 果） | 在A/D转换实验中，通过ADC单元的精确处理，所提供信号源转换的模拟输入量显示出了正常的结果。在D/A转换实验方面，我成功地在示波器上观测到了锯齿波和脉冲波。此外，通过进行扩展实验，我还将锯齿波转换成了三角波和梯形波。 | | |
| 小 结 | 通过本次实验，我深入学习了数字/模拟和模拟/数字转换的基本原理，并掌握了DAC0832芯片的操作方法。实验中，正确的线路连接确保了实验的顺利进行。我还学习并理解了模/数信号转换和数/模信号转换的基本理念，熟练掌握了ADC0809和DAC0832这两种转换芯片的使用技巧。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |

四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机系统与接口技术 | 实验课时 | 4课时 |
| 实验项目 | 8255并行接口实验 | 实验时间 | 2024年5月20日 |
| 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； 3. 掌握程序固化及脱机运行程序的方法。 | | |
| 实验环境 | 1. PC 机一台 2. TD-PITE 实验装置一套 | | |
| 实验内容（算法、程序、步骤和方法） | 1. **实验内容**   【基础实验：】  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。  【拓展实验：】  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。   1. **实验原理**   并行接口通过字节单位与I/O设备或被控制对象之间传递信息。CPU和接口之间的数据传输总是并行的，即可以同时传递8位、16位或32位数据。8255可编程外围接口芯片是Intel公司生产的通用并行I/O接口芯片，具有A、B、C三个并行接口，使用+5V单电源供电，能够在以下三种方式下工作：方式0——基本输入/输出方式，方式1——选通输入/输出方式，方式2——双向选通工作方式。8255的内部结构及引脚如图4.31所示，8255工作方式控制字和C口按位置位/复位控制字格式如图4.32所示。       1. **实验步骤**   **3.1基本输入输出实验：**   1. 按照参考手册给出的图所示的基本输入输出实验接线图连线。        1. 编写实验程序，经编译、链接无误后装入系统。   SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:SSTACK  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，观察 LED 灯的显示，验证程序功能。实验结果如图所示     拨动对应开关对应灯亮起  **3.2流水灯显示实验：**  将8255的A口和B口均设置为输出，数据灯D7～D0从左向右每次仅亮一个灯，循环显示；D15～D8与D7～D0相反，从右向左每次仅点亮一个灯，循环显示。实验步骤如下：   1. 按图连接实验线路。      1. 编写实验程序，经编译、链接无误后装入系统。   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，从中间向两边扩散显示。循环显示；D15～D8与D7～D0相反，从右向左每次仅点亮一个灯，循环显示 | | |
| （接上）  实验内容（算法、程序、步骤和方法） | **扩展实验一：**  (1)实验连线同基础实验二：      （2）编写相关代码如下  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE  START:  MOV DX, 0646H ; 初始化控制端口  MOV AL, 80H ; 数据设置，可能需要根据硬件调整  OUT DX, AL  ; 点亮LED  L1: MOV BX, 0001H ; 从第一个LED开始  MOV CX, 0010H ; 总共16个LED  CALL LIGHT\_UP\_LEDS  ; 闪烁所有LED三次  MOV CX, 3 ; 闪烁次数  L2: CALL FLASH\_ALL\_LEDS  LOOP L2  ; 重置并循环  JMP L1  ; 子程序：点亮LED  LIGHT\_UP\_LEDS:  MOV DX, 0640H  MOV AL, BL  OUT DX, AL  MOV DX, 0642H  MOV AL, BH  OUT DX, AL  SHL BX, 1 ; 移位，点亮下一个LED  CALL DELAY  LOOP LIGHT\_UP\_LEDS  RET  ; 子程序：闪烁所有LED  FLASH\_ALL\_LEDS:  MOV BX, 0FFFFH ; 所有LED设置为ON  MOV DX, 0640H  MOV AL, BL  OUT DX, AL  MOV DX, 0642H  MOV AL, BH  OUT DX, AL  CALL DELAY  MOV BX, 0000H ; 所有LED设置为OFF  MOV DX, 0640H  MOV AL, BL  OUT DX, AL  MOV DX, 0642H  MOV AL, BH  OUT DX, AL  CALL DELAY  RET  ; 延迟子程序  DELAY:  PUSH CX  MOV CX, 0F000H ; 调整延迟时间  DELAY\_LOOP:  NOP ; 空操作，用于延时  LOOP DELAY\_LOOP  POP CX  RET  CODE ENDS  END START  （3）编译链接载入系统  （4）查看数据灯的点亮顺序是否为依次点亮，并闪烁三次循环往复    **扩展实验二：**   1. 实验连线       这里注意到将KK1对应显示灯连接在PA0和PA1处，将PB0-PB7的灯与LED显示模块的D0-D7相链接。   1. 编写代码如下：   SSTACK SEGMENT STACK  DW 32 DUP(?) ; 定义堆栈空间  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE  START:  MOV DX, 0646H ; 初始化端口  MOV AL, 80H ; 设定初始值  OUT DX, AL ; 输出到端口  L1: MOV BL, 00H ; 计数器初始化  AA1: MOV DX, 0640H ; 设置输入端口  IN AL, DX ; 从端口读取值  CMP AL, 00H ; 检查是否有输入  JNZ AA3 ; 如果有输入，跳转到AA3  JMP AA1 ; 否则继续等待输入  AA3: INC BL ; 增加计数器  CMP BL, 10H ; 检查计数器是否达到15  JZ L1 ; 如果是，重置计数器  AA4: MOV DX, 0642H ; 设置显示端口  MOV AL, BL ; 将计数值放入AL  OUT DX, AL ; 输出到显示端口  CALL DELAY ; 调用延迟  JMP AA1 ; 返回等待更多输入  DELAY:  PUSH CX ; 保存CX寄存器  MOV CX, 0F000H ; 设置延迟循环计数  AA2: NOP ; 执行空操作以延时  LOOP AA2 ; 循环直到计数结束  POP CX ; 恢复CX寄存器  RET ; 返回调用位置  CODE ENDS  END START   1. 编译链接载入系统 2. 按下单脉冲开关KK1+，并观察D0-D3的数据灯显示状况，判断是否为计数 | | |
| 数据记录  和计算 | 基础实验1：    基础实验2：    扩展实验1：    扩展实验2： | | |
| 结 论  （结 果） | 基础实验一：实验目的是确保数据灯与相应的开关状态保持一致。当开关被激活时，相应的LED应亮起；反之，则熄灭。该实验旨在验证开关控制逻辑的正确实现。  基础实验二：该实验展示了流水灯效果，其中LED灯按照指定的顺序依次点亮和熄灭。通过此实验，可以观察到LED灯按预定的时间间隔和顺序逐一操作，从而验证时序控制的准确性。  扩展实验一：在此实验中，LED灯将依次点亮，直到所有灯均已亮起。随后，所有灯将一同闪烁三次，然后系统重置并重复之前的过程。该实验主要测试灯光控制和定时功能的实现。  扩展实验二：本实验中，数据灯的显示与单脉冲开关KK1+的输入次数直接相关。每次按压开关，相应的LED将根据输入次数以二进制形式更新其显示状态。实验的目的是验证计数器和显示逻辑的准确性。 | | |
| 小 结 | 在这次实验中，我有机会再次接触并学习新的接口技术，成功实现了LED流水灯的效果。看到LED灯按照设定的模式闪烁，我感受到了极大的成就感和满足感。通过本次课程的实践操作，我不仅深入理解了8255接口芯片的工作原理和广泛应用，还掌握了8255芯片在典型应用电路中的正确接法。这一经验极大地丰富了我的硬件操作技能和理论知识。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |

四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机系统与接口技术 | 实验课时 | 4课时 |
| 实验项目 | 8254定时/计数器应用实验 | 实验时间 | 2024年5月27日 |
| 实验目的 | 1. 掌握 8254 的工作方式及应用编程。 2. 掌握 8254 典型应用电路的接法。 | | |
| 实验环境 | 1. PC 机一台 2. TD-PITE 实验装置一套 | | |
| 实验内容（算法、程序、步骤和方法） | 1. **实验内容：**   【基础实验：】  1.定时应用实验。编写程序，应用8254的定时功能，产生一个1s的方波。  2.电子发声设计实验。根据实验提供的音乐频率表和时间表，编写程序控制8254，使其输出连接到扬声器上发出相应的乐曲。  【扩展实验：】  1. 应用8254的定时功能，只用一个通道实现输出1Hz的方波。  2.在“基础实验1”的基础上加入8255，实现当按动单次脉冲KK1+后改变示波器输出的方波的波形频率（大致为按动一次减少一倍，即第一次按动后为半秒一个方波）。   1. **实验原理：**   **2.1 8254芯片解析**  8254是由Intel公司生产的一款高级可编程间隔定时器，具备以下特点：  （1）多功能计数器：设备包括三个独立的16位计数器，每个计数器均能以二进制或十进制（BCD）模式进行计数。  （2）灵活的工作模式：每个计数器都可以被编程以六种不同的工作模式运行，提供广泛的应用灵活性。  （3）高频率计数能力：与其前身8253最高2MHz的计数频率相比，8254允许的最高计数频率达到10MHz，大幅提升性能。  （4）增强的命令功能：8254引入了读回命令，不仅可以读取当前计数器的内容，还能读取状态寄存器的信息，这是8253所不具备的。  （5）多样的计数脉冲输入：计数脉冲可以来源于规律的时钟信号，也可以是随机信号，这使得8254能适应各种不同的计时需求。  每个计数通道有3个接口信号：  （1）CLK时钟输入信号：该信号通过CLK引脚接收外部脉冲，并按照设定的二进制或十进制（BCD）格式对这些脉冲进行倒计数。这种计数可以处理不等间隔的脉冲，当脉冲间隔相等时，还可以用作计时器。  （2）GATE门控输入信号：GATE信号控制计数的开始和停止。当GATE为高电平时，允许计数；当为低电平时，暂停计数。这允许外部控制事件精确地控制计数操作的执行。  （3）OUT计数器输出信号：此信号表示计数的结果。OUT信号通常用于指示计数周期的结束，可以配置为在计数完成时产生一个脉冲或持续的输出状态变化。  **2.2 电子发生器实验原理：**  每个音符对应一个特定的频率，当将该频率的方波信号传递到扬声器时，就可以产生相应的音符声音。通过依次将一段乐曲中各音符对应频率的方波信号送入扬声器，即可演奏出整段乐曲。使用8254定时器的方式3——方波发生器，将相应频率的计数初值写入计数器，即可产生对应频率的方波。    例如，输入时钟频率为1MHz，如果要得到800Hz的输出频率，则计数初值应为1000000 ÷ 800。音符与频率的对应关系如表4.8所示。每个音符的演奏时间可以通过软件延时来控制。首先，根据CPU的频率确定单位延时时间。然后，确定每个音符演奏所需的单位时间数，将该值送入DL寄存器中，调用DALLY子程序即可。下面提供了乐曲《友谊地久天长》的实验参考程序。程序中，频率表记录了曲谱中各音符对应的频率值（B调、四分之二拍），时间表则记录了各个音符的相对发音时间（由曲谱中的节拍得出）。    频率表和时间表是逐项对应的。频率表的最后一项为0，作为重复的标志。根据频率表中的频率计算出对应的计数初值，然后依次写入8254计数器。将时间表中的相对时间值传入延时程序，以确定音符的演奏时间。  三、**实验步骤：**  **3.1定时应用实验：**   1. 实验接线如下图所示。       (2)编写实验程序能够产生1s的方波的代码，经编译、链接无误后装入系统。  A8254 EQU 0600H  B8254 EQU 0602H  C8254 EQU 0604H  CON8254 EQU 0606H  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:SSTACK  START: MOV DX, CON8254  MOV AL, 36H  OUT DX, AL  MOV DX, A8254  MOV AL, 0E8H  OUT DX, AL  MOV AL, 03H  OUT DX, AL  MOV DX, CON8254  MOV AL, 76H  OUT DX, AL  MOV DX, B8254  MOV AL, 0E8H  OUT DX, AL  MOV AL, 03H  OUT DX, AL  AA1: NOP  JMP AA1  CODE ENDS  END START   1. 运行程序，用示波器观察波形的方法   **3.2基础实验2(电子发生器实验)：**  （1）按照原理图连线：      （2）编写代码，并编译链接、然后装入系统  IOY3 EQU 06C0H  MY8254\_COUNT0 EQU IOY3+00H  MY8254\_COUNT1 EQU IOY3+02H  MY8254\_COUNT2 EQU IOY3+04H  MY8254\_MODE EQU IOY3+06H  STACK1 SEGMENT STACK  DW 256 DUP(?)  STACK1 ENDS  DATA SEGMENT  FREQ\_LIST DW 371,495,495,495,624,556,495,556,624  DW 495,495,624,742,833,833,833,742,624  DW 624,495,556,495,556,624,495,416,416,371  DW 495,833,742,624,624,495,556,495,556,833  DW 742,624,624,742,833,990,742,624,624,495  DW 556,495,556,624,495,416,416,371,495,0  TIME\_LIST DB 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 12, 1, 3, 6, 2  DB 4, 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 12, 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 12, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 6, 2, 4, 4, 12  DATA ENDS  CODE SEGMENT  ASSUME CS:CODE,DS:DATA  START: MOV AX,DATA  MOV DS,AX  MOV DX,MY8254\_MODE  MOV AL,36H  OUT DX,AL  BEGIN: MOV SI,OFFSET FREQ\_LIST  MOV DI,OFFSET TIME\_LIST  PLAY: MOV DX,0FH  MOV AX,4240H  DIV WORD PTR [SI]  MOV DX,MY8254\_COUNT0  OUT DX,AL  MOV AL,AH  OUT DX,AL  MOV DL,[DI]  CALL DALLY  ADD SI,2  INC DI  CMP WORD PTR [SI],0  JE BEGIN  JMP PLAY  DALLY PROC  D0: MOV CX,0010H  D1: MOV AX,0FF0H  D2: DEC AX  JNZ D2  LOOP D1  DEC DL  JNZ D0  RET  DALLY ENDP  CODE ENDS  END START  (3) 运行程序，听蜂鸣器是否放出声音 | | |
| （接上）  实验内容（算法、程序、步骤和方法） | **3.3 扩展实验1（编写使用一个通道能够产生1s的方波）：**  （1）接线如下    （2）代码如下  A8254 EQU 0600H  B8254 EQU 0602H  C8254 EQU 0604H  CON8254 EQU 0606H  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS: SSTACK  START: MOV DX, CON8254 ; 将8254的控制字寄存器地址加载到DX  MOV AL, 36H ; 设置控制字为36H，选择工作模式3  OUT DX, AL ; 将控制字写入8254的控制字寄存器  MOV DX, A8254 ; 将A通道的数据寄存器地址加载到DX  MOV AL, 0E8H ; 设置计数器初值低8位  OUT DX, AL ; 将初值低8位写入A通道的数据寄存器  MOV AL, 03H ; 设置计数器初值高8位  OUT DX, AL ; 将初值高8位写入A通道的数据寄存器  AA1: NOP ; 等待一段时间，控制方波的占空比  NOP ; 可以通过调整NOP的数量来调整占空比  NOP  JMP AA1 ; 跳转到AA1，形成一个简单的循环  MOV DX, CON8254 ; 关闭定时器，释放资源  MOV AL, 0H ; 将控制字设置为0，停止定时器  OUT DX, AL  CODE ENDS  END START  (3)打开示波器并运行查看波形  **3.4扩展实验2：**  (1)实验接线    (2)编写代码  A8254 EQU 0600H  B8254 EQU 0602H  C8254 EQU 0604H  CON8254 EQU 0606H  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS: SSTACK  START:  MOV BX, 4800H ; 初始化计数器初值为 4800H  L1: MOV DX, CON8254 ; 将 8254 控制字寄存器地址加载到 DX  MOV AL, 36H ; 设置控制字为 36H，选择工作模式3  OUT DX, AL ; 将控制字写入 8254 的控制字寄存器  MOV DX, A8254  MOV AL, BL ; 设置计数器初值低8位  OUT DX, AL  MOV AL, BH ; 设置计数器初值高8位  OUT DX, AL  L3: MOV DX, 0646H ; 检测脉冲开关状态的端口地址  MOV AL, 90H ; 发送读取命令  OUT DX, AL ; 发送命令到端口  L4: MOV AL, 0640H ; 读取脉冲开关状态  IN AL, DX ; 从端口读取状态  CMP AL, 00H ; 检查是否按下脉冲开关  JZ L3 ; 如果没按下，继续等待  SHR BX, 1  CALL DELAY ; 调用延迟函数，确保只处理一次脉冲  JMP L1 ; 回到主循环  DELAY: PUSH CX  MOV CX, 0F00H ; 设置延迟循环次数  AA2: PUSH AX  POP AX  AA1: PUSH AX  POP AX  LOOP AA2 ; 循环等待延迟结束  POP CX  RET  CODE ENDS  END START  (3)编译，链接，装入系统  （4）运行并观察示波器的输出，按下单次脉冲KK1+后继续观察示波器输出的方波的波形频率 | | |
| 数据记录  和计算 | 基础实验显示：    扩展实验1波形显示：    扩展实验2波形显示：（按动KK1后逐渐缩短时钟周期） | | |
| 结 论  （结 果） | 通过学习了解8254的工作方式和应用编程，以及典型的应用电路接法，我成功地应用了8254的定时功能，并确保了接线的正确性。在基础实验一中，我在示波器上观测到了1秒的方波输出，验证了定时器的正常工作。在基础实验二中，我成功发出了相应的乐曲，进一步验证了8254的功能。在扩展实验一中，我利用一个通道成功地产生了1赫兹的方波输出，展示了定时功能的灵活应用。在扩展实验二中，基于基础实验一的成功经验，我实现了按下单次脉冲开关后，示波器输出的方波波形频率的动态改变，进一步拓展了8254的应用场景。 | | |
| 小 结 | 这次实验给我留下了深刻的印象，因为它涉及了8254计数器的应用，这个计数器在人机交互中扮演着重要的角色。我们通过亲自编写代码、进行编译，并最终在硬件上完成实验。在实验过程中，我们通过拨动KK1+五次，成功地在电脑屏幕上输出了指定的结果。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |

四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 微机系统与接口技术 | 实验课时 | 4课时 |
| 实验项目 | 8259中断控制实验 | 实验时间 | 2024年6月3日 |
| 实验目的 | 1. 掌握 8259 中断控制器的工作原理。  2. 学习 8259 的应用编程方法。  3. 掌握 8259 级联方式的使用方法。 | | |
| 实验环境 | 1. PC 机一台 2. TD-PITE 实验装置一套 | | |
| 实验内容（算法、程序、步骤和方法） | **一、实验内容：**  【基础实验：】  1.单中断实验要求：单次脉冲输出与主片8259的IR7相连，每按动一次单次脉冲，产生一次外部中断，在显示屏上输出一个字符“7”。  2.级联实验要求：KK1+连接主片8259的IR7，KK2+连接从片8259的IR1，当按一次KK1+时，显示屏上显示字符“M7”，按一次KK2+时，显示字符“S1”，编写程序。  3.计数产生中断实验要求：编写程序，应用8254的计数功能，使用单次脉冲模拟计数，使每当按动“KK1+”5次后，通过8259产生一次计数中断，并在屏幕上显示一个字符“M”  【扩展实验：】  1. 在显示屏上以1s的固定间隔持续输出特定的字符。  2. 利用8255和8259a实现开关控制流水灯。（自行设计实验接线图，在8255基础实验2的流水灯循环显示的基础上，实现按一次KK1＋时，流水灯暂停；按一次KK1＋时，16位流水灯从暂停位置开始继续循环亮起。编写程序。）  3. 在扬声器发声实验基础上，按动KK1+控制扬声器发声的开和关。  **二、实验原理：**  内部中断主要用于处理程序异常和指令中断，例如除数为零、溢出等情况。Intel 386EX芯片内部的两片8259A中断控制芯片不仅加强了微处理器的异常管理能力，还提高了其对可屏蔽中断的处理效率。8259A是一种高效的可编程中断控制器，为微处理器提供了强大的中断管理功能。  在Intel 386EX芯片中，集成有中断控制单元（ICU），该单元由两个级联的8259中断控制器组成，分别作为主控制器和从控制器。这些控制器在功能上与工业标准的82C59A中断控制器相同，并采用相似的操作方法。级联配置中，从控制器的INT线连接到主控制器的IR2信号，实现两个8259芯片的协同工作。    在该系统中，单次脉冲开关与主控制器8259的IR7相连，当被触发时，会产生一次外部中断，导致显示屏上输出字符“7”。这种外部中断可以分为两类：非屏蔽中断和可屏蔽中断。非屏蔽中断用于处理系统意外或故障，如掉电、存储器读写错误等；而可屏蔽中断主要用于处理与外设的数据交换，如8259A中断控制器负责优先级排队等问题。   1. **实验内容：**   **3．1 基础实验1：**  实验接线图如下图所示，单次脉冲输出与主片 8259 的 IR7 相连，每按动一次单次脉冲，产生一次外部中断，在显示屏上输出一个字符“7”。      （1）按上图连接实验线路。  （2）编写实验程序，经编译、链接无误后装入系统。  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE  START: PUSH DS  MOV AX, 0000H  MOV DS, AX  MOV AX, OFFSET MIR7  MOV SI, 003CH  MOV [SI], AX  MOV AX, CS  MOV SI, 003EH  MOV [SI], AX  CLI  POP DS  MOV AL, 11H  OUT 20H, AL  MOV AL, 08H  OUT 21H, AL  MOV AL, 04H  OUT 21H, AL  MOV AL, 01H  OUT 21H, AL  MOV AL, 6FH  OUT 21H, AL  STI  AA1: NOP  JMP AA1  MIR7: STI  CALL DELAY  MOV AX, 0137H  INT 10H  MOV AX, 0120H  INT 10H  MOV AL, 20H  OUT 20H, AL  IRET  DELAY: PUSH CX  MOV CX, 0F00H  AA0: PUSH AX  POP AX  LOOP AA0  POP CX  RET  CODE ENDS  END START  （3）运行程序，重复按单次脉冲开关 KK1＋，显示屏会显示字符“7”，说明响应了中断。  **3.2基础实验2：**   1. 接线       (2)编写程序  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE  START: PUSH DS  MOV AX, 0000H  MOV DS, AX  MOV AX, OFFSET MIR7  MOV SI, 003CH  MOV [SI], AX  MOV AX, CS  MOV SI, 003EH  MOV [SI], AX  MOV AX, OFFSET SIR1  MOV SI, 00C4H  MOV [SI], AX  MOV AX, CS  MOV SI, 00C6H  MOV [SI], AX  CLI  POP DS  MOV AL, 11H  OUT 20H, AL  MOV AL, 08H  OUT 21H, AL  MOV AL, 04H  OUT 21H, AL  MOV AL, 01H  OUT 21H, AL  MOV AL, 11H  OUT 0A0H, AL  MOV AL, 30H  OUT 0A1H, AL  MOV AL, 02H  OUT 0A1H, AL  MOV AL, 01H  OUT 0A1H, AL  MOV AL, 0FDH  OUT 0A1H,AL  MOV AL, 6BH  OUT 21H, AL  STI  AA1: NOP  JMP AA1  MIR7: CALL DELAY  MOV AX, 014DH  INT 10H  MOV AX, 0137H  INT 10H  MOV AX, 0120H  INT 10H  MOV AL, 20H  OUT 20H, AL  IRET  SIR1: CALL DELAY  MOV AX, 0153H  INT 10H  MOV AX, 0131H  INT 10H  MOV AX, 0120H  INT 10H  MOV AL, 20H  OUT 0A0H, AL  OUT 20H, AL  IRET  DELAY: PUSH CX  MOV CX, 0F00H  AA0: PUSH AX  POP AX  LOOP AA0  POP CX  RET  CODE ENDS  END START  (3)编译、链接、装入系统  (4)运行程序，按动单次脉冲开关KK1+或KK2+，观察显示器的显示结果  **3.3 基础实验3：**  （1）根据实验原理图链接      (2)编写代码  A8254 EQU 06C0H  B8254 EQU 06C2H  C8254 EQU 06C4H  CON8254 EQU 06C6H  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:SSTACK  START: PUSH DS  MOV AX, 0000H  MOV DS, AX  MOV AX, OFFSET IRQ7  MOV SI, 003CH  MOV [SI], AX  MOV AX, CS  MOV SI, 003EH  MOV [SI], AX  CLI  POP DS  MOV AL, 11H  OUT 20H, AL | | |
| （接上）  实验内容（算法、程序、步骤和方法） | MOV AL, 08H  OUT 21H, AL  MOV AL, 04H  OUT 21H, AL  MOV AL, 01H  OUT 21H, AL  MOV AL, 6FH  OUT 21H, AL  MOV DX, CON8254  MOV AL, 10H  OUT DX, AL  MOV DX, A8254  MOV AL, 04H  OUT DX, AL  STI  AA1: NOP  JMP AA1  IRQ7: MOV DX, A8254  MOV AL, 04H  OUT DX, AL  MOV AX, 014DH  INT 10H  MOV AX, 0120H  INT 10H  MOV AL, 20H  OUT 20H, AL  IRET  CODE ENDS  END START  (3)编译、链接、装入系统  (4)运行程序，多次按动KK1+，观察显示器显示结果  **3.4 扩展实验1：**  (1)实验连线：    （2）编写代码  A8254 EQU 0600H  B8254 EQU 0602H  C8254 EQU 0604H  CON8254 EQU 0606H  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:SSTACK  START:  ; 初始化8254定时器  MOV DX, CON8254  MOV AL, 36H  OUT DX, AL  MOV DX, A8254  MOV AL, 0E8H  OUT DX, AL  MOV AL, 03H  OUT DX, AL  MOV DX, CON8254  MOV AL, 76H  OUT DX, AL  MOV DX, B8254  MOV AL, 0E8H  OUT DX, AL  MOV AL, 03H  OUT DX, AL  ; 设置中断向量  PUSH DS  MOV AX, 0000H  MOV DS, AX  MOV AX, OFFSET MIR7  MOV SI, 003CH  MOV [SI], AX  MOV AX, CS  MOV SI, 003EH  MOV [SI], AX  CLI  POP DS  ; 初始化8259中断控制器  MOV AL, 11H  OUT 20H, AL  MOV AL, 08H  OUT 21H, AL  MOV AL, 04H  OUT 21H, AL  MOV AL, 01H  OUT 21H, AL  MOV AL, 6FH  OUT 21H, AL  STI  ; 主循环  AA1: NOP  JMP AA1  ; 中断服务程序  MIR7:  STI  CALL DELAY  MOV AX, 014DH ; AH=01H，字符模式写字符，DL='M'  INT 10H  MOV AX, 0120H ; AH=01H，移动光标  INT 10H  MOV AL, 20H  OUT 20H, AL  IRET  ; 延时子程序  DELAY:  PUSH CX  MOV CX, 0F00H  AA0: PUSH AX  POP AX  LOOP AA0  POP CX  RET  CODE ENDS  END START  （3）编译，链接，装入系统  （4）计时，并查看显示器的输出  **3.5 扩展实验2**  （1）实验连线：    （2）编写代码  PPI\_BASE EQU 0600H ; 8255的基地址  CONTROL\_REG EQU PPI\_BASE + 3 ; 控制寄存器地址  ; 定义8255端口地址  PORTA EQU PPI\_BASE + 0  PORTB EQU PPI\_BASE + 1  PORTC EQU PPI\_BASE + 2  ; 定义中断控制器8259A的端口地址  PIC\_BASE EQU 020H  PIC\_CMD EQU PIC\_BASE + 0  PIC\_DATA EQU PIC\_BASE + 1  SSTACK SEGMENT STACK  DW 32 DUP(?)  SSTACK ENDS  CODE SEGMENT  ASSUME CS:CODE, SS:SSTACK, DS:CODE  START:  ; 初始化8255  MOV DX, CONTROL\_REG  MOV AL, 80H  OUT DX, AL ; 设置8255的控制寄存器  ; 初始化8259A中断控制器  MOV AL, 11H  OUT PIC\_CMD, AL ; 初始化命令字1  MOV AL, 08H  OUT PIC\_DATA, AL ; 设置中断向量地址  MOV AL, 04H  OUT PIC\_DATA, AL ; 设置IR位  MOV AL, 01H  OUT PIC\_DATA, AL ; 设置OCW3  ; 设置中断向量表  PUSH DS  MOV AX, 0  MOV DS, AX  MOV AX, OFFSET ISR  MOV SI, 0FCH  MOV [SI], AX  MOV AX, CS  MOV SI, 0FEH  MOV [SI], AX  POP DS  ; 使能中断  STI  ; 主程序循环  MAIN\_LOOP:  MOV AL, 01H ; 设置流水灯初始值  MOV DX, PORTA  OUT DX, AL  CALL DELAY  JMP MAIN\_LOOP  ; 中断服务程序  ISR:  CLI  PUSH AX  PUSH DX  PUSH DI  ; 读取开关状态  MOV DX, PORTC  IN AL, DX  TEST AL, 01H  JZ CONTINUE\_LIGHTS  XOR DI, DI ; 如果按下开关，DI置0，暂停  JMP END\_ISR  CONTINUE\_LIGHTS:  MOV DI, 01H ; 如果再次按下开关，DI置1，继续  END\_ISR:  MOV AL, 20H  OUT PIC\_CMD, AL ; 发送EOI信号  POP DI  POP DX  POP AX  STI  IRET  ; 延时程序  DELAY:  PUSH CX  MOV CX, 0FFFFH  DELAY\_LOOP:  LOOP DELAY\_LOOP  POP CX  RET  CODE ENDS  END START  3.编译，链接，装入系统  4. 按下KK1+，观察流水灯的运行状况，并重复实验观察  **扩展实验3：**   * 1. 实验接线     （2）编写程序如下  ; 定义常量  IOY3 EQU 06C0H  MY8254\_COUNT0 EQU IOY3+00H  MY8254\_COUNT1 EQU IOY3+02H  MY8254\_COUNT2 EQU IOY3+04H  MY8254\_MODE EQU IOY3+06H  PPI\_BASE EQU 0600H  CONTROL\_REG EQU PPI\_BASE + 3  PORTA EQU PPI\_BASE + 0  PORTB EQU PPI\_BASE + 1  PORTC EQU PPI\_BASE + 2  PIC\_BASE EQU 020H  PIC\_CMD EQU PIC\_BASE + 0  PIC\_DATA EQU PIC\_BASE + 1  ; 定义段  STACK\_SEG SEGMENT STACK  DW 256 DUP(?)  STACK\_SEG ENDS  DATA\_SEG SEGMENT  FREQ\_LIST DW 371,495,495,495,624,556,495,556,624  DW 495,495,624,742,833,833,833,742,624  DW 624,495,556,495,556,624,495,416,416,371  DW 495,833,742,624,624,495,556,495,556,833  DW 742,624,624,742,833,990,742,624,624,495  DW 556,495,556,624,495,416,416,371,495,0  TIME\_LIST DB 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 12, 1, 3, 6, 2  DB 4, 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 12, 4, 6, 2, 4, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 12, 4, 6, 2, 4, 4  DB 6, 2, 4, 4, 6, 2, 4, 4, 12  DATA\_SEG ENDS  CODE\_SEG SEGMENT  ASSUME CS:CODE\_SEG, DS:DATA\_SEG, SS:STACK\_SEG  START:  ; 初始化数据段  MOV AX, DATA\_SEG  MOV DS, AX  ; 初始化8255  MOV DX, CONTROL\_REG  MOV AL, 80H  OUT DX, AL  ; 初始化8259A中断控制器  MOV AL, 11H  OUT PIC\_CMD, AL  MOV AL, 08H  OUT PIC\_DATA, AL  MOV AL, 04H  OUT PIC\_DATA, AL  MOV AL, 01H  OUT PIC\_DATA, AL  ; 设置中断向量表  PUSH DS  MOV AX, 0  MOV DS, AX  MOV AX, OFFSET ISR  MOV SI, 0FCH  MOV [SI], AX  MOV AX, CS  MOV SI, 0FEH  MOV [SI], AX  POP DS  ; 使能中断  STI  ; 初始化8254计数器  MOV DX, MY8254\_MODE  MOV AL, 36H  OUT DX, AL  BEGIN:  MOV SI, OFFSET FREQ\_LIST  MOV DI, OFFSET TIME\_LIST  PLAY:  MOV DX, 0FH  MOV AX, 4240H  DIV WORD PTR [SI]  MOV DX, MY8254\_COUNT0  OUT DX, AL  MOV AL, AH  OUT DX, AL  MOV DL, [DI]  CALL DALLY  ADD SI, 2  INC DI  CMP WORD PTR [SI], 0  JE BEGIN  JMP PLAY  DALLY PROC  D0: MOV CX, 0010H  D1: MOV AX, 0FF0H  D2: DEC AX  JNZ D2  LOOP D1  DEC DL  JNZ D0  RET  DALLY ENDP  ISR PROC FAR  CLI  PUSH AX  PUSH DX  PUSH DI  ; 读取开关状态  MOV DX, PORTC  IN AL, DX  TEST AL, 01H  JZ DISABLE\_SOUND  JMP ENABLE\_SOUND  DISABLE\_SOUND:  ; 停止发声  MOV DX, MY8254\_COUNT0  MOV AL, 00H  OUT DX, AL  MOV AL, 00H  OUT DX, AL  JMP END\_ISR  ENABLE\_SOUND:  ; 继续发声  CALL PLAY  END\_ISR:  ; 发送EOI信号  MOV AL, 20H  OUT PIC\_CMD, AL  POP DI  POP DX  POP AX  STI  IRET  ISR ENDP  CODE\_SEG ENDS  END START  （3）编译链接载入系统  （4）按下KK1+，观察流水灯的运行状况，并重复实验观察 | | |
| 数据记录  和计算 | 基础实验一：    基础实验二：    基础实验三：    扩展实验一：    扩展实验二：      循环播放流水灯从中心向两边扩散，按下KK1+后流水灯停止，再次按下后继续从中心向两边扩散。  扩展实验三：  播放出友谊地久天长乐曲，按下KK1+后友谊地久天长乐曲暂停，再次按下后继续播放该曲目。 | | |
| 结 论  （结 果） | 通过对实验结果与理论预期的对比，可以清晰地看出，我的实验结果完全符合预期。实验的成功不仅验证了理论的正确性，也表明我已经达到了相关实验的目标：  基础实验一：显示结果完全符合要求，输出为数字“7”。  基础实验二：按照要求，当按下指定的KK1+或KK2+时，屏幕上正确地输出了指定的字符。  基础实验三：输出结果与理论相符，按下五次KK1+后，8259成功产生了一次计数中断，并在屏幕上显示了字符“M”。  对于扩展实验部分：  扩展实验一：实验结果与预期相似，在大约每隔1秒的间隔内，屏幕连续不断地输出了字符“M”。  扩展实验二：结果完全符合要求，按下KK1+后流水灯便会按照预定的模式循环运行或停止。  扩展实验三：结果完全符合要求，按下KK1+后友谊地久天长乐曲暂停，再次按下后继续播放该曲目。  这些实验不仅加深了我对8259中断控制器的理解，还增强了我在实际应用中使用这种技术的能力。 | | |
| 小 结 | 通过这次实验，我掌握了 8259 中断控制器的工作原理，学习了 8259 的应用编程方法，并熟悉了 8259 的级联方式的使用方法。在过程中，遇到的主要挑战是如何将 KK1+ 与 8259 芯片的触发挂钩。这个问题通过详细研究 8259 的中断配置和 KK1+ 的连接方式，以及反复调试和验证信号的触发机制，最终得以解决。这些知识的获得和应用，不仅提高了我的技术能力和理解深度，还为我在未来处理更复杂的技术问题奠定了坚实的基础。 | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |