

# 汇编语言程序设计



专 业：生物医学工程

年 级：2022级

班 级：1班

姓 名：张台忍

学 号：3022202299

邮 箱：ztr8526@gmail.com

**2024年4月13日**

# **一 实验目的**

熟悉集成开发环境的使用方法.

掌握基本指令的用法.

掌握汇编语言程序设计方法.

# **二 实验设备**

微机一台, 集成开发环境一套, 单片机开发仪

# **三 实验内容**

实验流程如图所示

A screenshot of a diagram

Description automatically generated

图1 实验流程图

## 1 建立汇编程序

在中, 参照实验0的方法, 建立工程, 然后新建汇编程序. 与实验0不同的是, 本次实验使用汇编语言进行编程, 因此本次实验应在中选择. 具体操作如图所示.

A screenshot of a computer

Description automatically generated

图2 文件创建

新建完工程之后, 即可开始代码的编写, 以实现特定的功能.

## 2 数码管的显示

本次实验想要实现的功能是在静态数码管上循环显示, 并且每个数字显示时间为. 要实现此功能, 首先要观察原理图, 进行硬件连接.

A diagram of a circuit

Description automatically generated

图3 数码管原理图

观察静态数码管的原理图可知, 静态数码管的管脚通过保护电阻和连接到排针上, 且当为低电平点亮. 因此如果我们在硬件电路中通过杜邦线把排针与单片机的口连接, 即可通过控制口的输出电平, 来实现数码管不同位置的点亮, 通过调节点亮的位置, 即可实现不同数字的显示, 再通过输出电平随时间的变化, 即可实现数值之间的切换.

根据以上分析, 在中画出电路图, 电路图如下图所示.

A screen shot of a computer

Description automatically generated

图4 电路图

观察数码管引脚结构, 可以得到当数码管输入电平与数码管显示数字的对应关系, 如下表所示.

表1 输入电平与显示数字对应关系

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

将电平的输入转化为十六进制数, 可得十六进制数与显示数字的对应关系如下表所示.

表2 显示数字与十六进制数对应关系

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

因为数码管的输入就是单片机口的输出, 这样就得到了单片机的输出电平, 接下来只需要将电平随时间变化输出即可.

在中编写语言和汇编语言代码, 代码如下:

1. #include <REGX51.H>
3. #include <intrins.h>
5. **void** DelayMs(unsigned **int** \_ms)
6. {
7. unsigned **char** i, j;
9. **while** (\_ms--)
10. {
11. \_nop\_();
12. i = 2;
13. j = 199;
14. **do**
15. {
16. **while** (--j);
17. } **while** (--i);
18. }
19. }

22. **void** main()
23. {
24. **char** a[]={0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x98};  //存储不同数字对应的十六进制数
25. **int** i=0;
27. **while**(1)
28. {
29. P1=a[i];
30. DelayMs(300);  //延时300ms
31. i++;  //显示下一个数字
32. **if**(i==9) i=0;  //当数字显示到9的时候, 回到数字1
33. }
34. }

语言代码

1. $NOMOD51
3. NAME    AMBERCA
4. P1  DATA    090H

7. ?PR?\_DelayMs?AMBERCA     SEGMENT CODE
8. ?PR?main?AMBERCA         SEGMENT CODE
9. ?DT?main?AMBERCA         SEGMENT DATA OVERLAYABLE
10. ?CO?AMBERCA              SEGMENT CODE
11. EXTRN   CODE (?C\_STARTUP) ; 外部引用，标准C库的启动代码
12. EXTRN   CODE (?C?COPY)      ; 外部引用，用于变量复制的函数
13. PUBLIC  main              ; main函数公开定义，使其可被其他文件引用
14. PUBLIC  \_DelayMs          ; \_DelayMs函数公开定义
16. RSEG  ?DT?main?AMBERCA         ; 分配一个数据段给main程序的AMBERCA部分
17. ?main?**BYTE**:
18. a?143:   DS   10   ; 定义一个大小为10个字节的数据空间
20. RSEG  ?CO?AMBERCA             ; 分配一个代码段给AMBERCA显示
21. \_?ix1000:
23. DB  0F9H               ; 定义AMBERCA的字形数据1
24. DB  0A4H               ; 定义AMBERCA的字形数据2
25. DB  0B0H               ; 定义AMBERCA的字形数据3
26. DB  099H               ; 定义AMBERCA的字形数据4
27. DB  092H               ; 定义AMBERCA的字形数据5
28. DB  082H               ; 定义AMBERCA的字形数据6
29. DB  0F8H               ; 定义AMBERCA的字形数据7
30. DB  080H               ; 定义AMBERCA的字形数据8
31. DB  098H               ; 定义AMBERCA的字形数据9
33. ; #include <REGX51.H>
34. ; #include <intrins.h>
35. ; **void** DelayMs(unsigned **int** \_ms)
36. RSEG  ?PR?\_DelayMs?AMBERCA
37. \_DelayMs:
38. USING   0
40. ;---- Variable '\_ms?040' assigned to Register 'R6/R7' ----
41. ; {
42. ?C0001:
43. ;     unsigned **char** i, j;
44. ;
45. ;     **while** (\_ms--)
46. MOV     A,R7
47. DEC     R7
48. MOV     R4,AR6
49. JNZ     ?C0013
50. DEC     R6
51. ?C0013:
52. ORL     A,R4
53. JZ      ?C0008
54. ;     {
55. ;         \_nop\_();
56. NOP
57. ;         i = 2;
58. ;---- Variable 'i?041' assigned to Register 'R5' ----
59. MOV     R5,#02H
60. ;         j = 199;
61. ;---- Variable 'j?042' assigned to Register 'R4' ----
62. MOV     R4,#0C7H
63. ;         **do**
64. ;         {
65. ?C0006:
66. ;             **while** (--j);
67. DJNZ    R4,?C0006
68. ;         } **while** (--i);
69. DJNZ    R5,?C0006
70. ;     }
71. SJMP    ?C0001
72. ; }
74. ?C0008:
75. RET
76. ; END OF \_DelayMs
78. ; **void** main()
79. RSEG  ?PR?main?AMBERCA
80. main:
81. USING   0
82. ; {
83. ;   **char** a[]={0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x98};
84. MOV     R0,#LOW (a?143)
85. MOV     R4,#HIGH (a?143)
86. MOV     R5,#00H
87. MOV     R3,#0FFH
88. MOV     R2,#HIGH (\_?ix1000)
89. MOV     R1,#LOW (\_?ix1000)
90. MOV     R6,#00H
91. MOV     R7,#09H
92. LCALL   ?C?COPY
93. ;   **int** i=0;
94. ;---- Variable 'i?144' assigned to Register 'R2/R3' ----
95. CLR     A
96. MOV     R3,A
97. MOV     R2,A
98. ?C0009:
99. ;   **while**(1)
100. ;   {
101. ;       P1=a[i];
102. MOV     A,#LOW (a?143)
103. ADD     A,R3
104. MOV     R0,A
105. MOV     A,@R0
106. MOV     P1,A
107. ;       DelayMs(300);
108. MOV     R7,#02CH
109. MOV     R6,#01H
110. LCALL   \_DelayMs
111. ;       i++;
112. INC     R3
113. CJNE    R3,#00H,?C0014
114. INC     R2
115. ?C0014:
116. ;       **if**(i==9) i=0;
117. MOV     A,R3
118. XRL     A,#09H
119. ORL     A,R2
120. JNZ     ?C0009
121. MOV     R2,A
122. MOV     R3,A
123. ;   }
124. SJMP    ?C0009
125. ; END OF main
126. END

对应的汇编语言代码

写好代码后点击, 生成文件. 先将文件放入中进行仿真, 观察现象, 操作如下所示.

A screenshot of a computer

Description automatically generated

图5 仿真文件

仿真结果如图所示.

A screenshot of a computer

Description automatically generated

图6 仿真现象

仿真结果符合预期, 即数码管显示数字, 并每次间隔. 逻辑正确, 因此将生成的文件烧录进入单片机, 实际观察现象. 观察得到相同的现象, 实验成功.

A close up of a circuit board

Description automatically generated

图7 实际现象

# 四 结果与讨论

本实验较为简单, 核心思路就是调节高低电平来实现不同数字的显示. 汇编语言程序可由生成, 通过代码的编写和阅读, 可加深对汇编语言及其原理的理解.