信息与软件工程学院上机实验报告

(第 2 次)

一、实验名称

13.1.2 实验二 多分支实验

二、实验目的及要求

1. 实验目的
   通过本次实验掌握 Keil uVision3 和 Proteus 开发环境以及它们联机调试单片机汇编语言和C51源程序方式方法, 熟悉单片机汇编语言指令, 掌握程序散转的方法, 实现程序的多分支转移。
2. 实验要求
   在本次实验前, 在自己的电脑安装 Keil uVision3 和 Proteus7.0 以上版本的软件, 使用 Proteus 软件画参考硬件原理图 (如图13-4)。

三、实验环境

- **仿真实验**：装有 Keil uVision3 以及 Proteus7.0 以上版本软件的笔记本或台式PC机。
- **元件清单**：1个STC89C52、1个共阳极的七段数码管7SEG-COM-AN-GRN, 一个+5V稳压电源。

四、实验内容

编写软件实现: 根据STC89C52单片机片内30H单元内容(00或01或02或03)进行散转, 1个数码管循环显示对应的数字。(为了便于观察你也可以增加一个循环, 循环取分支变量, 观察显示结果)

五、实验设计

1、实验步骤

实验流程图

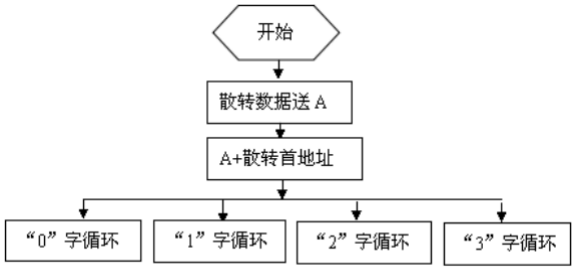


图 13-5 多分支实验框图

设计思想

- 本实验的核心是利用变址寻址指令（如 MOVC A, @A+PC）或跳转指令（JMP @A+DPTR）来实现程序的多分支跳转。
- 首先，在程序存储区（Code Area）定义一个数据表（数码管字型码表）。
- 主程序从片内30H单元读取一个值（00H~03H）到累加器A，这个值将作为查表的偏移量。
- 使用 MOVC 指令，将累加器 A 的内容作为偏移量，与程序计数器 PC（或DPTR，若表基址存入DPTR）相加，

从字型码表中获取对应的共阳极数码管字型码。

- 将获取的字型码输出到 P0 口（或其他连接数码管的端口），点亮数码管显示相应数字。
- 为了实现循环显示，设计一个主循环，不断地从片内30H单元获取数值（或内部循环产生0-3的变量）并执行查表和显示操作。

实现步骤

1. **初始化**：设置堆栈指针 SP。
2. **主循环**：
    - a. (方案一) 从片内RAM 30H单元获取分支数据（0-3）到累加器A。
    - b. (方案二) 或者设置一个循环变量（如R2），使其在0-3之间循环变化，并将R2的值送入A。
    - c. 将累加器 A 的值（偏移量）与后续字型码表的基地址相加，以计算目标字型码的地址。
    - d. 执行查表指令 MOVC A, @A+PC。
    - e. 将查表得到的字型码（如 0C0H, 0F9H, 0A4H, 0B0H 等）输出到 P0 端口。
    - f. 调用一个延时子程序，以便人眼能清晰观察到数码管显示的数字。
    - g. 跳转回主循环的起始处，继续下一个数字的显示。
3. **字型码表**：在主循环之后，定义字型码表（DB 伪指令），按0, 1, 2, 3的顺序存放共阳极字型码。

2、调试过程及实验结果

调试过程：

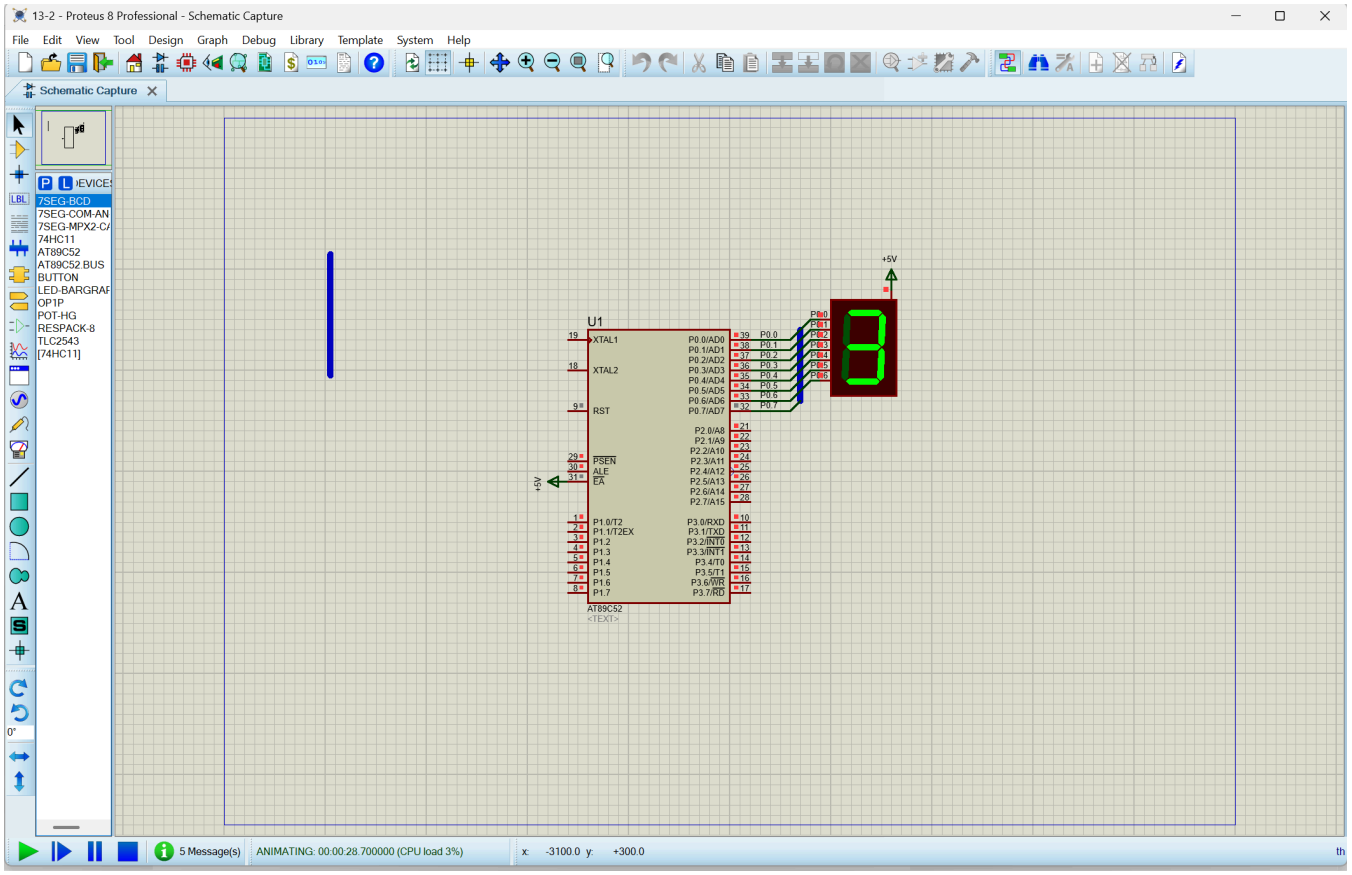
1. **硬件搭建**：在 Proteus 中按照图 13-4 搭建电路。使用一个 STC89C52，将 P0.0 至 P0.7 连接到 `7SEG-COM-AN-GRN` (共阳极七段数码管) 的 a 至 g, dp 引脚
2. **Keil-Proteus 联调设置**：在 Keil 中设置 "Options for Target"，在 "Debug" 标签下选择 "Use: Proteus VSM Simulator"。在 "Output" 标签下确保 "Create HEX File" 已勾选
3. **加载程序**：在 Proteus 的 STC89C52 元件属性中，加载 Keil 生成的 .hex 文件。
4. **程序执行**：
    - 程序（如附录代码）首先初始化 `NUMB` (30H) 单元为 00H。
    - 进入 `START` 循环，将 `NUMB` (00H) 送入 A。
    - 执行 `RL A` (A 变为 00H)，`JMP @A+DPTR`，跳转到 `TABADDR`+0，即 `SJMP DISP0`。
    - `DISP0` 将 0C0H (共阳极'0'的字型码) 存入 `FONT` 单元，并调用 `MJUP1` 显示在 P0 口。
    - 延时后，`INC NUMB` (NUMB 变为 01H)，R2 未减到 0，循环回 `START`。
    - 第二次循环，A=01H，`RL A` =02H，`JMP @A+DPTR` 跳转到 `TABADDR`+2，即 `SJMP DISP1`。
    - `DISP1` 将 0F9H (共阳极'1'的字型码) 显示在 P0 口。
    - 依此类推，直到 R2=0，程序重置 `NUMB` 为 00H，重新开始循环。

实验结果：

启动 Proteus 仿真后，实验现象与预期完全一致

1. **Proteus 仿真**：电路中的七段数码管循环显示数字 "0", "1", "2", "3"，然后又从 "0" 开始，每次显示之间有明显延时，便于观察。

2. **Keil 调试**：若在 Keil 中联调，在 `MOV P0, A` 处设置断点，可以看到 P0 端口的值依次变为 0C0H, 0F9H, 0A4H, 0B0H，这与实验指导书中给出的字型码一致



3、实验改进（选做）

4、总结

13-2 多分支实验.

本次实验让我掌握了程序的多分支转移的方法

我通过实验践理解了两种实现方式.

1. 跳转表：使用 JMP @A+DPTR 指令，搞根据累加器A的值跳转到不同的程序分支。

2. 查表法：使用 MOVC A, @A+PC 或 MOVC A, @A+DPTR 指令，从存储器中查找预设数据

通过调试，我加深了对共阳极数码管字型码的理解，并认识到P0口作为输出口需要上拉电阻

这个实验锻炼了我的硬件~~调试能力~~电路搭建能力和程序逻辑设计能力.

5、附录

```
/*实验二    分支程序，文件名：13-2.A51*/
        ORG 0000H
NUMB    EQU 30H
FONT    EQU 31H
INT:        MOV NUMB,#00H      ;需要显示数字送NUMB单元
        MOV R2,#04H
START:  MOV A,NUMB
        MOV DPTR,#TABADDR
```

```
        RL  A
        JMP @A+DPTR
TABADDR:SJMP    DISP0
        SJMP    DISP1
        SJMP    DISP2
        SJMP    DISP3
DISP0:  MOV FONT,#0C0H
        LJMP    MJUP1
DISP1:  MOV FONT,#0F9H
        LJMP    MJUP1
DISP2:  MOV FONT,#0A4H  ;
        LJMP    MJUP1
DISP3:  MOV FONT,#0B0H
/**************数码管显示子程序*****************/
MJUP1:  MOV     A,FONT
        MOV    P0,A          ;字型送P0口
        ACALL   DELAY
        INC     NUMB
        DJNZ    R2,START
        AJMP    INT
DELAY:  MOV    R7,#0FFH
DL:     MOV    R6,#0FFH
DL6:    DJNZ   R6,DL6
        DJNZ   R7,DL
        RET
        END
```