单片机学习笔记 (第4章)

1. 核心概念：数码管显示方式

1.1 静态显示 vs 动态显示

- **静态显示方式**: 指无论有多少个数码管，所有数码管都同时处于显示状态。
  - **特点**: 一直显示。亮度高，没有闪烁，软件控制比较容易。
  - **缺点**: 占用I/O口线非常多。例如，4位数码管静态显示需要占用4个8位的I/O口，一个单片机最多只能接4个。
- **动态显示方式**: 利用人眼的“视觉暂留”效应，不是一直显示。
  - **原理**: 通过程序控制，逐位地每隔一定时间轮流点亮各位显示器（扫描方式）。 只要扫描速度足够快，人眼就会误认为所有数码管是同时点亮的，达到同时显示的效果。
  - **优点**: 极大节省了I/O口线。所有数码管的段码线可以并联，由一个8位I/O口控制，而位选线则由另外的I/O口控制。

1.2 补充知识：使用译码器选择数码管

为了用更少的单片机引脚控制更多的数码管，我们通常使用**74LS138译码器。**

- **硬件连接**: 将单片机某个I/O口（如P2口）的任意3位连接到74LS138的C、B、A三个输入端。
- **输出控制**: 译码器的8个输出端（Y0-Y7）分别连接到8个数码管（M0-M7）的片选端（开关）。
- **软件实现**: 在程序中，向连接C、B、A的这3个引脚输出`000`到`111`的二进制数，就可以依次选中M0到M7数码管。 例如，在笔记中，我们通过`p2=weixuan[j]`（其中`weixuan`数组的值为0x00-0x07）来选择数码管。

1.3 数码管硬件知识

- **数码管排列**: 开发板上的数码管从左到右依次是 **M0, M1, M2, ..., M7**。
- **共阴极与共阳极**:
  - 开发板上的**小数码管**是**共阴极**接法。
  - 开发板上的**大数码管**是**共阳极**接法。
  - **共阴极原理**: 所有LED的阴极连接在一起，通常接地。当某个LED的阳极为高电平时，该段点亮。
  - **共阳极原理**: 所有LED的阳极连接在一起，通常接正电压。当某个LED的阴极为低电平时，该段点亮。

1.4 数码管段码

为了让数码管显示特定的数字或字符，需要给它对应的编码，这个编码称为“段码”或“字型码”。 习惯上，'a'段对应段码字节的最低位。

- **共阴极段码表 (0-F)**:

```
0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f, 0x6f, 0x77, 0x7c, 0x39,
0x5e, 0x79, 0x71
```

- **共阳极段码**:
  可以直接在共阴极段码的基础上加上**取反符号** `~` 即可得到。

2. 模块化编程思想

思考：如何避免每次都重写`delay()`延时函数?

答案: 将 `delay` 函数封装成一个公共模块，体现模块化的编程思想。

1. **创建公共资源文件夹**:
   - `delay.c`
   - `delay.h`
2. **头文件 `delay.h` 的标准写法**:
   使用条件编译指令 `#ifndef...#define...#endif` 来防止头文件被重复包含。

```
#ifndef _delay_h_
#define _delay_h_

void delay (unsigned int z,unsigned int w)
{
    unsigned int i,j;
    for(i=0;i<z;i++)
        for(j=0;j<w;j++);
}

#endif
```

3. **在Keil C51 IDE中的设置**:
   - 在项目中，右键点击`Source Group` -> `Add Files to Group`，除了添加自己编写的 `.c` 文件，还要把公共模块 `delay.c` 添加进来。
   - **注意**: 如果在添加文件时找不到 `.h` 文件，需要将文件过滤器从 "C Source file" 修改为 "All Files"。
   - 在主程序中使用时，通过 `#include "delay.h"` 来引入。（注意是双引号 `""` 而不是尖括号 `<>`）。
   - 在IDE的 `Options for Target` -> `C51` 选项卡中，将公共资源文件夹的路径添加到 `Include Paths` 中。

3. 课堂应用与代码实现

课堂应用6

1. 在数码管M0上显示任一数字或字符

```
#include <reg52.h>

void main() {
    // 共阴极段码表 (0-9, A, b, C, d, E, F)
    unsigned char code duanma[] =
{0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71}
;
```

```
    while(1) {
        p2 = 0x00; // 选中M0数码管
        p0 = duanma[5]; // 在P0口显示数字'5'的段码
    }
}
```

2. 在数码管M0和大数码管上同时显示相同的/不同的数字

```
#include <reg52.h>

void main() {
    unsigned char code duanma[] =
{0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f}; // 0-9

    while(1) {
        // === 同时显示相同的数字 '5' ===
        p2 = 0x00; // 选中M0数码管
        p0 = duanma[5]; // M0 (共阴) 显示 '5'
        p1 = ~duanma[5]; // 大数码管 (共阳) 显示 '5'

        /*
        // === 同时显示不同的数字 (M0显示'1', 大数码管显示'2') ===
        p2 = 0x00; // 选中M0
        p0 = duanma[1]; // M0显示'1'
        p1 = ~duanma[2]; // 大数码管显示'2'
        */
    }
}
```

3. 在数码管M0上循环显示0-9数字，每个数字显示0.5秒

```
#include <reg52.h>
#include "delay.h" // 引入模块化的延时函数头文件

#define uint unsigned int

void main() {
    uint i;
    uint code duanma[] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f}; //
0-9

    p2 = 0x00; // 始终选中M0数码管
    while(1) {
        for(i = 0; i < 10; i++) {
            p0 = duanma[i];
            delay(1000, 40); // 0.5秒延时
        }
    }
}
```

4. 在数码管M0上先显示0-7，灭3秒，再显示8-F，灭3秒，循环往复

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

// 模拟一个3秒的延时
void delay_3s() {
    uint j, i;
    // 根据笔记，j=427约为2秒，所以3秒约为 j=640
    for(j = 0; j < 640; j++) {
        for(i=0; i<8; i++) {
            delay(1000 / 500, 40 / 2);
        }
    }
}

void main() {
    uint i;
    uint code duanma[] =
{0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71}
;

    p2 = 0x00; // 始终选中M0
    while(1) {
        // 1. 显示 0-7，每个0.5秒
        for(i = 0; i < 8; i++) {
            p0 = duanma[i];
            delay(1000, 40);
        }

        // 2. 熄灭3秒
        p0 = 0x00;
        delay_3s();

        // 3. 显示 8-F，每个0.5秒
        for(i = 8; i < 16; i++) {
            p0 = duanma[i];
            delay(1000, 40);
        }

        // 4. 熄灭3秒
        p0 = 0x00;
        delay_3s();
    }
}
```

5. 在数码管M0和大数码管上同时显示00-11-22 ... 99数字

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

void main() {
    uint i;
    uint code duanma[] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f}; //
0-9

    p2 = 0x00; // 选中M0
    while(1) {
        for(i = 0; i < 10; i++) {
            p0 = duanma[i]; // M0 (共阴)
            p1 = ~duanma[i]; // 大数码管 (共阳)
            delay(1000, 80); // 延时1秒
        }
    }
}
```

动态显示课堂应用

1. 在一个管子上循环显示0-7后转移到下一个管子继续显示，每个字符0.125秒

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07}; // 选管子
uint code duanma[] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07}; // 0~7的段
码
uint i, j;

void main() {
    while(1) {
        for(j = 0; j < 8; j++) { // 外层循环，选择数码管
            p2 = weixuan[j];
            for(i = 0; i < 8; i++) { // 内层循环，显示0-7
                p1 = duanma[i];
                delay(1000, 40 / 4); // 延时0.125秒
            }
        }
    }
}
```

2. 走马数码，在8个数码管上依次显示0-7，每个字符0.25秒

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07}; // 选管子
uint code duanma[] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07}; // 0~7的段码
uint i;

void main() {
    while(1) {
        for(i = 0; i < 8; i++) {
            p2 = weixuan[i];
            p1 = duanma[i];
            delay(1000, 40 / 2); // 0.25秒延时

            // 问：delay的时间短了，每个数字周围部分有点亮（视觉暂留）。如何解决？
            // 答：消影。添加一句p1=0x00;
            p1 = 0x00;
        }
    }
}
```

3. 在8个数码管上动态显示0-7，使得肉眼看起来8个数字一起显示

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07};
uint code duanma[] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07};
uint i;

void main() {
    while(1) {
        for(i = 0; i < 8; i++) {
            p2 = weixuan[i]; // 选中第i个管子
            p0 = duanma[i];  // 送出第i个段码
            delay(100, 1);   // 极短的延时，人眼无法分辨
            p0 = 0x00;       // 消影操作，动态显示必需
        }
    }
}
```

4. 在4个连续的数码管上动态显示“2025”

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07};
```

```
// 问：若显示指定数字？如显示2025
// 答：修改duanma数组即可
uint code duanma[] = {0x5b, 0x3f, 0x5b, 0x6d}; // "2025" 对应的段码
uint i;

void main() {
    while(1) {
        for(i = 0; i < 4; i++) {
            // 问：让2025居中显示？
            // 答：weixuan[i+2];
            p2 = weixuan[i + 2]; // 从M2,M3,M4,M5显示
            p0 = duanma[i];
            delay(100, 1);
            p0 = 0x00;
        }
    }
}
```

5. 分时显示2个字符串“01234567”和“89AbCdEF”，每个字符串显示2秒

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07};
uint code duanma[] =
{0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71}
;
uint i, j;

void main() {
    while(1) {
        // 问题：j取2000是我们随机取得一个值，这个循环并不一定是2s的时间。如何确定正确的j？
        // 答：断点调试程序发现，j=2000对应的运行时间是9.38s。
        // 列出比例计算式 2000：9.38 = x：2，计算的x取427。
        // 注：j=427对应运行时间2s，这个需要记忆下来。

        // 循环1 负责显示串1 "01234567"
        for(j = 0; j < 427; j++) {
            for(i = 0; i < 8; i++) {
                p2 = weixuan[i];
                p0 = duanma[i]; // 提问：i溢出问题？ （答：duanma有16个元素，
duanma[i+8]不会溢出）
                delay(1000 / 500, 40 / 2); // 短延时用于动态显示
                p0 = 0x00; // 消影
            }
        }

        // 循环2 负责显示串2 "89AbCdEF"
        for(j = 0; j < 427; j++) {
            for(i = 0; i < 8; i++) {
                p2 = weixuan[i];
                p0 = duanma[i + 8];
                delay(1000 / 500, 40 / 2);
```

```
                p0 = 0x00;
            }
        }
    }
}
```

6. 分时显示4个字符串，每个字符串显示3秒

核心思路：多个字符串，若仿照上面的写法，多个for循环太麻烦。这里采用**三重循环**解决。

```
#include <reg52.h>
#include "delay.h"

#define uint unsigned int

uint code weixuan[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07};

// 定义4个字符串的段码
uint code string1[] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07}; // "01234567"
uint code string2[] = {0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71}; // "89AbCdEF"
uint code string3[] = {0x5b,0x3f,0x5b,0x6d,0x06,0x3f,0x3f,0x6f}; // "20251009"
uint code string4[] = {0x5b,0x3f,0x5b,0x7d,0x5b,0x3f,0x7f,0x7f}; // "20262088"

uint i, j, k;

void main() {
    while(1) {
        // 最外层循环 k，用于选择4个字符串中的一个
        for(k = 0; k < 4; k++) {
            // 中间层循环 j，控制每个字符串显示3秒 (2秒是427，3秒是 427 * 1.5 ≈ 640)
            for(j = 0; j < 640; j++) {
                // 最内层循环 i，负责8个数码管的动态扫描
                for(i = 0; i < 8; i++) {
                    p2 = weixuan[i];

                    // 根据k的值选择要显示的字符串
                    if(k == 0) p0 = string1[i];
                    else if(k == 1) p0 = string2[i];
                    else if(k == 2) p0 = string3[i];
                    else p0 = string4[i];

                    delay(100, 1); // 动态扫描的短延时
                    p0 = 0x00; // 消影
                }
            }
        }
    }
}
```