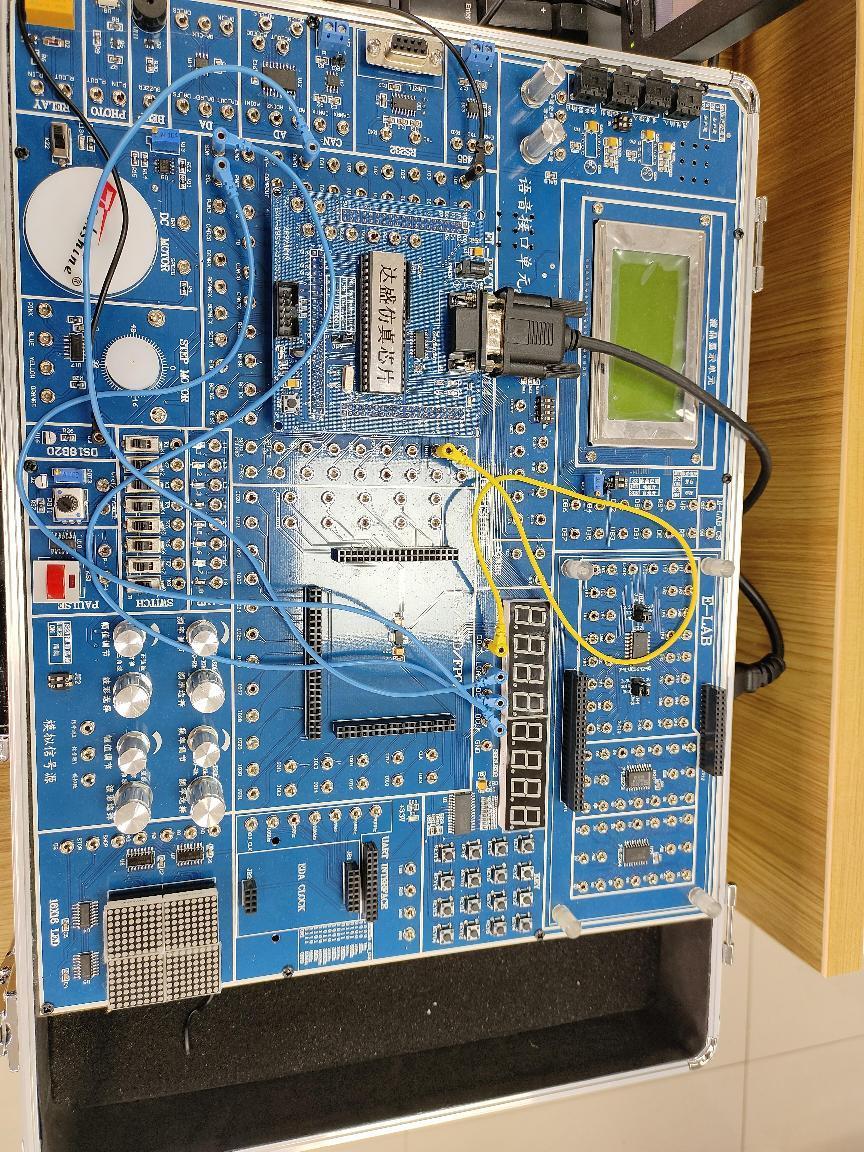
**实验报告五 键盘和温度测量**

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 学号 | 组内分工 | 工作量占百分比 |
| 李纪群 | 21211020 | 键盘程序的编写，实验报告撰写 | 50% |
| 冯哲熙 | 21211019 | 温度测量程序的编写，部分实验报告编写 | 50% |

一、电路图

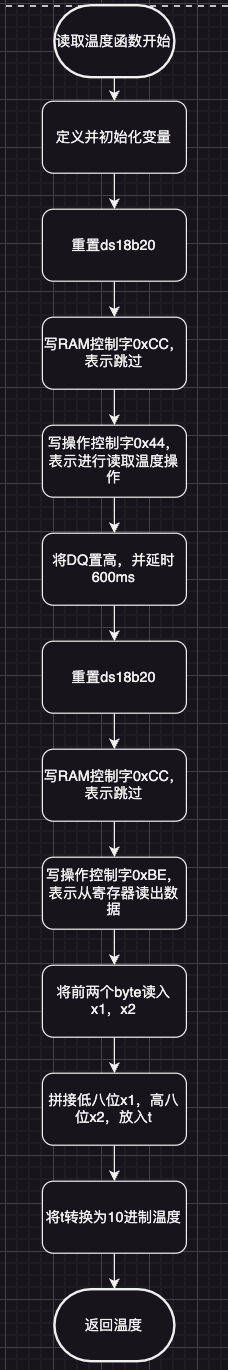


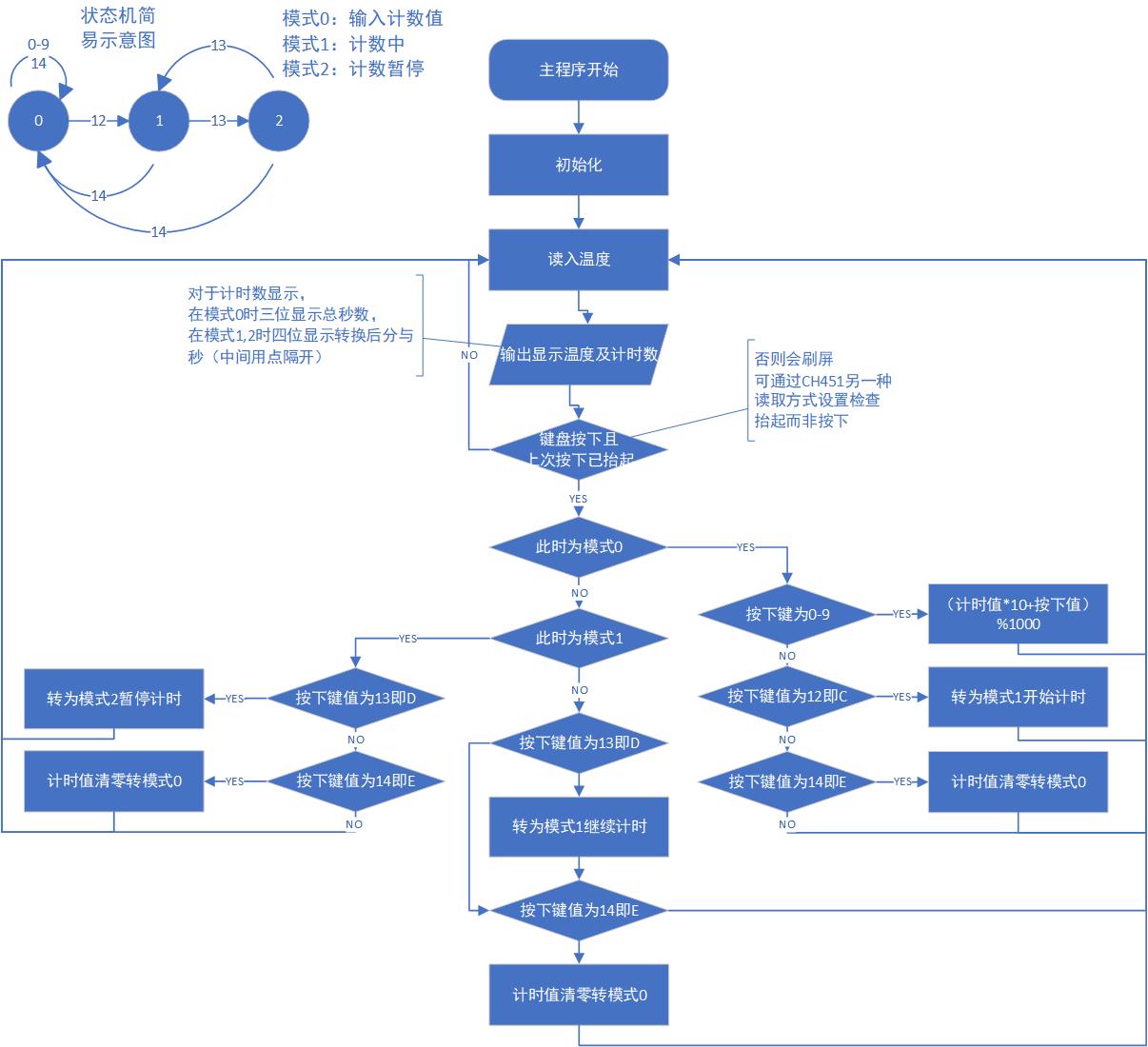
(P1.0-P1.2)CAP0,SDA,SCL->LOAD,DIN,CLK(输出控制ch451)

(P3.2)INT0->DOUT(CH451信号输出)

(P0.0)IO1->DQ(温度控制读入口)

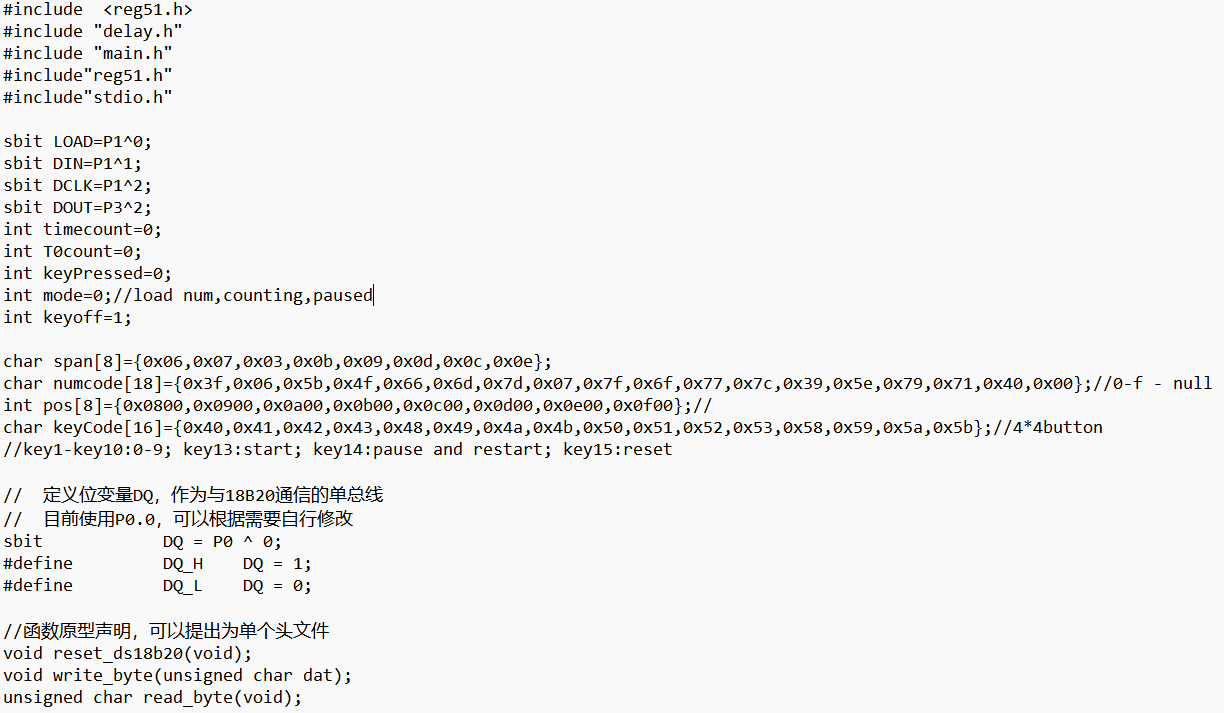
二、程序分析

1、流程图的形式给出程序设计的思路，



1. 代码及必要注释。

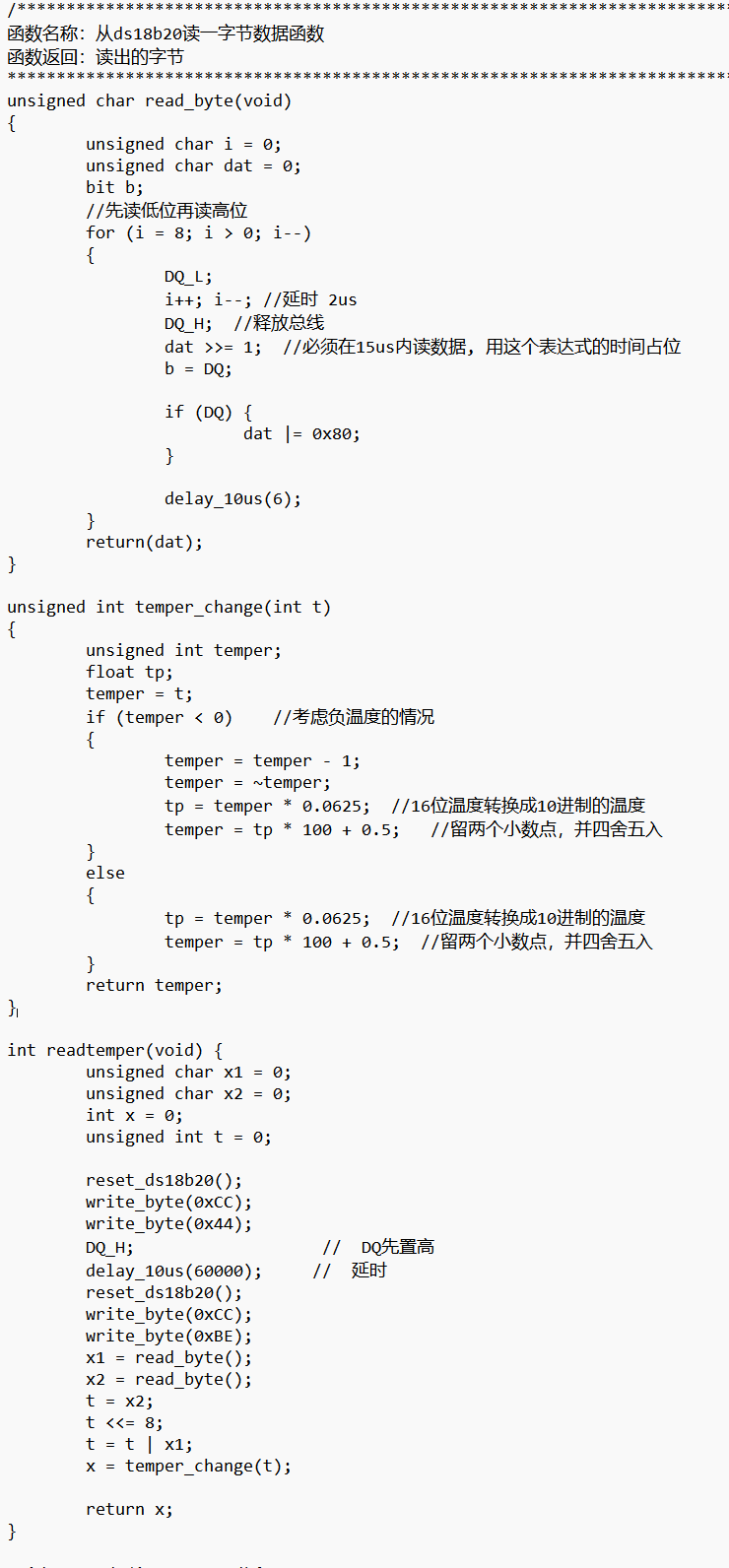
前置声明：



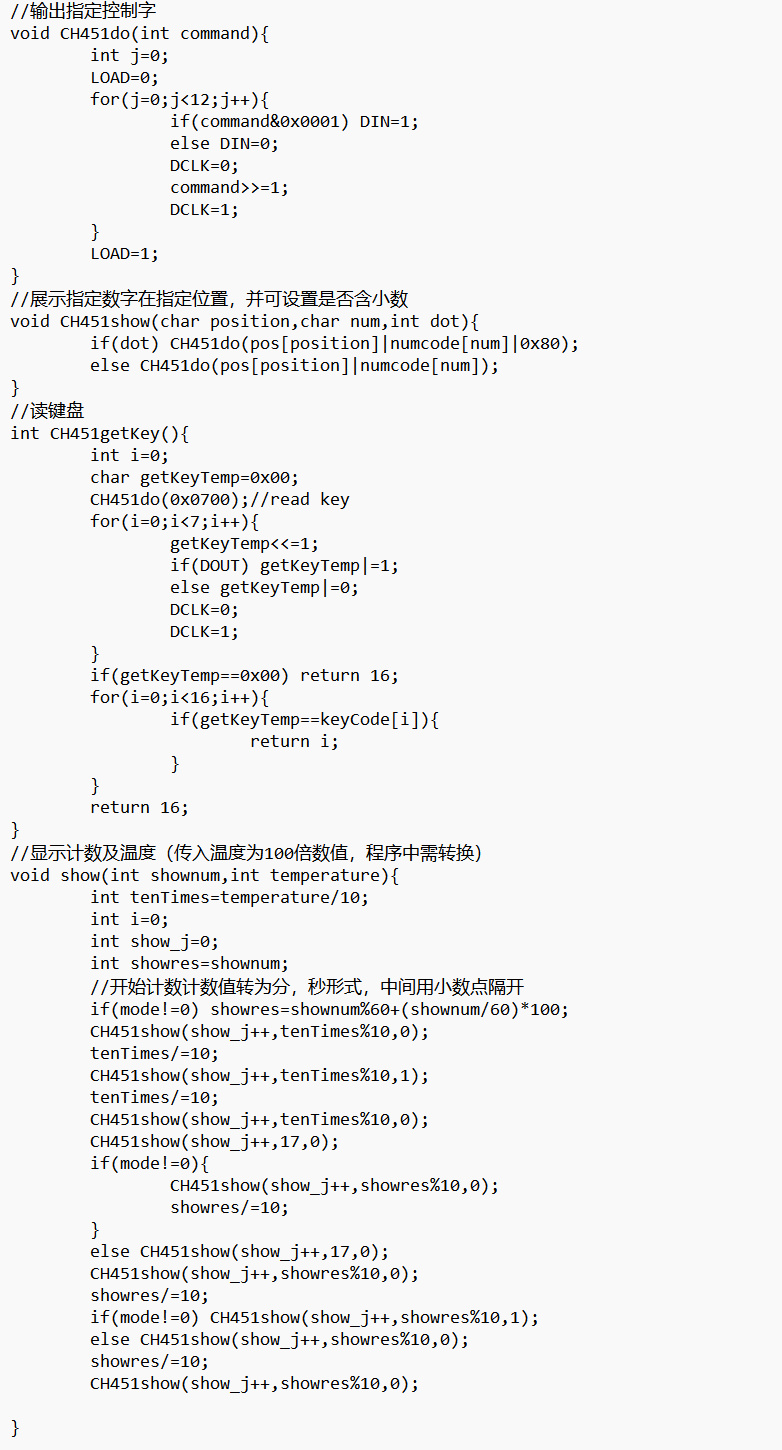
温度部分代码1：



温度部分代码2：



CH451部分代码：



主函数及中断：



1. 思考题（见PPT）

1：说明单线通讯的优点和缺点

答：一个芯片、仪器部件的接口数量是有限的，单线通讯能够最大程度地利用这些接口，使得这些有限地接口尽可能地与更多的设备进行联通，使得最终产生的集合体能够实现更多的功能，同时，单线通讯因为使用接口和连线较少，占用更少的资源的同时，在出现问题如器件损坏时，能够更简单地确定损坏部位（检查一个接口总比分别检查多个接口容易）。但是因为仅使用单一接口，单线通讯对于通讯的时序要求较高（如此次的温度），同时接收信息的速率低于多个接口，必须要一定的时钟周期后才能将所需信息全部接收转换。

2：影响单片机精确定时的因素有哪些？如何尽量避免这些因素的影响？

答：影响单片机精确定时的因素包括硬件的因素和程序的因素。

硬件因素如时钟源自身出现问题，外部的电磁干扰等，只能通过改变物理环境与器件来避免影响。

而程序因素如对于定时器的设置上的问题在思路正确时基本不会出现。以下讨论并给出部分可能出现的影响精确定时的程序上的情况。

一：所需的精确定时十分精确，如精确到1ms等，此时，若在中断中插入耗时过大的代码，会使得此计时器定时实际上两次中断之间的时间差变大，由理想中的定时时间变大变为实际上运行完中断中的代码所需的时间。故在定时间隔较小时最好将长程序尽量不要置于控制定时的中断中（若长代码置于另一中断，也要确保用于定时的中断优先级高于那一中断，可以打断进行计时）

二：此次实验所需的倒计时的功能，若考虑在暂停计时后又重新开始计时，并使得总的倒数时间还是相同，不会变多（比如考虑如下情况：10秒倒计时，在倒数至5.1秒时（此时显示6），暂停计时，重新开始计时，若是按照显示，从6s重新开始计时，则最终总的计时时间为10.9秒，比10秒多，在多次暂停时，这个变化还会进一步增加），此种意义上的精确定时，则要对于计时的划分进行一个讨论。若是要较为精确地从上一次计数时间开始继续进行计时，则需要存储当时计时的进度，要将秒划分地更加精细。此次实验中我们所采用的计时方法，将一秒分为20部分，每50ms进入一次中断（所能找到的最大的，一次计时器所能计时且整数倍后适配1s的计时值），暂停又重新开始时，从上次的计数次数开始继续计数，这样就将一次暂停产生的误差控制在了50ms内.若要继续缩小此误差，一种方法是使用更细分的计时方法，如20ms进入一次中断，50次后判定一次1秒。但此种方法频繁进入中断可能会带来如一所述的问题。另一种方法是在暂停时将计时器的TH和TL记录下来，再次开始时从此开始。

3：讨论如何去除键盘输入抖动带来的干扰？

答：首先声明，此次我们所使用的，检查键盘的方法为，在按下一个键后，需检查到按下键抬起，才可以进行下一个键的检查，学习通所给文档中有讲述在按键抬起而非按下再读出键值的方法，但是我们未采用。

以实验的实际使用感受来说，CH451已经对于键盘的抖动有进行消除，同时，阅读文档也可以看到其中有说明“为了防止因为按键抖动或者外界干扰而产生误码，CH451实行两次扫描，只有当两次键盘扫描的结果相同时，按键才会被确认有效。”此即为CH451对于键盘输入抖动的消除方法。双重检查的方法可以使得若是抖动，则两次结果不一致，结果无效。

若是仅靠我们编写程序来消除此种键盘抖动，则也是依照此思路，在按键值变化时，延时一定时间再次检查，若确实发生变化，则结果有效，读入，若变化后又变回来了，则说明为扰动，结果无效。

四、问题分析

1、实验过程中遇到的问题及解决方法。

一：按键按下后刷屏

解决方法：检测按键抬起，在抬起后下次的键盘读入才能算作有效输出进行处理。

二：对于小数点的显示，在显示小数点后会出现所有的偶数位的数显示部位全部变为CH451最左侧的问题

解决方法：似乎是写入小数，转化存在问题

原代码为CH451do(pos[position]|(numcode[num]|0x80))会出现问题，改为CH451do(pos[position]|numcode[num]|0x80)后又可以正常运行（CH451do为输入一个12位的指令，执行这个指令），猜测可能是因为可能存在强制类型转换，转换中间存在问题。

三：读出温度但是温度不正确

首先是读出温度时分高八位和第八位进行读出，读出后没有正确拼接导致了这个问题。其次没有注意到温度计的精度单位，没有正确地转换成十进制数字。

解决方法：加入了 一个温度转换函数，将读出来的数正确拼接之后，乘以0.0625就得到了正确的温度。

四：无法读出数据，关键字错误以及没有正确的设置延时。

解决方法：在发送命令使其读出温度后，设置一个延时，给他充足的时间读出温度。检查控制字是否正确，由于我们是使用单个ds18b20，所以不需要RAM定位之类的操作，在RAM操作控制字输出时全部写入控制字0xCC跳过即可。

1. 实验的收获或感想。

李纪群：

此次实验工作分工极其明确，分为温度部分与键盘部分，我负责了键盘程序的编写，以及计时控制部分的编写。

有了实验四的基础，键盘程序的编写并无那么困难，键盘按键的消除抖动CH451内部也帮我们做了，只要看文档理解了如何读取键盘输出的值便可以对代码如何编写有个七七八八的感觉了。所以关于键盘部分最令人头疼的，居然落到了debug上，就是问题分析中所说的那个，关于小数点显示的bug。C51的调试我没有使用过，功能上感觉找不太到像vs一样监视一个变量的数值的功能，所以在debug的时候感觉不太方便，因为我当时不太能确定传回CH451控制字是否正确……后面还是手动写了个传回显示的12位控制字测试才意识到可能是计算的公式会有问题。

至于计时的模式切换，因为已经完成微机实验，而微机实验中也有关于计时器的编写，而且因为纯用汇编，分段辨别更难，现在遇到相对清晰明了的C51，上手就更加轻松了。虽然早就有思路，但看到PPT中提到设计有限状态自动机来控制，我还是感觉眼前一亮，因为之前也没有注意，编译原理所学的知识在这里又实际运用了起来，各个课程间知识互通，互相辅助，然后自己动手实际运用设计，这一点还是挺有成就感的。

冯哲熙：

本次试验算是目前最有趣的一次实验，因为这是第一次完整的做了一个有实际功能的硬件链接和软件编写。通本次试验我不由得就觉得我自己已经具备了一定自己动手制作一个测温计的能力。以后放假没准会从网上购买一些相关硬件部件，自己看说明书进行编程做一个小温度计，非常有趣。

本次分工也是相当明确，分两部分，我完成的是温度计部分。本次试验也是花了不少时间反复通读文档。先大概了了解硬件的作用和工作原理，然后再仔细识别内容，找到自己所需的内容。内容比较散，所以需要反复看细心寻找。主要是所需内容也不是和明显，这种解决方法需要自己寻找然后搭配组合尝试，直到得出正确结果。

仔细看文档会发现，其实已经有一段将如何执行的顺序样例给了出来，能看出来整个读取温度的过程还是相当线性的，但是其实样例和我们的试验并不一样，还需要替换一些部分，不过通过文档所示样例我很快理解了试验原理和过程。由于我是周围最先做到这个试验的，我并没有得到别人的指点和借鉴，我遇到不懂和不对的地方的解决方法就是反复阅读文档，看看有没有之前遗漏或者是理解错误的地方。本实验的关键在于搞懂这个控制原理后，安装正确的顺序进行输出关键字，置控制位的电压，读取数据，已经温度转换。其中控制字也得确保是正确适配此次试验的，不可以忙不套用样例中的控制字。需要自己看不同控制字的区别，进行选择。

大致的流程顺序对了之后，还需要注意细节的实现。例如如何将原始的温度数据转换为10进制的数字，还需要注意温度计的精度单位。

总的来说本次试验在我和同伴2人合理分工之后并不是非常复杂，我们又学会了一些硬件和相关软件控制的知识。试验十分有趣，我们做得也比较快，在程序终于能正常运行时我们都有抑制不住的兴奋。