
电子装配与工艺 实验指导书

米月琴 周佳社

国家级电工电子教学基地

2022 年 11 月

目录

第一章 实验目的、任务、要求	3
§ 1.1 目的	3
§ 1.2 任务	3
§ 1.3 方式	3
§ 1.4 要求	3
§ 1.5 考核	4
第二章 正规化装配和焊接	4
§ 2.1 实验目的	4
§ 2.2 相关知识	10
§ 2.3 操作	14
第三章 基于单片机的编程实验	21
§ 3.1 跑马灯	21
§ 3.2 温度显示	27
§ 3.3 矩阵键盘	34
§ 3.4 收音机	39

第一章 实验目的、任务、要求

§ 1.1 目的

本实验旨在研发出由 51 单片机为主控制器,集成了 EEPROM 电路、多位数码管电路、LED 排灯电路、4x4 矩阵键盘电路、38 译码器电路、温控电路、LCD 显示电路、收音机调频电路、总线电路、接口电路等多种外围电路的实验电路板。

该电路板上各个元器件尽量采用不同的焊接方式,方便学生用于电子装配、硬件扩展、软件编程与调试等不同层级的实验。时间跨度可从大一到大三。

§ 1.2 任务

《电子装配与工艺》是高校教学实践课程的重要环节。

该课程主要以收音机、温度显示、矩阵键盘等为对象,通过正规化、规范化的强化训练,让学生了解并体会电子产品商业化生产的全过程。该课程的实施,是对传统理论知识教育环节的有力补充,同时,这种模拟化的装配调试,对学生工程师基本素质的训练、思维方式的成长,特别是在实际动手方面都起着重要作用。

§ 1.3 方式

参加课程的学生按选课计划进行实验。

- (1) 学生在每学期安排的上课时间选上课的学周。
- (2) 每个学生必须在规定时间内完成作品,按时归还领取的工具和钥匙。
- (3) 指导教师安排具体事宜。

§ 1.4 要求

为保证每位同学顺利完成本课程,基地规定以下学生守则:

1、在实践规定时间内,在老师指导下完成以下任务:

- (1) 清点工具箱内工具;
- (2) 清点元器件。

如清点中发现工具或材料有误,立即请指导教师核对;过时再缺少或损坏由学生自己负责。

2、装配和焊接操作必须严格按照工艺规范进行,凡操作不当造成损坏,照价赔偿。

3、提倡学生间互相帮助检查。严禁代装、代调,凡违反者,两者均按作弊处理。

-
- 4、保持场内安静、文明、整洁、卫生。禁止吸烟，禁止随地吐痰。
 - 5、自觉遵守场内安全、防火规定，防止事故发生。
 - 6、离开实验室前，做好工具材料整理工作，并打扫周围卫生。
 - 7、实习结束时，保持工具完整齐备，场地清洁。损坏或丢失工具者照价赔偿。

§ 1.5 考核

每位学生在选定的时间内完成整个实践过程，指导老师根据该学生在实践期间的考勤情况、认真程度、实践动手能力和实践效果综合情况进行评分。

在实践期限内，不认真努力，不听认指导教师分配和引导，而不能完成实践任务者，该课程将给予不及格分数。

学生考核成绩由以下几方面构成：

- | | |
|--------------------|------|
| 1. 思想态度(10%) | 10% |
| 2. 实践能力 50%) | |
| (1) 规范化 | 15% |
| (2) 操作能力 | 30% |
| (3) 故障排除能力 | 5% |
| 3. 实践产品质量及调试(15%) | |
| (1) 仪器使用 | 10% |
| (2) 总体质量(包括外型、音质等) | 5% |
| 4. 实践报告(25%) | |
| (1) 理论阐述 | 5% |
| (2) 操作步骤 | 15% |
| (3) 认识体会 | 5% |
| 总分 | 100% |

第二章 正规化装配和焊接

焊接是电子产品组装过程中的重要工艺。焊接质量的好坏，直接影响电子产品工作的稳定性和可靠性，是电子产品质量的关键。

本项目主要安排了焊接基本知识、焊接材料和工具的选用、手工焊接方法、步骤和要求、元器件的安装规范、自动焊接流程等方面的训练内容。

§ 2.1 实验目的

- 1.了解焊接原理和焊接条件，熟悉焊接材料的性能。
- 2.掌握焊接工具的结构、性能、用途以及选用、操作要求。
- 3.掌握元器件成型、插装、焊接工艺标准。
- 4.熟练掌握手工焊接要领、拆焊方法。
- 5.熟悉自动焊接流程和工艺要求

§ 2.2 元器件识别

装配前识别先器件的好坏和准确数据，是保证装配电子产品质量的重要步骤之一。

一、电阻

电阻在电路中用来限制电流、降低电压、分配电流、分配电压，可与电容组成电源退耦电路、低通电路、高通电路等，还可给晶体管等元件提供必要的工作条件（提供电压或电流）。

电阻分为固定电阻和可变电阻（即电位器）。其主要技术参数有：标称阻值、阻值误差、额定功率。

电阻的阻值标称有二种方法：一是直接在电阻上标出数据；二是用色环表示阻值。色环表示方法可在任意角度识别其阻值大小，使用很方便，被广泛使用。

常见的有四环电阻与五环精密电阻。

图 2-1 四环色环电阻紧靠电阻头的第一道色环表示阻值的第一位数，第二道色环表示第二位数，第三道色环表示幂的次方，第四道色环表示误差，其中色环的读法见表 1：

举例说明：如色环为“蓝红橙金”，其阻值为： $62 \times 10^3 = 62k\Omega \pm 5\%$ 。

当电流流过电阻器时，就要消耗电能，产生热量。在规定的温度条件下，电阻器长期工作时所允许承受的最大电功率叫电阻器的额定功率，单位是 W（瓦）。

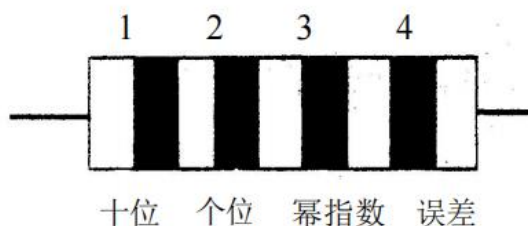


图 2-1 四环色环电阻识别示意图

表 1 四环色环对照表

颜色 序号	棕	红	橙	黄	绿	蓝	紫	灰	白	黑	金	银	无色
1	1	2	3	4	5	6	7	8	9	0	/	/	/
2	1	2	3	4	5	6	7	8	9	0	/	/	/
3	10 ¹	10 ²	10 ³	10 ⁴	10 ⁵	10 ⁶	10 ⁷	/	/	10 ⁰			
4	/	/	/	/	/	/	/	/	/	/	±5%	±10%	±20%

精密五色环电阻器的读法：

前三环为有效数字，第四环为倍率，第五环为误差环。色环读法见表 2：

表 2 五环色环对照表

第一环表示 第 1 个字 a		第二环表示 第 2 个字 b		第三环表示 第 3 个字 c		倍数 d		电阻 公差	
颜色	数字	颜色	数字	颜色	数字	颜色	倍数	颜色	公差
黑色	0	黑色	0	黑色	0	黑色	1	银色	±10%
棕色	1	棕色	1	棕色	1	棕色	10	金色	±5%
红色	2	红色	2	红色	2	红色	100	棕色	±1%
橙色	3	橙色	3	橙色	3	橙色	1,000	红色	±2%
黄色	4	黄色	4	黄色	4	黄色	10,000	橙色	±3%
绿色	5	绿色	5	绿色	5	绿色	100,000	绿色	±.5%
蓝色	6	蓝色	6	蓝色	6	蓝色	1,000,000	蓝色	±.25%
紫色	7	紫色	7	紫色	7	银色	0.01	紫色	±.1%
灰色	8	灰色	8	灰色	8	金色	0.1		
白色	9	白色	9	白色	9				

四环的色标电阻= $a b \cdot 10^d \Omega$



五环的色标电阻= $abc \cdot 10^d \Omega$



二、电容

电容是由两个绝缘介质隔开的金属极板组成的。在电子电路中，它可以用来隔离直流、耦合交流信号，与电阻器或电感线圈组成低通或高通电路，能将交流信号旁路，与电感线圈组成串联或并联谐振电路等。

电容的种类非常多，但最常用电容有瓷片电容、电解电容、聚酯电容等。如图 2-2。

电容的主要技术参数有标称容值、容值误差、额定电压、绝缘电阻。

1. 瓷介电容容量较小，容量范围一般在 $1\text{pF}\sim 1\mu\text{F}$ 之间。形似圆饼状，其表示方法有：

(1) 直接表示法用 $\mu\text{F} = 10^{-6}$ 法拉， $\text{nF} = 10^{-9}$ 法拉， $\text{pF} = 10^{-12}$ 法拉 来表示电容容量量级单位。

举例：“3P”=3pF，“ 0.01μ ”= $0.01\mu\text{F}$ ，“4n7”=4.7nF=4700pF

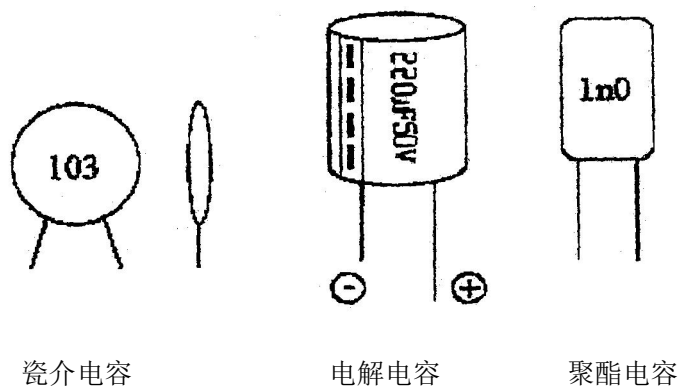


图 2-2 电容的识别

(2) 不标单位的直接表示法

举例：“3”=3pF，“27”=27pF，“0.047”= $0.047\mu\text{F}$

(3) 数码表示法

一般用三位数表示，前两位表示容量有效数字，第三位表示幂指数，即“0”的个数。默认单位为 pF。

举例：“203”= $20 \times 10^3 = 0.02\mu\text{F}$ ，“223”= $22 \times 10^3 = 0.022\mu\text{F}$

“104”= $10 \times 10^4 = 0.1\mu\text{F}$ ，“103”= $10 \times 10^3 = 0.01\mu\text{F}$

2. 电解电容容量较大，一般在 $0.1\mu\text{F}\sim 9999\mu\text{F}$ 之间，形似圆柱状，具有极性区分：

(1) 新电解电容以管脚长短为标志：长脚为正极，短脚为负极。

(2) 在外壳封装上有极性标志。

(3) 容量标识在塑封外壳上，例如：

“1 μ F50V” 代表：容量 1 μ F，耐压值 50V。

容值误差指示了电容器容值的允许误差量，等于电容器实际容值与标称容值之差除以标称容值所得的百分数。

额定电压是指在一定环境温度下，电容器长时间可靠地工作所能承受的最大直流电压，通常简称“耐压”。

当电容器两端加上直流电压 U 长时间充电后，电容支路仍有电流 I 存在，电流 I 叫做电容器漏电电流，则绝缘电阻为 $R = \frac{U}{I}$ 。绝缘电阻越小，漏电越严重，引起的能量损耗也就越大。

三、电感线圈

电感线圈是根据电磁感应原理制成的器件，其用途极为广泛，是电子线路中主要的元件之一。电感线圈分为两大类，一类是应用“自感”作用的线圈，另一类是应用“互感”作用的变压器。

电感线圈有阻碍交流通过的作用，而对稳定的直流电流却不起作用。电感线圈和电容器配合使用，可起调谐、滤波、选频、退藕等作用。电感线圈用符号 L 表示。电感量的基本单位为亨利（H），简称亨，其它常用单位有（mH）和微亨（uH），其换算关系如下：

$$1H=1000mH$$

$$1mH=1000uH$$

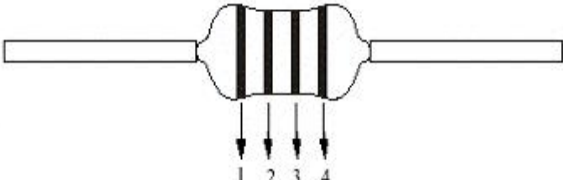
四、色码电感

使用颜色环带（或色点）表示电感线圈性能的小型电感，称为色码电感。色码电感以铁氧体磁芯为基体，外表进行涂覆，适用频率一般在 10KHZ—200MHZ，它的工作电流可分为 50mA,150mA,300mA,700mA,1.6A 等档位。结构有卧式和立式两种。它们主要用作高频滤波电感、回路电感等。

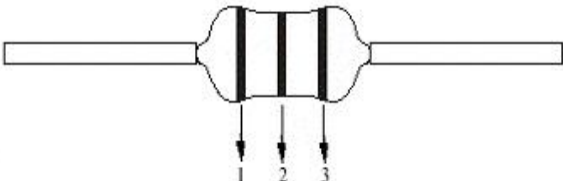
色码电感值判断方法如下图所示。

色碼				
顏色	1 色環	2 色環	3 倍率	4 標稱電感值
黑	0	0	1	$\pm 20\%$
棕	1	1	10	-
紅	2	2	100	-
橙		3	3	1000
黃	4	4	-	-
綠	5	5	-	-
藍	6	6	-	-
紫	7	7	-	-
灰	8	8	-	-
白	9	9	-	-
金	-	-	0.1	$\pm 5\%$
銀	-	-	0.01	$\pm 10\%$

(1) LGA0307, LGA0410



(2) LGA0204, LGA0202



五、三极管

图 2-3 所示为三极管的几种常见外形，其共同特征就是具有三个电极，这就是“三极管”简称的来历。通俗来讲，三极管内部为由 P 型半导体和 N 型半导体组成的三层结构，根据分层次序分为 NPN 型和 PNP 型两大类。

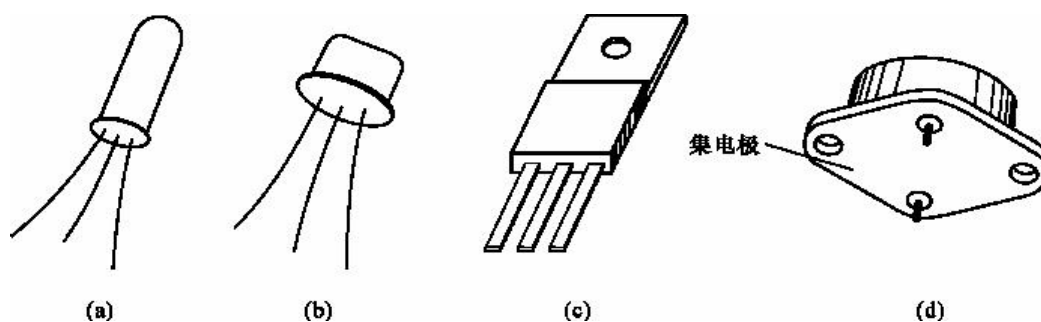


图 2-3 常见三极管类型

用万用表测试三极管的方法：

(1) 判别基极和管子的类型

选用欧姆档的 $R \times 100$ （或 $R \times 1K$ ）档，先用红表笔接一个管脚，黑表笔接另一个管脚，可测出两个电阻值，然后再用红表笔接另一个管脚，重复上述步骤，又测得一组电阻值，这

样测 3 次，其中有一组两个阻值都很小的，对应测得这组值的红表笔接的为基极，且管子是 PNP 型的；反之，若用黑表笔接一个管脚，重复上述做法，若测得两个阻值都小，对应黑表笔为基极，且管子是 NPN 型的。

（2）判别集电极

因为三极管发射极和集电极正确连接时 β 大（表针摆动幅度大），反接时 β 就小得多。因此，先假设一个集电极，用欧姆档连接，（对 NPN 型管，发射极接黑表笔，集电极接红表笔）。测量时，用手捏住基极和假设的集电极，两极不能接触，若指针摆动幅度大，而把两极对调后指针摆动小，则说明假设是正确的，从而确定集电极和发射极。

（3）电流放大系数 β 的估算

选用欧姆档的 R*100（或 R*1K）档，对 NPN 型管，红表笔接发射极，黑表笔接集电极，测量时，只要比较用手捏住基极和集电极（两极不能接触），和把手放开两种情况小指针摆动的大小，摆动越大， β 值越高。

六、二极管

二极管种类有很多，按照所用的半导体材料，可分为锗二极管（Ge 管）和硅二极管（Si 管）。根据其不同用途，可分为检波二极管、整流二极管、稳压二极管、开关二极管等。

变容二极管是一个特殊的二极管，它的 PN 结电容随 PN 结的偏压增大而减小，改变变容二极管 PN 的电压来达到本振频率的改变。

二极管的识别很简单，小功率二极管的 N 极（负极），在二极管外表大多采用一种色圈标出来，有些二极管也用二极管专用符号来表示 P 极（正极）或 N 极（负极），也有采用符号标志为“P”、“N”来确定二极管极性的。发光二极管的正负极可从引脚长短来识别，长脚为正，短脚为负。

§ 2.3 相关知识

一、焊接的基本知识

1. 焊接原理

采用锡铅焊料进行的焊接称为锡焊。锡焊是最早得到广泛应用的一种电子产品的布线连接方法。

锡焊的原理是：通过加热的烙铁将固态焊锡丝加热熔化，再借助于助焊剂的作用，使其流入被焊金属之间，待冷却后形成牢固可靠的焊接点。

2. 锡焊过程

(1) 润湿

润湿过程是指已经熔化了了的焊料，借助毛细管力沿着母材金属表面细微的凹凸和结晶的间隙向四周漫流，从而在被焊母材表面形成附着层，使焊料与母材金属的原子相互接近，达到原子引力起作用的距离。

引起润湿的环境条件：被焊母材的表面必须是清洁的，不能有氧化物或污染物。

(2) 扩散

伴随着润湿的进行，焊料与母材金属原子间的相互扩散现象开始发生，使熔化的焊料与母材中的原子相互越过接触面进入对方的晶格点阵，原子的移动速度与数量决定于加热的温度与时间。

(3) 形成金属化合物

由于焊料与母材相互扩散，在两种金属之间形成了一个中间层——金属化合物。要获得良好的焊点，被焊母材与焊料之间必须形成金属化合物，从而使母材达到牢固的结合状态。

3. 锡焊必须具备的条件

(1) 焊件必须具有良好的可焊性；(2) 焊件表面必须保持清洁

(3) 合格的焊料；(4) 要使用合适的助焊剂

(5) 合理的焊点；(6) 焊件要加热到适当的温度

(7) 合适的焊接时间

二、焊接材料

1. 焊料

在一般的电子焊接中，主要使用锡铅合金焊料，俗称焊锡。焊锡的熔点为 183℃左右，一般形状为管状，里面填有助焊剂，焊锡丝的直径有 0.5mm、0.8mm、1.0mm、1.2mm、1.5mm、2.0mm、2.5 mm、3 mm 等规格。



2. 助焊剂

锡焊中常采用松香作为助焊剂，松香加热到 70℃时熔化后就能去除金属表面的氧化物。将松香溶入酒精可制成松香水，将其涂在敷铜版上起到防氧化和助焊的作用。

3. 阻焊剂

是一种耐高温的涂料，它可使焊接只在需要焊接的点上进行，而将不需要焊接的部分保护起来。应用阻焊剂可以防止桥连、短路等现象发生，减少返修，提高劳动生产率，节约焊料，并可使焊点饱满，减少虚焊发生，提高了焊接质量。

三、焊接工具

1. 电烙铁

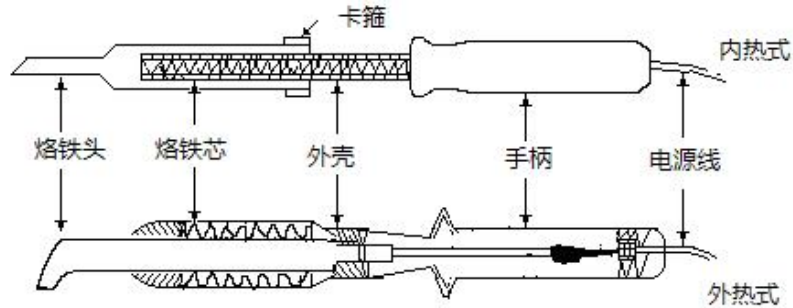
(1) 电烙铁的类型和结构

直热式电烙铁



内热式

外热式



内部结构

吸锡电烙铁



调温及恒温电烙铁



电焊台



热风枪



(2) 电烙铁的选用

电烙铁的种类和功率的选择

焊件及工作性质	选用电烙铁
一般元器件、细导线、印制电路板	20W内热式，30W外热式，恒温式
集成电路	20W内热式，恒温式，储能式
MOS管	储能式
焊片，电位器，2~8W电阻，大功率晶体管	35W内热式，调温式，50~75W外热式
8W以上大电阻，直径2mm以上导线等较大元器件	100W内热式，150~200W外热式
金属板	300W以上外热式
SMT表贴元器件	恒温式，电焊台

2. 其它工具

(1) 尖嘴钳：它的主要作用是在连接点上网绕导线、元件引脚及对元件引脚成型等。

(2) 偏口钳：又称斜口钳、剪线钳，主要用于剪切导线，剪掉元器件多余的引脚。不要用偏口钳剪切螺钉、较粗的钢丝，以免损坏钳口。

(3) 剥线钳：用于剥去导线绝缘皮。

(4) 镊子：主要用途是摄取微小器件；在焊接时夹持被焊件以防止其移动和帮助散热。

(5) 螺丝刀：又称改锥，分为十字改锥、一字改锥。主要用于拧动螺钉及调整可调元器件的可调部分。

(6) 小刀：主要用来刮去导线和元件引脚上的绝缘物和氧化物，使之易于上锡。

另外还有用来清洁的钢刷、纱布、砂纸及锉刀；用来拆焊的吸锡器和吸锡线；用来放置电烙铁的烙铁架以及松香盒；防静电手腕带、防静电指套等。

§ 2.4 操作

一、操作前准备

1. 安全检查

查看电烙铁手柄上的螺钉是否紧固，电源线有无破损，烙铁头是否松动。用万用表检查电源线有无短路、开路，电烙铁是否漏电。

2. 烙铁头上锡处理

给电烙铁通电，将烙铁头压在松香上，等到松香冒烟较大时，取出烙铁头，迅速与焊锡丝接触，直到烙铁头上均匀镀上一层焊锡为止。

3. 检查吸锡海绵

检查吸锡海绵是否有水和清洁，若没水，请加入适量的水。

4. 防静电检查

保证焊接人员戴防静电手腕带、绝缘手套、防静电工作服。

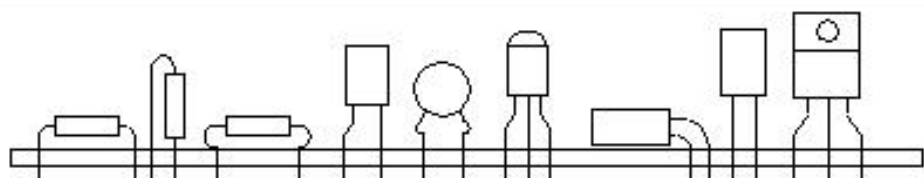
5. 熟悉所印制电路板的装配图

按图纸配料检查元器件型号、规格及数量是否符合图纸上的要求。

二、元器件引脚成型与插装

1. 去除氧化层

2. 引脚成型



元器件引脚成型示意图



元器件引脚弯曲度

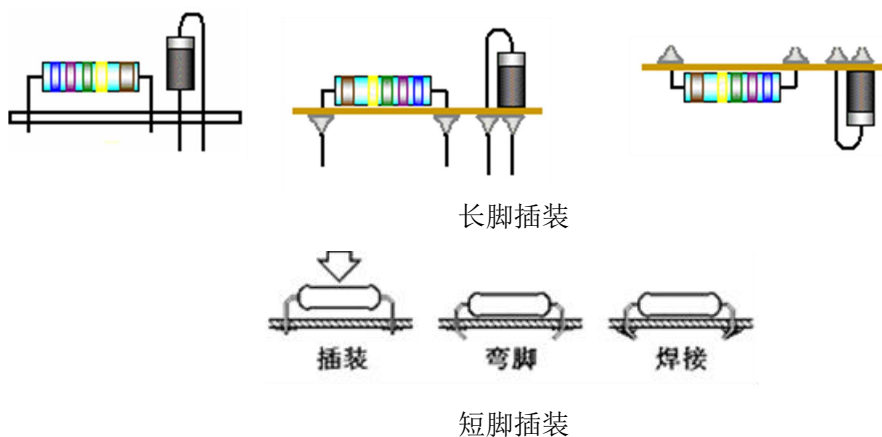
①手工插装、焊接，应该先插装那些需要机械固定的元器件，如功率器件的散热器、支架、卡子等，然后再插装需焊接固定的元器件。插装时不要用手直接碰元器件引脚和印制板上铜箔。手工插焊遵循先低后高，先小后大的原则。

②元器件插装要求做到整齐、美观、稳固，元器件应插装到位，无明显倾斜、变形现象；卧式安装的元器件，尽量使两端的引脚的长度对称，元器件放在两孔中央。

③插装有极性的元器件，按线路板上的丝印进行插装，不得插反和插错；对于有空间位置限制的元器件，应尽量将元器件放在丝印范围内。

④各种元器件的安装，应该使它们的标记（用色码或字符标注的数值、精度等）朝左和朝下，并注意标记方向的一致性（从左到右），以符合阅读习惯。

（3）长短脚的插焊技巧

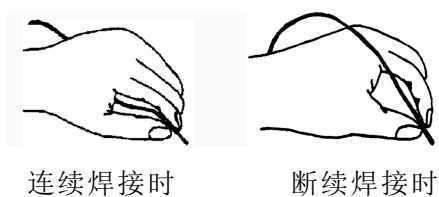


三、焊接要领

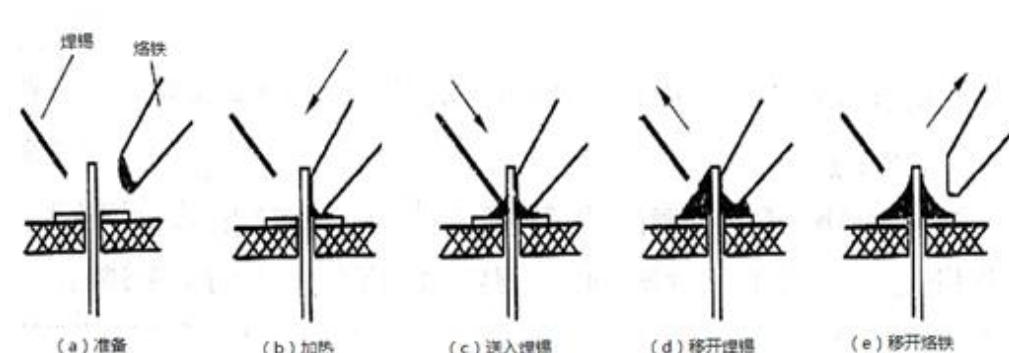
1. 电烙铁的握法



2. 焊锡丝的拿法



3 操作步骤



4. 注意事项

（1）保持烙铁头的清洁

-
- (2) 焊锡用量要适中
 - (3) 焊剂用量要适度
 - (4) 掌握好烙铁温度和焊接时间
 - (5) 焊接时焊件不能晃动

四、常用元器件的焊接要求

1. 电阻器的焊接

按图将电阻器准确地装入规定位置，并要求标记向上，字向一致。装完一种规格再装另一种规格，尽量使电阻器的高低一致。焊接后将露在印制电路板表面上多余的引脚齐根剪去。

2. 电容器的焊接

将电容器按图纸要求装入规定位置，并注意有极性的电容器其“+”与“-”极不能接错。电容器上的标记方向要易看得见。先装玻璃釉电容器、金属膜电容器、瓷介电容器，最后装电解电容器。

3. 二极管的焊接

正确辨认正、负极后按要求装入规定位置，型号及标记要易看得见。焊接立式二极管时，对最短的引脚焊接时，时间不要超过 2 秒钟。

4. 三极管的焊接

按要求将 e、b、c 三根引脚装入规定位置。焊接时间应尽可能地短些，焊接时用镊子夹住引脚，以帮助散热。焊接大功率三极管时，若需要加装散热片，应将接触面平整，打磨光滑后再紧固，若要求加垫绝缘薄膜片时，千万不能忘记管脚与线路板上焊点需要连接时，要用塑料导线。

5. 集成电路的焊接

将集成电路插装在印制线路板上，按照图纸要求，检查集成电路的型号、引脚位置是否符合要求。焊接时先焊集成电路边沿的两只引脚，以使其定位，然后再从左到右或从上至下进行逐个焊接。焊接时，烙铁一次沾取锡量为焊接 2-3 只引脚的量，烙铁头先接触印制电路的铜箔，待焊锡进入集成电路引脚底部时，烙铁头再接触引脚，接触时间以不超过 3 秒钟为宜，而且要使焊锡均匀包住引脚。焊接完毕后要查一下，是否有漏焊、碰焊、虚焊之处，并清理焊点处的焊料。

五、贴片元件焊接技巧

1. 焊接贴片元件常用工具

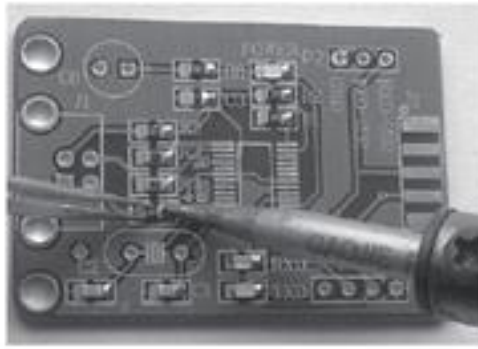
- (1) 电烙铁 (2) 热风枪 (3) 放大镜
- (4) 镊子 (5) 吸锡带

2. 贴片元件手工焊接步骤

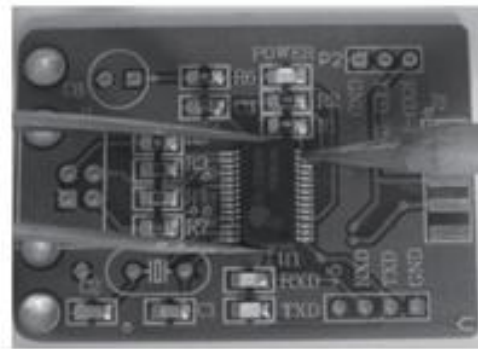
(1) 焊前准备

清洗焊盘，然后在焊盘上涂上助焊剂。

(2) 固定贴片元件

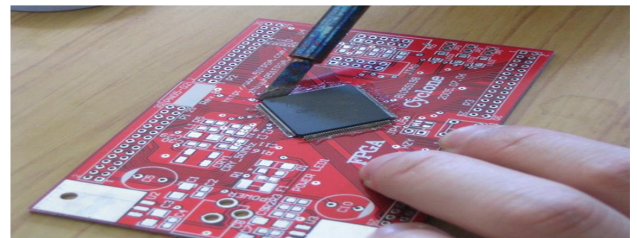
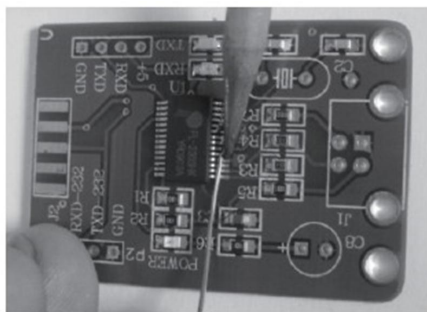


对管脚少的元件进行单脚固定焊接



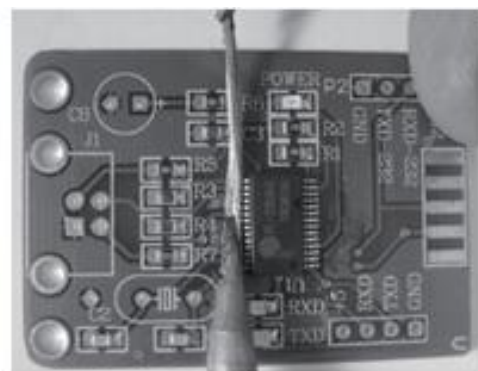
对管脚多的元件进行对角线定位焊接

(3) 焊接剩下的管脚



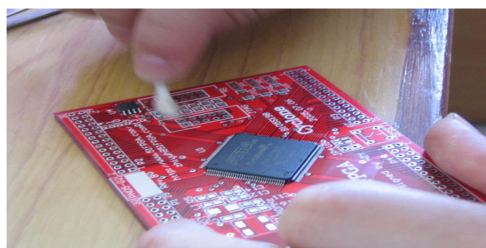
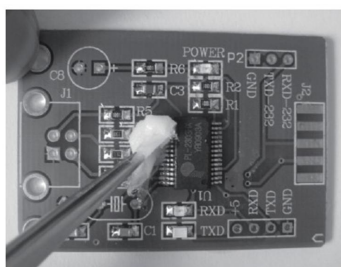
对管脚较多的贴片芯片进行拖焊或拉焊

(4) 清除多余焊锡



用自制的吸锡带吸去芯片管脚上多余的焊锡

(5) 酒精清洗印制电路板



3. 用热风枪吹焊贴片元件

(1) 吹焊小贴片元件的方法

吹焊小贴片元件一般采用小嘴喷头，热风枪的温度调至 2~3 档，风速调至 1~2 档。待温度和气流稳定后，便可用手指钳夹住小贴片元件，使热风枪的喷头离欲拆卸的元件 2~3cm，并保持垂直，在元件的上方均匀加热，待元件周围的焊锡熔化后，用手指钳将其取下。如果焊接小元件，要将元件放正，若焊点上的锡不足，可用烙铁在焊点上加注适量的焊锡，焊接方法与拆卸方法一样，只要注意温度与气流方向即可。

(2) 吹焊贴片集成电路的方法

先在芯片的表面涂上适量的助焊剂，吹焊时可采用大嘴喷头，热风枪的温度可调至 3~4 档，风量可调至 2~3 档，风枪的喷头离芯片 2.5cm 左右为宜。吹焊时应在芯片上方均匀加热，直到芯片底部的锡珠完全熔解，此时应用手指钳将整个芯片取下。

六、焊接后续工作及印制电路板清洗

1. 焊接后续工作规范

(1) 手工焊完后，先检查一遍所焊元器件有无错误，有无焊接质量缺陷，确认无误后将已焊接的线路板或部件转入下道工序的生产。

(2) 将未用完的材料或元器件分类放回原位，将桌面上残余的锡渣或杂物扫入指定的周转盒中；将工具归位放好；保持台面整洁。

(3) 关掉电源，按照电烙铁使用要求放好电烙铁，并做好防氧化保护工作。

(4) 工作人员应先洗净手后才能喝水或吃饭，以防锡珠对人体的危害。

2. 印制电路板清洗

第一步：用牙刷将焊接完成的印制电路板上的焊锡渣、松香、灰尘等污渍去除。在清洗过程中，操作员只允许用手套拿住印制电路板的两侧，在清洗剂未干透之前，不要用手触摸印制电路板，以出现手指纹。

第二步：在首次清理后，使用防静电牙刷进行二次清理。操作时需在清洗台上进行操作，印制电路板呈 30° 至 45° 放置，清洗时要求方向一致，自上而下地进行操作，且等清洗剂完全挥发后再放回存放区。

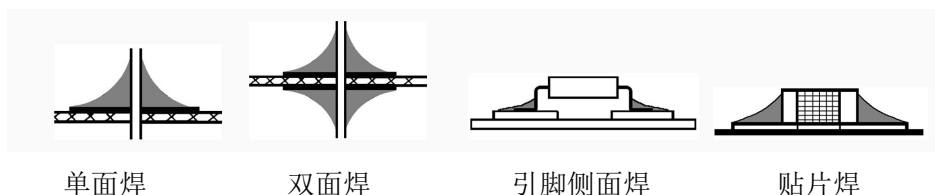
七、焊接质量检查

1. 合格焊点的标准

(1) 焊点接触良好，无虚焊

(2) 焊点要有足够的机械强度

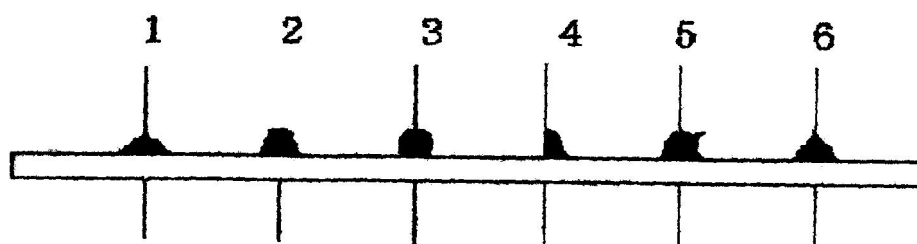
(3) 焊点表面整齐、美观



2. 焊点质量检查

- (1) 是否有错焊、漏焊、虚焊。
- (2) 有没有连焊、焊点是否有拉尖现象。
- (3) 焊盘有没有脱落、焊点有没有裂纹。
- (4) 焊点外形润湿应良好，焊点表面是不是光亮、圆润。
- (5) 焊点周围是无有残留的焊剂。
- (6) 焊接部位有无热损伤和机械损伤现象。

焊接前先要将焊接物和焊接处清洁，然后在焊接处镀上锡，才能焊得快，焊得牢，不虚焊。在焊接时，要注意焊接时间，过短会焊不牢，过长会烫坏元器件。焊点要光滑透亮，如果焊点成豆腐渣形，则说明焊点温度不适当。以下举例一些正确和错误的焊点：



焊点的不同形态

- (1) 正确焊点，全焊成为光滑小山丘。
2. 不正确焊点，元件线没有出头。
3. 不正确焊点，焊锡多、中间空，虚焊。
4. 不正确焊点，半焊，振动易脱焊。
5. 不正确焊点，烙铁从边缘撤走时带出一个小尖峰。
6. 正确焊点，烙铁头从元件引脚撤走。

正确焊接的顺序是：

- (1) 先将烙铁头放在两个焊接面处加热；
- (2) 将焊锡丝顺烙铁头方向熔化到焊接面；

当焊锡融化量适当后，抽走焊锡丝；

待焊接面的焊锡均匀熔化后，再抽走烙铁头。

在焊接完毕后，用斜口钳将剩余管脚剪去，便完成焊接工作。

八、拆焊方法

1. 拆卸引脚少的元器件

对于电阻、电容、晶体管等引脚不多的元器件，可使用电烙铁进行拆焊。方法是一边用电烙铁依次熔化各焊点的焊锡，一边用镊子或尖嘴钳夹住元器件或引脚，稍微用力拉拔即可拆下。拆焊时要掌握好时间和用力，时间长了容易烫坏元器件或者使印制电路板焊盘翘起、剥离；拉拔力量过猛容易拉断元器件引脚。特别注意在一个焊点上不要反复拆焊，否则很容易使焊盘脱落，造成印制电路板损坏。拆焊后要对焊点的位置进行清理，检查是否因拆焊造成电路短路或开路。

2. 拆卸引脚多的元器件

采用专用烙铁头或拆焊热风枪等专用工具，同时加热各个引脚，等到各引脚的焊锡全部熔化后，用镊子或尖嘴钳夹住元器件，轻轻用力拉拔即可拆下。对表面安装元件用热风枪拆焊更方便。

项目	考核内容	配分	评价标准	评分记录
焊接用具的准备	1. 检查电烙铁及烙铁头上锡 2. 阅读电路原理图和印制电路板装配图，按材料清单检查元器件型号及数量	10 10	1. 能对电烙铁进行安全检查，会给烙铁头正确上锡 2. 能看懂电路原理图和印制电路板装配图，能按材料清单检查元器件	
元器件加工成型、插装与焊接	1. 元器件引脚加工成型 2. 元器件插装 3. 五步焊接法	10 10 10	1. 引脚成型符合规范，错一个扣2分 2. 元器件插装符合规范，错一个扣2分 3. 能用五步焊接法正确焊接，操作不规范扣5分，不合格焊点，每个扣1分	
焊点检查	判断焊接质量	10	能正确判断焊点质量，误判、错判，每个扣2分	
拆焊	1. 根据元件类型选择拆焊工具 2. 实用工具进行常见元件的拆焊	10 10	1. 工具选择不当，每件扣2分 2. 操作错误，扣5分；元件、焊盘损坏，每个扣1分	
贴片焊接	1. 用电烙铁拆焊和焊接贴片元件 2. 用热风枪拆焊和焊接贴片元件	10 10	1. 操作错误，扣5分 2. 元件、焊盘损坏，每个扣1分 3. 焊点不合格，每个扣1分	

。

第三章 基于单片机的编程实验

§ 3.1 跑马灯

一 实验目的

学习编程首先要实现的最简单控制就是 CPU 的 IO 口控制了, LED 的控制是最简单的 IO 口控制, 这里通过控制发光二极管(LED)实现跑马灯的效果

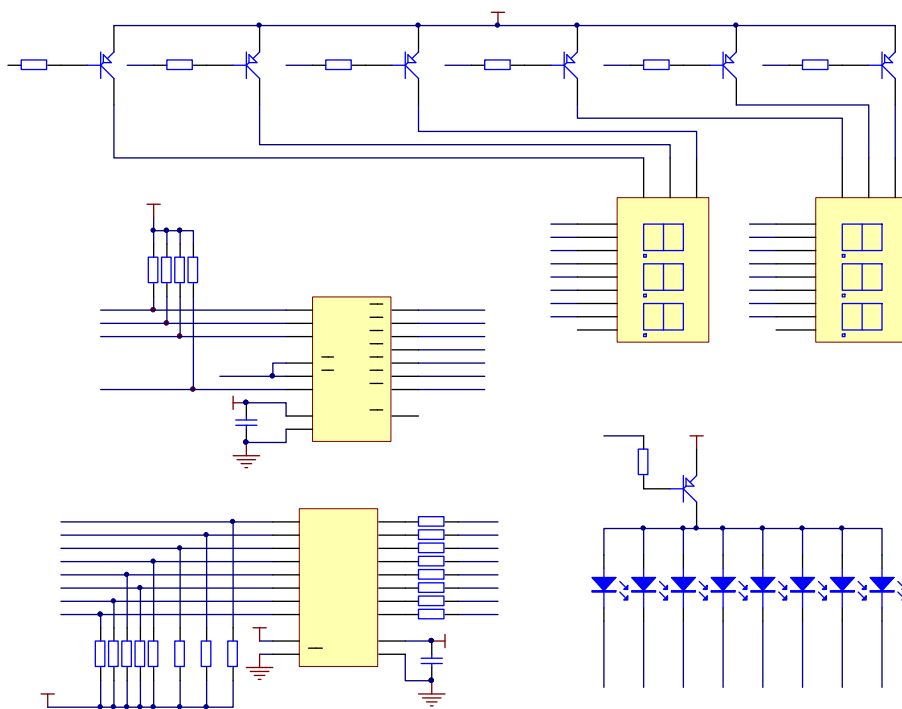
二 实验设备

window 10 操作系统电脑一台;

51 单片机实验板一套;

三 硬件原理

硬件原理图如下:



(1) 74HC245 原理

74HC245 是三态输出的八组总线控制器,其工作功能表如下: 74HC245 的 19 脚为芯片的使能端,低电平有效; 1 脚为输出方向控制端, 1 脚接 VCC,则数据从 A 端输入, B 端输出, 1 脚接 GND,则数据从 B 端输入, A 端输出。

其工作功能表如下:

功能表:

Enable \overline{G}	Direction Control DIR	Operation
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

(2) 74HC138 原理

74HC138 为 3 线-8 线译码器，可将地址端（A、B、C）的二进制编码在一个对应的输出端以低电平译出。其功能表如下：

Inputs					Outputs							
Enable		Select										
G1	$\overline{G2}^*$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

$$/G2^*=G2A+G2B$$

H=高电平

L=低电平

X=任意

G1 为 6 脚，接 P13(ADDR3),G2A(E1)和 G2B(E2)同 P14 连接，A(A0),B(A1),C(A2)分别和 P10(ADDR0)，P11(ADDR1)，P12(ADDR2)连接。

(3) 整体控制原理

74HC245 的数据从 CPU 的 P0 口输入到 A 口，B 口输出数据到 LED 的负极端，所以，控制时，直接给 P0 口赋值即可。

74HC138 的控制需要先是能芯片，也就是给 P14 和 ADDR3 赋值：P14=0;ADDR3=1。然后地址端使 Y6 输出低电平，即 LED6=0，这时三极管 8550 导通，使 LED 正端全部加了 VCC。接着通过 P0 口给相应的 LED 负端高低电平即可实现 LED 的亮灭。

四 软件实现

1 点亮一个 LED

```
/*
*****

标题： 点亮一个 LED

效果： LED 灯被点亮

*****

*/

#include<reg52.h>

sbit ADDR0 = P1^0;      //74HC138:A0
sbit ADDR1 = P1^1;      //74HC138:A1
sbit ADDR2 = P1^2;      //74HC138:A2

sbit ADDR3 = P1^3;      //74HC138:E3 高电平有效
sbit EN = P1^4;          //74HC138:E1,E2 低电平有效

sbit LED0 = P0^0;        //LED 发光二极管

void main()
{
    ADDR3 = 1;
    EN = 0;               //选通 74HC138

    ADDR0 = 0;
    ADDR1 = 1;
    ADDR2 = 1;            //74HC138: Y6 =0, Y0-Y5, Y7 =1
    while(1)
    {
        LED0 = 0;        // 点亮 LED
    }
}
```

2 LED 灯闪烁

```
/*
*****
```

*

标题：LED 灯闪烁效

果： LED 不断亮灭

*/

#include<reg52.h>

#include

sbit ADDR0 = P1^0; //74HC138:A0

sbit ADDR1 = P1^1; //74HC138:A1

sbit ADDR2 = P1^2; //74HC138:A2

sbit ADDR3 = P1^3; //74HC138:E3 高电平有效

sbit EN = P1^4; //74HC138:E1,E2 低电平有效

sbit LED0 = P0^0; //LED 发光二极管

/******

* 函数名称：delayms()

函数功能：实现延时大概延时 1ms

输入参数：t:0~65535

输出参数：无

*** / void delayms(unsigned int t)

{

unsigned char i; //定义延时时间变量

unsigned int j; for(i=t;i>0;i--)

for(j=110;j>0;j--);

}

```

void main()
{
    ADDR3 = 1;

    EN = 0;           //选通 74HC138

    ADDR0 = 0;

    ADDR1 = 1;
    ADDR2 = 1;        //74HC138: Y6 =0,   Y0-Y5,
                      Y7 =1

    while(1)
    {
        LED0 = 0;     //点亮 LED

        delayms(1000);

        LED0 = 1;     //关闭 LED

        delayms(1000);

    }
}

```

3 跑马灯

```

/*****
*
* 标题： 跑马灯
* 效果： 8 个 LED 灯循环点亮
*****/

*/

#include<reg52.h>

sbit ADDR0 = P1^0;    //74HC138:A0
sbit ADDR1 = P1^1;    //74HC138:A1
sbit ADDR2 = P1^2;    //74HC138:A2

sbit ADDR3 = P1^3;    //74HC138:E3 高电平有效
sbit EN = P1^4;       //74HC138:E1,E2 低电平有

```

效

```
/******  
* 函数名称: delayms()  
    函数功能: 实现延时大概延时 1ms  
    输入参数: t:0~65535  
    输出参数: 无  
*****  
***/ void delayms(unsigned int t)  
{  
    unsigned char i;      //定义延时时间变量  
    unsigned int j; for(i=t;i>0;i--)  
        for(j=110;j>0;j--);  
}  
  
void main()  
{  
    unsigned char i;      //定义一个变量  
    ADDR3 = 1;  
    EN = 0;               //选通 74HC138  
    ADDR0 = 0;  
    ADDR1 = 1;  
    ADDR2 = 1;            //74HC138: Y6 =0, Y0-Y5, Y7 =1  
    P0 = 0xff;            //先关闭所有的 LED  
    while(1)  
    { for(i=0;i  
        <8;i++)  
        {  
            P0 = P0 & ~(0x1<<i);    //右移动一个一个点亮 LED  
            delayms(1000);  
        }  
    }  
}
```

```

    }
    P0 = 0xff;          // 关闭所有
                        的 LED
    delayms(500);
    P0 = 0x00;          // 打开所有
                        的 LED
    delayms(500);
    P0 = 0xff;          // 关闭所
                        有的 LED

    delayms(500);

    for(i=8;i>0;i--)
    {
        P0 = P0 & ~(0x1<<i);    //左移动一个一个点亮 LED

        delayms(1000);
    }

    P0 = 0xff;          //关闭所有的 LED

    delayms(500);
}
}

```

五 测试结果

硬件分别连接好 P3, P4, P5, P6 的 1 和 2 脚, 然后烧写程序。实验中列了三个控制 LED 的程序, 第一个是点亮一个 LED 灯; 第二个是一个 LED 灯不断的亮灭; 第三个是从左到右一个一个点亮 LED, 然后关闭所有 LED 显示, 再打开全部 LED 显示, 接着关闭, 然后在从右到左一个一个点亮 LED, 最后熄灭全部 LED, 如此不断反复循环。

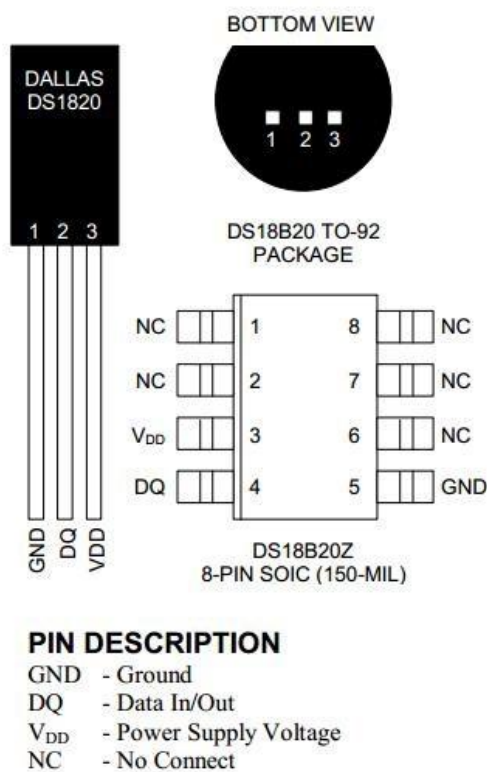
§ 3.2 温度显示

一 实验目的

DS18B20 是 DALLAS 公司生产的单总线式数字温度传感器, 它具有微型化、低功耗、高性能、搞干扰能力强、易配处理器等优点, 特别适用于构成多点温度测控系统, 可直接将温度转化成串行数字信号给单片机处理。可搭配数码管进行显示。

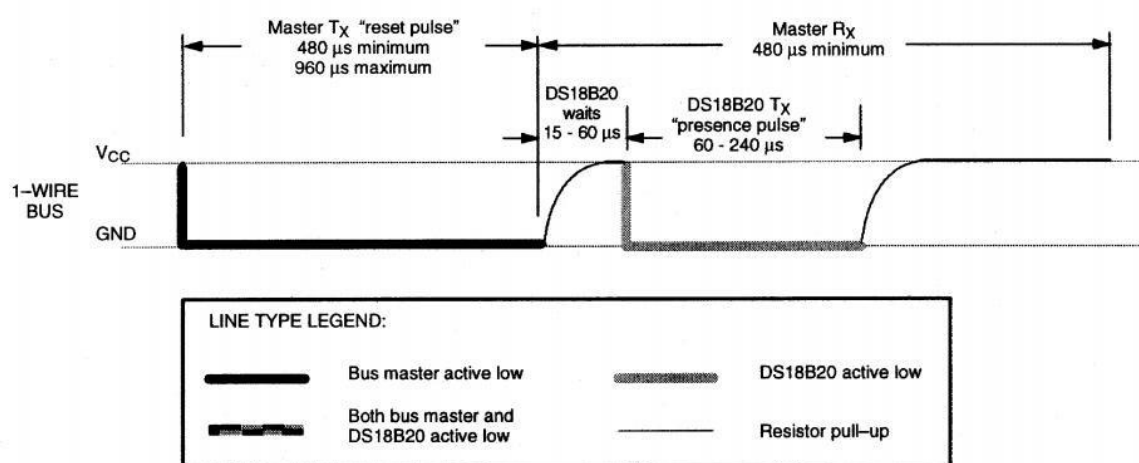
DS18B20 是 DALLAS 公司生产的单总线(1-Wire)式数字温度传感器, 它具有微型化、低

功耗、高性能、搞干扰能力强、易配处理器等优点，特别适用于构成多点温度测控系统，可直接将温度转化成串行数字信号（提供 9 位二进制数字）给单片机处理，且在同一总线上可以挂接多个传感器芯片。它具有 3 引脚 TO-92 小体积封装形式，温度测量范围为 $-55^{\circ}\text{C}\sim+125^{\circ}\text{C}$ ，可编程为 9 位~12 位 A/D 转换精度，测温分辨率可达 0.0625°C ，被测温度用符号扩展的 16 位数字量方式串行输出，其工作电源既可在远端输入，也可采用寄生电源方式产生，多个 DS18B20 可以并联到 3 根或 2 根线上，CPU 另需一根端口线就能与多个 DS18B20 通信，占用微处理器的端口数少，可节省大量的引线 and 逻辑电路。以上特点使 DS18B20 非常适用于远距离多点温度检测系统中。DS18B20 的外形及引脚封装：



DS18B20 的工作协议过程

(1) 初始化和 I2C 的寻址类似，1-Wire 总线开始也需要检测这条总线上是否存在 DS18B20 这个器件。如果这条总线上存在 DS18B20，总线会根据时序要求返回一个低电平脉冲，如果不存在的话，也就不会返回脉冲，即总线保持为高电平，所以习惯上称之为检测存在脉冲。此外，获取存在脉冲不仅仅是检测是否存在 DS18B20，还要通过这个脉冲过程通知 DS18B20 准备好，单片机要对它进行操作了，如下图：



图中，实粗线是我们的单片机 IO 口拉低这个引脚，虚粗线是 DS18B20 拉低这个引脚，细线是单片机和 DS18B20 释放总线后，依靠上拉电阻的作用把 IO 口引脚拉上去，51 单片机释放总线就是给高电平。存在脉冲检测过程，首先单片机要拉低这个引脚，持续大概 480us 到 960us 之间的时间即可，然后，单片机释放总线，就是给高电平，DS18B20 等待大概 15 到 60us 后，会主动拉低这个引脚大概是 60 到 240us，而后 DS18B20 会主动释放总线，这样 IO 口会被上拉电阻自动拉高。

(2)ROM 操作指令

I₂C 总线上可以挂多个器件，通过不同的器件地址来访问不同的器件。同样，1-Wire 总线也可以挂多个器件，但是它只有一条线，如何区分不同的器件呢？

在每个 DS18B20 内部都有一个唯一的 64 位长的序列号，这个序列号值就存在 DS18B20 内部的 ROM 中。开始的 8 位是产品类型编码（DS18B20 是 0x10），接着的 48 位是每个器件唯一的序号，最后的 8 位是 CRC 校验码。DS18B20 可以引出去很长的线，最长可以到几十米，测不同位置的温度。单片机可以通过和 DS18B20 之间的通信，获取每个传感器所采集到的温度信息，也可以同时给所有的 DS18B20 发送一些指令。这些指令相对来说比较复杂，而且应用很少，所以这里不介绍了。我们这里只讲一条总线上只接一个器件的指令和程序。Skip ROM（跳过 ROM）：0xCC。当总线上只有一个器件的时候，可以跳过 ROM，不进行 ROM 检测。

(3)RAM 存储器操作指令

Read Scratchpad（读暂存寄存器）：0xBE

这里要注意的是，DS18B20 的温度数据是 2 个字节，我们读取数据的时候，先读取到的是低字节的低位，读完了第一个字节后，再读高字节的低位，直到两个字节全部读取完毕。

Convert Temperature（启动温度转换）：0x44

当我们发送一个启动温度转换的指令后，DS18B20 开始进行转换。从转换开始到获取温度，DS18B20 是需要时间的，而这个时间长短取决于 DS18B20 的精度。前边说 DS18B20 最高可以用 12 位来存储温度，但是也可以用 11 位，10 位和 9 位一共四种格式。位数越高，精度越高，9 位模式最低位变化 1 个数字温度变化 0.5 度，同时转换速度也要快一些，如下图

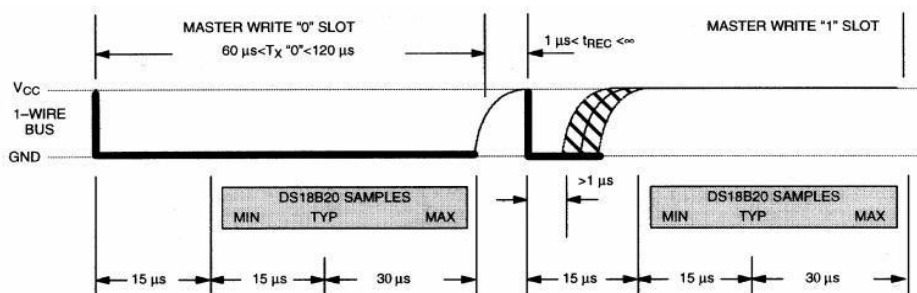
DS18B20 转换时间:

R1	R0	Thermometer Resolution	Max Conversion Time
0	0	9-bit	93.75ms
0	1	10-bit	187.5ms
1	0	11-bit	375ms
1	1	12-bit	750ms

其中寄存器 R1 和 R0 决定了转换的位数，出厂默认值就 11，也就是 12 位表示温度，最大的转换时间是 750ms。当启动转换后，至少要再等 750ms 之后才能读取温度，否则读到的温度有可能是错误的值。这就是为什么很多读 DS18B20 的时候，第一次读出来的是 85 度，这个值要么是没有启动转换，要么是启动转换了，但还没有等待一次转换彻底完成，读到的是一个错误的的数据。

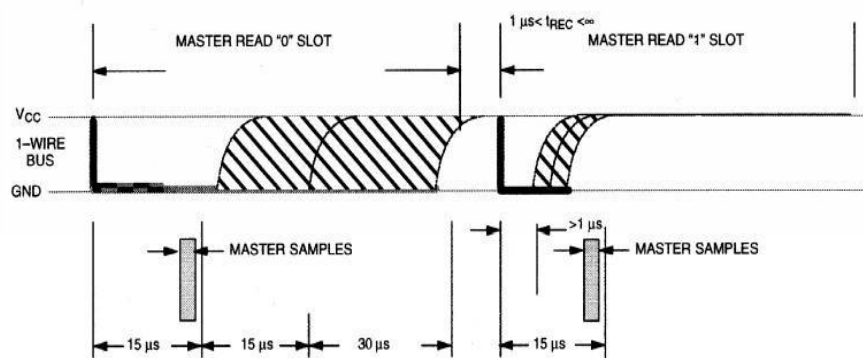
(4)DS18B20 的位读写时序

DS18B20 的位写入时序如下图:



当要给 DS18B20 写入 0 的时候，单片机直接将引脚拉低，持续时间大于 60us 小于 120us 就可以了。图上显示的意思是，单片机先拉低 15us 之后，DS18B20 会在从 15us 到 60us 之间的时间来读取这一位，DS18B20 最早会在 15us 的时刻读取，典型值是在 30us 的时刻读取，最多不会超过 60us，DS18B20 必然读取完毕，所以持续时间超过 60us 即可。当要给 DS18B20 写入 1 的时候，单片机先将这个引脚拉低，拉低时间大于 1us，然后马上释放总线，即拉高引脚，并且持续时间也要大于 60us。和写 0 类似的是，DS18B20 会在 15us 到 60us 之间来读取这个 1。可以看出来，DS18B20 的时序比较严格，写的过程中最好不要有中断打断，但是在两个“位”之间的间隔，是大于 1 小于无穷的，那在这个时间段，我们是可开中断来处理其它程序的。

DS18B20 的位读时序如下图:



当要读取 DS18B20 的数据的时候，我们的单片机首先要拉低这个引脚，并且至少保持 1us 的时间，然后释放引脚，释放完毕后要尽快读取。从拉低这个引脚到读取引脚状态，不能超过 15us。大家从上图中可以看出来，主机采样时间，也就是 MASTER SAMPLES，是在 15us 之内必须完成的。

(5) 数据处理

DS18B20 的高速暂存存储器由 9 个字节组成，其分配如下图所示：



当温度转换命令发布后经转换所得的温度值以二字节补码形式存放在高速暂存存储器的第 0 和第 1 个字节。单片机可通过单线接口读取该数据，读取时低位在前，高位在后。

温度℃	数据输出（二进制）	数据输出（十六进制）
+125	00000000 11111010	00FA
+25	00000000 00110010	0032
+1/2	00000000 00000001	0001
0	00000000 00000000	0000
-1/2	11111111 11111111	FFFF
-25	11111111 11001110	FFCE
-55	11111111 10010010	FF92

DS18B20 温度数据表

上表是 DS18B20 温度采集转化后得到的 12 位数据，存储在 DS18B20 的两个 8 比特的 RAM 中，二进制中的前面 5 位是符号位，如果测得的温度大于或等于 0，这 5 位为 0，只要将测得的数值乘以 0.0625 即可得到实际温度；如果温度小于 0，这 5 位为 1，测得的数值需要取反加 1 再乘以 0.0625 即可得到实际温度。

温度转换计算方法实例：

当 DS18B20 采集到 +125℃ 的实际温度后，输出为 07D0H，则：实际温度
 $= 07D0H \times 0.0625 = 2000 \times 0.0625 = 1250C$ 。

当 DS18B20 采集到-55℃的实际温度后,输出为 FC90H,则应先将 11 位数据取反加 1 得 370H (符号位不变,也不作为计算),则:实际温度=370H×0.0625=880×0.0625=550C。

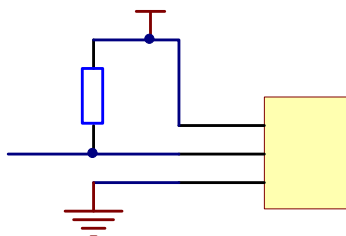
二 实验设备

window 10 操作系统电脑一台;

51 单片机实验板一套;

三 硬件原理

板子的硬件原理图如下:



四 软件实现

/*****

标题： 数码管温度显示

效果： 显示 ds18b20 的温度值

*****/

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

```
#include"ds18b20.h"
```

```
sbit ADDR0 = P1^0;    //74HC138:A0
```

```
sbit ADDR1 = P1^1; //74HC138:A1
```

```
sbit ADDR2 = P1^2; //74HC138:A2
```

```
sbit ADDR3 = P1^3; //74HC138:E3 高电平有效
```

```
sbit EN = P1^4;           //74HC138:E1,E2 低电平有效
```

```
unsigned char code display_code[] = {    //使用关键字 code 存储到程序空间 Flash
中, 节省单片机 RAM 的使用量
```

[illegible]

```

0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90, 0x88, 0x83, 0xa7, 0xa1, 0x86, 0x8e
} :

```

```

/*****
函数名称: delays()
函数功能: 实现延时大概延时 1ms
输入参数: t:0~65535
输出参数: 无
*****/
void delays(unsigned int t)
{
    unsigned char i;    //定义延时时间变量
    unsigned int j;
    for(i=t;i>0;i--)
        for(j=110;j>0;j--);
}

void main()
{
    unsigned char bai,shi,ge;
    unsigned char i=0;    //定义一个临时变量
    unsigned int temp;    //温度变量

    ADDR3 = 1;
    EN = 0;    //选通 74HC138

    P0 = 0xff;    //先关闭数码管显示
    while(1)
    {
        temp=Get_Tmp();

        bai = temp/100;    //百位
        shi = temp%100/10;    //十位
        ge = temp%10;    //个位

        switch(i)
        {
            case 0:    //第一个数码管
                ADDR0 = 0;
                ADDR1 = 0;
                ADDR2 = 0;    //74HC138: Y0 =0, Y1-Y7 =1
                i++;
                P0 = display_code[ge];
                break;
            case 1:    //第二个数码管
                ADDR0 = 1;
                ADDR1 = 0;

```

```

        ADDR2 = 0;           //74HC138: Y1 =0,  Y0,Y2-Y7 =1
        i++;
        P0 = display_code[shi];
        break;
    case 2:                   //第三个数码管
        ADDR0 = 0;
        ADDR1 = 1;
        ADDR2 = 0;           //74HC138: Y2 =0,  Y0,Y1,Y3-Y7 =1
        i=0;
        P0 = display_code[bai];
        break;
    }
    delayms(1); //delayms(500);
    P0 = 0xff;               //关闭数码管显示，也就是消影;这个语句非常重要，如果
    不加，会导致数码管显示乱码
}
}

```

五 测试结果

烧写编译好的镜像到板子中，可以从数码管显示出当前的环境温度。

§ 3.3 矩阵键盘

一 实验目的

按键在日常生活中随处可见:固定电话，手机，键盘.....,按键的工作原理一般是按键按下，然后执行相应的操作。板子中使用了 **4x4** 矩阵键盘的形式。每个按键对应一个值，按键按下时，数码管对应显示 0-F

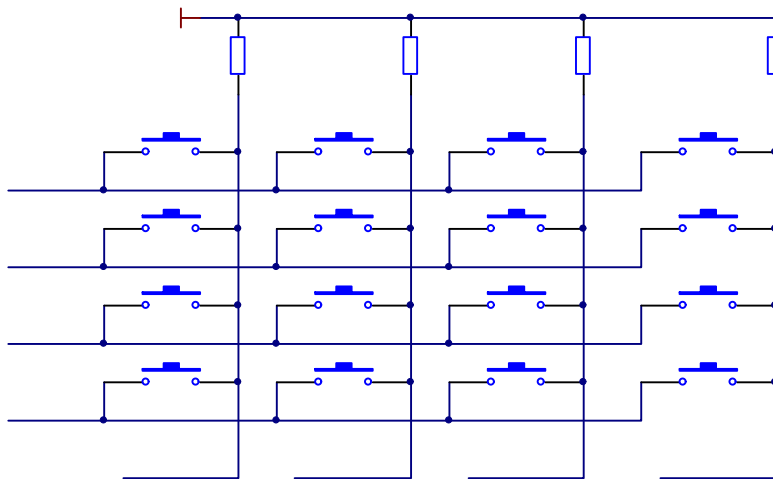
二 实验设备

window 10 操作系统电脑一台;

51 单片机实验板一套;

三 硬件原理

按键的硬件原理图如下:



原理图中，P24-P27 接了 4.7K 的上拉电阻，此时，如果设置 P20-P23 中的一个位为低电平，则分别检查 P24-P27 的状态，比如设置 P20=0，然后检测 S1-S4(P24-P27)的状态，如果 S1 按键按下，则检测到 P24 为低电平，代表 S1 按键按下，其他的 S2，S3 和 S4 的原理是一样的，这样的检测按键的工作是独立按键方式。矩阵键盘的方式，是在独立按键的基础上，分别设置 P20-P23 为低电平，然后检测 P2-P27 的状态来确认按键按下。

四 软件实现

```

/*****
标题： 矩阵键盘数码管显示 0-F
效果： 每个按键对应一个值，按键按下时，数码管对应显示 0-F
*****/
#include<reg52.h>

sbit ADDR0 = P1^0;    //74HC138:A0
sbit ADDR1 = P1^1;    //74HC138:A1
sbit ADDR2 = P1^2;    //74HC138:A2

sbit ADDR3 = P1^3;    //74HC138:E3 高电平有效
sbit EN = P1^4;       //74HC138:E1,E2 低电平有效

unsigned char code display_code[] = { //使用关键字 code 存储到程序空间 Flash
中,节省单片机 RAM 的使用量
/*0   1   2   3   4   5   6   7   8   9   A   b   c   d   E
F*/
0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90, 0x88, 0x83, 0xa7, 0xa1, 0x86, 0x
8e
};

unsigned int num=0;

```

```

unsigned char i=0; //定义一个临时变量

unsigned char key_value=0;//计数值变量

#define KeyPort P2

/*****
函数名称: delaysms()
函数功能: 实现延时大概延时 1ms
输入参数: t:0~65535
输出参数: 无
*****/
void delaysms(unsigned int t)
{
    unsigned char i;    //定义延时时间变量
    unsigned int j;
    for(i=t;i>0;i--)
        for(j=110;j>0;j--);
}

/*****
函数名称: display()
函数功能: 实现数码管显示
输入参数: 无
输出参数: 无
*****/
void display()
{
    switch(i)
    {
        case 0:                //第一个数码管
            ADDR0 = 0;
            ADDR1 = 0;
            ADDR2 = 0;          //74HC138: Y0 =0,  Y1-Y7 =1
            i=0;
            P0 = display_code[key_value];
            break;
    }
    delaysms(1); //delaysms(500);
    P0 = 0xff;      //关闭数码管显示, 也就是消影;这个语句非常重要, 如果不加, 会导致数码管显示乱码
}

```

```

/*****
函数名称: keyscan()
函数功能: 实现按键按下检测键码
输入参数: 无
输出参数: 无
*****/
void keyscan()
{
    unsigned char temp;
    //开始检测第一行
    KeyPort = 0xfe;
    temp = KeyPort;
    temp = temp & 0xf0;
    if (temp != 0xf0)                //按键按下
    {
        delays(10);                //延时消抖动
        temp = KeyPort;            //再次赋值
        if (temp != 0xf0)            //确认按键已经按下
        {
            temp = KeyPort;        //把按键按下的值付给 temp
            switch(temp)            //判断 temp 的值
            {
                case 0xee: key_value = 1; break;
                case 0xde: key_value = 2; break;
                case 0xbe: key_value = 3; break;
                case 0x7e: key_value = 15; break;
            }
        }
    }
    //结束检测第一行

    //开始检测第二行
    KeyPort = 0xfd;
    temp = KeyPort;
    temp = temp & 0xf0;
    if (temp != 0xf0)                //按键按下
    {
        delays(10);                //延时消抖动
        temp = KeyPort;            //再次赋值
        if (temp != 0xf0)            //确认按键已经按下
        {
            temp = KeyPort;        //把按键按下的值付给 temp
            switch(temp)            //判断 temp 的值
            {

```

```

        case 0xed: key_value = 4; break;
        case 0xdd: key_value = 5; break;
        case 0xbd: key_value = 6; break;
        case 0x7d: key_value = 14; break;
    }
}
}
//结束检测第二行

//开始检测第三行
KeyPort = 0xfb;
temp = KeyPort;
temp = temp&0xf0;
if(temp!=0xf0)                //按键按下
{
    delays(10);                //延时消抖动
    temp = KeyPort;            //再次赋值
    if(temp!=0xf0)            //确认按键已经按下
    {
        temp = KeyPort;        //把按键按下的值付给 temp
        switch(temp)           //判断 temp 的值
        {
            case 0xeb: key_value = 7; break;
            case 0xdb: key_value = 8; break;
            case 0xbb: key_value = 9; break;
            case 0x7b: key_value = 13; break;
        }
    }
}
//结束检测第三行

//开始检测第四行
KeyPort = 0xf7;
temp = KeyPort;
temp = temp&0xf0;
if(temp!=0xf0)                //按键按下
{
    delays(10);                //延时消抖动
    temp = KeyPort;            //再次赋值
    if(temp!=0xf0)            //确认按键已经按下
    {
        temp = KeyPort;        //把按键按下的值付给 temp
        switch(temp)           //判断 temp 的值
        {

```

```

        case 0xe7: key_value = 10; break;
        case 0xd7: key_value = 11; break;
        case 0xb7: key_value = 12; break;
        case 0x77: key_value = 0; break;
    }
}
}
//结束检测第四行
}

void main()
{
    ADDR3 = 1;
    EN = 0;           //选通 74HC138

    ADDR0 = 0;
    ADDR1 = 1;
    ADDR2 = 1;        //74HC138 Y6 = 0; Y1~Y5, Y7 = 1

    P0 = 0xff;        //程序运行前，关闭数码管显示

    while(1)
    {
        keyscan();
        display();
    }
}

```

五 测试结果

独立按键的测试结果是，按键启动或者停止定时器，定时器开启时，数码管数值自动加 1，停止时，数码管数值保持不变；数码管按下加键时，数码管数值加 1，按下减按键时，数码管减 1。矩阵键盘的测试结果是，按键按下时，数码管对应显示相应的数字，和 0-F 分别对应。

§ 3.4 收音机

一 实验目的

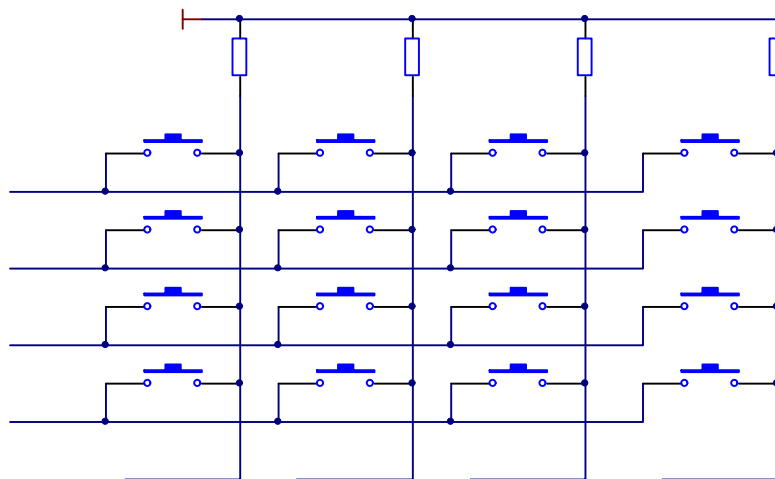
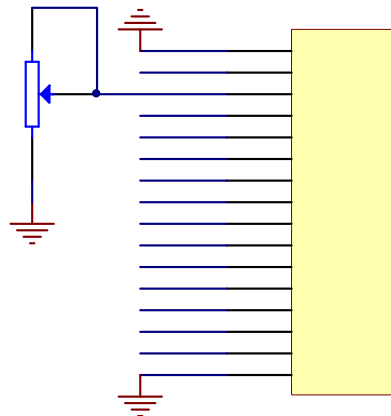
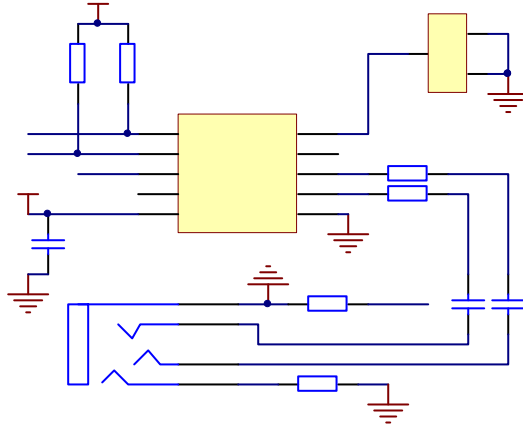
采用 AT89S52 微控制器对收音机进行控制，由 1602 液晶模块作显示，用 TEA5767 的高灵敏度，低电压和低功耗的全集成的优点作收音模块芯片，通过合理设计，制作一款完善的数字调谐收音机。

二 实验设备

window 10 操作系统电脑一台；

51 单片机实验板一套；

三 硬件原理



收音机的调频可以通过按键 s1、s2、s3、s4 来实现，手动增加或减少时，按一下调一

个频率，自动增加或减少是按着自动往上调或往下调。调频的结果会在 LCD1602 上显示出来。

四 软件设计

```
/******  
*  
标题：    TEA5767+LCD1602 显示  
效果：    TEA5767+LCD1602 收音机  
*****  
*/  
  
#include<reg52.h>  
#include"delay.h"  
#include"iic.h"  
#include"tea5767.h"  
#include"lcd1602.h"  
#include"ds18b20.h"  
#include"ds1302.h"  
  
//收音调整端口  
sbit s0=P2^0;  
sbit s1=P2^4;    //手动减小键  
sbit s2=P2^5;    //手动增加键  
sbit s3=P2^6;    //自动减小键  
sbit s4=P2^7;    //自动增加键  
  
extern unsigned long frequency;    //tea5767 的收音频率值，在 tea5767.c 中定义  
  
extern unsigned char second,minute,hour,week,day,month,year;    //DS1302 的秒、分、  
时、星期、日、月、年 在 ds1302.c 中定义  
  
unsigned code table[]={'.','C'};  
unsigned code table1[]={ '0','1','2','3','4','5','6','7','8','9'};  
  
/******  
*  
函数名称：shouyin_display(unsigned long sy)  
函数功能：收音显示函数  
输入参数：sy  
输出参数：无  
*****  
*/  
  
void shouyin_display(unsigned long sy)
```

```

{
    write_com(0x80+0x40+2);
    write_data('F');
    delayms(1);
    write_com(0x80+0x40+3);
    write_data('M');
    delayms(1);
    write_com(0x80+0x40+4);
    write_data(':');
    delayms(1);
    write_com(0x80+0x40+5);
    write_data(table1[sy/100000]);
    delayms(1);
    write_com(0x80+0x40+6);
    write_data(table1[sy%100000/10000]);
    delayms(1);
    write_com(0x80+0x40+7);
    write_data(table1[sy%100000%10000/1000]);
    delayms(1);
    write_com(0x80+0x40+8);
    write_data('.');
    delayms(1);

    write_com(0x80+0x40+9);
    write_data(table1[sy%100000%10000%1000/100]);
    delayms(1);

    write_com(0x80+0x40+10);
    write_data('M');
    delayms(1);
    write_com(0x80+0x40+11);
    write_data('H');
    delayms(1);
    write_com(0x80+0x40+12);
    write_data('z');
    delayms(1);
}

```

```

/*****

```

*

函数名称: shoukey()

函数功能: 收音按键函数

输入参数: 无

输出参数：无

*/

```
void shoukey()
{
    if(s1==0)
    {
        delayms(1);
        if(s4==0)
        {
            search(0);
            while(!s4);
        }
    }

    if(s2==0)
    {
        delayms(1);
        if(s5==0)
        {
            search(1);
            while(!s5);
        }
    }

    if(s3==0)
    {
        delayms(1);
        if(s6==0)
        {
            auto_search(0);
            while(!s6);
        }
    }

    if(s4==0)
    {
        delayms(1);
        if(s7==0)
        {
            auto_search(1);
            while(!s7);
        }
    }
}
```

```

    }

    /*****
*
    函数名称: Display_18b20(uint t)
    函数功能: 温度显示函数
    输入参数: 无
    输出参数: 无
    *****/
*/
void Display_18b20(uint t)    //显示程序
{
    uchar A1,A2,A3;
    uchar i;
    A1 = t/100;    //百位
    A2 = t%100/10;    //十位
    A3 = t%10;    //个位
    write_com(0x80+0x0a);
    write_data(table1[A1]);
    delayms(5);

    write_com(0x80+0x0b);
    write_data(table1[A2]);
    delayms(5);

    i=0;
    write_com(0x80+0x0c);
    write_data(table[0]);
    delayms(5);

    write_com(0x80+0x0d);
    write_data(table1[A3]);
    delayms(5);

    write_com(0x80+0x0e);
    write_data('^');
    delayms(5);

    i=1;
    write_com(0x80+0x0f);
    write_data(table[1]);
    delayms(5);
}

```

```

/*****
*
函数名称: Display_1302()
函数功能: ds1302 显示函数
输入参数: 无
输出参数: 无
*****/

*/
void Display_1302()    //显示程序
{
    write_com(0x80+0x00);
    write_data(table1[hour/10%10]);
    delayms(1);
    write_com(0x80+0x01);
    write_data(table1[hour%10]);
    delayms(1);

    write_com(0x80+0x02);
    write_data(':');
    delayms(1);

    write_com(0x80+0x03);
    write_data(table1[minute/10%10]);
    delayms(1);
    write_com(0x80+0x04);
    write_data(table1[minute%10]);
    delayms(1);

    write_com(0x80+0x05);
    write_data(':');
    delayms(1);

    write_com(0x80+0x06);
    write_data(table1[second/10%10]);
    delayms(1);
    write_com(0x80+0x07);
    write_data(table1[second%10]);
    delayms(1);
}

void main()
{
    s0=0;
    InitDS1302();

```

```
    lcd1602_init();
    iic_init();
    init_tea5767();

    while(1)
    {
        DS1302ReadTime();
        Display_1302();    //显示程序
        Display_18b20(Get_Tmp()); //温度显示程序
        shouyin_display(frequency); //89700
        shoukey();
    }
}
```

五 实验结果

由 LCD 液晶模块作显示，按键用作调频，用 TEA5767 的高灵敏度，低电压和低功耗的全集成的优点作收音模块芯片，通过合理软硬件设计，实现了一款完善的数字调谐收音机。