康耘电子硬件工程师培训教材
嵌入式高级班培训教材

硬件工程师
培训教材

西安康耘电子有限责任公司
Xi’an Canwin Electronic Co.,Ltd.

http://www.canwinelec.com

版权声明

本手册版权归西安康耘电子有限责任公司所有，未经康耘电子同意，任何单位和个人不得擅自抄录本手册或全部以任何形式用于商业目的，但可以自由传播。本手册所介绍相关软件版权均归相关公司所有，这里只供学习使用，若进行实际商业性开发使用请与相关公司联系购买正版软件。

本手册编制过程中个别电路及程序参考了相关资料，在手册中都给出了说明，感谢这些资料的提供者！

目　录

第一部分 扩充知识

第1章　常用电路元件

1.1 电阻、电容与二极管

1、电阻

在选择电阻器的阻值时，应根据设计电路时理论计算电阻值，在最靠近标称值系列中选用。普通电阻器（不包括精密电阻器）阻值标称系列值见下表，实际电阻器的阻值是表中的数值乘以 10^n（n 为整数）。

允许偏差（%）	阻 值（Ω）
±5%	1.0、1.1、1.2、1.3、1.5、1.6、1.8、2.0、2.2、2.4、2.7、3.0、3.3、3.6、3.9、4.3、4.7、5.1、5.6、6.2、6.8、7.5、8.2、9.1
±10%	1.0、1.2、1.5、1.8、2.2、2.7、3.3、3.9、4.7、5.6、6.8、8.2
±20%	1.0、1.5、2.2、3.3、4.7、6.8

电阻器额定功率标称系列值

电阻器类型	额定功率（W）
线绕电阻器	0.05、0.125、0.25、0.5、1、2、4、8、10、16、25、40、50 、75、100、150、250、500
非线绕电阻器	0.05、0.125、0.25、0.5、1、2、5、10、25、50、100

2、快速识别色环电阻

目前，国产或进口电视机、收录机广泛采用色环电阻，其优点是在装配、调试和修理过程中，不用拨动元件，即可在任意角度看清色环，读出阻值，使用很方便。以往杂志上都介绍过色环电阻识读法，按其方法读数时，要进行换算，较麻烦，这里介绍一种快速识别阻值的方法。

带有四个色环的其中第一、二环分别代表阻值的前两位数；第三环代表倍率；第四环代表误差。快速识别的关键在于根据第三环的颜色把阻值确定在某一数量级范围内，例如是几点几 K、还是几十几 K 的，再将前两环读出的数"代"进去，这样就可很快读出数来。

下面介绍掌握此方法的几个要点：

1） 熟记第一、二环每种颜色所代表的数。

可这样记忆：棕 1，红 2，橙 3，黄 4，绿 5，蓝 6，紫 7，灰 8，白 9，黑 0。这样连起来读，多复诵几遍便可记住。

记准记牢第三环颜色所代表的 阻值范围，这一点是快识的关键。具体是：

金色：几点几 Ω

黑色：几十几 Ω

棕色：几百几十 Ω

红色：几点几 kΩ

橙色：几十几 kΩ

黄色：几百几十 kΩ

绿色：几点几 MΩ

蓝色：几十几 MΩ

从数量级来看，大体上可把它们划分为三个大的等级，即：

金、黑、棕色是欧姆级的；

红橙、黄色是千欧级的；

绿、蓝色则是兆欧级的。

这样划分一下是为了便于记忆。

2）当第二环是黑色时，第三环颜色所代表的则是整数，即几，几十，几百 kΩ等，这是读数时的特殊情况，要注意。例如第三环是红色，则其阻值即是整几kΩ的。

3）记住第四环颜色所代表的误差，即：金色为 5%；银色为 10%；无色为 20%。

下面举例说明：

例 1：当四个色环依次是黄、橙、红、金色时，因第三环为红色、阻值范围是几点几kΩ的，按照黄、橙两色分别代表的数"4"和"3"代入,，则其读数为 43 kΩ。第环是金色表示误差为 5%。

例 2：当四个色环依次是棕、黑、橙、金色时，因第三环为橙色，第二环又是黑色，阻值应是整几十 kΩ的，按棕色代表的数"1"代入，读数为 10 kΩ。第四环是金色，其误差为 5%。

3、晶体二极管的种类

晶体二极管按其组成的材料分可为 : 锗二极管、硅二极管、砷化嫁二极管 (发光二极管) 。而按用途分可为 : 整流二极管、稳压二极管、开关二极管、发光二极管、检波二极管、变容二极管等。

常用晶体二极管的电路符号如下图所示。

(a) 一般二极管　(b) 稳压二极管　(c) 发光二极管　(d) 光电二极管

4、电容

1）单位： 电容的基本单位是：F （法），此外还有 μF（微法）、pF（皮法），另外还有一个用的比较少的单位，那就是：nF（），由于电容 F 的容量非常大，所以我们看到的一般都是 μF、nF、pF 的单位，而不是 F 的单位。 他们之间的具体换算如下： 1F＝1000000μF　　1μF=1000nF=1000000pF

2）电容的耐压 单位：V（伏特）

每一个电容都有它的耐压值，这是电容的重要参数之一。普通无极性电容的标称耐压值有：63V、100V、160V、250V、400V、600V、1000V 等，有极性电容的耐压值相对要比无

极性电容的耐压要低，一般的标称耐压值有：4V、6.3V、10V、16V、25V、35V、50V、63V、80V、100V、220V、400V 等。

3）电容的种类

电容的种类有很多，可以从原理上分为：无极性可变电容、无极性固定电容、有极性电容等，从材料上可以分为：CBB 电容（聚乙烯），涤纶电容、瓷片电容、云母电容、独石电容、电解电容、钽电容等。下面是各种电容的优缺点：

无感 CBB 电容

2 层聚丙乙烯塑料和 2 层金属箔交替夹杂然后捆绑而成。 无感，高频特性好，体积较小 ，不适合做大容量，价格比较高，耐热性能较差。

CBB 电容

2 层聚乙烯塑料和 2 层金属箔交替夹杂然后捆绑而成。 有感，其他同上。

瓷片电容

薄瓷片两面渡金属膜银而成。 体积小，耐压高，价格低，频率高（有一种是高频电容） 易碎！容量低

云母电容

云母片上镀两层金属薄膜 容易生产，技术含量低。 体积大，容量小，（几乎没有用了）

独石电容

体积比 CBB 更小，其他同 CBB，有感

电解电容

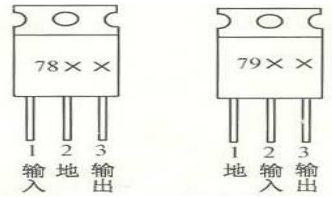
两片铝带和两层绝缘膜相互层叠，转捆后浸泡在电解液（含酸性的合成溶液）中。 容量大。 高频特性不好。

钽电容

用金属钽作为正极，在电解质外喷上金属作为负极。 稳定性好，容量大，高频特性好。 造价高。（一般用于关键地方）

5、集成三端稳压器

型 号	功 能	型 号	功 能
7805	+5V 1A 稳压器	7905	-5V 1A 稳压器
7806	+6V 1A 稳压器	7906	-6V 1A 稳压器
7808	+8V 1A 稳压器	7908	-8V 1A 稳压器
7809	+9V 1A 稳压器	7909	-9V 1A 稳压器
7812	+12V 1A 稳压器	7912	-12V 1A 稳压器
7815	+15V 1A 稳压器	7915	-15V 1A 稳压器
7818	+18V 1A 稳压器	7918	-18V 1A 稳压器
7824	+24V 1A 稳压器	7924	-24V 1A 稳压器



1.2 功率电子器件

1.2.1 功率电子器件及其应用要求

功率电子器件大量被应用于电源、伺服驱动、变频器、电机保护器等功率电子设备。这些设备都是自动化系统中必不可少的，因此，我们了解它们是必要的。

近年来，随着应用日益高速发展的需求，推动了功率电子器件的制造工艺的研究和发展，功率电子器件有了飞跃性的进步。器件的类型朝多元化发展，性能也越来越改善。大致来讲，功率器件的发展，体现在如下方面：

1．器件能够快速恢复，以满足越来越高的速度需要。以开关电源为例，采用双极型晶体管时，速度可以到几十千赫；使用 MOSFET 和 IGBT，可以到几百千赫；而采用了谐振技术的开关电源，则可以达到兆赫以上。

2．通态压降（正向压降）降低。这可以减少器件损耗，有利于提高速度，减小器件体积。

3．电流控制能力增大。电流能力的增大和速度的提高是一对矛盾，目前最大电流控制能力，特别是在电力设备方面，还没有器件能完全替代可控硅。

4．额定电压：耐压高。耐压和电流都是体现驱动能力的重要参数，特别对电力系统，这显得非常重要。

5．温度与功耗。这是一个综合性的参数，它制约了电流能力、开关速度等能力的提高。目前有两个方向解决这个问题，一是继续提高功率器件的品质，二是改进控制技术来降低器件功耗，比如谐振式开关电源。

总体来讲，从耐压、电流能力看，可控硅目前仍然是最高的，在某些特定场合，仍然要使用大电流、高耐压的可控硅。但一般的工业自动化场合，功率电子器件已越来越多地使用 MOSFET 和 IGBT，特别是 IGBT 获得了更多的使用，开始全面取代可控硅来做为新型的功率控制器件。

1.2.2 功率电子器件

一、整流二极管

二极管是功率电子系统中不可或缺的器件，用于整流、续流等。目前比较多地使用如下三种选择：

1．高效快速恢复二极管。压降 0.8-1.2V，适合小功率，12V 左右电源。

2．高效超快速二极管。0.8-1.2V，适合小功率，12V 左右电源。

3．肖特基势垒整流二极管 SBD。0.4V，适合 5V 等低压电源。缺点是其电阻和耐压的平方成正比，所以耐压低（200V 以下），反向漏电流较大，易热击穿。但速度比较快，通态压降低。

目前 SBD 的研究前沿，已经超过 1 万伏。

二、大功率晶体管 GTR

分为：

单管形式。电流系数：10-30。

双管形式——达林顿管。电流倍数：100-1000。饱和压降大，速度慢。

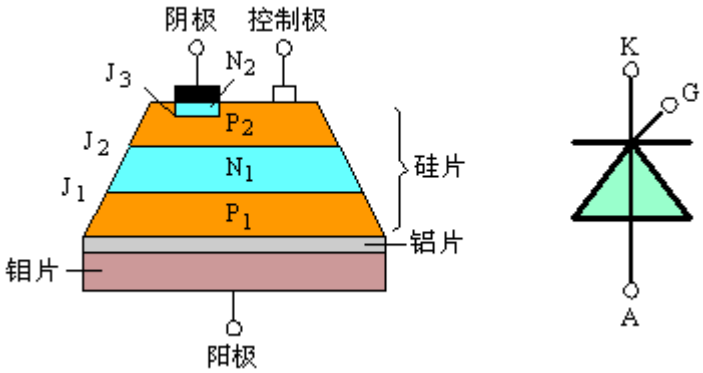
实际比较常用的是达林顿模块，它把 GTR、续流二极管、辅助电路做到一个模块内。在较早期的功率电子设备中，比较多地使用了这种器件。

这种器件的制造水平是 1800V/800A/2KHz、600V/3A/100KHz 左右（参考）。

三、晶闸管（可控硅 SCR）

晶闸管（Thyristor）是晶体闸流管的简称，又可称做可控硅整流器，以前被简称为可控硅；晶闸管是 PNPN 四层半导体结构，它有三个极：阳极，阴极和门极；晶闸管工作条件为：加正向电压且门极有触发电流；其派生器件有：快速晶闸管，双向晶闸管，逆导晶闸管，光控晶闸管等。它是一种大功率开关型半导体器件，在电路中用文字符号为“V”、“VT”表示（旧标准中用字母“SCR”表示）。

晶闸管具有硅整流器件的特性，能在高电压、大电流条件下工作，且其工作过程可以控制、被广泛应用于可控整流、交流调压、无触点电子开关、逆变及变频等电子电路中。



晶闸管的工作原理：

晶闸管 T 在工作过程中，它的阳极 A 和阴极 K 与电源和负载连接，组成晶闸管的主电路，晶闸管的门极 G 和阴极 K 与控制晶闸管的装置连接，组成晶闸管的控制电路。

晶闸管的工作条件：

1. 晶闸管承受反向阳极电压时，不管门极承受何种电压，晶闸管都处于关断状态。

2. 晶闸管承受正向阳极电压时，仅在门极承受正向电压的情况下晶闸管才导通。

3. 晶闸管在导通情况下，只要有一定的正向阳极电压，不论门极电压如何，晶闸管保持导通，即晶闸管导通后，门极失去作用。

4. 晶闸管在导通情况下，当主回路电压（或电流）减小到接近于零时，晶闸管关断。

注意事项：

①一般小功率晶闸管不需加散热片，但应远离发热元件，如大功率电阻、大功率三极管以及电源变压器等。对于大功率晶闸管，必须按手册申的要求加装散热装置及冷却条件，以保证管子工作时的温度不超过结温。

②晶闸管在使用中发生超越和短路现象时，会引发过电流将管子烧毁。对于过电流，一般可在交流电源中加装快速保险丝加以保护。快速保险丝的熔断时间极短，一般保险丝的额定电流用晶闸管额定平均电流的 1.5 倍来选择。

③交流电源在接通与断开时，有可能在晶闸管的导通或阻断对出现过压现象，将管子击穿。对于过电压，可采用并联 RC 吸收电路的方法。因为电容两端的电压不能突变，所以只要在晶闸管的阴极及阳极间并取 RC 电路，就可以削弱电源瞬间出现的过电压，起到保护晶闸管的作用。当然也可以采用压敏电阻过压保护元件进行过压保护。

可控硅在大电流、高耐压场合还是必须的，但在常规工业控制的低压、中小电流控制中，已逐步被新型器件取代。

目前的研制水平在 12KV/8000A 左右（参考）。

由于可控硅换流电路复杂，逐步开发了门极关断晶闸管 GTO。制造水平达到 8KV/8KA，频率为 1KHz 左右。

无论是 SCR 还是 GTO，控制电路都过于复杂，特别是需要庞大的吸收电路。而且，速度低，因此限制了它的应用范围拓宽。

集成门极换流晶闸管 IGCT 和 MOS 关断晶闸管之类的器件在控制门极前使用了 MOS 栅，从而达到硬关断能力。

四. 功率 MOSFET

又叫功率场效应管或者功率场控晶体管。

其特点是驱动功率小，速度高，安全工作区宽。但高压时，导通电阻与电压的平方成正比，因而提高耐压和降低高压阻抗困难。

适合低压 100V 以下，是比较理想的器件。

目前的研制水平在 1000V/65A 左右（参考）。商业化的产品达到 60V/200A/2MHz、500V/50A/100KHz。是目前速度最快的功率器件。

五. IGBT

IGBT(Insulated Gate Bipolar Transistor)，绝缘栅双极型功率管，是由 BJT(双极

型三极管)和 MOS(绝缘栅型场效应管)组成的复合全控型电压驱动式电力电子器件，兼有MOSFET 的高输入阻抗和 GTR 的低导通压降两方面的优点。GTR 饱和压降低，载流密度大，但驱动电流较大;MOSFET 驱动功率很小，开关速度快，但导通压降大，载流密度小。IGBT 综合了以上两种器件的优点，驱动功率小而饱和压降低。非常适合应用于直流电压为 600V 及以上的变流系统如交流电机、变频器、开关电源、照明电路、牵引传动等领域。

这种器件的特点是集 MOSFET 与 GTR 的优点于一身。输入阻抗高，速度快，热稳定性好。通态电压低，耐压高，电流大。

目前这种器件的两个方向：一是朝大功率，二是朝高速度发展。大功率 IGBT 模块达到 1200-1800A/1800-3300V 的水平（参考）。速度在中等电压区域（370-600V），可达到 150-180KHz。

它的电流密度比 MOSFET 大，芯片面积只有 MOSFET 的 40%。但速度比 MOSFET 低。

尽管电力电子器件发展过程远比我们现在描述的复杂，但是 MOSFET 和 IGBT，特别是 IGBT 已经成为现代功率电子器件的主流。因此，我们下面的重点也是这两种器件。

1.3 数字电位器

目前很多的电器产品及其仪器仪表所用的电位器是通过机械滑臂改变在电阻膜或线绕电阻体上的位置，以达到改变电位器阻值的大小，这种电位器称为模拟电位器。

数字电位器是利用微电子技术制成的集成电路，它是依靠电阻阵列和多路模拟开关的组合来完成阻值的变化。它没有可动的滑臂，而是通过按钮输入信号，或是通过数字输入信号来改变数字电位器的阻值。

数字电位器有易失和非易失两种，对于易失电位器，一旦器件掉电，电位器的调节位置将丢失。这时，有可能需要使用外部EEPROM、闪存或其他类型的非易失存储器。非易失数字电位器芯片内包含了这种存储器，一般为EEPROM，在断电时用于存储电位器的设置。

不同型号的数字电位器阻值不同，一般有 1 kΩ、2 kΩ、10 kΩ、50 kΩ、100 kΩ等。

此外，数字电位器需要在加电条件下进行调节和工作，没有加电的情况下不具备电阻功能。

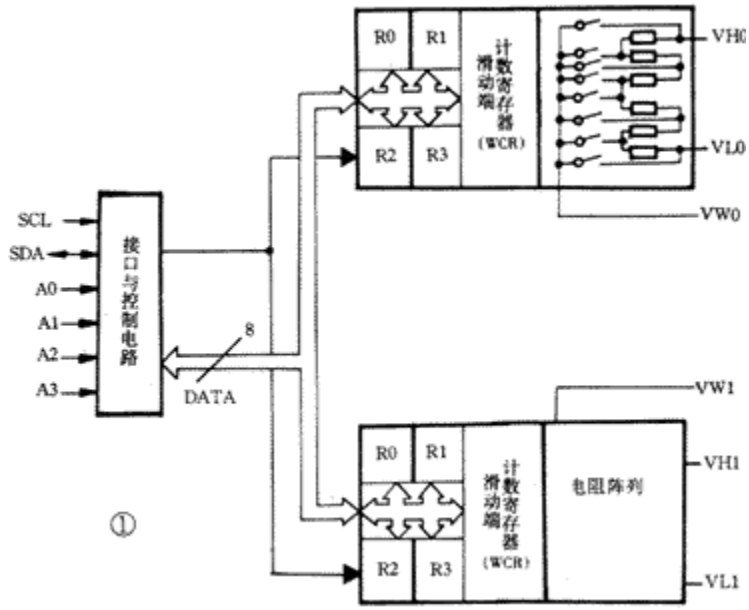
非易失性数字电位器 X9221

X9221 是美国 XICOR 公司新近研制出的功能独特的电子数控电位器。X9221 在一片 CMOS 集成电路内集成有 2 个非易失性数控电位器(E2POT)，其调节过程可以由微处理器(μP)或微控制器(μC)经二线总线接口进行控制。这种二线接口数字电位器具有如下许多优点：(1)调节精度高；(2)不易受诸如振动、污染、潮湿等影响；(3)无机械磨损；(4)接口引脚少；(5)集成度高；(6)数据可读写；(7)具有配置寄存器及数据寄存器；(8)多电平量存储功能，特别适用于音频系统；(9)易于软件控制；(10)采用设计人员熟悉的 I2C 通信协议；(11)体积小巧，易于装配。它适用于家庭影院系统、音频环绕控制、音响功放、有线电视设备等。

X9221 内含滑动端计数寄存器(WCR)及数据寄存器。它的每个 E2POT 可存储 4 个滑动端位置；每个电位器有 64 个抽头；工作电压 Vcc 则为 4.5～5.5 或 2.7～5.5V。

内部结构

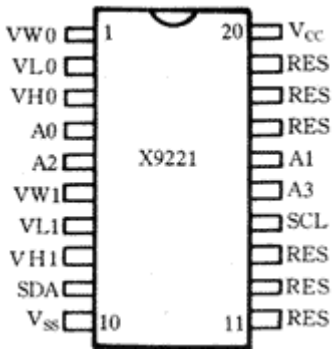
X9221 片内包含 2 个电阻阵列(或称电位器或 E2POT)和 I2C 接口电路。X9221 的功能方框图如图 1 所示。



每个电阻阵列内又包含63个电阻单元、64个电子开关、一个滑动端计数寄存器(WCR)、4个8位数据寄存器(R0～R3)、递增/递减逻辑电路、级联控制逻辑电路以及64选1译码电路。

在相邻的两个电阻单元之间以及两个端点处共设64个可以被滑动端访问的抽头。滑动端在阵列中的位置可由用户通过二线串行总线(I2C)接口控制。每个电阻阵列配置一个滑动端计数寄存器和4个数据寄存器，这4个数据寄存器可以由用户程序直接写入和读出。

滑动端计数寄存器的内容控制滑动端在电阻阵列中的位置。数据寄存器的内容可以传送到滑动端计数寄存器，以设置滑动端的位置。当前滑动端的位置可以被传送到与它相关联的4个数据寄存器中的任何一个之中。也就是说，WCR可以直接被写入，或者也可以把起辅助作用的4个数据寄存器之一的内容转移到WCR中来改变其内容。这些数据寄存器和WCR都可以由微电脑来读出或写入。



X9221 中的每一个电阻阵列的主体部分是 63 只串联连接的集成电阻器。电阻串联支路的两端 VH 和 VL 就相当于一个机械电位器的两个固定端；串联支路中的电阻器之间的连接

点以及两个端点，都可以经过场效应管开关连通到滑动端 VW 上。在同一时刻只能有一只开关闭合，究竟哪一只闭合由滑动端计数寄存器 WCR 内容确定。只有 WCR 中的低 6 位被译码，才能选择和使能 64 选 1 的开关接通。

引脚功能

X9221 共有 20 个外接引脚。它有 DIP、SOIC 和 TSSOP 三种封装形式。其引脚排列如图 3 所示。各引脚的功能如表 1 所示。

表 1　引脚功能

引　　脚	符 号	说　　明
B14	SCL	I^2C 总线串行时钟
⑨	SDA	I^2C 总线串行数据
④、⑤、B15　、B16	A0-A3	设置器件从属地址低 4 位
③、⑧	VH0-VH1	电位器终端，等效于机械电位器的上端
②、⑦	VL0-VL1	电位器终端，等效于机械电位器的下端
①、⑥	VW0-VW1	电位器滑动端，等效于机械电位器中心抽头
B20	Vcc	系统电源正极
⑩	Vss	系统地
B11　、B12　、B17～B19	RES	保留，无连接

器件寻址　在开始条件的后面，主器件必须输出它所要访问的从器件的地址。从器件的高4位地址是器件类型识别码，器件的类型不同，识别码也就不同，并且识别码是固定不变的。对于数控电位器X9221来说，这个识别码固定为0101。格式如下：

0	1	0	1	A2	A1	A0	A0
器件类型识别码				器件地址			

从器件的低4位是该器件的编程地址，该地址由A0～A3引脚的连接状态来定义。借助于器件的编程地址，主机可以识别一个系统中类型相同的多个器件(在此可以识别16片X9221)。每次通信的开始，X9221都把接收到的地址与自己的地址(含识别码和编程地址)相比较。如果是所有的8位地址都相符，则X9221做出一个应答响应。

I3	I2	I1	I0	0	P0	R1	R0
指令码				电位器选择		寄存器选择	

指令结构

X9221 共有 9 条指令，指令的长度为两个字节或三个字节不等。每条指令的第一个字节为地址字节，第二个字节为指令字节。在指令字节中又包含指令码和寄存器指针信息，即 4 个高位 I0～I3 是指令码；紧接着的两位 0 和 P0 选择 2 个电位器中的哪一个，最后两位 R1 和 R0 选择 4 个寄存器中的哪一个。

表 2　指令集

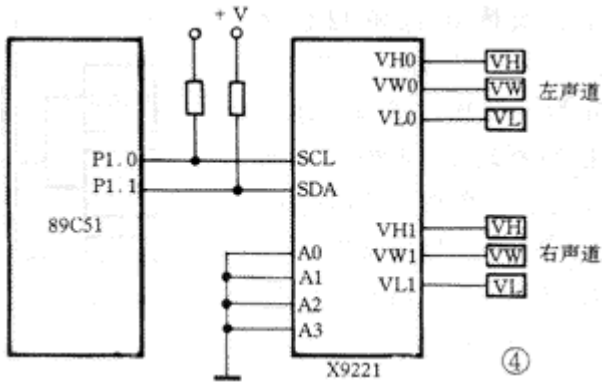
指令名	指令字节								操　　作
	I3	I2	I1	I0	0	P0	R1	R0	
RW	1	0	0	1	0	1/0	N/A	N/A	读出由 P1P 0 指定 WCR 的内容
WW	1	0	1	0	0	1/0	N/A	N/A	写人新值到 P1P 0 指定 WCR 中

RDR	1	0	1	1	0	1/0	1/0	1/0	读出由 P1P0R1R 0 指定数据寄存器内容
WDR	1	1	0	0	0	1/0	1/0	1/0	写入新值到 P1P 0R1R0 指定数据寄存器中
XDW	1	1	0	1	0	1/0	1/0	1/0	传输由 P1P 0R1R0 指定数据寄存器内容到 P1P0 指定的 WCR 中
XWD	1	1	1	0	0	1/0	1/0	1/0	传输由 P1P 0 指定的 WCR 内容到 P1P 0R1R0 指定的数据寄存器中
GXDW	0	0	0	1	N/A	N/A	1/0	1/0	传输由 R1R0 指定所有的 4 个数据寄存器的内容到与它们相关的 WCR 中
GXWD	1	0	0	0	N/A	N/A	1/0	1/0	传输所有的 WCR 内容到与它们相关的由 R1R0 指定的 4 个数据寄存器中
IDW	0	0	1	0	0	1/0	N/A	N/A	使能由 P1P 0 指定的 WCR 递增/递减操作

注：1/0 表示数据 0 或 1；N/A 表示没有使用。

应用举例

电路中，用一片 X9221 的两个电位器分别控制双声道立体声系统的左声道和右声道，以实现对传统音响设备的模糊控制、智能控制以及遥控。控制器件选一片 89C51 单片机，仅用 P1.0 和 P1.1 两个端口与 X9221 的 SDA 和 SCL 相连。通常在 SDA 和 SCL 线上需要设置上拉电阻，该阻值取决于连接到总线上的所有器件的总容量。按图 4 中的情况，等效容量约为 18pF。



如果所选用的单片机具有内部上拉电阻，则外部上拉电阻可以省略。当应用系统中只有一片X9221时，其4位地址A0～A1引脚可以都连接到地，此时编程地址定义为0000。

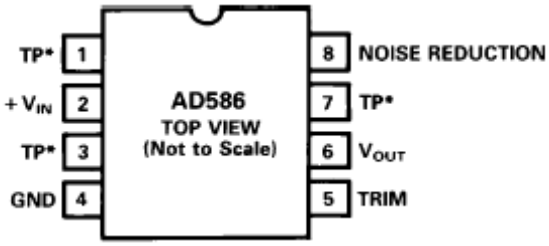
1.4 基准电源芯片

在需要高精度电压基准的场合，如高精度 A/D 转换器、D/A 转换器、传感器等应用电路中，使用低精度的 78 系列或 79 系列集成稳压器无法满足要求，这时就需要采用基准电压芯片来供给精密电压，以实现高精度的转换或测量。

最常用的基准电压芯片输出为±5V，因为大多数的 A/D 和 D/A 转换器采用 5V 电压基准。也有 3V、10V 等标准。

1、 AD586

AD586 为高精度 5V 参考电压源，最大偏差为±2.0mV，其优良的性能使其得到了广泛应用。其引脚图如下：



引脚功能如下：

1、3、7 脚为内部为内部测试端，使用该芯片时不需连接；

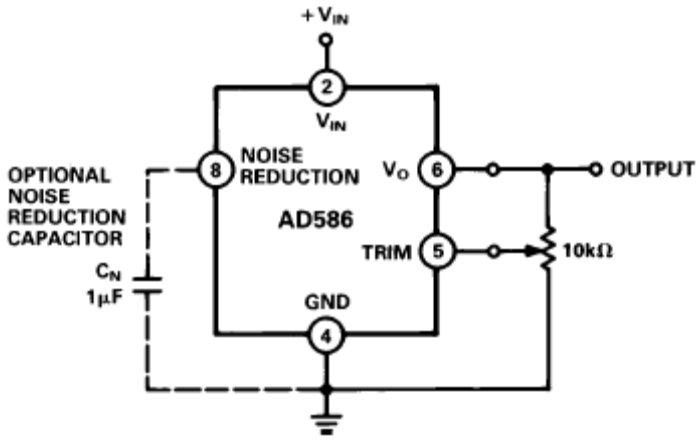
2 脚为正电源输入端，要求电压在 10.8V～36V 之间，推荐采用 12V 或 15V；

4 脚接地；

8 脚为噪声抑制端，使用时可在 8 脚及 4 脚之间接一个 1uF 的电容；

6 脚为精密电压输出端；

5 脚一般不用，在需要 5V 以上电压基准输出时，可采用下图方式连接，这样输出电压最大可达 5V+300mV。



AD586 应用非常简单，只需要 2、4 脚输入正确范围（10.8V～36V）的电压，6 脚就会输出+5V 的精确电压，精度只取决于应用哪种器件类型。不同类型器件的参数见下表：

型号	误差	温度漂移	温度范围	封装类型
AD586JN	20 mV	25 ppm/°C	0°C to +70°C	N-8
AD586JQ	20 mV	25 ppm/°C	0°C to +70°C	Q-8
AD586JR	20 mV	25 ppm/°C	0°C to +70°C	SO-8
AD586KN	5 mV	15 ppm/°C	0°C to +70°C	N-8
AD586KQ	5 mV	15 ppm/°C	0°C to +70°C	Q-8
AD586KR	5 mV	15 ppm/°C	0°C to +70°C	SO-8
AD586LN	2.5 mV	5 ppm/°C	0°C to +70°C	N-8
AD586LR	2.5 mV	5 ppm/°C	0°C to +70°C	SO-8
AD586MN	2 mV	2 ppm/°C	0°C to +70°C	N-8
AD586AR	5 mV	15 ppm/°C	–40°C to +85°C	SO-8
AD586BR	2.5 mV	5 ppm/°C	–40°C to +85°C	SO-8

AD586LQ	2.5 mV	5 ppm/°C	0°C to +70°C	Q-8
AD586SQ	10 mV	20 ppm/°C	–55°C to +125°C	Q-8
AD586TQ	2.5 mV	10 ppm/°C	–55°C to +125°C	Q-8
AD586JCHIPS	20 mV	25 ppm/°C	0°C to +70°C	

2、MAX873、MAX875 和 MAX876

MAX873、MAX875 和 MAX876 分别是 2.5V、5V 和 10V 的高精度电压基准芯片，而且功耗很低。它们的最大误差分别是 1.5mV、2mV 和 3mV，而且电压输入范围宽。

引脚如下：

1、8 脚为测试脚，无需连接；

7 脚为空管脚；

2 脚为正电源输入，MAX873 的范围是 4.5V～18V，MAX875 的范围是 7V～18V，MAX876 的范围是 12V～18V；

4 脚接地；

6 脚为基准电压输出。

MAX873、MAX875 和 MAX876 应用非常简单，只需要 2、4 脚输入正确范围的电压，6 脚就会输出所需的精确电压，精度只取决于应用哪种器件类型。

1.5 多路模拟开关

当需要对多个模拟量进行模数变换时，由于模数转换器（A/D 转换器）的价格较贵，通常不是每个模拟量输入通道设置一个 A/D，而是多路输入模拟量共用一个 A/D，中间经过多路转换开关切换，即多路模拟开关。

多路模拟开关最重要的部分是电子开关 AS，它是用数字电子逻辑控制模拟信号通、断的一种电路，通常是由双极型晶体管（BJT）、结型场效应晶体管（J-FET）或金属氧化物半导体场效应管（MOS-FET）等类型组成的电子开关。

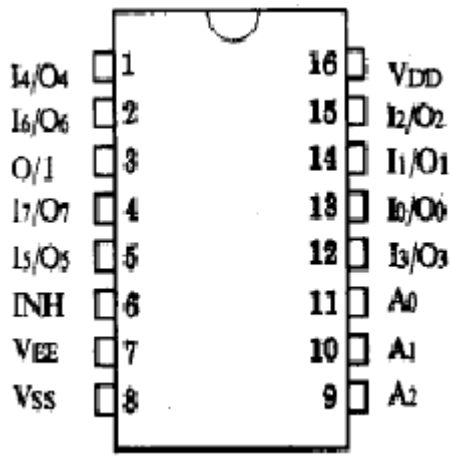
多路模拟开关芯片种类很多，这里介绍两种常用的芯片。

1、CD4051

CD4051 芯片允许双向使用，即可以用于从多路到单个的转换，也可以用于从单个到多个的转换。它有 3 个二进制控制输入端 A2、A1、A0 和一个禁止输入端 INH，用 3 位二进制信号来选择 8 个通道中的一个通道。当 INH=1 时，通道断开，禁止模拟量输入；当

INH=0 时，通道接通，允许模拟量输入。

该多路开关的输入电平范围广，数字量输入为 3~15V，模拟量输入可达 15V。



CD4051引脚图

CD4051真值表

输 入 状 态				被 选 通 道
INH	A2	A1	A0	
0	0	0	0	IO0
0	0	0	1	IO1
0	0	1	0	IO2
0	0	1	1	IO3
0	1	0	0	IO4
0	1	0	1	IO5
0	1	1	0	IO6
0	1	1	1	IO7
1	X	X	X	无

电源电压范围：3V～15V；输入电压范围：0V～VDD。

工作温度范围：M类：－55℃～125℃；E 类：－40℃～85℃。

模拟开关 CD4051 使用的 4 个要点

1、使用单电源时，CD4051 的 VEE 可以和 GND 相连。

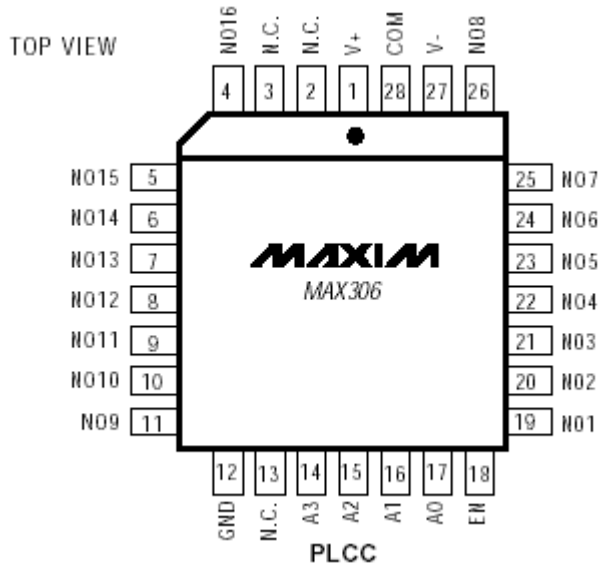
2、强烈建议 A，B，C 三路片选端要加上拉电阻。

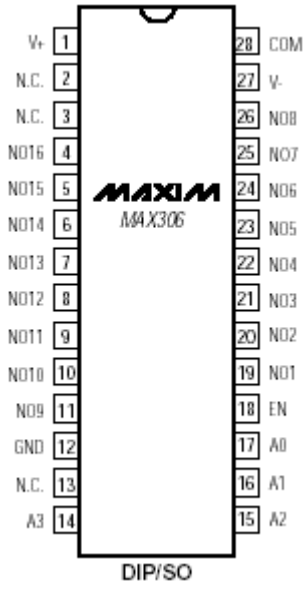
3、CD4051 的公共输出端不要加滤波电容（并联到地），否则不同通道转换后的电压经电容冲放电后会引起极大的误差。

4、禁止输出端（INH）为高电平时，所有输出切断，所以在应用时此端接地。作音频信号切换时，最好在输入输出端串入隔直电容。

2、MAX306

MAX306 是美信公司的一款高性能的 16 通道模拟多路开关。它支持很宽的电源范围：单电源供电时 4.5V~30V；双电源供电时±4.5V~±20V。





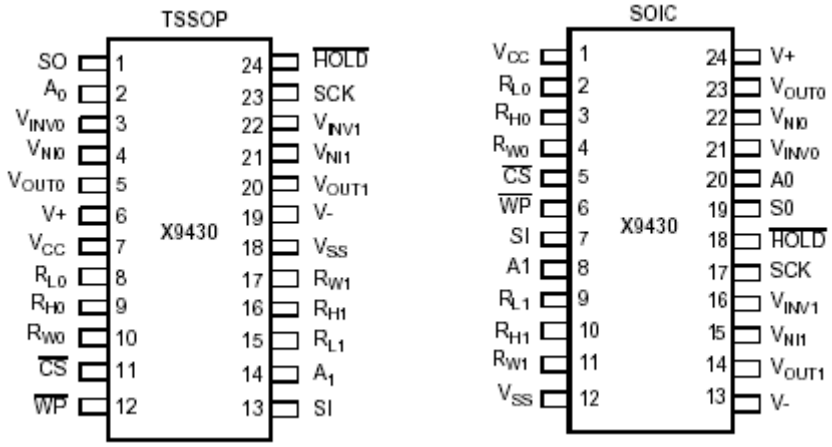
MAX306 PIN	NAME	FUNCTION
1	V+	Positive Supply Voltage Input
2, 3, 13	N.C.	No Internal Connections
4–11	NO16–NO9	Analog Inputs–bidirectional
12	GND	Ground
14–17	A3–A0	Address Inputs
18	EN	Enable Inputs
19–26	NO1–NO8	Analog Inputs–bidirectional
27	V-	Negative Supply Voltage Input
28	COM	Output–bidirectional

A3	A2	A1	A0	EN	ON Switch
X	X	X	X	0	None
0	0	0	0	1	1
0	0	0	1	1	2
0	0	1	0	1	3
0	0	1	1	1	4
0	1	0	0	1	5
0	1	0	1	1	6
0	1	1	0	1	7
0	1	1	1	1	8
1	0	0	0	1	9
1	0	0	1	1	10
1	0	1	0	1	11
1	0	1	1	1	12
1	1	0	0	1	13
1	1	0	1	1	14
1	1	1	0	1	15
1	1	1	1	1	16

1.6 可编程运算放大器

1、可编程运算放大器 X9430

X9430 是可编程运算放大器。它采用了 Xicor 公司的精密模拟技术将模拟电路、数字电路及非挥发性电路集成到一个芯片上，构成了一个新颖的混合信号器件。高增益、偏置及低功耗可编程。X9340 包括两个通用运算放大器，可通过串行外围接口(SPI)或 I2C 串行接口写入控制字对其进行编程。X9430 的技术指标与工业标准的 741，301A 以及 OP07 相同。X9430 的可编程参数存贮在非易失存储器中，这样即使在掉电时，也可保存数值。同时片内含有 16 字节 E2PROM，其中 4 字节用于存放运放参数。



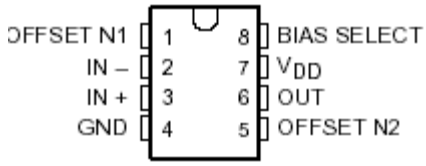
引 脚	描 述	引 脚	描 述
SCK	串行时钟	RW0 - RW1	电位计（滑动端）
SI	串行输入	VNI(0,1), VINV(0,1)	输入电压
SO	串行输出	VOUT0, VOUT1	运放输出
A0 - A1	器件地址	WP	硬件写保护
CS	片选	V+,V-	运放模拟电源
HOLD	保持	VCC	数字信号电源
RH0 - RH1,	电位计（终端）	VSS	数字信号地
RL0 - RL1			

2、可编程低功耗运算放大器 TLC271

TLC271 是美国 TI 公司研制的运算放大器，其输入阻抗高达 1012kΩ，而输入失调电压仅为 0.1μV/V。因此，用户可以利用 TLC271 选择功耗和交流性能的最佳组合并用于各种电路中。

用户在使用偏置选择时，可将 TLC271 与双极－场效应晶体管以及 NFET 等一起应用于各种电路，且可选择 TLC271（10mV）与 TLC271B（2mV）之间的各种失调电压级别。而高输入阻抗，低位移电流及优越的共模抑制比和电源电压抑制比性能则使 TLC271 成为用户使用不断升级的程序的最佳选择。

TLC271 的功耗很小，具有许多与双级技术相关的性能。TLC271 可广泛应用于各种电路，如转换程序的连接、等效计算、信号放大块、有源滤波器以及信号缓存等。TLC271 可使用直流电源，故而是应用于遥控电路和蓄电池供电电路的理想选择。TLC271 有 C－后缀型装置、I－后缀型装置和 M－后缀型装置三种形式。下图所示为 TLC271 的引脚图。



工作条件

	C-后缀型		I-后缀型		M-后缀型		单位
	最小值	最大值	最小值	最大值	最小值	最大值	
电源电压 V_{DD}	3	16	4	16	5	16	V
共模输入电压 V_{DD}=5V 时	-0.2	3.5	-0.2	3.5	0	3.5	V
V_{DD}=10V	-0.2	8.5	-0.2	8.5	0	8.5	
工作温度，TA	0	70	-40	85	-55	125	℃

1.7 电压/电流变换器（V/I）

在测控系统中，信号在远距离传输时，外部干扰非常严重。通常，采用电压/电流变换器，将电压信号转换成电流信号（4~20mA）进行传输，以减弱信号的衰减及干扰。因为

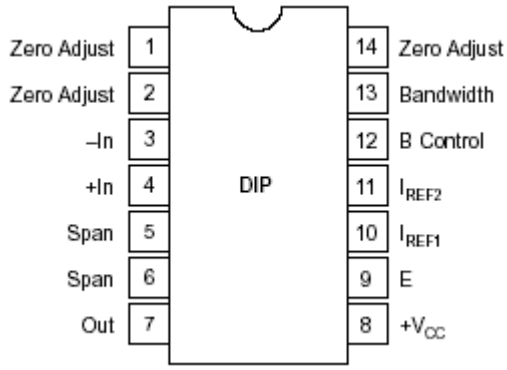
传输线上的压降、分布电容、接触电阻等对电流信号不起作用，外部电动机、继电器、开关等电磁干扰也不会影响传输线上的变化。

将电压变换为电流信号可采用运算放大器来实现，这里不予讨论。本节主要介绍集成电压/电流变换器。

集成电压/电流变换器种类很多，常用的有 XTR101、XTR105、XTR110、AD420、AD421、AD693、AD694 等。本节主要介绍 XTR101。

XTR101 是美国 BB 公司的产品，该芯片将微弱的电压信号输入转变为标准的电流信号（4~20mA）输出。在-40~+85℃温度范围内以 12 为精度完成电阻参数到 4~20mA 的电流转换。

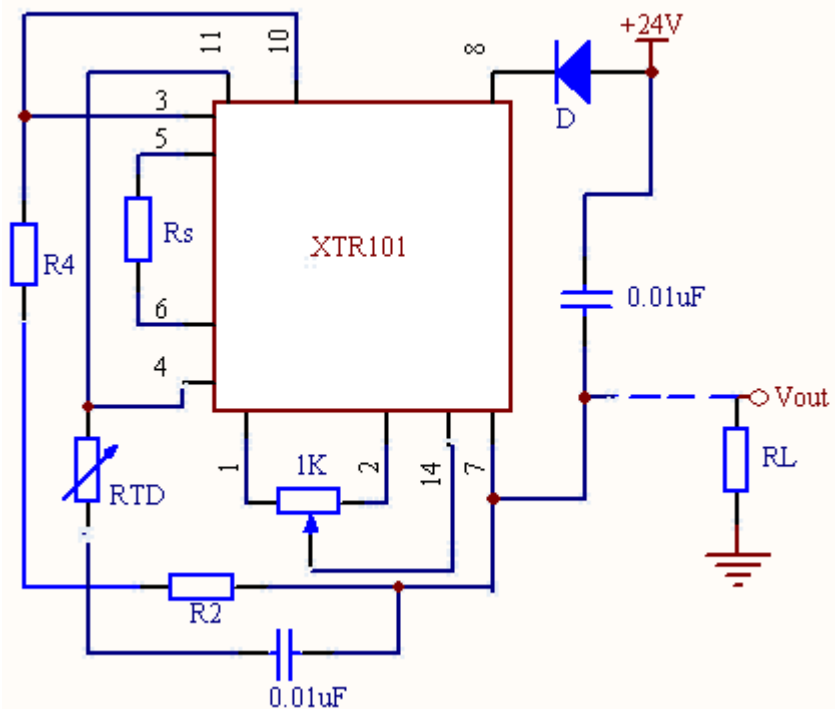
XTR101 是一种低漂移、双线输入/输出的微电路集成器件。它由精密测量放大器、压控输出电流源和双匹配参考电流组成。这种结合非常适用于各种传感器，如热电偶、RTD、热敏电阻和应变计电桥等的遥控信号调节。其引脚如下：



引脚号	功能描述	引脚号	功能描述
1、2	调零	9	三极管发射级
3、4	电压输入	10	恒流 1mA 输出
5、6	量程调节	11	恒流 1mA 输出
7	变换电流输出	12、13	频带控制
8	正电源	14	调零

传感器的电压信号由 3、4 叫输入；5、6 脚外接电阻 Rs 可以调节输出满幅值；1、2、14 脚外接电位器组成初始调零电路；10、11 脚分别输出两个 1mA 恒流，可用于传感器供电；8 脚接电源正端；7 脚通过负载电阻 R_L 接电源负端；12、8、9 脚可外接 NPN 功率三极管。

下图为 XTR101 与铂热电阻传感器 RTD 的接线图。



1.8 模拟信号放大器

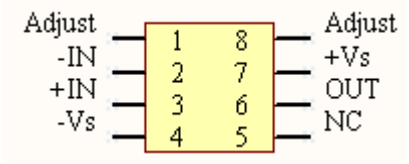
随着新技术、新工艺、新材料的发展，集成运放的精度越来越高，性能指标越来越好，品种也越来越多。集成运放分为通用型（如 F007、uA471 等）和专用型两类。专用型有低漂移型（如 DG725、OP07、5G7650 等）、高阻型（如 LF356、CA3410、5G28 等）及低功耗型（如 LM4250、uA735）等。此外，还有单电源的集成运放（如 LM324、DG324）。使用时应根据实际需要选择运放类型，选择的依据是其性能参数。运放的主要参数有：差模输入电阻、输出电阻、输入失调电压、电流以及温漂、开环差模增益、共模抑制比和最大输出电压幅度等。

1.8.1 集成运算放大器 OP07

由于经传感器变换后的模拟信号有时是很微弱的微伏级信号（如热电偶），而通用运放一般都具有毫伏级的失调电压和每度数微伏的温度漂移，显然是不能用于放大微弱信号的。在设计中，需要采用高精度运算放大器或测量放大器。下面介绍一种常用的超低失调电压、超低温漂的典型运算放大器 OP07，供设计放大器时使用。

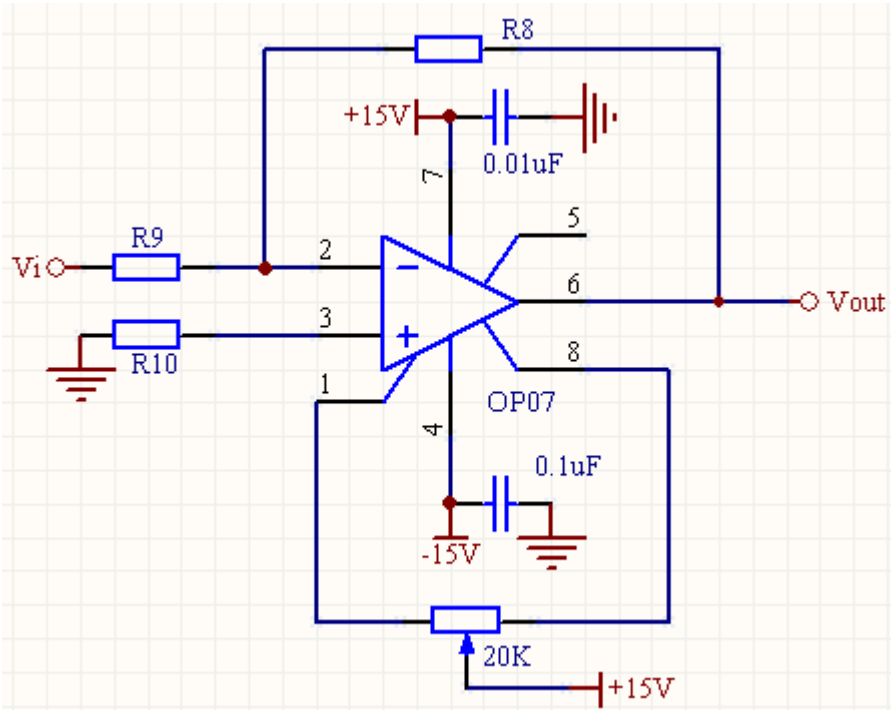
OP07 有 A、D、C、E 各档，它是高精度运算放大器，具有极低的失调电压（10uV）和偏置电流（0.7nA），它的温漂系数为 0.2uV/℃，长期稳定性指标为 0.2uV/月。OP07 具有较高的共模输入范围（±14V），共模抑制比 CMRR=126dB 以及极宽的供电电源范围（3～

18V 和-3～-18V）。



引脚图

1、8 脚：调零引脚；
2 脚：反向输入端；
3 脚：同向输入端；
4 脚：负电源；
5 脚：空引脚；
6 脚：输出引脚；
7 脚：正电源；
8 脚：调零引脚。



基本连接图

1.8.2 测量放大器

测量放大器又叫仪表放大器（简称 IA）。它不仅能满足放大要求，而且具有精确的增益标定，因此又称数据放大器。

1. 通用 IA

通用 IA 由三个运算放大器 A1、A2、A3 组成，如图 12-8 所示。 其中，A1 和 A2 组

成具有对称结构的差动输入 / 输出级，差模增益为 1+2*R*1/*R*G，而共模增益仅为 1。A3 将 A1、A2 的差动输出信号转换为单端输出信号。 A3 的共模抑制精度取决于四个电阻 *R* 的匹配精度。 通用 IA 的电压放大倍数为

$$A_{\mathrm{u}}=\frac{u_0}{u_{11}-u_{12}}=-(1+\frac{2R_1}{R_G})$$

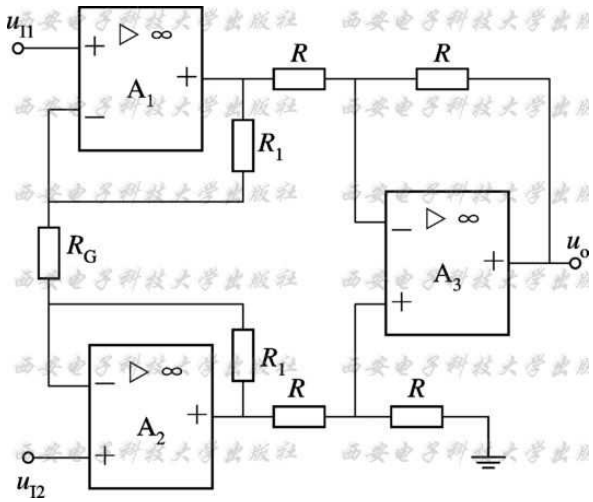


图　通用 IA 的结构

2. IA 的技术指标

测量放大器最重要的技术指标有： 非线性度、 偏置漂移、 建立时间以及共模抑制比等， 这些指标均为放大器增益的函数。

（1） 非线性度： 它的定义为放大器输出、 输入实际特性曲线与理想特性曲线（直线）的相对偏差。 在增益 *G*=1 时，一个 12 位（bit）系统的非线性度若为±0.025%， 则在 *G*＝500 时， 其非线性度可达±0.1%， 相当于系统精度降低到 10 倍以下。

（2） 偏置漂移：它是指工作温度变化 1℃时， 相应的直流偏置变化量。一个放大器的分辨率主要被直流偏置的不可预料性所限制。 放大器的偏置漂移一般为 1～50 μV/℃，也与增益 *G* 有关。如一个有 2 μV/℃漂移的放大器，当 *G*=1000、Δ*t*=10℃时， 其输出端将产生 20 mV 的偏置电压。这个数字相当于 12 位 ADC 在输入范围为 0～10 V 时的八个 LSB 值。值得注意的是，一般厂家只给出典型值，而最大值可以是典型值的 3～4 倍。

（3） 建立时间： 放大器的建立时间定义为从输入阶跃信号起，到输出电压达到满足给定误差（典型值为±0.01%）的稳定值为止所需用的时间。 一般 IA 的增益 *G*＞200，精度约为±0.01%， 建立时间约为 50～100 μs， 而高增益 IA 在同样精度下的建立时间可达 350 μs。因此，在数据采集系统中决定信号传输能力的往往是 IA 而不是 ADC。

（4） 恢复时间：放大器的恢复时间是指从断掉输入 IA 的过载信号起，到 IA 的输出信号恢复至稳定值时（与输入信号相应）的时间。

（5） 共模抑制比： IA 的共模抑制比定义为差模电压放大倍数 *A*d 与共模电压放大倍数 *A*c 比值的对数单位，即

$$\mathrm{CMR} = 20\lg\frac{A_{\mathrm{d}}}{A_{\mathrm{c}}}$$

3.单片集成测量放大器 AD620

单片机集成测量放大器种类很多，常用的有 AD521、AD526、AD620、AD623、AD624、AD626 等。这里介绍 AD620。

AD620 是只用一个外部电阻就能设置放大倍数为 1~1000 的低功耗高精度测量放大器。它体积小，为 8 脚 DIP 或 SOIC 封装，引脚排列如下：

1、8 脚：外部电阻 Rg 接入端，用于设置放大倍数；

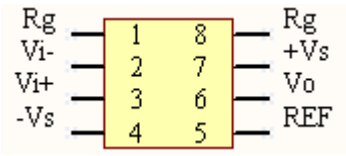
2、3 脚：模拟信号输入端；

6 脚：放大器输出端；

5 脚：参考地；

7 脚：正电源输入端，如接+12V；

4 脚：负电源输入端，如接-12V。



AD620 主要特点如下：

（1）只用一只外部电阻（跨接在 1 脚和 8 脚之间）就能设置放大倍数 G=1~1000；

（2）低失调电压、低失调电压漂移和低噪声性能，确保高增益精密放大。它的最大输入失调电压为 50uV，最大输入失调电压漂移为 1uV/℃，最大输入偏置电流为 2nA。G=10 时，共模抑制比大于 93dB；

（3）低功耗，供电电压范围为 2.3～±18V，最大供电电流仅为 1.3mA；

（4）体积小，只有 8 个引脚。

第2章　存储器类型及扩展

2.1 基础知识

1、半导体存储器的类型

半导体存储器按制造工艺分，可分为双极型和 MOS 型两大类；按存取方式分，又可分为随机存取存储器 RAM 和只读存储器 ROM 两大类；RAM 根据存储电路的性质不同，又可分为静态 RAM（SRAM）和动态 RAM（DRAM），ROM 按其性能不同，又可分为掩模式 ROM、熔炼式可编程 PROM、可用紫外线擦除、可编程的 EPROM 和可用电擦除、可编程的 E^2PROM。

目前大容量存储器主要采用闪存芯片。

2、半导体存储器的主要特点及主要性能指标

半导体存储器具有体积小、速度快、耗电少、价格低的优点。

半导体存储器主要有以下几个主要性能指标：

（1）　存储容量：存储器所能存储二进制数码的数量，即所含存储元的总数

（2）　存取时间（读写周期）：从启动一次存储器操作到完成该操作所经历的时间

（3）　功耗：每个存储元消耗功率的大小

（4）　可靠性；对电磁场及温度变化等的抗干扰能力。

3、试比较动态 RAM 与静态 RAM 的优缺点？

动态 RAM 集成度高、功耗低、价格低；但由于它是以电容上的电荷存储信息，必须定时刷新，所以接口电路比较复杂；

静态 RAM 速度快，但由于是用双稳电路存储信息，集成度较低、功耗较大、成本较高。

4、设计存储器接口应考虑哪些主要问题？

在设计存储器接口时除了要考虑存储器的地址空间外，还要考虑

存储器与 CPU 的时序配合问题：慢速存储器要能够向 CPU 申请延长总线传输周期；

CPU 总线的负载能力：大系统中，考虑到总线驱动能力不够，需要在接口中加入驱动器/缓冲器；

存储芯片的选择：选择芯片类型时根据存储信息类型的不同决定选择 RAM 或 ROM；选择芯片具体型号时，在满足容量要求的情况下，尽量选择容量大、集成度高的芯片。

5、当 CPU 与低速存储器接口时，通常采用什么方法进行速度匹配？举例说明。

当 CPU 与低速存储器接口时，通常由低速存储器向 CPU 发出“等待申请”信号，使 CPU 在正常的读/写周期之外再插入一个或几个等待周期，这样就使指令的时钟周期数增加了。

例如，在 8086CPU 的引脚上提供了一根 READY 信号，CPU 在每个总线周期的 T_3 时钟周期和插入的等待周期 T_W 中检测 READY，若 READY＝0，就在 T_3 或当前的 T_W 之后插入一个等待周期，在等待周期中继续检测 READY 信号。所以慢速存储器在与 CPU 接口时，只要能在 T_3 中（CPU 检测前）使 READY＝0，就可以让 CPU 延长总线传输周期。通过控制 READY 维持为低电平的时间长短可以控制插入等待周期的个数。

6、存储芯片的选择与接口电路有何关系？挑选时应注意哪些问题？

存储芯片的选用和存储器接口设计直接相关：不同类型、不同型号的芯片构成的存储器，其接口方法和复杂程度都不同。

在选择时一般要根据存储器的存放对象、总体性能、芯片类型和特征等方面综合考虑。

7、片选控制译码有哪几种常用方法？其中哪几种方法存在地址重叠问题？

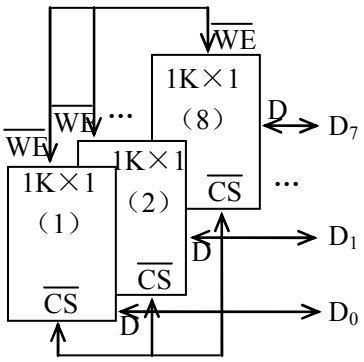
片选控制译码有线选法、全译码法、部分译码法和混合译码法。其中线选法、部分译码法和混合译码法都存在地址重叠的问题。

8、用 1024×1 位的 RAM 芯片组成 16K×8 位的存储器，需要多少个芯片？分为多少组？共需多少根地址线？地址线如何分配？试画出与 CPU 的连接框图。

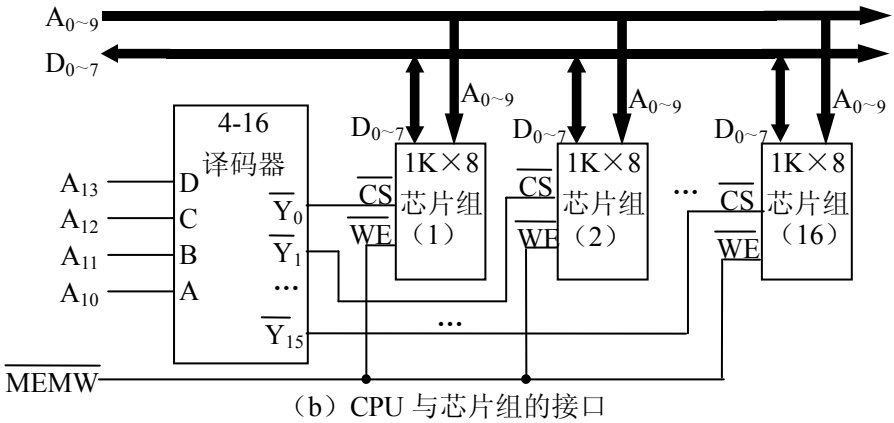
用 1024×1 位的 RAM 芯片构成 16K×8 位的存储器，共需要 16×8＝128 片；8 片为一组，共分为 16 组；共需要 14 根地址线；其中低 10 根作低位地址，实现片内单元的选择，高 4 位进行译码，产生片选信号，从 16 组中选中一组作为当前读写操作的对象。

首先进行芯片扩展，由 8 片 1024×1 位的芯片组成一个 1024×8 位的芯片组，除数据线之外，将一组中 8 个芯片的同名引脚连在一起（包括：低位地址 $A_{0\sim9}$、读写控制信号、片选信号），如图 a 所示：

然后将 CPU 的存储器读写控制信号与芯片组的读写控制相连；低位地址 $A_{0\sim9}$ 与芯片组的低位地址 $A_{0\sim9}$ 相连；再设计译码电路，产生 16 个译码输出信号，分别与 16 组的片选信号相连，如图（b）所示。



（a）芯片扩展



（b）CPU 与芯片组的接口

9、DRAM 接口电路与 SRAM 接口电路的主要区别是什么？

DRAM 和 SRAM 相比，由于存储原理和芯片结构上的区别，使之在与 CPU 接口时有两个特殊的问题要考虑：一是由于 DRAM 芯片中的存储元是靠栅极电容上的电荷存储信息的，时间一长，信息就会丢失，所以必须定时刷新；二是由于 DRAM 芯片集成度高，存储容量大，使得引脚数量不够用，所以地址输入一般采用两路复用锁存方式。

10、DRAM 控制器一般由哪几个主要部分组成？各自功能是什么？

DRAM 控制器的组成，及各部分的主要功能如下：

（1） 地址多路开关：把来自 CPU 的地址转换成行地址和列地址分两次送出；

（2） 刷新定时器：定时提出刷新请求；

（3） 刷新地址计数器：提供刷新地址，每刷新一行，计数器自动加 1，全部行刷新一遍后自动回零；

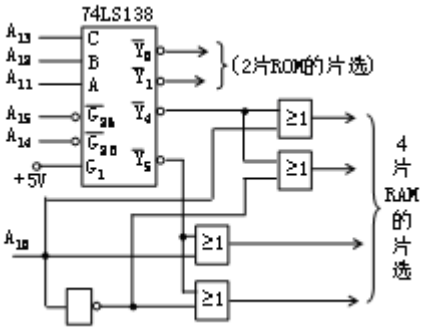
（4） 仲裁电路：当来自 CPU 的访问存储器请求和来自刷新定时器的刷新请求同时产生时，对二者的优先权进行裁定；

（5） 时序发生器：提供行地址选通信号 $\overline{RAS}$、列地址选通信号 $\overline{CAS}$ 和读写允许信号 $\overline{WE}$。

11、当构成存储器的存储芯片容量不一致时，如何进行地址译码电路设计？举例说明。

当构成存储器的存储芯片容量不一致时，有两种方法可共选择。一是用各自的译码电路分别译码产生各自的片选信号；二是分两次译码来实现。实际中采用第 2 种方法居多，这种方法首先按芯片容量大的进行一次译码，将一部分输出作为大容量芯片的片选信号；另外一部分输出则与其他相关地址一起进行二次译码，产生小容量芯片的片选信号。

例如：用 2KB 的 ROM 和 1KB 的 RAM 构成 4KB 的 ROM(0000H～0FFFH)和 4KB 的 RAM(2000H～2FFFH)，设系统有 16 根地址线，则芯片选择信号的产生如图所示：



12、Cache 结构中，地址索引机构的作用是什么？一般用什么构成？

地址索引机构中存放着与高速缓冲存储器中内容相关的高位地址，当访问 Cache 命中时，用来和地址总线上的低位地址一起形成访问 Cache 的地址。

为了保证 Cache 机构访问的快速性，地址索引机构一般采用按内容存取的相联存储器（CAM），它是一种 TTL 器件，本身读写的时间延迟极小，且全部比较一次完成。

2.2 闪存

NOR 和 NAND 是现在市场上两种主要的非易失闪存技术。Intel 于 1988 年首先开发出 NOR flash 技术，彻底改变了原先由 EPROM 和 EEPROM 一统天下的局面。紧接着，1989 年，东芝公司发表了 NAND flash 结构，强调降低每比特的成本，更高的性能，并且象磁盘一样可以通过接口轻松升级。但是经过了十多年之后，仍然有相当多的硬件工程师分不清 NOR 和 NAND 闪存。

相“flash 存储器”经常可以与相“NOR 存储器”互换使用。许多业内人士也搞不清楚 NAND 闪存技术相对于 NOR 技术的优越之处，因为大多数情况下闪存只是用来存储少量的代码，这时 NOR 闪存更适合一些。而 NAND 则是高数据存储密度的理想解决方案。

NOR 的特点是芯片内执行(XIP, eXecute In Place)，这样应用程序可以直接在 flash 闪存内运行，不必再把代码读到系统 RAM 中。NOR 的传输效率很高，在 1～4MB 的小容量时具有很高的成本效益，但是很低的写入和擦除速度大大影响了它的性能。

NAND 结构能提供极高的单元密度，可以达到高存储密度，并且写入和擦除的速度也很快。应用 NAND 的困难在于 flash 的管理和需要特殊的系统接口。

1、性能比较

flash 闪存是非易失存储器，可以对称为块的存储器单元块进行擦写和再编程。任何 flash 器件的写入操作只能在空或已擦除的单元内进行，所以大多数情况下，在进行写入操作之前必须先执行擦除。NAND 器件执行擦除操作是十分简单的，而 NOR 则要求在进行擦除前先要将目标块内所有的位都写为 0。

由于擦除 NOR 器件时是以 64～128KB 的块进行的，执行一个写入/擦除操作的时间为 5s，与此相反，擦除 NAND 器件是以 8～32KB 的块进行的，执行相同的操作最多只需要 4ms。

执行擦除时块尺寸的不同进一步拉大了 NOR 和 NADN 之间的性能差距，统计表明，对于给定的一套写入操作(尤其是更新小文件时)，更多的擦除操作必须在基于 NOR 的单元中进行。这样，当选择存储解决方案时，设计师必须权衡以下的各项因素。

- NOR 的读速度比 NAND 稍快一些。
- NAND 的写入速度比 NOR 快很多。
- NAND 的 4ms 擦除速度远比 NOR 的 5s 快。
- 大多数写入操作需要先进行擦除操作。
- NAND 的擦除单元更小，相应的擦除电路更少。

2、接口差别

NOR flash 带有 SRAM 接口，有足够的地址引脚来寻址，可以很容易地存取其内部的每一个字节。

NAND 器件使用复杂的 I/O 口来串行地存取数据，各个产品或厂商的方法可能各不相同。8 个引脚用来传送控制、地址和数据信息。

NAND 读和写操作采用 512 字节的块，这一点有点像硬盘管理此类操作，很自然地，基于 NAND 的存储器就可以取代硬盘或其他块设备。

3、容量和成本

NAND flash 的单元尺寸几乎是 NOR 器件的一半，由于生产过程更为简单，NAND 结构可以在给定的模具尺寸内提供更高的容量，也就相应地降低了价格。

NOR flash 占据了容量为 1～16MB 闪存市场的大部分，而 NAND flash 只是用在 8～128MB 的产品当中，这也说明 NOR 主要应用在代码存储介质中，NAND 适合于数据存储，NAND 在 CompactFlash、Secure Digital、PC Cards 和 MMC 存储卡市场上所占份额最大。

4、可靠性和耐用性

采用 flash 介质时一个需要重点考虑的问题是可*性。对于需要扩展 MTBF 的系统来说，Flash 是非常合适的存储方案。可以从寿命(耐用性)、位交换和坏块处理三个方面来比较 NOR 和 NAND 的可靠性。

1）寿命(耐用性)

在 NAND 闪存中每个块的最大擦写次数是一百万次，而 NOR 的擦写次数是十万次。NAND 存储器除了具有 10 比 1 的块擦除周期优势，典型的 NAND 块尺寸要比 NOR 器件小 8 倍，每个 NAND 存储器块在给定的时间内的删除次数要少一些。

2) 位交换

所有 flash 器件都受位交换现象的困扰。在某些情况下(很少见，NAND 发生的次数要比 NOR 多)，一个比特位会发生反转或被报告反转了。

一位的变化可能不很明显，但是如果发生在一个关键文件上，这个小小的故障可能导致系统停机。如果只是报告有问题，多读几次就可能解决了。

当然，如果这个位真的改变了，就必须采用错误探测/错误更正(EDC/ECC)算法。位反转的问题更多见于 NAND 闪存，NAND 的供应商建议使用 NAND 闪存的时候，同时使用 EDC/ECC 算法。

这个问题对于用 NAND 存储多媒体信息时倒不是致命的。当然，如果用本地存储设备来存储操作系统、配置文件或其他敏感信息时，必须使用 EDC/ECC 系统以确保可*性。

3）坏块处理

NAND 器件中的坏块是随机分布的。以前也曾有过消除坏块的努力，但发现成品率太低，代价太高，根本不划算。

NAND 器件需要对介质进行初始化扫描以发现坏块，并将坏块标记为不可用。在已制成的器件中，如果通过可*的方法不能进行这项处理，将导致高故障率。

5、易于使用

可以非常直接地使用基于 NOR 的闪存，可以像其他存储器那样连接，并可以在上面直接运行代码。

由于需要 I/O 接口，NAND 要复杂得多。各种 NAND 器件的存取方法因厂家而异。在使用 NAND 器件时，必须先写入驱动程序，才能继续执行其他操作。向 NAND 器件写入信息需要相当的技巧，因为设计师绝不能向坏块写入，这就意味着在 NAND 器件上自始至终都必须进行虚拟映射。

6、软件支持

当讨论软件支持的时候，应该区别基本的读/写/擦操作和高一级的用于磁盘仿真和闪存管理算法的软件，包括性能优化。

在 NOR 器件上运行代码不需要任何的软件支持，在 NAND 器件上进行同样操作时，通常需要驱动程序，也就是内存技术驱动程序(MTD)，NAND 和 NOR 器件在进行写入和擦除操作时都需要 MTD。

使用 NOR 器件时所需要的 MTD 要相对少一些，许多厂商都提供用于 NOR 器件的更高级软件，这其中包括 M-System 的 TrueFFS 驱动，该驱动被 Wind River System、Microsoft、QNX Software System、Symbian 和 Intel 等厂商所采用。

驱动还用于对 DiskOnChip 产品进行仿真和 NAND 闪存的管理，包括纠错、坏块处理和损耗平衡。

2.3 闪存卡

闪存卡（Flash Memory Card）是利用闪存（Flash Memory）技术达到存储电子信息的存储器，一般应用在手机，数码相机，掌上电脑，MP3，MP4，GPS 等小型数码产品中作为存储介质，所以样子小巧，有如一张卡片，所以称之为闪存卡。根据接口标准的不同，闪存卡大概有 SmartMedia（SM 卡）、Compact Flash（CF 卡）、MultiMediaCard（MMC 卡）、Secure Digital（SD 卡）、Memory Stick（记忆棒）、XD-Picture Card（XD 卡）这些闪存卡虽然外观尺寸各异、 但是存储的基本技术原理都是相同的。

尺寸和规格，可以参考下表：

类型	SM卡	CF卡	
		Type I	Type II
长(mm)	45	43	43
宽(mm)	37	36	36
高(mm)	0.76	3.3	5
工作电压	3.3或5V	3.3和5V	3.3和5V
接口	22针	50针 CF Type I	50针 CF Type II

类型	MMC卡	记忆棒	XD卡	SD卡
长(mm)	32	50	25	32
宽(mm)	24	21.5	20	24
高(mm)	1.4	0.28	1.7	2.1
工作电压	2.7～3.6	2.7～3.6		2.7～3.6
接口	7针	10针	18针	9针

2.3.1 SD 卡

SD 卡（Secure Digital Memory Card）是一种基于半导体快闪记忆器的新一代记忆设备。SD 卡由日本松下、东芝及美国 SanDisk 公司于 1999 年 8 月共同开发研制。拥有高记忆容量、快速数据传输率、极大的移动灵活性以及很好的安全性。相比 CF 卡，SD 卡要小得多，它的外形跟邮票差不多大小，所以使用 SD 卡的数码单反更容易实现小型化。目前，SD 卡已经在消费类数码相机取得了绝对的市场占有率（除了索尼和奥林巴斯的数码相机，其他品牌的数码相机都采用 SD 卡作为存储介质），而在数码单反领域，使用 SD 卡的数码单反也越来越多，尤其是小型数码单反都改用 SD 卡作为存储介质。

1、结构

SD 记忆卡具有机械式写入保护开关，以免至关重要的数据被意外丢失。卡两侧的导槽可防止其插反了方向，一个凹口可防止器械掉落或撞击时，卡跳出其插孔。肋条可保护金属触点，以减少静电所引起的损坏可能性，或触碰损坏，如擦伤等。为了与多媒体卡向上兼容，1.4mm 导槽可保护插孔，并可接受 SD 记忆卡或多媒体卡。

2、容量

SD 卡由于其体积较小，所以在同一时期，其最大容量总是落后于 CF 卡。目前，SD 卡的主流容量在 1GB～4GB，而目前市面上销售的最大容量的 SD 卡则达到了 8GB。当然，

8GB 远不是 SD 卡能够达到的容量极限，未来还会出现容量更大的 SD 卡。

3、兼容性

SD 卡分为两种，一种是标准 SD 卡，另一种则是 SDHC 卡。SDHC 中的 HC 是 High Capacity（高容量）的缩写，其主要特点是：采用 FAT32 分区，所以可以制造超过 4GB 容量的存储卡（标准 SD 卡最大容量为 4GB，而 SDHC 卡的容量则从 4GB 开始起步）。SDHC 与标准 SD 卡外形尺寸完全相同，但由于它采用了 FAT32 分区，所以，并不是能使用 SD 卡的数码单反都支持 SDHC 标准（某些较老的支持 SD 卡的数码相机可能只支持较传统 SD 卡使用的 FAT12/16 分区格式。但是，随着 SDHC 的普及，新出品的数码单反都对其提供了良好的支持。SDHC 卡在存储卡的包装和标签上都提供了显著的标识，所以识别起来并不困难。而且，SD 卡与 MMC 卡保持着向上兼容，也就是说，MMC 可以被新的 SD 设备存取，兼容性则取决于应用软件，但 SD 卡却不可以被 MMC 设备存取。（SD 卡外型采用了与 MMC 厚度一样的导轨式设计，以使 SD 设备可以适合 MMC）

另外，手机上使用的 Mini SD 和 Micro SD（又叫 TransFlash，TF 卡）可以视为 SD 卡的一种“变形”，它们都是 SD 卡的一种“微缩”版本，加上各自的适配器即可和 SD 卡保持完全的兼容。

4、速度

SD 卡的速度表示方法跟 CF 卡一样，也是以“倍数”作为衡量标准。而对于 SDHC 卡，还定义了一种新的速度表示方法。SDHC 包括 CLASS2、4、6 三种规格，其中，CLASS2 表示其写入速度不低于 2MB/s，CLASS4 则表示写入速度不低于 4MB/s，以此类推。尽管这个速度看上去并不是非常快，但需要注意的是，这是最低写入速度，而不是最快的写入速度，所以是一种非常保守的表示方法。其中，最高档的 CLASS6 的 SDHC 卡主要针对专业级的数码单反设计，在高速连拍下尤其能发挥威力。

2.3.2 CF 卡

CF 卡（Compact Flash card）是 1994 年由 SanDisk 最先推出的。CF 卡具有 PCMCIA-ATA 功能，并与之兼容；CF 卡重量只有 14g，仅纸板火柴般大小（42.8mm x 36.4mm x 3.3mm），是一种固态存储产品，也就是工作时没有运动部件。CF 卡采用闪存（flash memory）技术，是一种稳定的存储解决方案，不需要电池来维持其中存储的数据。对所保存的数据来说，CF 卡比传统的磁盘驱动器安全性和保护性都更高；比传统的磁盘驱动器及III型 PC 卡的可靠性高 5 到 10 倍，而且 CF 卡的用电量仅为小型磁盘驱动器的 5%。这些优异的条件使得大多数数码相机选择 CF 卡作为其首选存储介质。

虽然最初 CF 卡是采用 Flash Memory 的存贮卡，但随着 CF 卡的发展，各种采用 CF 卡规格的非 Flash Memory 卡也开始出现，CFA（CF 标准协会）后来又发展出了 CF+的规格，使 CF 卡的范围扩展到非 Flash Memory 的其它领域，包括其它 I/O 设备和磁盘存贮器，以及一个更新物理规格的 Type II 规格（IBM 的 Microdrive 就是 Type II 的 CF 卡），Type II 和原来的 Type I 相比不同之处在于 Type II 厚度为 5mm。

CF 卡同时支持 3.3 伏和 5 伏的电压，任何一张 CF 卡都可以在这两种电压下工作，这使得它具有广阔的使用范围。CF 存贮卡的兼容性还表现在它把 Flash Memory 存贮模块与控制器结合在一起，这样使用 CF 卡的外部设备就可以做得比较简单，而且不同的 CF 卡都可以用单一的机构来读写，不用担心兼容性问题，特别是 CF 卡升级换代时也可以保证旧设备的兼容性。

CF 卡有相当多的平台支持，包括 DOS，Windows 3.x，Windows 95，Windows 98，Windows CE，OS/2，Apple System 7，Linux 和许多种 UNIX 都能够支持。

CF 卡作为世界范围内的存储行业标准，保证 CF 产品的兼容，保证 CF 卡的向后兼容性；随着 CF 卡越来越被广泛应用，各厂商积极提高 CF 卡的技术，促进新一代体小质轻、低能耗先进移动设备的推出，进而提高工作效率。CFA 总部在加拿大的 Palo Alto，其成员有权免费得到 CF 卡、CF 商标和 CF 技术详情。CFA 成员包括 3COM，佳能、柯达、惠普、日立、IBM、松下、摩托罗拉、NEC、SanDisk、精工（爱普生）和 Socket Communications 等 120 多个。而且其中的主要数码相机生产研发厂商已经成立了一个专门组织，从事于 CF 产品的开发。

目前主流的 CF 卡都支持 CF4.0 及以上的标准，最高容量做到了 32GB，而且很多都支持 PIO6 和 UDMA6 的传送模式，最高可以达到 300X 的读写速度；这个速度应该是存储卡里面最出色的速度了。

CF 卡有以下缺点：

1）体积较大。与其他种类的存储卡相比，CF 卡的体积略微偏大，这也限制了使用 CF 卡的数码相机体积，所以现下流行的超薄数码相机大多放弃了 CF 卡，而改用体积更为小巧的 SD 卡。

2）功耗较高。相对其他几种规格的存储卡，CF 卡的耗电最大，通常都在 100mA 左右。

第3章　开关电源技术

3.1 开关电源原理

开关电源就是用通过电路控制开关管进行高速的导通与截止．将直流电转化为高频率的交流电提供给变压器进行变压，从而产生所需要的一组或多组电压的电源。

开关电源由以下几个部分组成：

一、主电路

从交流电网输入、直流输出的全过程，包括： 1、输入滤波器：其作用是将电网存在的杂波过滤，同时也阻碍本机产生的杂波反馈到公共电网。 2、整流与滤波：将电网交流电源直接整流为较平滑的直流电，以供下一级变换。 3、逆变：将整流后的直流电变为高频交流电，这是高频开关电源的核心部分，频率越高，体积、重量与输出功率之比越小。4、输出整流与滤波：根据负载需要，提供稳定可靠的直流电源。

二、控制电路

一方面从输出端取样，经与设定标准进行比较，然后去控制逆变器，改变其频率或脉宽，达到输出稳定.

开关电源省去了笨重的变压器。具有体积小，效率高，成本低。广泛用于各种电子设备中的电源。转化为高频交流电的原因是高频交流在变压器变压电路中的效率要比 50Hz 高很多．所以开关变压器可以做的很小，而且工作时不是很热！成本很低．如果不将 50Hz 变为高频那开关电源就没有意义

开关电源大体可以分为隔离和非隔离两种,隔离型的必定有开关变压器,而非隔离的未必一定有。

开关电源的工作原理是:

1.交流电源输入经整流滤波成直流;

2.通过高频 PWM(脉冲宽度调制)信号控制开关管,将那个直流加到开关变压器初级上;

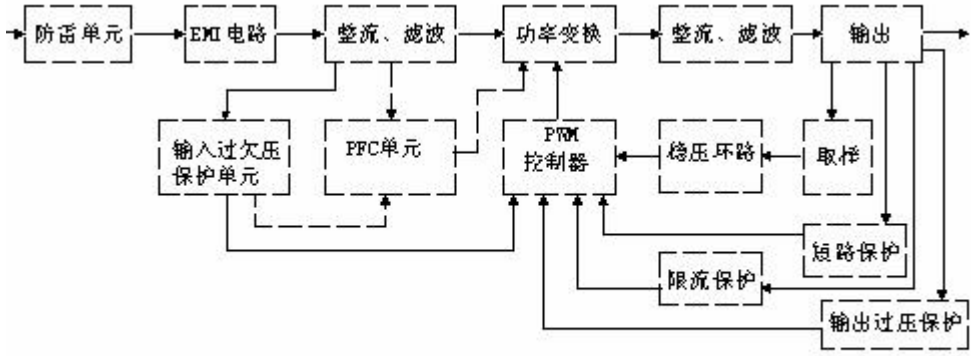
3.开关变压器次级感应出高频电压,经整流滤波供给负载;

4.输出部分通过一定的电路反馈给控制电路,控制 PWM 占空比,以达到稳定输出的目的。

3.2 开关电源的电路组成

开关电源的主要电路是由输入电磁干扰滤波器（EMI）、整流滤波电路、功率变换电路、PWM 控制器电路、输出整流滤波电路组成。辅助电路有输入过欠压保护电路、输出过欠压保护电路、输出过流保护电路、输出短路保护电路等。

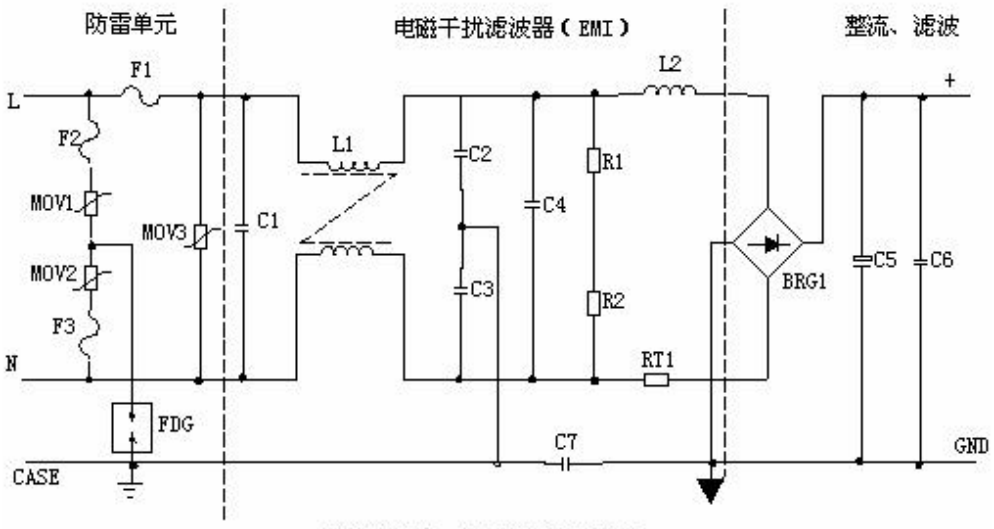
开关电源的电路组成方框图如下：



开关电源电路方框图

3.2.1 输入电路的原理及常见电路

1、AC 输入整流滤波电路原理



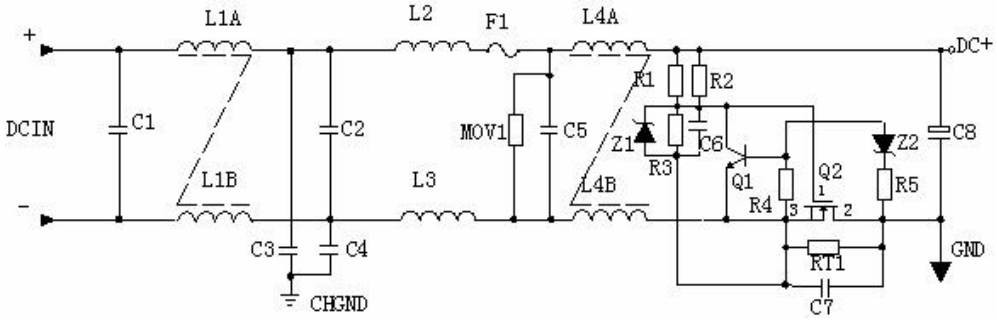
输入滤波、整流回路原理图

① 防雷电路：当有雷击，产生高压经电网导入电源时，由 MOV1、MOV2、MOV3：F1、F2、F3、FDG1 组成的电路进行保护。当加在压敏电阻两端的电压超过其工作电压时，其阻值降低，使高压能量消耗在压敏电阻上，若电流过大，F1、F2、F3 会烧毁保护后级电路。

② 输入滤波电路：C1、L1、C2、C3 组成的双 π 型滤波网络主要是对输入电源的电磁噪声及杂波信号进行抑制，防止对电源干扰，同时也防止电源本身产生的高频杂波对电网干扰。当电源开启瞬间，要对 C5 充电，由于瞬间电流大，加 RT1（热敏电阻）就能有效的防止浪涌电流。因瞬时能量全消耗在 RT1 电阻上，一定时间后温度升高后 RT1 阻值减小（RT1 是负温系数元件），这时它消耗的能量非常小，后级电路可正常工作。

③ 整流滤波电路：交流电压经 BRG1 整流后，经 C5 滤波后得到较为纯净的直流电压。若 C5 容量变小，输出的交流纹波将增大。

2、 DC 输入滤波电路原理



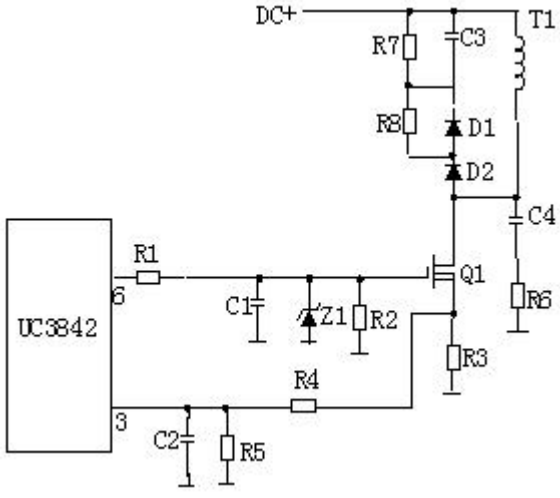
① 输入滤波电路：C1、L1、C2 组成的双 π 型滤波网络主要是对输入电源的电磁噪声及杂波信号进行抑制，防止对电源干扰，同时也防止电源本身产生的高频杂波对电网干扰。C3、C4 为安规电容，L2、L3 为差模电感。

② R1、R2、R3、Z1、C6、Q1、Z2、R4、R5、Q2、RT1、C7 组成抗浪涌电路。在起机的瞬间，由于 C6 的存在 Q2 不导通，电流经 RT1 构成回路。当 C6 上的电压充至 Z1 的稳压值时 Q2 导通。如果 C8 漏电或后级电路短路现象，在起机的瞬间电流在 RT1 上产生的压降增大，Q1 导通使 Q2 没有栅极电压不导通，RT1 将会在很短的时间烧毁，以保护后级电路。

3.2.2 功率变换电路

1、 MOS 管的工作原理：目前应用最广泛的绝缘栅场效应管是 MOSFET（MOS 管），是利用半导体表面的电声效应进行工作的。也称为表面场效应器件。由于它的栅极处于不导电状态，所以输入电阻可以大大提高，最高可达 105 欧姆，MOS 管是利用栅源电压的大小，来改变半导体表面感生电荷的多少，从而控制漏极电流的大小。

2、 常见的原理图：



3、工作原理：

R4、C3、R5、R6、C4、D1、D2 组成缓冲器，和开关 MOS 管并接，使开关管电压应力减少，EMI 减少，不发生二次击穿。在开关管 Q1 关断时，变压器的原边线圈易产生尖峰电压和尖峰电流，这些元件组合一起，能很好地吸收尖峰电压和电流。从 R3 测得的电流峰值信号参与当前工作周波的占空比控制，因此是当前工作周波的电流限制。当 R5 上的电压达到 1V 时，UC3842 停止工作，开关管 Q1 立即关断 。

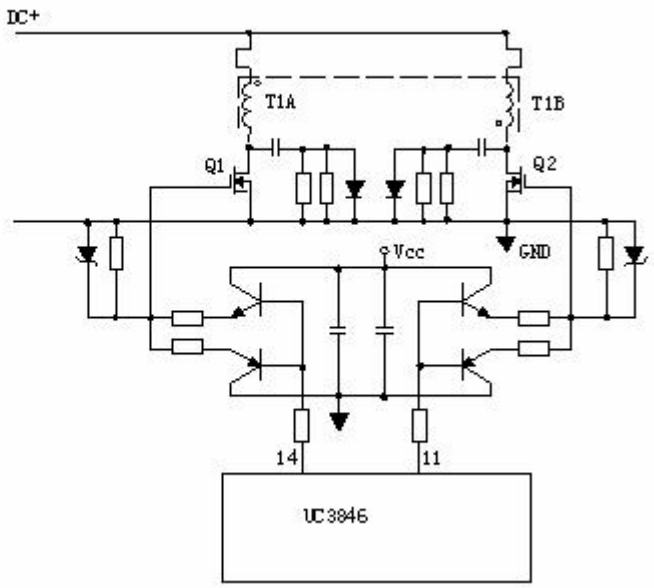
R1 和 Q1 中的结电容 CGS、CGD 一起组成 RC 网络，电容的充放电直接影响着开关管的开关速度。R1 过小，易引起振荡，电磁干扰也会很大；R1 过大，会降低开关管的开关速度。Z1 通常将 MOS 管的 GS 电压限制在 18V 以下，从而保护了 MOS 管。

Q1 的栅极受控电压为锯形波，当其占空比越大时，Q1 导通时间越长，变压器所储存的能量也就越多；当 Q1 截止时，变压器通过 D1、D2、R5、R4、C3 释放能量，同时也达到了磁场复位的目的，为变压器的下一次存储、传递能量做好了准备。IC 根据输出电压和电流时刻调整着⑥脚锯形波占空比的大小，从而稳定了整机的输出电流和电压。

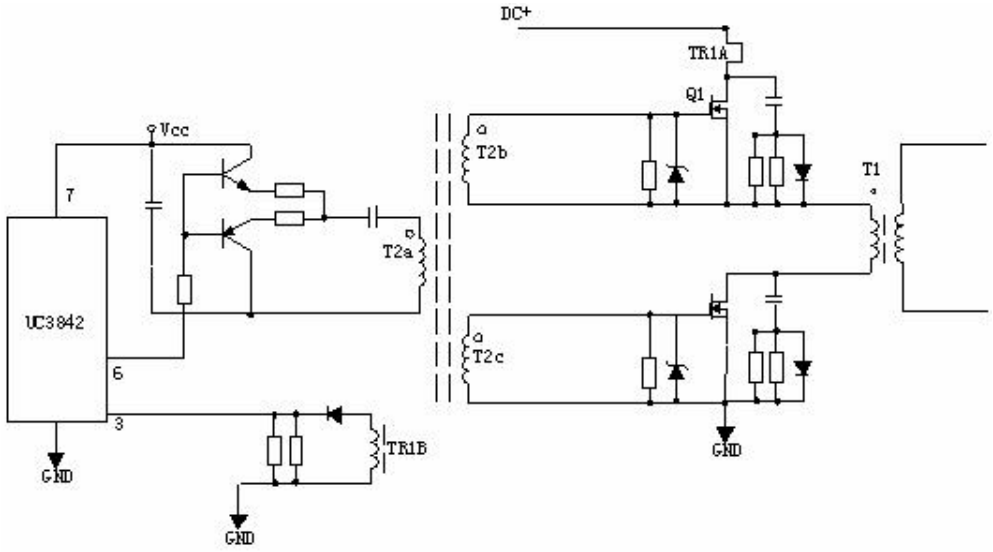
C4 和 R6 为尖峰电压吸收回路。

4、推挽式功率变换电路

Q1 和 Q2 将轮流导通。

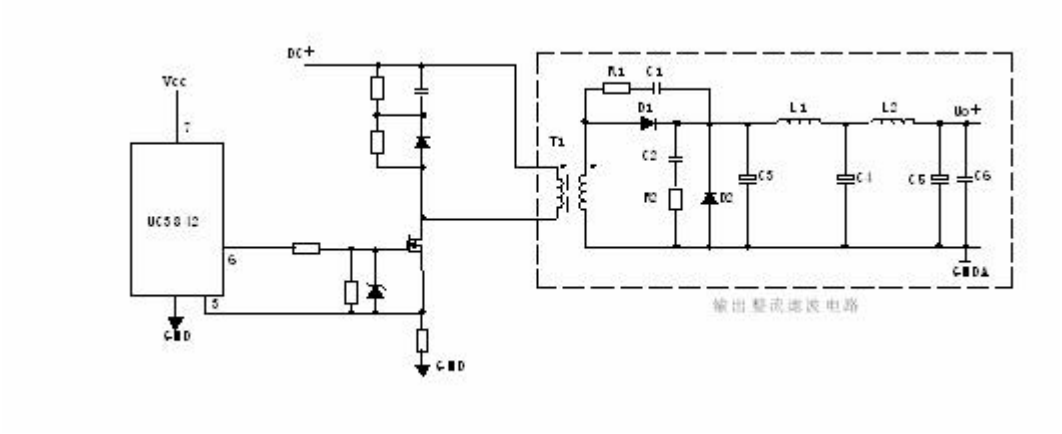


5、有驱动变压器的功率变换电路：T2 为驱动变压器，T1 为开关变压器，TR1 为电流环。



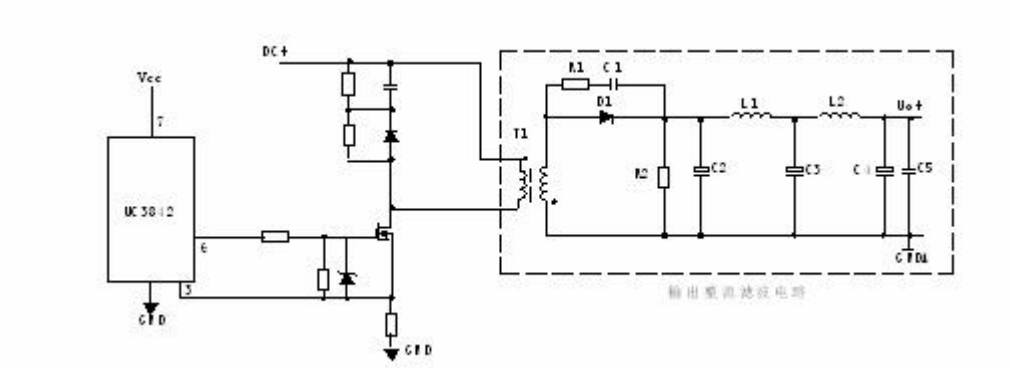
3.2.3 输出整流滤波电路

1、 正激式整流电路



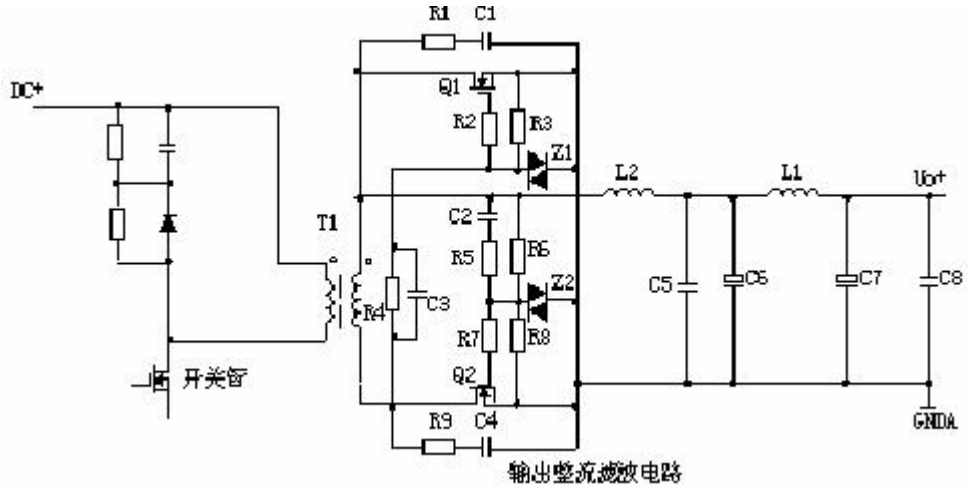
T1 为开关变压器，其初极和次极的相位同相。D1 为整流二极管，D2 为续流二极管，R1、C1、R2、C2 为削尖峰电路。L1 为续流电感，C4、L2、C5 组成 π 型滤波器。

2、 反激式整流电路



T1 为开关变压器，其初极和次极的相位相反。D1 为整流二极管，R1、C1 为削尖峰电路。L1 为续流电感，R2 为假负载，C4、L2、C5 组成 π 型滤波器。

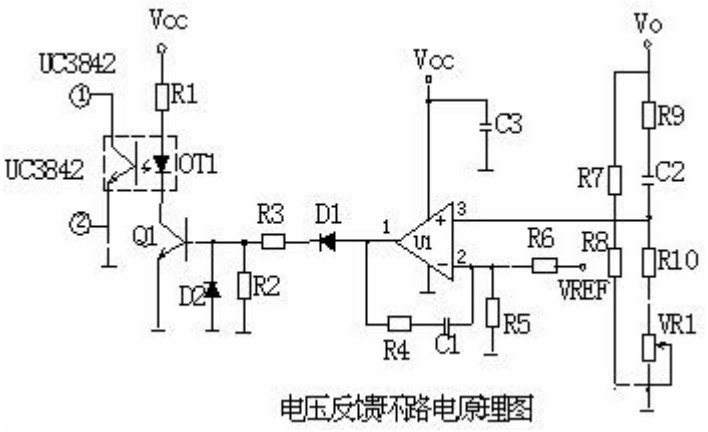
3、 同步整流电路



工作原理：当变压器次级上端为正时，电流经 C2、R5、R6、R7 使 Q2 导通，电路构成回路，Q2 为整流管。Q1 栅极由于处于反偏而截止。当变压器次级下端为正时，电流经 C3、R4、R2 使 Q1 导通，Q1 为续流管。Q2 栅极由于处于反偏而截止。L2 为续流电感，C6、L1、C7 组成 π 型滤波器。R1、C1、R9、C4 为削尖峰电路。

3.2.4 稳压环路原理

1、反馈电路原理图



电压反馈环路电原理图

2、工作原理：

当输出 U0 升高，经取样电阻 R7、R8、R10、VR1 分压后，U1③脚电压升高，当其超过 U1②脚基准电压后 U1①脚输出高电平，使 Q1 导通，光耦 OT1 发光二极管发光，光电三极管导通，UC3842①脚电位相应变低，从而改变 U1⑥脚输出占空比减小，U0 降低。

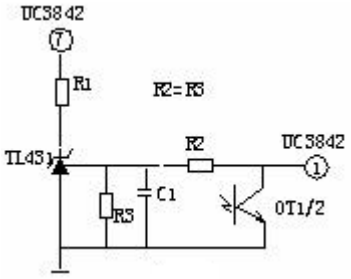
当输出 U0 降低时，U1③脚电压降低，当其低过 U1②脚基准电压后 U1①脚输出低电平，Q1 不导通，光耦 OT1 发光二极管不发光，光电三极管不导通，UC3842①脚电位升高，从而改变 U1⑥脚输出占空比增大，U0 降低。周而复始，从而使输出电压保持稳定。调节 VR1 可改变输出电压值。

反馈环路是影响开关电源稳定性的重要电路。如反馈电阻电容错、漏、虚焊等，会产生自激振荡，故障现象为：波形异常，空、满载振荡，输出电压不稳定等。

3.2.5 短路保护电路

1、在输出端短路的情况下，PWM 控制电路能够把输出电流限制在一个安全范围内，它可以用多种方法来实现限流电路，当功率限流在短路时不起作用时，只有另增设一部分电路。

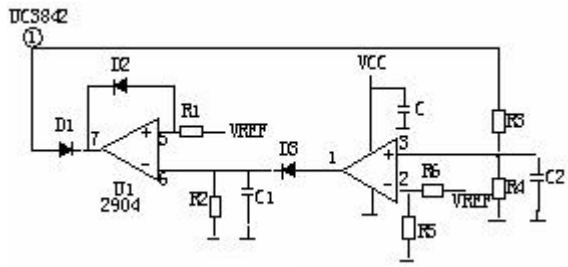
2、短路保护电路通常有两种，下图是小功率短路保护电路，其原理简述如下：



当输出电路短路，输出电压消失，光耦 OT1 不导通，UC3842①脚电压上升至 5V 左右，R1 与 R2 的分压超过 TL431 基准，使之导通，UC3842⑦脚 VCC 电位被拉低，IC 停止工作。UC3842 停止工作后①脚电位消失，TL431 不导通 UC3842⑦脚电位上升，UC3842

重新启动，周而复始。当短路现象消失后，电路可以自动恢复成正常工作状态。

3、下图是中功率短路保护电路，其原理简述如下：



当输出短路，UC3842①脚电压上升,U1 ③脚电位高于②脚时，比较器翻转①脚输出高电位，给 C1 充电，当 C1 两端电压超过⑤脚基准电压时 U1⑦脚输出低电位，C3842①脚低于 1V，UCC3842 停止工作，输出电压为 0V，周而复始，当短路消失后电路正常工作。R2、C1 是充放电时间常数，阻值不对时短路保护不起作用。

4、 下图是常见的限流、短路保护电路。其工作原理简述如下：

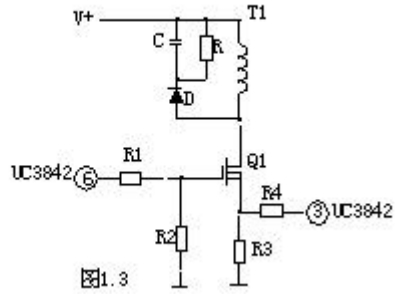
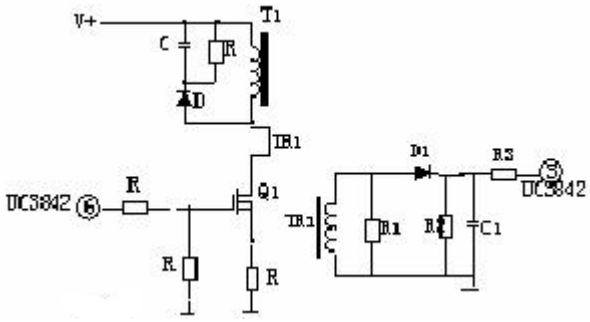


图1.3

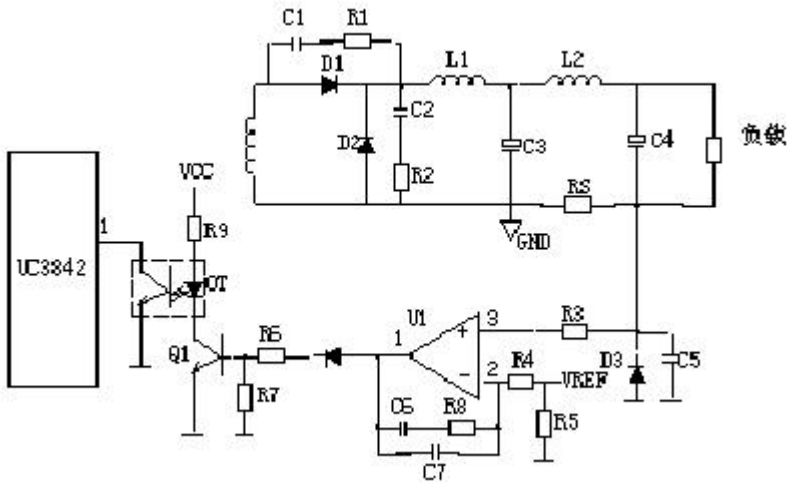
当输出电路短路或过流，变压器原边电流增大，R3 两端电压降增大，③脚电压升高，UC3842⑥脚输出占空比逐渐增大，③脚电压超过 1V 时，UC3842 关闭无输出。

5、下图是用电流互感器取样电流的保护电路，有着功耗小，但成本高和电路较为复杂，其工作原理简述如下：



输出电路短路或电流过大，TR1 次级线圈感应的电压就越高，当 UC3842③脚超过 1 伏，UC3842 停止工作，周而复始，当短路或过载消失，电路自行恢复。

3.2.6 输出端限流保护

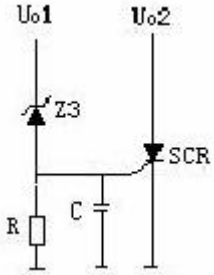


上图是常见的输出端限流保护电路，其工作原理简述如上图：当输出电流过大时，RS（锰铜丝）两端电压上升，U1③脚电压高于②脚基准电压，U1①脚输出高电压，Q1 导通，光耦发生光电效应，UC3842①脚电压降低，输出电压降低，从而达到输出过载限流的目的。

3.2.7 输出过压保护电路的原理

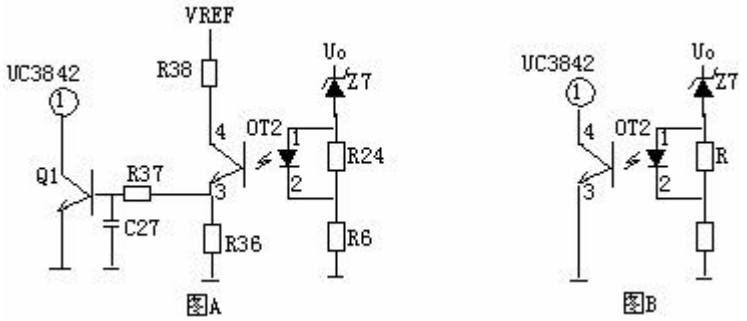
输出过压保护电路的作用是：当输出电压超过设计值时，把输出电压限定在一安全值的范围内。当开关电源内部稳压环路出现故障或者由于用户操作不当引起输出过压现象时，过压保护电路进行保护以防止损坏后级用电设备。应用最为普遍的过压保护电路有如下几种：

1、可控硅触发保护电路



如上图，当 Uo1 输出升高，稳压管（Z3）击穿导通，可控硅（SCR1）的控制端得到触发电压，因此可控硅导通。Uo2 电压对地短路，过流保护电路或短路保护电路就会工作，停止整个电源电路的工作。当输出过压现象排除，可控硅的控制端触发电压通过 R 对地泄放，可控硅恢复断开状态。

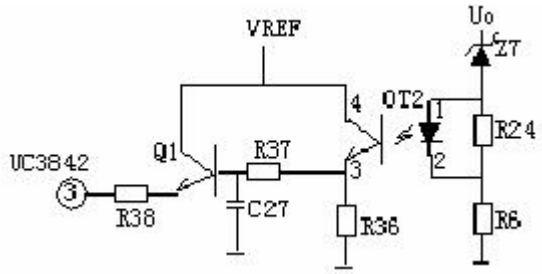
2、光电耦合保护电路



如上图，当 Uo 有过压现象时，稳压管击穿导通，经光耦（OT2）R6 到地产生电流流过，光电耦合器的发光二极管发光，从而使光电耦合器的光敏三极管导通。Q1 基极得电导通，3842 的③脚电降低，使 IC 关闭，停止整个电源的工作，Uo 为零，周而复始。

3、输出限压保护电路

输出限压保护电路如下图，当输出电压升高，稳压管导通光耦导通，Q1 基极有驱动电压而道通，UC3842③电压升高，输出降低，稳压管不导通，UC3842③电压降低，输出电压升高。周而复始，输出电压将稳定在一范围内（取决于稳压管的稳压值）。



4、输出过压锁死电路

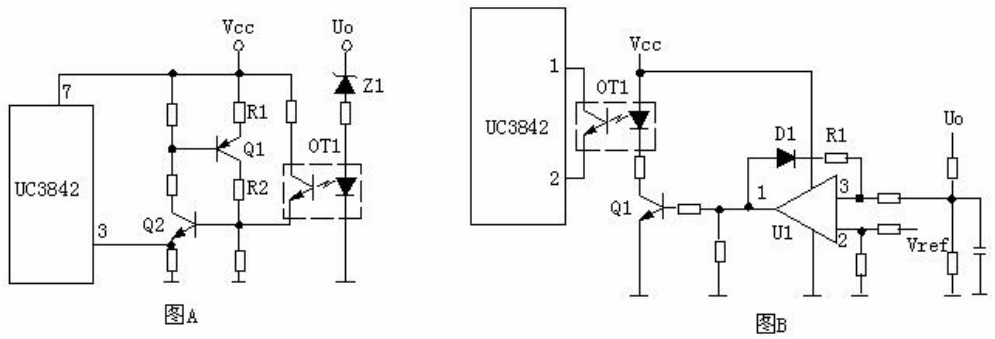
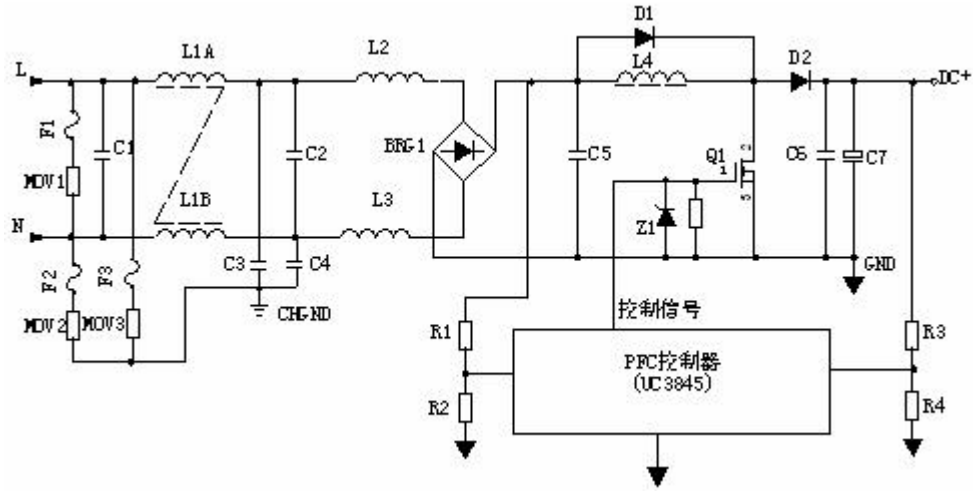


图 A 的工作原理是，当输出电压 Uo 升高，稳压管导通，光耦导通，Q2 基极得电导通，由于 Q2 的导通 Q1 基极电压降低也导通，Vcc 电压经 R1、Q1、R2 使 Q2 始终导通，UC3842③脚始终是高电平而停止工作。在图 B 中，UO 升高 U1③脚电压升高，①脚输出高电平，由于 D1、R1 的存在，U1①脚始终输出高电平 Q1 始终导通，UC3842①脚始终是低电平而停止工作。

3.2.8 功率因数校正电路

1、原理示意图：

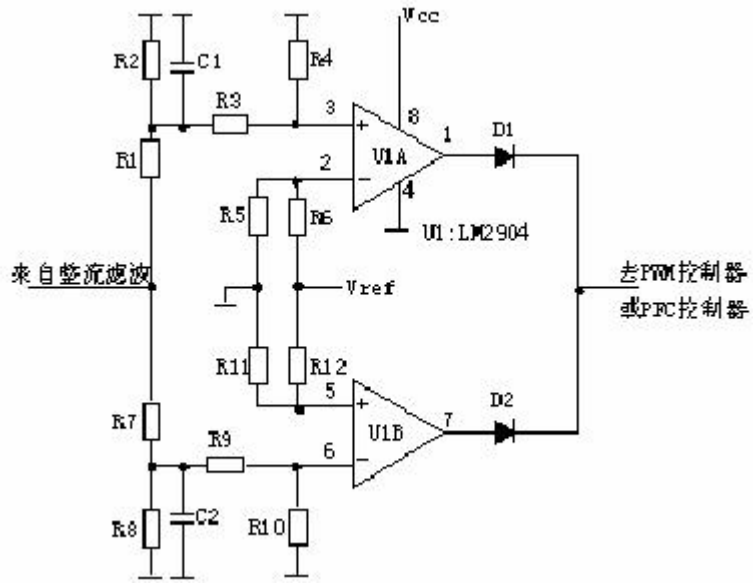


2、工作原理：

输入电压经 L1、L2、L3 等组成的 EMI 滤波器，BRG1 整流一路送 PFC 电感，另一路经 R1、R2 分压后送入 PFC 控制器作为输入电压的取样，用以调整控制信号的占空比，即改变 Q1 的导通和关断时间，稳定 PFC 输出电压。L4 是 PFC 电感，它在 Q1 导通时储存能量，在 Q1 关断时施放能量。D1 是启动二极管。D2 是 PFC 整流二极管，C6、C7 滤波。PFC 电压一路送后级电路，另一路经 R3、R4 分压后送入 PFC 控制器作为 PFC 输出电压的取样，用以调整控制信号的占空比，稳定 PFC 输出电压。

3.2.9 输入过欠压保护

1、 原理图：



2、 工作原理：

AC 输入和 DC 输入的开关电源的输入过欠压保护原理大致相同。保护电路的取样电压均来自输入滤波后的电压。

取样电压分为两路，一路经 R1、R2、R3、R4 分压后输入比较器 3 脚，如取样电压高于 2 脚基准电压，比较器 1 脚输出高电平去控制主控制器使其关断，电源无输出。另一路经 R7、R8、R9、R10 分压后输入比较器 6 脚，如取样电压低于 5 脚基准电压，比较器 7 脚输出高电平去控制主控制器使其关断，电源无输出。

第 4 章　总线技术

任何一个微处理器都要与一定数量的部件和外围设备连接，但如果将各部件和每一种外围设备都分别用一组线路与 CPU 直接连接，那么连线将会错综复杂，甚至难以实现。为了简化硬件电路设计、简化系统结构，常用一组线路，配置以适当的接口电路，与各部件和外围设备连接，这组共用的连接线路被称为总线。采用总线结构便于部件和设备的扩充，尤其制定了统一的总线标准则容易使不同设备间实现互连。

按总线的功能分类：地址总线、数据总线、控制总线。

通常所说的总线都包括上述三个组成部分：地址总线（AB）用来传送地址信息，数据总线（DB）用来传送数据信息，控制总线（CB）用来传送各种控制信号。

微机中总线一般有内部总线、系统总线和外部总线。内部总线是微机内部各外围芯片与处理器之间的总线，用于芯片一级的互连；而系统总线是微机中各插件板与系统板之间的总线，用于插件板一级的互连；外部总线则是微机和外部设备之间的总线，微机作为一种设备，通过该总线和其他设备进行信息与数据交换，它用于设备一级的互连。

另外，从广义上说，计算机通信方式可以分为并行通信和串行通信，相应的通信总线被称为并行总线和串行总线。并行通信速度快、实时性好，但由于占用的口线多，不适于小型化产品；而串行通信速率虽低，但在数据通信吞吐量不是很大的微处理电路中则显得更加简易、方便、灵活。串行通信一般可分为异步模式和同步模式。

随着微电子技术和计算机技术的发展，总线技术也在不断地发展和完善，而使计算机总线技术种类繁多，各具特色。下面仅对微机各类总线中目前比较流行的总线技术分别加以介绍。

4.1 内部总线

1. I2C 总线

I2C（Inter-IC）总线 10 多年前由 Philips 公司推出，是近年来在微电子通信控制领域广泛采用的一种新型总线标准。它是同步通信的一种特殊形式，具有接口线少，控制方式简化，器件封装形式小，通信速率较高等优点。在主从通信中，可以有多个 I2C 总线器件同时接到 I2C 总线上，通过地址来识别通信对象。

2. SPI 总线

串行外围设备接口 SPI（serial peripheral interface）总线技术是 Motorola 公司推出的一种同步串行接口。Motorola 公司生产的绝大多数 MCU（微控制器）都配有 SPI 硬件接口，如 68 系列 MCU。SPI 总线是一种三线同步总线，因其硬件功能很强，所以，与 SPI 有关的软件就相当简单，使 CPU 有更多的时间处理其他事务。

3. SCI 总线

串行通信接口 SCI（serial communication interface）也是由 Motorola 公司推出的。

它是一种通用异步通信接口 UART，与 MCS-51 的异步通信功能基本相同。

4. IIS 总线

IIS(Inter-ICSoundbus)又称 I2S，是菲利浦公司提出的串行数字音频总线协议。目前很多音频芯片和 MCU 都提供了对 IIS 的支持。

IIS 总线只处理声音数据。其他信号(如控制信号)必须单独传输。为了使芯片的引出管脚尽可能少，IIS 只使用了三根串行总线。这三根线分别是：提供分时复用功能的数据线、字段选择线(声道选择)、时钟信号线。

5.单总线

近年来，美国的达拉斯半导体公司（DALLASSEMICONDUCTOR）推出了一项特有的单总线（1－Wire Bus）技术。该技术与上述总线不同，它采用单根信号线，既可传输时钟，又能传输数据，而且数据传输是双向的，因而这种单总线技术具有线路简单，硬件开销少，成本低廉，便于总线扩展和维护等优点。

单总线适用于单主机系统，能够控制一个或多个从机设备。主机可以是微控制器，从机可以是单总线器件，它们之间的数据交换只通过一条信号线。当只有一个从机设备时，系统可按单节点系统操作；当有多个从机设备时，系统则按多节点系统操作。

4.2 系统总线

1. ISA 总线

ISA（industrial standard architecture）总线标准是 IBM 公司 1984 年为推出 PC/AT 机而建立的系统总线标准，所以也叫 AT 总线。它是对 XT 总线的扩展，以适应 8/16 位数据总线要求。它在 80286 至 80486 时代应用非常广泛，以至于现在奔腾机中还保留有 ISA 总线插槽。ISA 总线有 98 只引脚。

2. EISA 总线

EISA 总线是 1988 年由 Compaq 等 9 家公司联合推出的总线标准。它是在 ISA 总线的基础上使用双层插座，在原来 ISA 总线的 98 条信号线上又增加了 98 条信号线，也就是在两条 ISA 信号线之间添加一条 EISA 信号线。在实用中，EISA 总线完全兼容 ISA 总线信号。

3. VESA 总线

VESA（video electronics standard association）总线是 1992 年由 60 家附件卡制造商联合推出的一种局部总线，简称为 VL(VESA local bus)总线。它的推出为微机系统总线体系结构的革新奠定了基础。该总线系统考虑到 CPU 与主存和 Cache 的直接相连，通常把这部分总线称为 CPU 总线或主总线，其他设备通过 VL 总线与 CPU 总线相连，所以 VL 总线被称为局部总线。它定义了 32 位数据线，且可通过扩展槽扩展到 64 位，使用 33MHz 时钟频率，最大传输率达 132MB/s，可与 CPU 同步工作。是一种高速、高效的局部总线，

可支持 386SX、386DX、486SX、486DX 及奔腾微处理器。

4. PCI 总线

PCI（peripheral component interconnect）总线是当前最流行的总线之一，它是由 Intel 公司推出的一种局部总线。它定义了 32 位数据总线，且可扩展为 64 位。PCI 总线主板插槽的体积比原 ISA 总线插槽还小，其功能比 VESA、ISA 有极大的改善，支持突发读写操作，最大传输速率可达 132MB/s，可同时支持多组外围设备。 PCI 局部总线不能兼容现有的 ISA、EISA、MCA（micro channel architecture）总线，但它不受制于处理器，是基于奔腾等新一代微处理器而发展的总线。

5. Compact PCI

以上所列举的几种系统总线一般都用于商用 PC 机中，在计算机系统总线中，还有另一大类为适应工业现场环境而设计的系统总线，比如 STD 总线、 VME 总线、PC/104 总线等。这里仅介绍当前工业计算机的热门总线之一——Compact PCI。

Compact PCI 的意思是“坚实的 PCI”，是当今第一个采用无源总线底板结构的 PCI 系统，是 PCI 总线的电气和软件标准加欧式卡的工业组装标准，是当今最新的一种工业计算机标准。Compact PCI 是在原来 PCI 总线基础上改造而来，它利用 PCI 的优点，提供满足工业环境应用要求的高性能核心系统，同时还考虑充分利用传统的总线产品，如 ISA、STD、VME 或 PC/104 来扩充系统的 I/O 和其他功能。

4.3 外部总线

1. RS-232-C 总线

RS-232-C 是美国电子工业协会 EIA（Electronic Industry Association）制定的一种串行物理接口标准。RS 是英文“推荐标准”的缩写，232 为标识号，C 表示修改次数。RS-232-C 总线标准设有 25 条信号线，包括一个主通道和一个辅助通道，在多数情况下主要使用主通道，对于一般双工通信，仅需几条信号线就可实现，如一条发送线、一条接收线及一条地线。RS-232-C 标准规定的数据传输速率为每秒 50、75、 100、150、300、600、1200、2400、4800、9600、19200 波特。RS-232-C 标准规定，驱动器允许有 2500pF 的电容负载，通信距离将受此电容限制，例如，采用 150pF/m 的通信电缆时，最大通信距离为 15m；若每米电缆的电容量减小，通信距离可以增加。传输距离短的另一原因是 RS-232 属单端信号传送，存在共地噪声和不能抑制共模干扰等问题，因此一般用于 20m 以内的通信。

2. RS-485 总线

在要求通信距离为几十米到上千米时，广泛采用 RS-485 串行总线标准。RS-485 采用平衡发送和差分接收，因此具有抑制共模干扰的能力。加上总线收发器具有高灵敏度，能检测低至 200mV 的电压，故传输信号能在千米以外得到恢复。 RS-485 采用半双工工作方式，任何时候只能有一点处于发送状态，因此，发送电路须由使能信号加以控制。RS-485 用于多点互连时非常方便，可以省掉许多信号线。应用 RS-485 可以联网构成分布式系统，其允许最多并联 32 台驱动器和 32 台接收器。

3. IEEE-488 总线

上述两种外部总线是串行总线，而 IEEE-488 总线是并行总线接口标准。IEEE-488 总线用来连接系统，如微计算机、数字电压表、数码显示器等设备及其他仪器仪表均可用 IEEE-488 总线装配起来。它按照位并行、字节串行双向异步方式传输信号，连接方式为总线方式，仪器设备直接并联于总线上而不需中介单元，但总线上最多可连接 15 台设备。最大传输距离为 20 米，信号传输速度一般为 500KB/s，最大传输速度为 1MB/s。

4. USB 总线

USB 是英文 UniversalSerialBus 的缩写，中文含义是“通用串行总线”。它是一种应用在 PC 领域的新型接口技术。自从 1995 年 PC 机带有 USB 接口，1998 年 USB 接口逐步走进大规模实用阶段。

这几年，随着大量支持 USB 的个人电脑的普及，USB 逐步成为 PC 机的标准接口已经是大势所趋。在主机(host)端，最新推出的 PC 机几乎 100%支持 USB；而在外设(device)端，使用 USB 接口的设备也与日俱增，例如数码相机、扫描仪、游戏杆、磁带和软驱、图像设备、打印机、键盘、鼠标等等。

USB 设备之所以会被大量应用，主要具有以下优点：

1）可以热插拔，告别“并口和串口先关机，将电缆接上，再开机”的动作。

2）系统总线供电，低功率设备无需外接电源，采用低功耗设备，并可提供 5V/500mA 电源。

3）支持设备众多，支持多种设备类，例如鼠标，键盘，打印机等。

4）扩展容易，可以连接多个设备，最多可扩 127 个。

5）高速数据传输，USB1.1 是 12Mb/s，USB2.0 高达 480Mb/S。

6）方便的设备互连，USBOTG 支持点对点通信，例如数码相机和打印机直接互连，无需 PC。

当然，USB 设备也有其缺点，包括：

1）供电能力，如果外设的供电电流大于 500mA 时，设备必须外接电源。

2）传输距离，USB 总线的连线长度最大为 5m，即便是用 HUB 来扩展，最远也不超过 30 米。

4.4 CAN 总线

4.4.1 ＣＡＮ总线简介及其特点

ＣＡＮ网络（Ｃｏｎｔｒｏｌｌｅｒ Ａｒｅａ Ｎｅｔｗｏｒｋ）是现场总线技术的一种，它是一种架构开放、广播式的新一代网络通信协议，称为控制器局域网现场总线。ＣＡＮ网络原本是德国Ｂｏｓｃｈ公司为欧洲汽车市场所开发的。ＣＡＮ推出之初是用于汽车内部测量和执行部件之间的数据通信。例如汽车刹车防抱死系统、安全气囊等。对机动车辆总线和对现场总线的需求有许多相似之处，即能够以较低的成本、较高的实时处理

能力在强电磁干扰环境下可靠地工作。因此ＣＡＮ总线可广泛应用于离散控制领域中的过程监测和控制，特别是工业自动化的底层监控，以解决控制与测试之间的可靠和实时数据交换。

ＣＡＮ总线有如下基本特点：

＊ ＣＡＮ协议最大的特点是废除了传统的站地址编码，代之以对数据通信数据块进行编码，可以多主 方式工作；

＊ ＣＡＮ采用非破坏性仲裁技术，当两个节点同时向网络上传送数据时，优先级低的节点主动停止数据发送，而优先级高的节点可不受影响地继续传输数据，有效避免了总线冲突；

＊ ＣＡＮ采用短帧结构，每一帧的有效字节数为８个（ＣＡＮ技术规范２．０Ａ），数据传输时间短，受干扰的概率低，重新发送的时间短；

＊ ＣＡＮ的每帧数据都有ＣＲＣ效验及其他检错措施，保证了数据传输的高可靠性，适于在高干扰环境中使用；

＊ ＣＡＮ节点在错误严重的情况下，具有自动关闭总线的功能，切断它与总线的联系，以使总线上其它操作不受影响；

＊ ＣＡＮ可以点对点、一点对多点（成组）及全局广播集中方式传送和接受数据；

＊ ＣＡＮ总线直接通讯距离最远可达１０ｋｍ／５Ｋｂｐｓ，通讯速率最高可达１Ｍｂｐｓ／４０ｍ；

＊ 采用不归零码（ＮＲＺ—Ｎｏｎ－Ｒｅｔｕｒｎ－ｔｏ－Ｚｅｒｏ）编码／解码方式，并采用位填充（插入）技术。

4.4.2 ＣＡＮ总线通信介质访问控制方式

ＣＡＮ采用了的３层模型：物理层、数据链路层和应用层。ＣＡＮ支持的拓扑结构为总线型。传输介质为双绞线、同轴电缆和光纤等。采用双绞线通信时，速率为１Ｍｂｐｓ／４０ ｍ，５０Ｋｂｐｓ／１０ｋｍ,结点数可达１１０个。

ＣＡＮ的通信介质访问为带有优先级的ＣＳ－ＭＡ／ＣＡ。采用多主竞争方式结构：网络上任意节点均可以在任意时刻主动地向网络上其它节点发送信息，而不分主从，即当发现总线空闲时，各个节点都有权使用网络。在发生冲突时，采用非破坏性总线优先仲裁技术：当几个节点同时向网络发送消息时，运用逐位仲裁原则，借助帧中开始部分的表示符，优先级低的节点主动停止发送数据，而优先级高的节点可不受影响的继续发送信息，从而有效地避免了总线冲突，使信息和时间均无损失。例如，规定０的优先级高，在节点发送信息时，ＣＡＮ总线作与运算。每个节点都是边发送信息边检测网络状态，当某一个节点发送１而检测到０时，此节点知道有更高优先级的信息在发送，它就停止发送信息，直到再一次检测到网络空闲。

ＣＡＮ的传输信号采用短帧结构（有效数据最多为８个字节），和带优先级的ＣＳ－ＭＡ／ＣＡ通信介质访问控制方式，对高优先级的通信请求来说，在１Ｍｂｐｓ通信速率时，最长的等待时间为０．１５ｍｓ，完全可以满足现场控制的实时性要求。ＣＡＮ突出的差错检验机理，如５种错误检测、出错标定和故障界定；ＣＡＮ传输信号为短帧结构，

因而传输时间短，受干扰概率低。这些保证了出错率极低，剩余错误概率为报文出错率的４。７×１０－１１。另外，ＣＡＮ节点在严重错误的情况下，具有自动关闭输出的功能，以使总线上其它节点的操作不受其影响。因此，ＣＡＮ具有高可靠性。

ＣＡＮ的通信协议主要有ＣＡＮ总线控制器完成。ＣＡＮ控制器主要由实现ＣＡＮ总线协议部分和微控制器接口部分电路组成。通过简单的连接即可完成ＣＡＮ协议的物理层和数据链路层的所有功能，应用层功能由微控制器完成。ＣＡＮ总线上的节点即可以是基于微控制器的智能节点，也可以是具有ＣＡＮ接口的Ｉ／Ｏ器件。

4.4.3 应用技术

１、 系统组成

ＣＡＮ总线用户接口简单，编程方便。ＣＡＮ总线属于现场总线的范畴，ＣＡＮ总线系统的一般组成模式如图 4-1 所示。

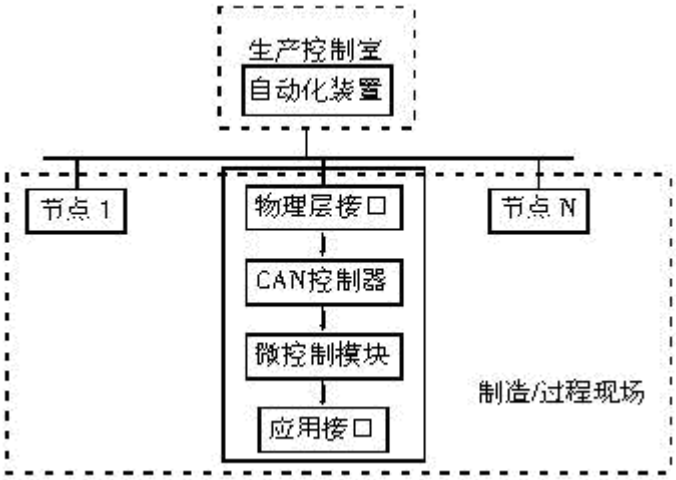


图 4-1　ＣＡＮ总线系统的一般组成模式

网络拓扑结构采用总线式结构。这种网络结构结构简单、成本低，并且采用无源抽头连接，系统可靠性高。通过ＣＡＮ总线连接各个网络节点，形成多主机控制器局域网（ＣＡＮ）。信息的传输采用ＣＡＮ通信协议，通过ＣＡＮ控制器来完成。各网络节点一般为带有微控制器的智能节点完成现场的数据采集和基于ＣＡＮ协议的数据传输，节点可以使用带有在片ＣＡＮ控制器的微控制器，或选用一般的微控制器加上独立的ＣＡＮ控制器来完成节点功能。传输介质可采用双绞线、同轴电缆或光纤。如果需要进一步提高系统的抗干扰能力，还可以在控制器和传输介质之间加接光电隔离，电源采用ＤＣ－ＤＣ变换器等措施。这样可方便构成实时分布式测控系统。

２、 ＣＡＮ总线的物理层设计

ＣＡＮ总线协议对物理层没有严格定义，给使用者较大的灵活性，同时也给设计者带来了困难。ＣＡＮ总线物理层的设计原则是：针对ＣＴＸ０、ＣＴＸ１的两种输出状态（显性（Ｄａｍｉｎａｎｔ）、隐性（Ｒｅｃｅｓｓｉｖｅ）），总线应具有两种不同电平，接

收端呈现（显性、隐性）两种状态，如图 4- 2 所示。

这样不要求总线必须是数字逻辑电平，只要是能够呈现两种电平（显性和隐性）的模拟量，满足上述设计原则就可以。

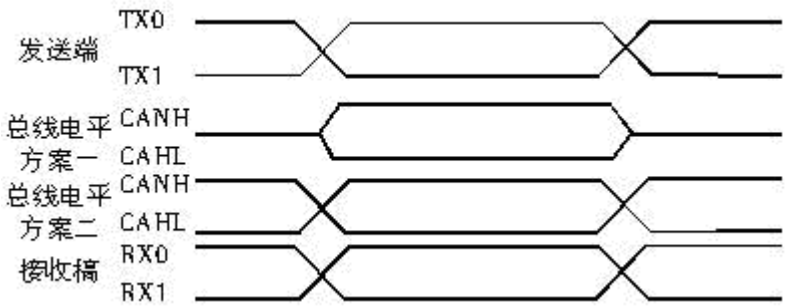


图 4-2 CAN 总线电平示意图

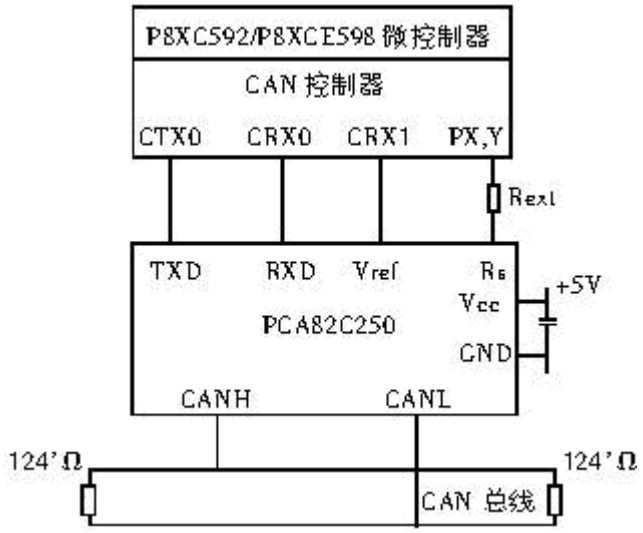


图 4-3 CAN 总线物理接口示意

ＣＡＮ控制器芯片的片内输出驱动器和输入比较器可编程，它可方便地提供多种发送类型，诸如：单线总线、双线总线（差分）和光缆总线。它可以直接驱动总线，若网络的规模比较大，节点数比较多，需要外加总线驱动元件，以增大输出电流。如图 4-3 采用了ＣＡＮ收发器作为ＣＡＮ控制器和物理总线之间的接口，提供向总线的差动发送能力和对ＣＡＮ控制器的差动接收能力。

一般在驱动芯片和ＣＡＮ控制器之间加入光电耦合器,增加抗干扰能力。ＣＡＮ总线的速度将由光电耦合器的速度决定。比如：用４Ｎ２７光耦，因为它的响应速度比较慢，ＣＡＮ网络的位速度只能达到几十Ｋｂｉｔ／ｓ。如果采用６Ｎ１３７高速光电耦合器，ＣＡＮ网络速度可以达到和电阻网络驱动时的速度一样。另外，物理层的设计要注意电缆的终端阻抗匹配，这直接影响了ＣＡＮ总线能否正常工作和网络性能。

3 、 应用软件设计

ＣＡＮ控制器其内部硬件实现了ＣＡＮ总线物理层和数据链路层的所有协议内容，有关ＣＡＮ总线的通信功能均由ＣＡＮ控制器自动管理执行。ＣＡＮ控制器对于ＣＰＵ来说，是以确保双方独立工作的存储影像外围设备出现的。ＣＡＮ控制器的地址域由控制段和报文缓存器组成，在初始化向下加载期间，控制段可被编程以配置通信参数。ＣＡＮ总线上的通信也通过此段由ＣＰＵ控制，被发送的报文必须写入发送缓存器，成功接收后，ＣＰＵ可以从接收缓存器读取报文，然后释放它，以备下次使用。对于在片的ＣＡＮ控制器，它与ＣＰＵ之间的接口一般借助于４个特殊寄存器：ＣＡＮ地址寄存器、数据寄存器、控制寄存器、状态寄存器。对于单独的ＣＡＮ控制器，ＭＣＵ可以通过其地址／数据总线对其寄存器直接寻址，就像ＭＣＵ对一般外部ＲＡＭ寻址一样。通过对这些寄存器编程操作，可很方便控制ＣＡＮ控制器完成通讯功能。

4.5 以太网

以太网是当今现有局域网采用的最通用的通信协议标准，组建于七十年代早期。Ethernet(以太网）是一种传输速率为 10Mbps 的常用局域网（LAN）标准。在以太网中，所有计算机被连接一条同轴电缆上，采用具有冲突检测的载波感应多处访问（CSMA/CD）方法，采用竞争机制和总线拓朴结构。基本上，以太网由共享传输媒体，如双绞线电缆或同轴电缆和多端口集线器、网桥或交换机构成。在星型或总线型配置结构中，集线器/交换机/网桥通过电缆使得计算机、打印机和工作站彼此之间相互连接。

以太网具有的一般特征概述如下：

共享媒体：所有网络设备依次使用同一通信媒体。

广播域：需要传输的帧被发送到所有节点，但只有寻址到的节点才会接收到帧

CSMA/CD：以太网中利用载波监听多路访问/冲突检测方法（Carrier Sense Multiple Access/Collision Detection）以防止 twp 或更多节点同时发送。

MAC 地址：媒体访问控制层的所有 Ethernet 网络接口卡（NIC）都采用 48 位网络地址。这种地址全球唯一。

Ethernet 基本网络组成：

共享媒体和电缆：10BaseT（双绞线），10Base-2（同轴细缆），10Base-5（同轴粗缆）。

转发器或集线器：集线器或转发器是用来接收网络设备上的大量以太网连接的一类设备。通过某个连接的接收双方获得的数据被重新使用并发送到传输双方中所有连接设备上，以获得传输型设备。

网桥：网桥属于第二层设备，负责将网络划分为独立的冲突域获分段，达到能在同一个域/分段中维持广播及共享的目标。网桥中包括一份涵盖所有分段和转发帧的表格，以确保分段内及其周围的通信行为正常进行。

交换机：交换机，与网桥相同，也属于第二层设备，且是一种多端口设备。交换机所支持的功能类似于网桥，但它比网桥更具有的优势是，它可以临时将任意两个端口连接在

一起。交换机包括一个交换矩阵，通过它可以迅速连接端口或解除端口连接。与集线器不同，交换机只转发从一个端口到其它连接目标节点且不包含广播的端口的帧。

以太网协议：IEEE 802.3 标准中提供了以太帧结构。当前以太网支持光纤和双绞线媒体支持下的四种传输速率：

10 Mbps – 10Base-T Ethernet（802.3）

100 Mbps – Fast Ethernet（802.3u）

1000 Mbps – Gigabit Ethernet（802.3z)）

10 Gigabit Ethernet – IEEE 802.3ae

4.6 无线通信技术

目前使用较广泛的近距无线通信技术是蓝牙(Bluetooth)，无线局域网 802.11(Wi-Fi)和红外数据传输(IrDA)。同时还有一些具有发展潜力的近距无线技术标准，它们分别是：ZigBee、超宽频(Ultra WideBand)、短距通信(NFC)、WiMedia、GPS、DECT、无线 1394 和专用无线系统等。它们都有其立足的特点，或基于传输速度、距离、耗电量的特殊要求;或着眼于功能的扩充性;或符合某些单一应用的特别要求;或建立竞争技术的差异化等。但是没有一种技术可以完美到足以满足所有的需求。

1、蓝牙技术

“蓝牙”（Bluetooth）技术使用高速跳频和时分多址等先进技术，在近距离内最廉价地将几台数字化设备（各种移动设备、固定通信设备、计算机及其终端设备、各种数字数据系统，如数字照相机、数字摄像机等，甚至各种家用电器、自动化设备）呈网状链接起来。蓝牙技术将是网络中各种外围设备接口的统一桥梁，它消除了设备之间的连线，取而代之以无线连接。

蓝牙技术的规范及特点 ：

蓝牙的标准是 IEEE802.15，工作在 2.4GHz 频带，带宽为 1Mb/s。以时分方式进行全双工通信，其基带协议是电路交换和分组交换的组合。一个跳频频率发送一个同步分组，每个分组占用一个时隙，使用扩频技术也可扩展到 5 个时隙。同时，蓝牙技术支持 1 个异步数据通道或 3 个并发的同步话音通道，或 1 个同时传送异步数据和同步话音的通道。每一个话音通道支持 64kb/s 的同步话音；异步通道支持最大速率为 721kb/s，反向应答速率为 57. 6 kb/s 的非对称连接，或者是 432. 6 kb/s 的对称连接。

依据发射输出电平功率不同，蓝牙传输有 3 种距离等级：Class1 为 100m 左右；Class2 约为 10m；Class3 约为 2-3m。一般情况下，其正常的工作范围是 10m 半径之内。在此范围内，可进行多台设备间的互联。

蓝牙技术的特点包括：1、采用跳频技术，数据包短，抗信号衰减能力强；2、采用快速跳频和前向纠错方案以保证链路稳定，减少同频干扰和远距离传输时的随机噪声影响；3、使用 2.4GHzISM 频段，无须申请许可证；4、可同时支持数据、音频、视频信号；5、采用 FM 调制方式，降低设备的复杂性。

蓝牙匹配规则及使用注意：

蓝牙技术作为一种标准开放性无线接入技术，用户在使用时必须了解和遵守其标准技术规范。

两个蓝牙设备在进行通讯前，必须将其匹配在一起，以保证其中一个设备发出的数据信息只会被经过允许的另一个设备所接受。

蓝牙技术将设备分为两种：主设备和从设备。

蓝牙主设备特点：主设备一般具有输入端。在进行蓝牙匹配操作时，用户通过输入端可输入随机的匹配密码来将两个设备匹配。蓝牙手机、安装有蓝牙模块的 PC 等都是主设备。（例如：蓝牙手机和蓝牙 PC 进行匹配时，用户可在蓝牙手机上任意输入一组数字，然后在蓝牙 PC 上输入相同的一组数字，来完成这两个设备之间的匹配。）

蓝牙主设备特点：从设备一般不具备输入端。因此从设备在出厂时，在其蓝牙芯片中，固化有一个 4 位或 6 位数字的匹配密码。蓝牙耳机、优士通 UD 笔等都是从设备。（例如：蓝牙 PC 与 UD 笔匹配时，用户将 UD 笔上的蓝牙匹配密码正确的输入到蓝牙 PC 上，完成 UD 笔与蓝牙 PC 之间的匹配。）

主设备与主设备之间、主设备与从设备之间，是可以互相匹配在一起的；而从设备与从设备是无法匹配的。例如：蓝牙 PC 与蓝牙手机可以匹配在一起；蓝牙 PC 也可以与 UD 笔匹配在一起；而 UD 笔与 UD 笔之间是不能匹配的。

一个主设备，根据其类型的不同，可匹配一个或多个其他设备。例如：一部蓝牙手机，一般只能匹配 7 个蓝牙设备。而一台蓝牙 PC，可匹配十多个或数十个蓝牙设备。

在同一时间，蓝牙设备之间仅支持点对点通讯。

2. IrDA

红外线数据协会 IrDA(Infrared Data Association)成立于 1993 年。起初，采用 IrDA 标准的无线设备仅能在 1m 范围内以 115.2 kb/s 速率传输数据，很快发展到 4Mb/s 以及 16Mb/s 的速率。

IrDA 是一种利用红外线进行点对点通信的技术，是第一个实现无线个人局域网(PAN)的技术。目前它的软硬件技术都很成熟，在小型移动设备，如 PDA、手机上广泛使用。事实上，当今每一个出厂的 PDA 及许多手机、笔记本电脑、打印机等产品都支持 IrDA。

IrDA 的主要优点是无需申请频率的使用权，因而红外通信成本低廉。并且还具有移动通信所需的体积小、功耗低、连接方便、简单易用的特点。此外，红外线发射角度较小，传输上安全性高。

IrDA 的不足在于它是一种视距传输，两个相互通信的设备之间必须对准，中间不能被其它物体阻隔，因而该技术只能用于 2 台(非多台)设备之间的连接。而蓝牙就没有此限制，且不受墙壁的阻隔。IrDA 目前的研究方向是如何解决视距传输问题及提高数据传输率。

3 、Wi-Fi 技术

Wi-Fi(Wireless Fidelity,无线高保真)也是一种无线通信协议，正式名称是 IEEE802.11b，与蓝牙一样，同属于短距离无线通信技术。Wi-Fi 速率最高可达 11Mb/s。虽然在数据安全性方面比蓝牙技术要差一些，但在电波的覆盖范围方面却略胜一筹，可达 100 m 左右。

Wi-Fi 是以太网的一种无线扩展，理论上只要用户位于一个接入点四周的一定区域内，

就能以最高约 11Mb/s 的速度接入 Web。但实际上，如果有多个用户同时通过一个点接入，带宽被多个用户分享，Wi-Fi 的连接速度一般将只有几百 kb/s 的信号不受墙壁阻隔，但在建筑物内的有效传输距离小于户外。

WLAN 未来最具潜力的应用将主要在 SOHO、家庭无线网络以及不便安装电缆的建筑物或场所。目前这一技术的用户主要来自机场、酒店、商场等公共热点场所。Wi-Fi 技术可将 Wi-Fi 与基于 XML 或 Java 的 Web 服务融合起来，可以大幅度减少企业的成本。例如企业选择在每一层楼或每一个部门配备 802.11b 的接入点，而不是采用电缆线把整幢建筑物连接起来。这样一来，可以节省大量铺设电缆所需花费的资金。

最初的 IEEE802.11 规范是在 1997 年提出的，称为 802.11b，主要目的是提供 WLAN 接入，也是目前 WLAN 的主要技术标准，它的工作频率也是 2.4GHz，与无绳电话、蓝牙等许多不需频率使用许可证的无线设备共享同一频段。随着 Wi-Fi 协议新版本如 802.11a 和 802.11g 的先后推出，Wi-Fi 的应用将越来越广泛。速度更快的 802.11g 使用与 802.11b 相同的正交频分多路复用调制技术。它工作在 2.4GHz 频段，速率达 54Mb/s。根据最近国际消费电子产品的发展趋势判断，802.11g 将有可能被大多数无线网络产品制造商选择作为产品标准。

微软推出的桌面操作系统 WindowsXP 和嵌入式操作系统 WindowsCE，都包含了对 Wi-Fi 的支持。其中，WindowsCE 同时还包含对 Wi-Fi 的竞争对手蓝牙等其它无线通信技术的支持。由于投资 802.11b 的费用降低，许多厂商介入这一领域。Intel 推出了集成 WLAN 技术的笔记本电脑芯片组，不用外接无线网卡，就可实现无线上网。

4、NFC 技术

NFC(Near Field Communication，近距离无线传输)是由 Philips、NOKIA 和 Sony 主推的一种类似于 RFID(非接触式射频识别)的短距离无线通信技术标准。和 RFID 不同，NFC 采用了双向的识别和连接。在 20cm 距离内工作于 13.56MHz 频率范围。

NFC 最初仅仅是遥控识别和网络技术的合并，但现在已发展成无线连接技术。它能快速自动地建立无线网络，为蜂窝设备、蓝牙设备、Wi-Fi 设备提供一个“虚拟连接”，使电子设备可以在短距离范围进行通讯。NFC 的短距离交互大大简化了整个认证识别过程，使电子设备间互相访问更直接、更安全和更清楚，不用再听到各种电子杂音。

NFC 通过在单一设备上组合所有的身份识别应用和服务，帮助解决记忆多个密码的麻烦，同时也保证了数据的安全保护。有了 NFC，多个设备如数码相机、PDA、机顶盒、电脑、手机等之间的无线互连，彼此交换数据或服务都将有可能实现。

此外 NFC 还可以将其它类型无线通讯(如 Wi-Fi 和蓝牙)“加速”，实现更快和更远距离的数据传输。每个电子设备都有自己的专用应用菜单，而 NFC 可以创建快速安全的连接，而无需在众多接口的菜单中进行选择。与知名的蓝牙等短距离无线通讯标准不同的是，NFC 的作用距离进一步缩短且不像蓝牙那样需要有对应的加密设备。

同样，构建 Wi-Fi 家族无线网络需要多台具有无线网卡的电脑、打印机和其它设备。除此之外，还得有一定技术的专业人员才能胜任这一工作。而 NFC 被置入接入点之后，只要将其中两个靠近就可以实现交流，比配置 Wi-Fi 连结容易得多。

NFC 有三种应用类型：

设备连接。除了无线局域网，NFC 也可以简化蓝牙连接。比如，手提电脑用户如果想

在机场上网，他只需要走近一个 Wi-Fi 热点即可实现。

实时预定。比如，海报或展览信息背后贴有特定芯片，利用含 NFC 协议的手机或 PDA，便能取得详细信息，或是立即联机使用信用卡进行票卷购买。而且，这些芯片无需独立的能源。

移动商务。飞利浦 Mifare 技术支持了世界上几个大型交通系统及在银行业为客户提供 Visa 卡等各种服务。索尼的 FeliCa 非接触智能卡技术产品在中国香港及深圳、新加坡、日本的市场占有率非常高，主要应用在交通及金融机构。

总而言之，这项新技术正在改写无线网络连接的游戏规则，但 NFC 的目标并非是完全取代蓝牙、Wi-Fi 等其他无线技术，而是在不同的场合、不同的领域起到相互补充的作用。所以如今后来居上的 NFC 发展态势相当迅速!

5、ZigBee 技术

ZigBee 主要应用在短距离范围之内并且数据传输速率不高的各种电子设备之间。ZigBee 名字来源于蜂群使用的赖以生存和发展的通信方式，蜜蜂通过跳 ZigZag 形状的舞蹈来分享新发现的食物源的位置、距离和方向等信息。

ZigBee 联盟成立于 2001 年 8 月。2002 年下半年，Invensys、Mitsubishi、Motorola 以及 Philips 半导体公司四大巨头共同宣布加盟 ZigBee 联盟，以研发名为 ZigBee 的下一代无线通信标准。到目前为止，该联盟大约已有 27 家成员企业。所有这些公司都参加了负责开发 ZigBee 物理和媒体控制层技术标准的 IEEE 802.15.4 工作组。

ZigBee 联盟负责制定网络层以上协议。目前，标准制订工作已完成。ZigBee 协议比蓝牙、高速率个人区域网或 802.11x 无线局域网更简单实用。

ZigBee 可以说是蓝牙的同族兄弟，它使用 2.4 GHz 波段，采用跳频技术。与蓝牙相比，ZigBee 更简单、速率更慢、功率及费用也更低。它的基本速率是 250kb/s，当降低到 28kb/s 时，传输范围可扩大到 134m，并获得更高的可靠性。另外，它可与 254 个节点联网。可以比蓝牙更好地支持游戏、消费电子、仪器和家庭自动化应用。人们期望能在工业监控、传感器网络、家庭监控、安全系统和玩具等领域拓展 ZigBee 的应用。

ZigBee 技术特点主要包括以下几个部分：

数据传输速率低。只有 10kb/s～250kb/s，专注于低传输应用。

功耗低。在低耗电待机模式下，两节普通 5 号干电池可使用 6 个月以上。这也是 ZigBee 的支持者所一直引以为豪的独特优势。

成本低。因为 ZigBee 数据传输速率低，协议简单，所以大大降低了成本; 积极投入 ZigBee 开发的 Motorola 以及 Philips，均已在 2003 年正式推出芯片，飞利浦预估，应用于主机端的芯片成本和其它终端产品的成本比蓝牙更具价格竞争力。

网络容量大。每个 ZigBee 网络最多可支持 255 个设备，也就是说每个 ZigBee 设备可以与另外 254 台设备相连接。

有效范围小。有效覆盖范围 10～75m 之间，具体依据实际发射功率的大小和各种不同的应用模式而定，基本上能够覆盖普通的家庭或办公室环境。

工作频段灵活。使用的频段分别为 2.4GHz、868MHz(欧洲)及 915MHz(美国)，均为

免执照频段。

根据 ZigBee 联盟目前的设想，ZigBee 的目标市场主要有 PC 外设(鼠标、键盘、游戏操控杆)、消费类电子设备(TV、VCR、CD、VCD、DVD 等设备上的遥控装置)、家庭内智能控制(照明、煤气计量控制及报警等)、玩具(电子宠物)、医护(监视器和传感器)、工控(监视器、传感器和自动控制设备)等非常广阔的领域。

6、UWB 技术

超宽带技术 UWB(Ultra Wideband)是一种无线载波通信技术，它不采用正弦载波，而是利用纳秒级的非正弦波窄脉冲传输数据，因此其所占的频谱范围很宽。

UWB 可在非常宽的带宽上传输信号，美国 FCC 对 UWB 的规定为：在 3.1～10.6GHz 频段中占用 500MHz 以上的带宽。由于 UWB 可以利用低功耗、低复杂度发射/接收机实现高速数据传输，在近年来得到了迅速发展。它在非常宽的频谱范围内采用低功率脉冲传送数据而不会对常规窄带无线通信系统造成大的干扰，并可充分利用频谱资源。基于 UWB 技术而构建的高速率数据收发机有着广泛的用途。

UWB 技术具有系统复杂度低，发射信号功率谱密度低，对信道衰落不敏感，低截获能力，定位精度高等优点，尤其适用于室内等密集多径场所的高速无线接入，非常适于建立一个高效的无线局域网或无线个域网(WPAN)。

UWB 主要应用在小范围、高分辨率、能够穿透墙壁、地面和身体的雷达和图像系统中。除此之外，这种新技术适用于对速率要求非常高(大于 100 Mb/s)的 LANs 或 PANs。

UWB 最具特色的应用将是视频消费娱乐方面的无线个人局域网(PANs)。现有的无线通信方式，802.11b 和蓝牙的速率太慢，不适合传输视频数据;54 Mb/s 速率的 802.11a 标准可以处理视频数据，但费用昂贵。而 UWB 有可能在 10 m 范围内，支持高达 110 Mb/s 的数据传输率，不需要压缩数据，可以快速、简单、经济地完成视频数据处理。

具有一定相容性和高速、低成本、低功耗的优点使得 UWB 较适合家庭无线消费市场的需求：UWB 尤其适合近距离内高速传送大量多媒体数据以及可以穿透障碍物的突出优点，让很多商业公司将其看作是一种很有前途的无线通信技术，应用于诸如将视频信号从机顶盒无线传送到数字电视等家庭场合。当然，UWB 未来的前途还要取决于各种无线方案的技术发展、成本、用户使用习惯和市场成熟度等多方面的因素。

第 5 章　常用传感器

传感器是一种以测量为目的，以一定精度把被测量（温度、压力、速度等）转换为与之有确定关系的、易于处理的电量信号输出的装置。如果传感器进一步对此输出信号进行处理，转换成标准统一信号（例如：4-20mA 或 1-5V；0-10mA 或 0-5V 等）时，此时的传感器一般称为变送器。

5.1 传感器分类

通常传感器按下列原则进行分类。

1、按被检测量分类

按被检测量分类，可分为物理量传感器，化学量传感器，生物量传感器。

类	组	具体名称
物理量传感器	力学量传感器	压力传感器　力传感器　力矩传感器　速度传感器　加速度传感器　流量传感器　位移传感器　位置传感器　尺度传感器　密度传感器　粘度传感器　硬度传感器　浊度传感器
	热学量传感器	温度传感器　热流传感器　热导率传感器
	光学量传感器	可见光传感器　红外光传感器　紫外光传感器　照度传感器　色度传感器　图像传感器　亮度传感器
	磁学量传感器	磁场强度传感器　磁通传感器
	电学量传感器	电流传感器　电压传感器　电场强度传感器
	声学量传感器	声压传感器　噪声传感器　超声波传感器　声表面波传感器
	射线传感器	x 射线传感器　β 射线传感器　γ 射线传感器　辐射剂量传感器
化学量传感器	离子传感器	离子活度传感器　离子浓度传感器　成分传感器　PH 值传感器
	气体传感器	气体分压传感器　气体浓度传感器
	湿度传感器	露点传感器　水分传感器
生物量传感器	生化量传感器	酶式葡萄糖传感器　酶式尿素传感器　酶式胆固醇传感器　免疫血型传感器　微生物 BOD 传感器　微生物谷氨酸传感器　血液 PH 传感器　血氧传感器　血液二氧化碳传感器　血钾传感器　血钠传感器　血钙传感器
	生理量传感器	体压传感器　脉搏传感器　心音传感器　体温传感器　血流

传感器　呼吸传感器　血容量传感器心电图传感器　脑电图

传感器　肌电图传感器　视网膜电图传感器

2. 按物理原理分类

这种分类方法是以传感器的物理原理作为分类依据。可分为压阻式、压电式、电感式、电容式、应变式、霍尔式……；这种分类方法有利于传感器专业工作者从原理和设计上作归纳性的分析和研究。

3. 按能量的传递方式分类

按能量的传递方式分类，传感器可分为有源传感器和无源传感器两大类。有源传感器将非电量转换为电量。无源传感器本身并不是一个换能器，被测非电量仅对传感器中的能量起控制或调节作用，所以它必须具有辅助能源——电源。

5.2 温度传感器

温度传感器种类很多，主要有热电偶传感器、热电阻型温度传感器、热敏电阻、晶体管温度传感器、PN 结温度传感器、热膨胀型温度传感器、示温涂料型温度传感器、辐射式温度传感器、热释电式温度传感器、电容型温度传感器、光纤温度传感器。

5.2.1 热敏电阻

热敏电阻是对温度敏感的半导体元件，主要特征是随着外界环境温度的变化，其阻值会相应发生较大改变。电阻值对温度的依赖关系称为阻温特性。热敏电阻根据温度系数分为两类：正温度系数热敏电阻和负温度系数热敏电阻。由于特性上的区别，应用场合互不相同。

正温度系数热敏电阻简称 PTC（是 Positive Temperature Coefficient 的缩写），超过一定的温度(居里温度)时，它的电阻值随着温度的升高呈阶跃性的增高。

负温度系数热敏电阻简称 NTC（是 Negative Temperature Coefficient 的缩写），它的阻值是随着温度的升高而下降的。主要是以锰、钴、镍和铜等金属氧化物为主要材料，采用陶瓷工艺制造而成的。这些金属氧化物材料都具有半导体性质，因为在导电方式上完全类似锗、硅等半导体材料。

热敏电阻的规格是在摄氏 25 度时的阻值，如某热敏电阻的规格为 10K 欧，表示该电阻在 25℃时呈现的阻值为 10K。热敏电阻的测温范围一般在-50℃～+350℃。

应用设计：

电子温度计、电子万年历、电子钟温度显示、电子礼品；

* 冷暖设备、加热恒温电器；
* 汽车电子温度测控电路；
* 温度传感器、温度仪表；
* 医疗电子设备、电子盥洗设备；

* 手机电池及充电电器。

某种 NTC 热敏电阻分度表实测如下：

温度	阻值	温度	阻值	温度	阻值
-8℃	1.65KΩ	0℃	1.7KΩ	2℃	1.76KΩ
5℃	1.80KΩ	10℃	1.88KΩ	15℃	1.94KΩ
20℃	2.04KΩ	25℃	2.11KΩ	30℃	2.20KΩ
35℃	2.26KΩ	40℃	2.33KΩ	45℃	2.40KΩ
50℃	2.52KΩ	55℃	2.60KΩ	60℃	2.76KΩ
65℃	2.82KΩ	70℃	2.99KΩ		

热敏电阻信号调理电路如图 5-1 所示。

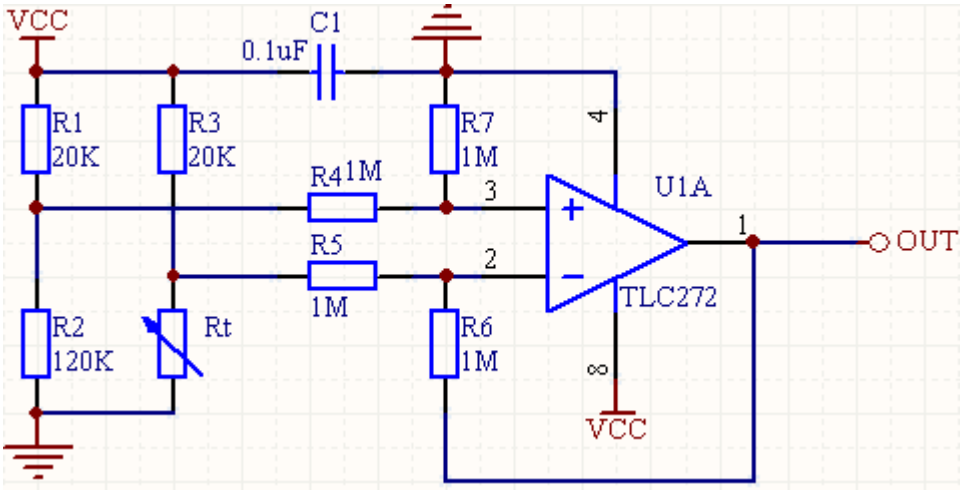


图 5-1 热敏电阻信号调理电路

热敏电阻 Rt 与电阻 R1\R2\R3 组成电桥电路，当温度变化时，Rt 的阻值发生变化，引起电桥输出，经运放 TLC272 差动放大后，送 A/D 转换电路。

5.2.2 热电偶

热电偶的原理基于热电效应，如图 5-2(a)所示。图中，将两种不同导体 A、B 两端连接在一起组成闭合回路，并使两端处于不同的温度环境， 在回路中会产生热电动势而形

成电流，这一现象称为热电效应。 这样的两种不同导体的组合称为热电偶， 相应的电动势和电流称为热电动势和热电流， 导体 *A*、*B* 称为热电极， 置于被测温度（*T*）的一端称为工作端(热端)，另一端（*T*0）称为参考端(冷端)。 实验证明， 热电动势与热电偶两端的温度差成比例，即

$$E_{AB}(T, T_0)=K(T-T_0)$$

式中，*K* 与导体的电子浓度有关

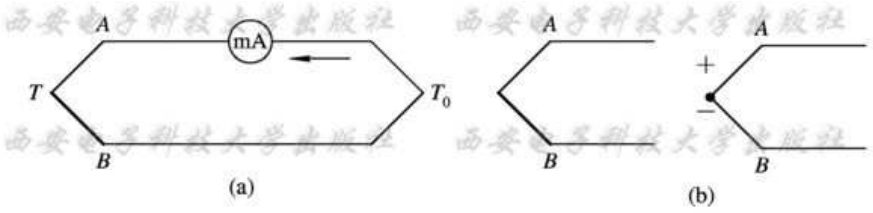


图 5-2 （a） 热电效应； （b）热电偶的电路符号

常用的热电偶材料有

热电偶分度号	热电极材料
正极	负极
S 铂铑	10 纯铂
R 铂铑	13 纯铂
B 铂铑	30 铂铑 6
K 镍铬	镍硅
T 纯铜	铜镍
J 铁	铜镍
N 镍铬硅	镍硅
E 镍铬	铜镍

由于热电偶材料价格昂贵，本身不适合进行远距离测量，因此需要使用补偿导线来延长测量距离。

图 5-3 补偿导线的使用

补偿导线是特种导线，用于热电偶和二次（基地）仪表间的信号传输，能够消除热电偶冷端温度变化引起的测量误差，保证仪表对介质温度的精确测量。用它们将热电偶与测量装置联接，以补偿热电偶连接处的温度变化所产生的误差。

与热电阻和热敏电阻相同，热电偶也采用分度表形式给出其温度与热电势的关系。

附录 E　镍铬–镍硅 K 热电偶分度表（自由端温度为 0℃）

工作端温度/℃	热电势/mV	工作端温度/℃	热电势/mV	工作端温度/℃	热电势/mV	工作端温度/℃	热电势/mV
-270	-6.458	0	0.000	270	10.971	540	22.350
-260	-6.441	10	0.397	280	11.382	550	22.776
-250	-6.404	20	0.798	290	11.795	560	23.203
-240	-6.344	30	1.203	300	12.209	570	23.629
-230	-6.262	40	1.612	310	12.624	580	24.055
-220	-6.158	50	2.023	320	13.040	590	24.480
-210	-6.035	60	2.436	330	13.457	600	24.905
-200	-5.891	70	2.851	340	13.874	610	25.330
-190	-5.730	80	3.267	350	14.293	620	25.755
-180	-5.550	90	3.682	360	14.713	630	26.179
-170	-5.354	100	4.096	370	15.133	640	26.602
-160	-5.141	110	4.509	380	15.554	650	27.025
-150	-4.913	120	4.920	390	15.975	660	27.447
-140	-4.669	130	5.328	400	16.397	670	27.869
-130	-4.411	140	5.735	410	16.820	680	28.289
-120	-4.138	150	6.138	420	17.243	690	28.710
-110	-3.852	160	6.540	430	17.667	700	29.129
-100	-3.554	170	6.941	440	18.091	710	29.548
-90	-3.243	180	7.340	450	18.516	720	29.965
-80	-2.920	190	7.739	460	18.941	730	30.382
-70	-2.587	200	8.138	470	19.366	740	30.798
-60	-2.243	210	8.539	480	19.792	750	31.213
-50	-1.889	220	8.940	490	20.218	760	31.628
-40	-1.527	230	9.343	500	20.644	770	32.041
-30	-1.156	240	9.747	510	21.071	780	32.453
-20	-0.778	250	10.153	520	21.497	790	32.865
-10	-0.392	260	10.561	530	21.924	800	33.275

可以看出，热电偶的温度与其热电势不呈线性关系，因此在使用时需进行线性补偿或采用查表方式进行温度计算。

由于热电势很小（mV 级），在进行测量时需进行放大等信号处理。下图是使用 K 型热电偶，将 0℃～500℃的温度转换为 0V～5V 电压的电路。除放大外还有基准节点温度补偿电路和断线检测电路，而线性化处理由微处理器进行。

热电偶的输出电压极小，每 1℃约为 40uV，因此，运放要采用高灵敏度运放，电路中采用 AD707J 运放。

K 型热电偶 500℃（满度）的感应电势为 20.64mV，运放增益 Av 应为

Av=5V/20.64mV=242。电路中，R7=1K，R5=232K，Rp3=20K 决定增益的大小，调节 Rp3 使增益在 233 和 253 之间。R3 和 C1 为低通滤波器，时间常数越大，消除噪声越好，但响应速度变慢。另外，R3 增大，运放的输入偏置电流要产生偏移电压，因此，r3 阻值不能过大。AD707 偏置输入电流为 2.5nA、R3=10K 时，产生 25uV 的偏移电压。

基准节点温度补偿采用温度传感器 LM35D，它的每 1℃温度相应输出电压为 10mV，用电阻分压，并用电位器 Rp1 调整使其 A 点电压为 40.44uV/℃，进行基准节点的温度补偿。

R2 是检测传感器断线的电阻，热电偶断线时，运放输出就要超出范围。然而，因接有电阻 R2，热电偶的内阻要产生偏移电压。例如，热电阻内阻为 200（包括 Rp1 和 R6 的阻值），R2=100M 时，也要产生 24uV 的偏移电压。另外，断线时运放输入偏置电流要经过 R2，因此，不能采用输入偏置电流较大的运放。

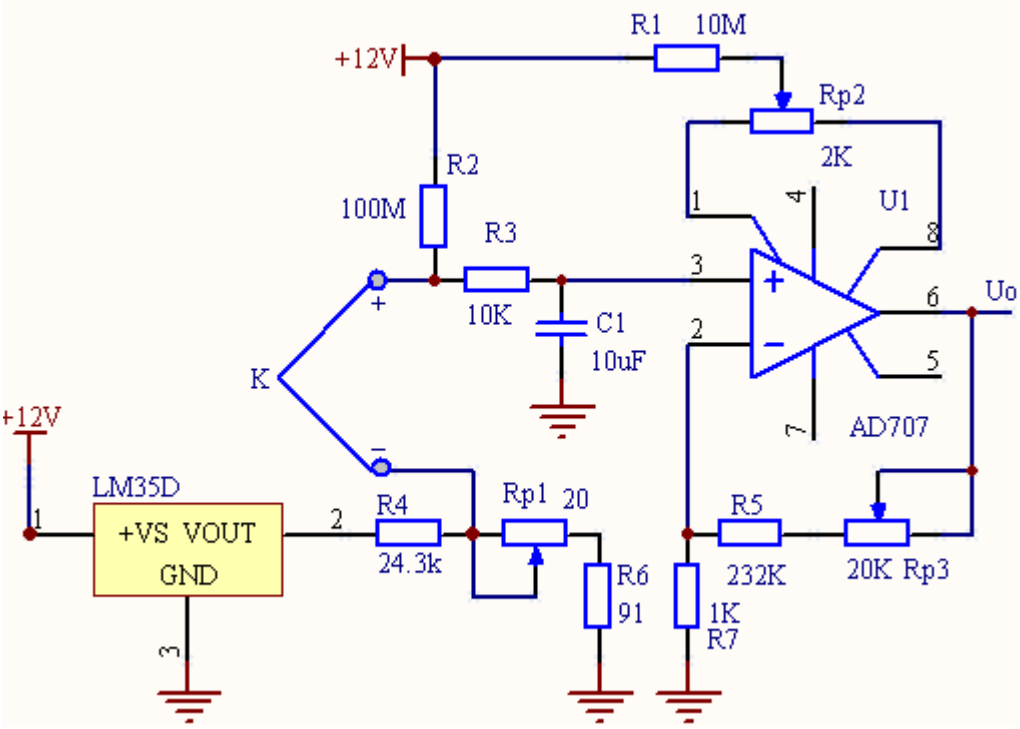


图 5-4 热电偶信号调理电路

5.2.3 其它常用温度传感器

1、热电阻

与热敏电阻（半导体材料制成）不同，一般用铂（Pt）或铜（Cu）材料制成，只有正温度系数。调理方法与热敏电阻类似，而且也是采用分都表方式表示其电阻变化与温度的

关系。一般常用 Pt100 和 Pt1000 等。

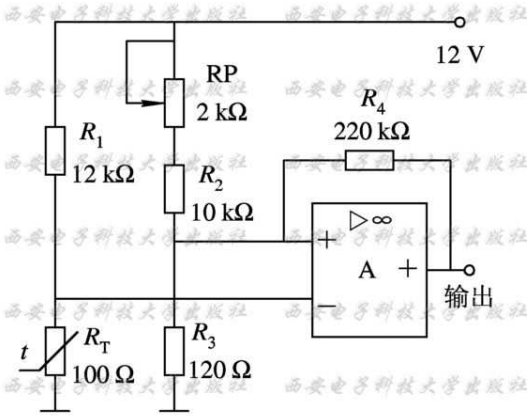


图 5-5 二线式铂热电阻接线实例

2、模拟集成温度传感器

模拟集成温度传感器实质上是一种半导体集成电路它是利用晶体管的 B-E 结压降的不饱和值 Vbe 与热力学温度 T 和通过发射极电流 I 的下述关系实现对温度的检测。

$$Vbe = \frac{kIT}{q} \ln I$$

式中， k 是波尔兹曼常数，q 是电子电荷的绝对值。

集成温度传感器具有线性好、精度高、灵敏度高、体积小、使用方便等特点，得到了广泛应用。其输出形式分为电压输出和电流输出两种。一般测温范围为－50℃～150℃左右。

常用的集成温度传感器有 AD590、AN6701、LM35、LM335/LM336/LM334 等。

3、数字温度传感器

数字温度传感器直接将温度信号变为数字，微处理器可直接读取其温度数据，进行简单计算后即可获得温度值。精度非常高，象 DALLAS 公司的 DS18B20 就是一款数字温度传感器，其测温范围一般在－30℃～150℃左右，精度可达 0.0625℃，而且接口简单。该传感器应用非常广泛，详细介绍在《单片机培训教材》中描述。

5.3 光电式传感器

5.3.1 光与光电效应

光是一种电磁波， 其频谱如图 5-6 所示。 可见光只是电磁波谱中的一小部分，波长在 780～380 nm 之间， 红光频率最低， 紫光频率最高。 光的频率越高， 携带的能量越大。

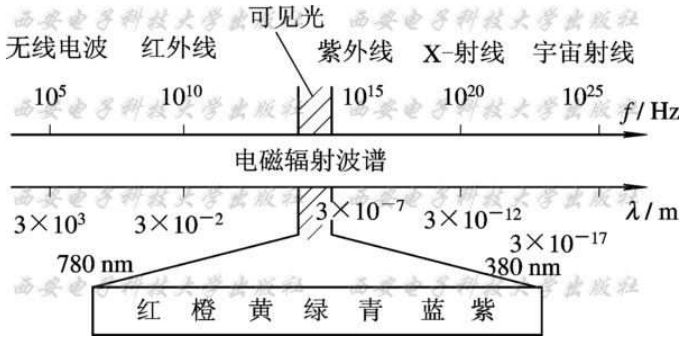


图 5-6 光的频谱

光电式传感器是将光量的变化转变为电量的变化的一种变化器，其理论基础是光电效应。在光线的作用下，能使电子逸出物体表面的现象称为外光电效应；在光线作用下，能使物体的电阻率改变的现象称为内光电效应；在光线的作用下能使物体产生一定方向的电动势的现象称为阻挡层光电效应。

由于光电元件反应快、结构简单、而且有较高的可靠性等优点，因此，它在自动化系统中得到了非常广泛的应用。光电元件是构成光电式传感器最主要的部件。

5.3.2. 紫外线传感器

紫外线传感器是一种专门用来检测紫外线的光电器件。它的光谱响应为 85～260 nm，对紫外线特别敏感，尤其对燃烧时产生的紫外线反应更为强烈，甚至可以检测 5m 以内打火机火焰发出的紫外线。 它除了会受到高压水银灯、γ 射线、 闪电及焊接弧光的干扰外，对可见光不敏感。 此外， 它还具有灵敏度高、 受光角度宽（视角范围达 120°）、 响应速度快的特点。 因此， 紫外线传感器主要用作火灾报警敏感元件， 故又称它为火灾报警传感器。它可以广泛地用于石油、气体燃料的火灾报警，也可以用于宾馆、饭店、办公室、仓库等重要场合的火灾报警。

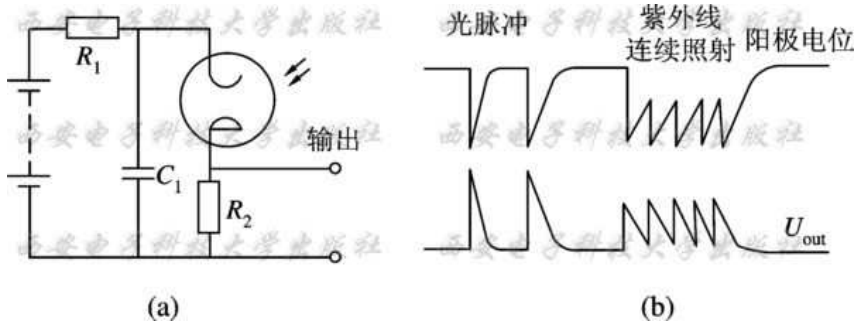


图 5-7 紫外线传感器基本电路及输出波形

（a） 基本电路； （b） 输出波形

当入射紫外线光通量低于某值时， 从输出端可以得到与入射光量成正比的脉冲数，但若光通量大于此值时， 由于电容的放电， 管内电流就饱和了。 因此紫外线传感器适合作光电开关，不适合作精密的紫外线测量。

5.33.光敏电阻

光敏电阻又称光导管，是一种均质半导体光电元件，当光照射时其电阻值降低。将其与一电阻串联并接到电源上， 便可把光信号变成电信号。

按光谱特性及最佳工作波长范围分类， 可有紫外光、 可见光及红外光光敏电阻类。CdS 光敏电阻覆盖了紫外光和可见光范围，其典型结构如图 5-8 所示。 将 CdS 粉末烧结在陶瓷衬底上，形成一层 CdS 膜，用两根引线引出。为防止光敏电阻芯片受潮，均需采用密封结构，常用金属外壳、塑料或防潮涂料等密封。

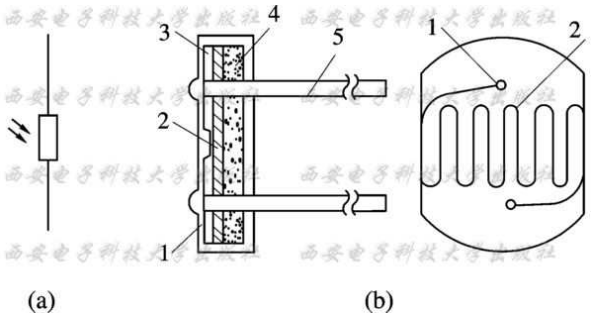


图 5-8 光敏电阻的结构 (a) 电路符号； (b) 结构图

5.3.4 光敏管

1、光敏二极管

光敏二极管与普通半导体二极管的主要区别在于 PN 结面积较大、 距表面较浅， 上电极较小， 利于接受光照射以提高光电转换效率。 如前所述， 它的工作机理是光生电动势效应， 即当受到光照时，半导体本征载流子浓度增加，在 P 区和 N 区均为少数载流子，在 PN 结势垒作用下，分别向对方区域漂移。 此时若将两端短路，便构成短路光电流；若两端开路或接负载， 则输出光生电势； 若加外电场， 则反向饱和电流增加。

光敏二极管正向伏安特性与普通二极管相似，光电流不明显；反向特性受光照控制。因此， 光敏二极管一般加反向偏置电压，利用反向饱和电流随光照强弱而变化进行工作。

光敏二极管的种类很多。 按制作材料来分， 有硅光敏二极管(2CU、2DU 类)， 锗光敏二极管（2AU 类）； 按不同峰值波长来分，有近红外光硅光敏二极管， 如对红外光最敏感的锂漂移性硅光敏二极管， 蓝光光敏二极管等； 其他还有用于激光的 PIN 型硅光敏二极管（日本产 SPD、 S、MP、 MBC、 SP、 PP、 TP、 PD、 PH、 TPS、 M 等多种系列，国产 2CU101、 2CU201 等）和灵敏度更高的雪崩光敏二极管等。 国产光敏二极管一般有 2CU 和 2DU 两种， 常用 2CU 型。

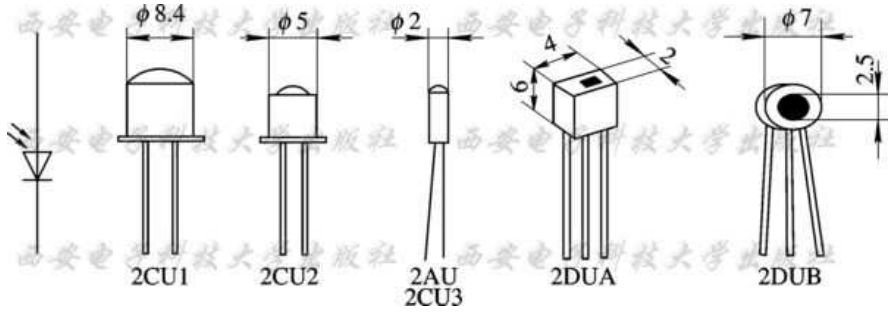


图 5-9 光敏二极管电路符号和外形

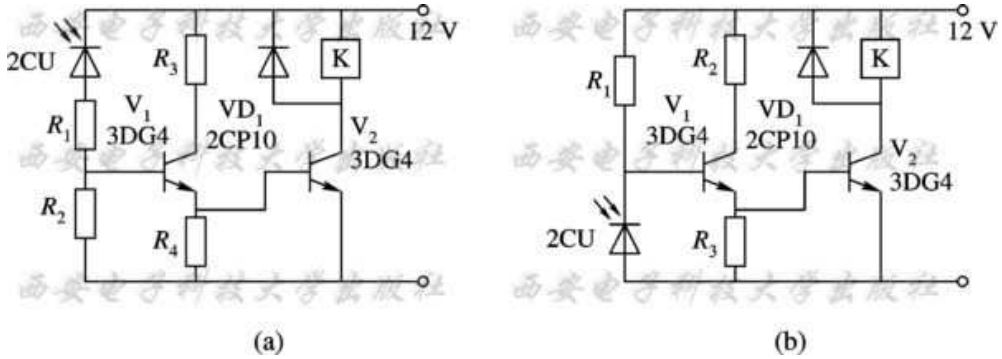


图 5-10 光敏二极管应用电路（a) 亮通电路； （b) 暗通电路

2、光敏晶体管

光敏晶体管与普通晶体管类似，但发射区较小，当光照射到发射结上时，产生基极光电流 *I*L，集电极电流 *I*C＝ *β I*L， 显然集电极电流 *I*C 正比于照射光的强度。

光敏三极管的电路符号及基本应用电路如图 5-11 所示。 国产光敏三极管的型号主要有 3AU、 3DU、 ZL 系列，日本型号有 TPS、 PT、 PPT、 PH、 PS、 PN、 T 等系列。

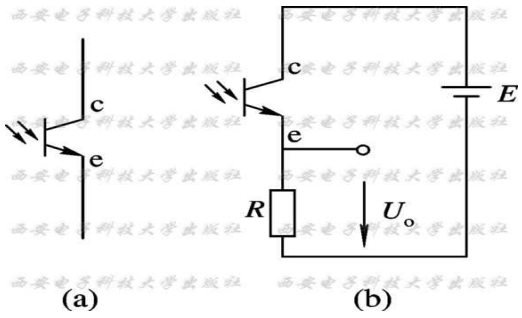


图 5-11 光敏三极管电路符号及应用电路 （a） 电路符号； （b） 基本应用电路

5.3.5 热释电传感器（PIR）

压电陶瓷类电介质在电极化后能保持极化状态， 称为自发极化。自发极化随温度升高而减小， 在居里点温度降为零。因此，当这种材料受到红外辐射而温度升高时， 表面

电荷将减少，相当于释放了一部分电荷，故称为热释电。将释放的电荷经放大器可转换为电压输出。这就是热释电传感器的工作原理。

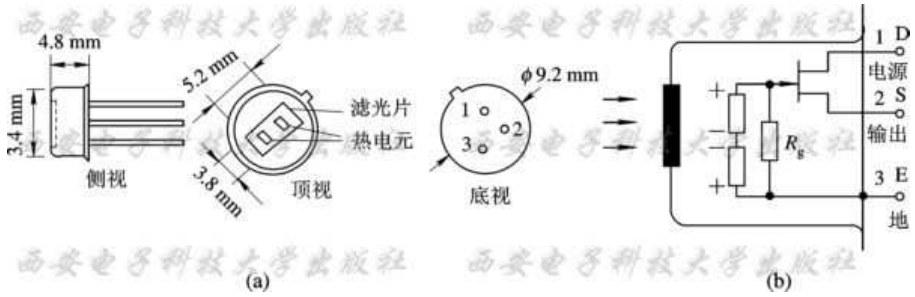


图 5-12　LN074B 型热释电传感器的外形及内部组成（a） 外形图；　（b）内部组成图

热释电传感器又称人体红外传感器， 被广泛应用于防盗报警、 来客告知及非接触开关等红外领域。 当辐射继续作用于热释电元件， 使其表面电荷达到平衡时， 便不再释放电荷。 因此， 热释电传感器不能探测恒定的红外辐射。

5.3.5 光电检测的组合形式

光电传感器按输出信号有开关型和模拟型， 开关型用于转速测量、 模拟开关、 位置开关等；模拟型用于光电式位移计、 光电比色计等。 光电检测必须具备光源、 被测物和光电元件。 按照光源、被测物和光电元件三者的关系，光电传感器可分为四种类型。

（1） 被测物发光：被测物为光源，可检测发光物的某些物理参数。如光电比色高温计、光照度计等。

（2） 被测物反光：可检测被测物体表面性质参数或状态参数， 如光洁度计和白度计等。

（3） 被测物透光：可检测被测物与吸收光或透射光特性有关的某些参数，如浊度计和透明度计等。

（4） 被测物遮光： 检测被测物体的机械变化， 如测量物体的位移、 振动、 尺寸、位置等。

1、光电耦合器

如图 5-13 所示，光电耦合器是把发光器件和光敏器件组装在同一蔽光壳体内，或用光导纤维把二者连接起来构成的器件。 当输入端加电信号，发光器件发光，光敏器件受光照后， 输出光电流， 实现以光为媒介质的电信号传输，从而实现输入和输出电流的电气隔离， 所以可用它代替继电器，变压器和斩波器等。 它广泛应用于隔离线路、开关电路、数模转换、逻辑电路、 长线传输、 过流保护、 高压控制等方面。

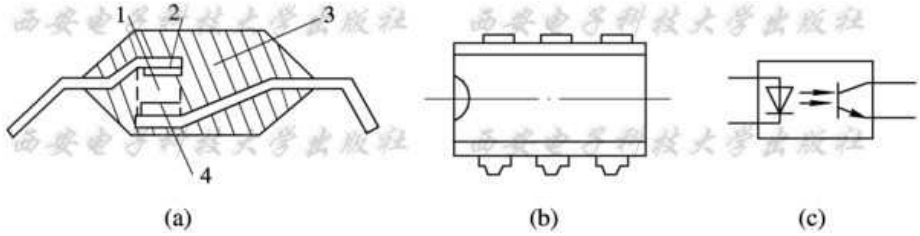


图 5-13 光电耦合器（a） 结构； （b） 外形； （c） 图形符号

2、光断续器

（1） 直射型光断续器：如图 5-14 所示，主要用于光电控制和光电计量等电路中及检测物体的有无、运动方向、 转速等。

（2） 反射型光断续器： 如图 5-15 所示，主要用于光电式接近开关、 光电自动控制、物体识别等。

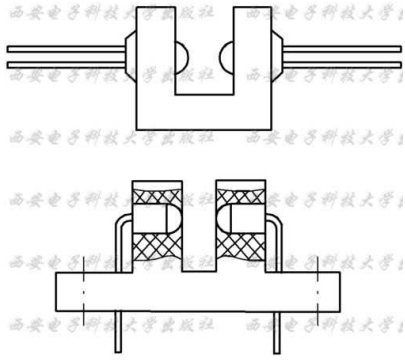


图 5-14 直射型光断续器

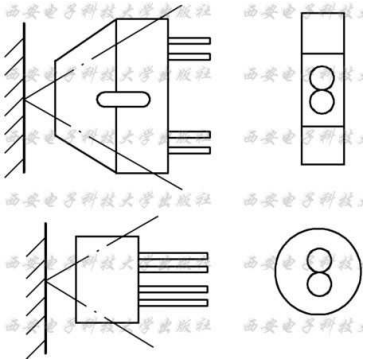


图 5-15 反射型光断续器

5.4 超声波传感器

1、超声波介绍

人耳能够听到的机械波，频率在 16 Hz～20 kHz 之间，称为声波。人耳听不到的机械波， 频率高于 20 kHz 的称为超声波； 频率低于 16 Hz 的称为次声波。超声波的频率越高，就越接近光学的反射、折射等特性。

超声波可分为纵波、横波和表面波。质点的振动方向和波的传播方向一致的波称为纵波，它能在固体、液体和气体中传播。质点的振动方向和波的传播方面相垂直的波称为横波， 它只能在固体中传播。质点的振动介于横波和纵波之间，沿着表面传播，振幅随着深度的增加而迅速衰减的波称为表面波。

超声波在介质中的传播速度取决于介质密度、介质的弹性系数及波型。一般来说， 在同一固体中横波声速为纵波声速的一半左右，而表面波声速又低于横波声速。当超声波在某一介质中传播，或者从一种介质传播到另一介质时，遵循如下一些规律：

（1） 传播速度：超声波的传播速度与波长及频率成正比， 即声速为

$C=\lambda f$

（2） 超声波的衰减：超声波在介质中传播时，由于声波的扩散、散射及吸收，能量按指数规律衰减。如平面波传播时的衰减公式可写作 $Ix= I0e\text{-}2\,\alpha x$。其中，$I0$ 为声源处的声强; Ix 为距声源 x 处的声强; α 为衰减系数（单位为 1×10-3dB/mm），水和一般低衰减材料的的取值 α 为 1～4。

（3） 超声波的反射与折射： 当超声波从一种介质传播到另一种介质时， 在两种介质的分界面上，会发生反射与折射。同样遵循反射定律和折射定律：入射角与反射角、折射角的正弦比等于入射波速与反射波速、折射波速之比。

（4） 超声波的波形转换： 若选择适当的入射角， 使纵波全反射， 那么在折射中只有横波出现； 如果横波也全反射， 那么在工件表面上只有表面波存在。

2、超声波换能器

超声波换能器也称为超声波探头， 即超声波传感器。 按原理有压电式、 磁致伸缩式、 电磁式等， 其中压电式最常用。 压电式利用压电材料的逆压电效应制成超声波发射头， 利用压电效应制成超声波接收头。 按照不同的应用目的， 超声波传感器有不同的结构形式。

3、空气中传播的超声波传感器及其基本电路

1）遥控用超声波传感器

超声波遥控电路采用专用的在空气中传播的超声发射器（用符号 T 表示）与接收器(用符号 R 表示)成对配套使用。 超声波传感器的结构采用双压电陶瓷晶片结构。 将双压电陶瓷晶片固装在基座上，为了增强其效果，在压电晶片上面加装了锥形喇叭，最后将其装在金属壳体中并伸出两根引线。它所发射的超声波采用固定的中心频率，谐振频率 $f0$ 一般为 40kHz。这种传感器有一种单峰特性，即在中心频率 $f0$ 处灵敏度最高，输出信号幅度最大，接收器的接收灵敏度最高， 而在中心频率两侧则迅速衰减。由于超声波接收器具有很好的选频特性， 因此在组成电路系统时，不必另设选频网络。由于发射器需要发射出强度较高的超声波信号，所以它的灵敏度大于 100 dB。接收器应能良好地接收超声波信号，因此它的灵敏度大于－60 dB。

2）超声波发射电路

图 5-16 是由数字集成电路构成的超声波振荡电路，振荡器产生的高频电压通过耦合电容 CP 供给超声波振子 MA40S2S。CC4049 的 H1 和 H2 产生与超声波频率相对应的高频电压信号， H3～H6 进行功率放大，再经过耦合电容 CP 传给超声波振子 MA40S2S。超声波振子若长时间加直流电压，会使传感器特性明显变差， 因此，一般用交流电压通过耦合电容 CP 供给传感器。该电路通过调节 R 可改变振荡频率：

$$f_0=\frac{1}{2.2RC}(\mathrm{Hz})$$

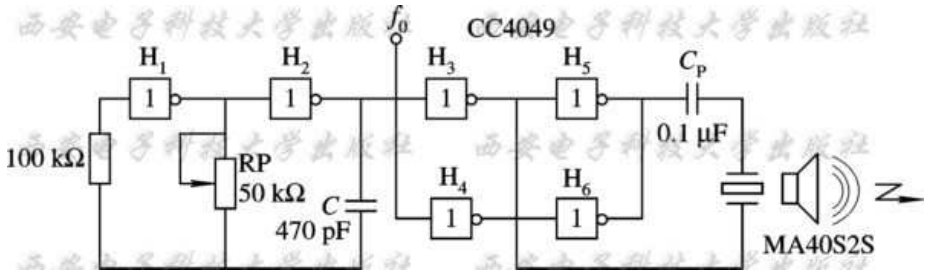


图 5-16 数字式超声波振荡电路

3）超声波接收电路

由于超声波传感器接收到的信号极其微弱，因此，一般要接几十 dB 以上的高增益放大器。

如图 5-17 所示，采用 NPN 晶体管 V 进行放大构成超声波接收电路，超声波传感器采用 MA40S2R。 超声波传感器一般用于检测反射波，它远离超声波发生源，能量衰减较大，只能接收到几 mV 左右的微弱信号。因此，实际应用时要加多级放大器。

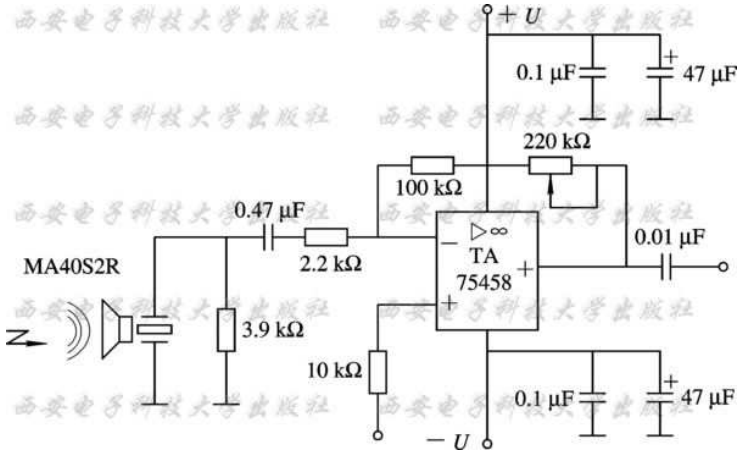


图 5-17 集成运放超声波接收电路

5.5 压力传感器

压力传感器的主要类别有电位器式、应变式、霍尔式、电感式、压电式、压阻式、电容式及振弦式等，测量范围为 7×10-5～5×108 Pa；信号输出有电阻、电流、电压、频率等形式。 压力测量系统一般由传感器、测量线路和测量装置以及辅助电源所组成。常见的信号测量装置有电流表、 电压表、 应变仪以及计算机等。

1、压力测量电路

由压力传感器及运算放大器组成的压力测量电路如图 5-18 所示。 图中， 压力传感器采用 43 系列。43 系列是一种小型压阻式压力传感器，有绝对压力和表压力两大类。标准量程系列有 0～5 Psi 和 0～250 Psi，共分 7 挡， 每挡有 A、B、C 三种等级。1 Psi=6.895

×103 Pa，Psi 是磅/英寸 2(bf/in2)单位。

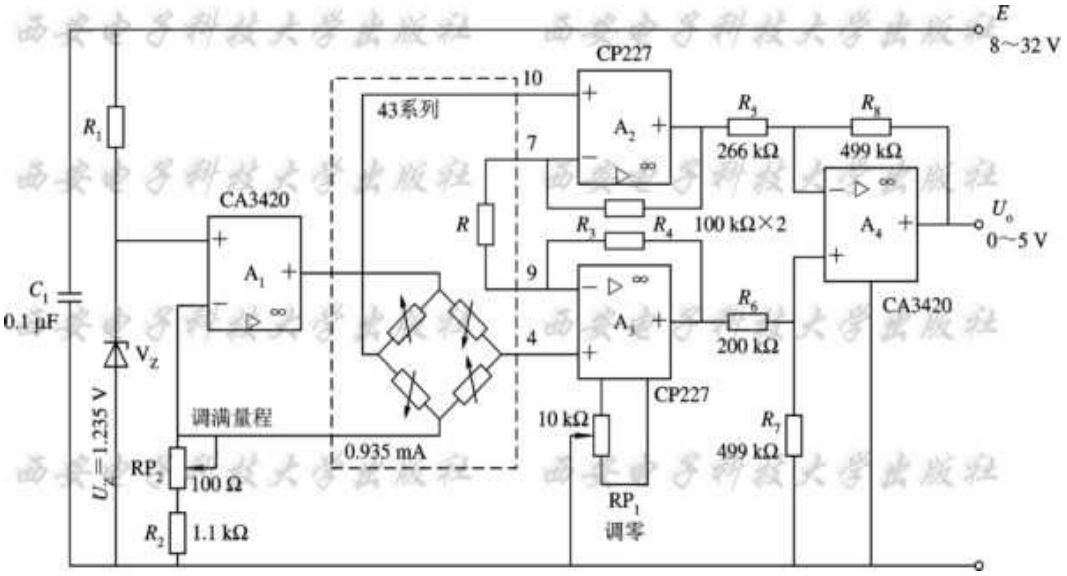


图 5-18 压力测量电路

供给传感器的恒流源电流 *I*0=*U*z/*R*2 可通过 *R*2 来调整，从而调节传感器的灵敏度。该电路的电流为 0.996 mA。43 系列满量程输出为 100 mV， 经 A2、A3 和 A4 放大后，要求相应的输出为 0～5 V，以此来决定放大器的放大倍数。 调节调零电位器 RP1， 使在零压力时输出为 0 V。调节电位器 RP2，使在满量程时， 输出为 5 V。

2、压力变送器电路

图 5-19 为一个用集成变送器 XTR101 将传感器输出电压转为 4～20 mA 电流的二线制变送器电路。 二线制即信号、负载和电源串联， 信号需要远距离传送（可能达几百米），该系统采用直流 24 V 电源。负载可以用串联电流表来指示，也可以用如图所示的 *R*L=240 Ω转换为 0.96～4.8 V 的电压输出来指示。

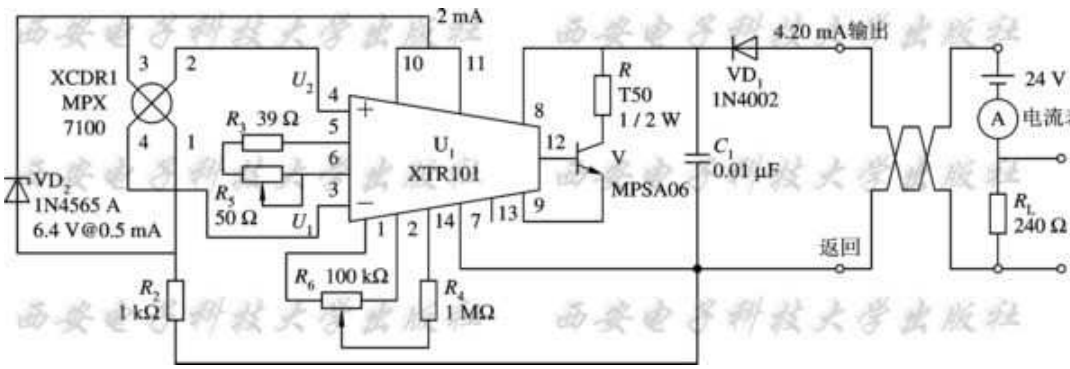


图 5-19 4~20 mA 压力变送器电路

5.6 气体检测电路

1、高灵敏度氢气(煤气泄漏)报警器电路

应用 3DOH 氢敏传感器制作的高灵敏度的氢气报警器电路如图 5-20 所示。由于家用管道煤气中的氢气含量大约为 40%，所以本电路也可以用来作为家用管道煤气的泄漏报警器。

图 5-20 中，场效应管 V1(3DJ6D)接成恒流源形式，作为 3DOH 氢敏传感器内部钯栅 MOS 场效应管的漏极负载，使流过 3DOH 漏极 D 的电流恒定不变，约为几百微安。V2(3DJ6D)也接成恒流源形式，为 3DOH 内部测温二极管提供几百微安的恒定电流。

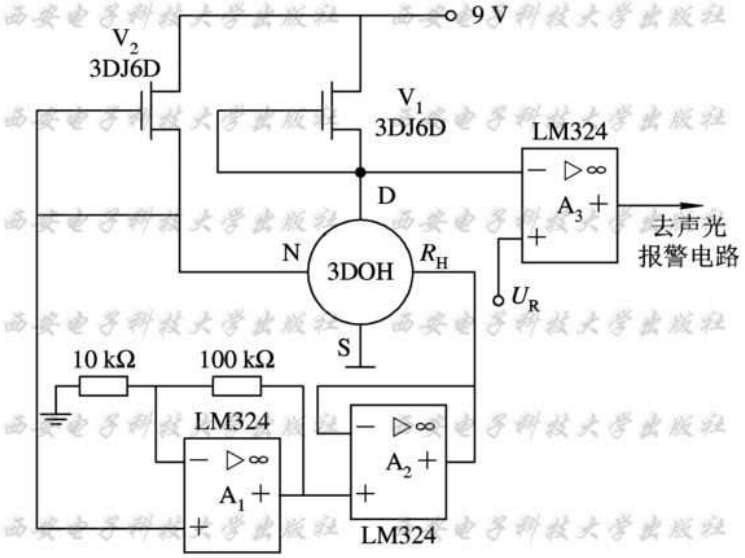


图 5-20 高灵敏度氢气报警器电路

2、可燃性气体泄漏报警器电路

可燃性气体泄漏报警器的电路如图 5-21 所示。图中所用气敏器件为 TGS813，它对一般可燃性气体(如氢气、一氧化碳、 丙烷、 乙醇等)均具有较高的灵敏度， 能用来检测煤气、 液化石油气、 天然气等的泄漏， 也可用来检测如冬季取暖时煤炭燃烧不完全而产生的过量一氧化碳等，以防意外事故的发生。

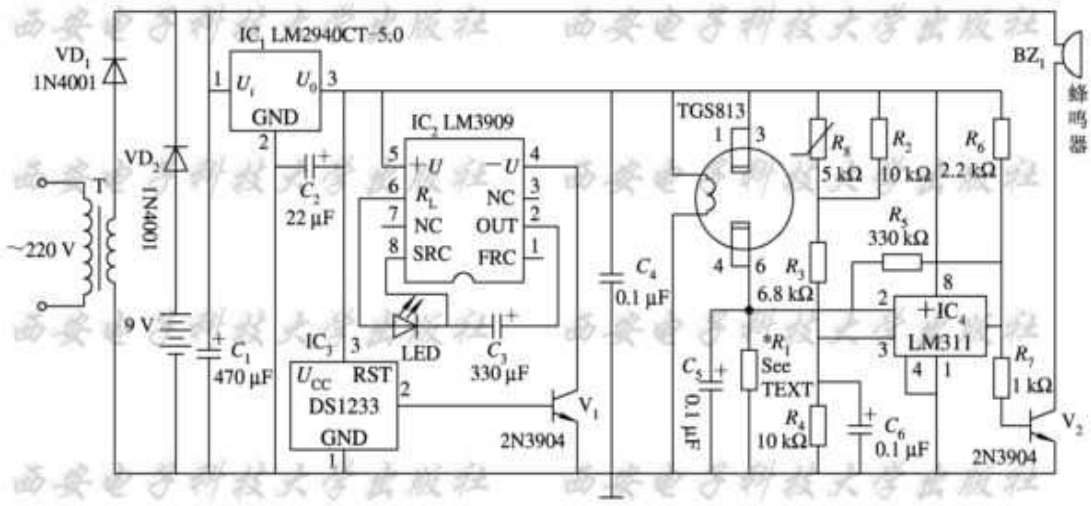


图 5-21 可燃性气体泄漏报警器电路

3、便携式缺氧监控器电路

图 5-22 为便携式缺氧监控器电路。图中用来检测氧气浓度的传感器为原电池式氧气传感器(也称为伽伐尼电池式氧传感器)。原电池式氧气传感器的负极采用金、铂等贵金属制成， 正极采用铅等普通易氧化金属制成，隔膜采用氧气穿透性良好的聚乙烯或氟烯脂制成厚度为 10～20 μm 的薄膜，电解液为酸性电解液。 其工作原理是：氧气穿过隔膜时起化学反应，从而形成电流，电流的大小与氧气的浓度成比例， 通过对电流的检测可以知道氧气的浓度。 这种传感器在 0～100%氧气浓度范围内有线性输出， 在检测空气中的氧气浓度时可输出电压约 50 mV。 调节电位器 RP4 可设定氧气浓度小于 18%时比较器 IC2 输出高电压，通过 VD2 和 *R*6 驱动晶体管 V1 导通，使接于 V1 集电极的蜂鸣器鸣响报警。由于采用液晶显示， 故用 4 节 450 mA/h 的充电电池可连续工作约 100 h。

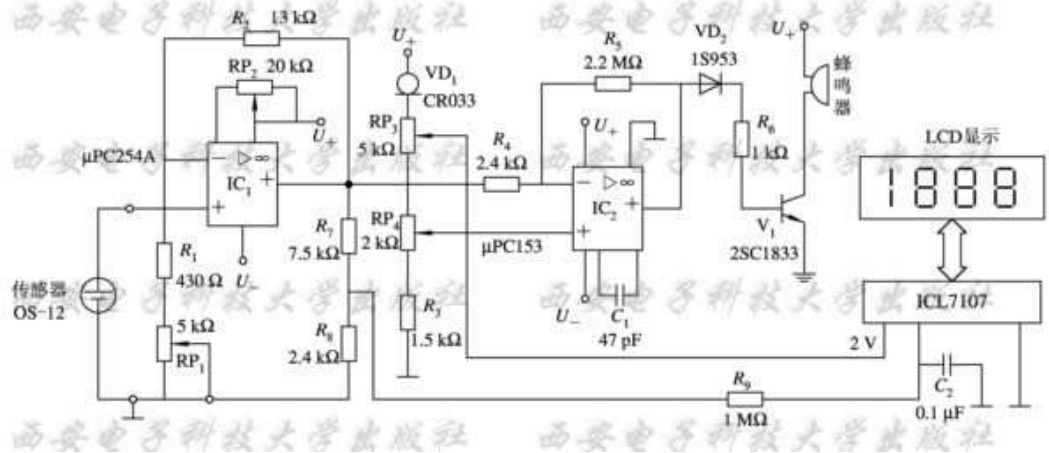


图 5-22 便携式缺氧监控器电路

4、几种常用的半导体气敏器件

UL-281、UL-282、UL-206 和 QM-N10 是几种常用的半导体气敏器件， 它们分别用

于一氧化碳、酒精、烟雾和可燃性气体的检测与报警。表 10-5 中列出了它们的基本参数。

型号 参数及应用	UL－281	UL－282	UL－206	QM－N10
检测气体	一氧化碳	酒精	烟雾	可燃气体
灵敏度 R_0/R_x（R_0 为在空气中的电阻值，R_x 为在某一浓度待测气体中的电阻值）	一氧化碳为 50×10^{-6} 时的灵敏度大于 1000×10^{-6} 酒精灵敏度，1000×10^{-6} 氢气灵敏度	$\frac{R_0}{R_x}>5$ R_x 为在 200×10^{-6} 酒精气体中的电阻值	$\frac{R_0}{R_x}>3$ R_x 为在 700×10^{-6} 烟雾中的电阻值	$\frac{R_0}{R_x}>8$ R_x 为在丁烷浓度为 0.3%时的电阻值
工作电压/V (AC 或 DC)	10±1	15±1.5	15±1.5	—
加热电压/V	清洗：5.5±0.55 工作：0.8±0.08	5±0.5	5±0.5	5±0.5
加热电流/mA	清洗：170～190 工作：25～35	160～180	160～180	—
工作环境温度/℃	－10～50	－10～50	－10～50	－20～50
工作环境湿度/%RH	不大于 95	不大于 95	小于 95	小于 95

5.7 湿度检测技术

1、阻抗式湿度传感器应用电路

阻抗式湿度传感器的应用电路如图 5-23 所示，它适用于 UD-08、CGS-2 等湿度传感器，电路较为简单，精度在±3%RH 左右。当使用其他类型湿度传感器时，应适当调整图中参数。

2、电容式湿度传感器应用电路

电容式湿度传感器应用电路如图 5-24 所示。这种电路适用于 MC-2 等湿度传感器，其灵敏度为 2 mV/%RH。 电路由两个时基电路组成。 第一个时基电路 IC 1 及其外围电路组成多谐振荡器， 由 $R1$、$R2$、$C1$ 提供 20 ms 的脉冲触发第二个时基电路。 第二个时基电路 IC2 及其外围电路是一个可变脉宽发生器， 其脉冲宽度取决于湿敏器件 MC-2 的电容值大小。

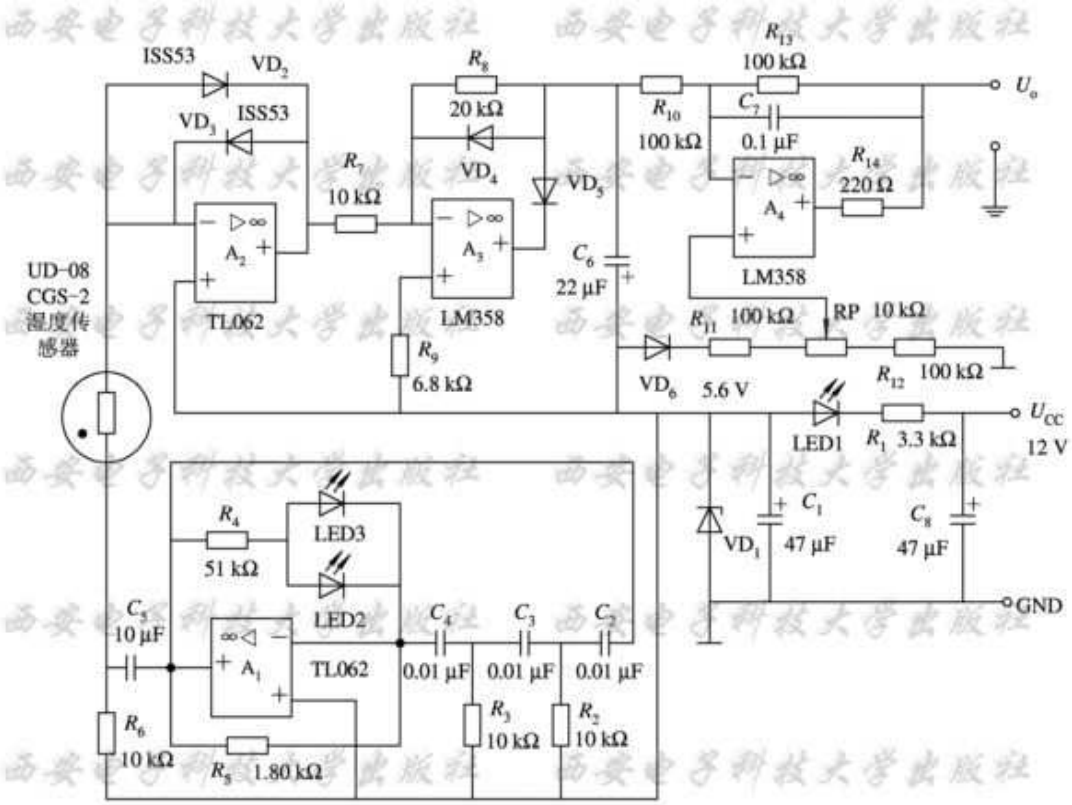


图 5-23 阻抗式湿度传感器应用电路

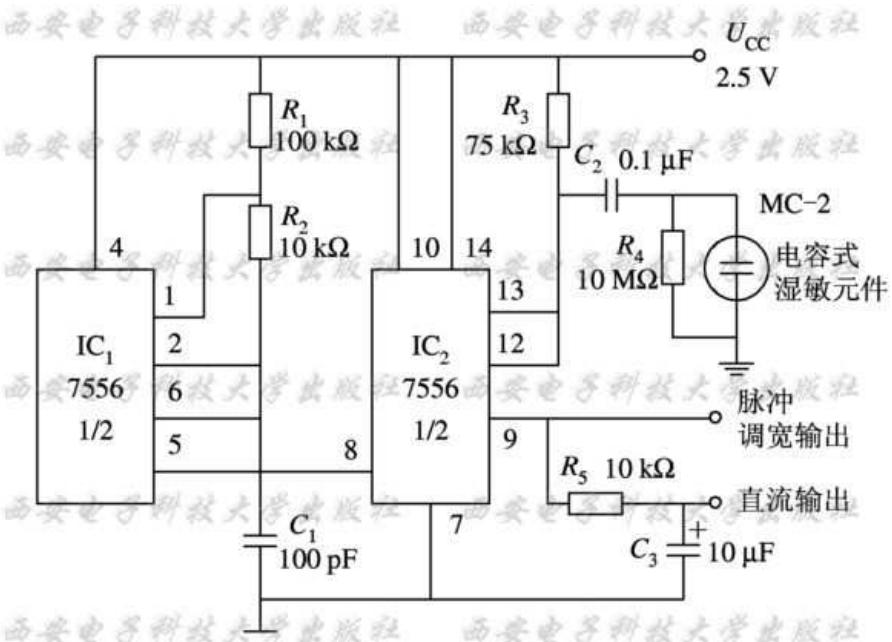


图 5-24 电容式湿度传感器应用电路

5.8 干扰的抑制技术

1、抑制干扰的方法

（1） 消除或抑制干扰源： 如使产生干扰的电气设备远离检测装置； 对继电器、 接触器、 断路器等采取触点灭弧措施或改用无触点开关； 消除电路中的虚焊、 假接等。

（2） 破坏干扰途径： 提高绝缘性能， 采用变压器、 光电耦合器隔离以切断“路”径； 利用退耦、 滤波、 选频等电路手段引导干扰信号转移； 改变接地形式消除共阻抗耦合干扰途径； 对数字信号可采用甄别、 限幅、 整形等信号处理方法或选通控制方法切断干扰途径。

（3） 削弱接收电路对干扰的敏感性： 例如电路中的选频措施可以削弱对全频带噪声的敏感性， 负反馈可以有效削弱内部噪声源， 其他如对信号采用绞线传输或差动输入电路等。

常用的抗干扰技术有屏蔽、 接地、 浮置、 滤波、 隔离技术等。

2、屏蔽技术

1） 静电屏蔽

众所周知， 在静电场作用下， 导体内部各点等电位， 即导体内部无电力线。 因此，若将金属屏蔽盒接地， 则屏蔽盒内的电力线不会传到外部， 外部的电力线也不会穿透屏蔽盒进入内部。 前者可抑制干扰源， 后者可阻截干扰的传输途径。 所以静电屏蔽也叫电场屏蔽， 可以抑制电场耦合的干扰。

为了达到较好的静电屏蔽效果， 应注意以下几个问题：

（1） 选用铜、 铝等低电阻金属材料作屏蔽盒。

（2） 屏蔽盒要良好地接地。

（3） 尽量缩短被屏蔽电路伸出屏蔽盒之外的导线长度。

2） 电磁屏蔽

电磁屏蔽主要是抑制高频电磁场的干扰， 屏蔽体采用良导体材料（铜、 铝或镀银铜板)， 利用高频电磁场在屏蔽导体内产生涡流的效应， 一方面消耗电磁场能量， 另一方面涡电流产生反磁场抵消高频干扰磁场， 从而达到磁屏蔽的效果。 当屏蔽体上必须开孔或开槽时， 应注意避免切断涡电流的流通途径。 若把屏蔽体接地， 则可兼顾静电屏蔽。若要对电磁线圈进行屏蔽， 屏蔽罩直径必须大于线圈直径一倍以上， 否则将使线圈电感量减小，Q 值降低。

3） 磁屏蔽

对低频磁场的屏蔽， 要用高导磁材料， 使干扰磁感线在屏蔽体内构成回路， 屏蔽体以外的漏磁通很少， 从而抑制了低频磁场的干扰作用。 为保证屏蔽效果， 屏蔽板应有一定的厚度， 以免磁饱和或部分磁通穿过屏蔽层而形成漏磁干扰。

3、接地技术

1） 电气、电子设备中的地线

接地起源于强电技术。 为保障安全， 将电网零线和设备外壳接大地， 称为保安地

线。 对于以电能作为信号的通信、 测量、 计算控制等电子技术来说， 把电信号的基准电位点称为“地”， 它可能与大地是隔绝的， 称为信号地线。 信号地线分为模拟信号地线和数字信号地线两种。 另外从信号特点来看， 还有信号源地线和负载地线。

4、其他抑制干扰的措施

在仪表中还经常采用调制、 解调技术， 滤波和隔离（一般用变压器作前隔离， 光电耦合器作后隔离）技术。 通过调制、 选频放大、 解调、 滤波， 只放大输出有用信号， 抑制无用的干扰信号。 滤波的类型有低通滤波、 高通滤波、 带通滤波、 带阻滤波等， 起选频作用。 隔离主要防止后级对前级的干扰。

第6章　遥控技术

遥控技术主要分为红外遥控、无线遥控、超声波遥控和微波遥控等几类。其中，以红外遥控和无线遥控方式使用居多。

6.1 红外遥控

红外遥控技术是一种利用红外线进行点对点通信的技术，其相应的软件和硬件技术都已比较成熟。它在技术上的主要优点是：1、无需专门申请特定频率的使用执照；2、具有移动通信设备所必需的体积小、功率低的特点；3、传输速率适合于家庭和办公室使用的网络；4、信号无干扰，传输准确度高；5、成本低廉。

1、单通道红外遥控电路

在不需要多路控制的应用场合，可以使用由常规集成电路组成的单通道红外遥控电路。这种遥控电路不需要使用较贵的专用编译码器，因此成本较低。

单通道红外遥控发射电路如图 6-1 所示。在发射电路中使用了一片高速 CMOS 型四重二输入“与非”门 74HC00。其中“与非”门 3、4 组成载波振荡器，振荡频率 f0 调在 38kHz 左右；“与非”门 1、2 组成低频振荡器，振荡频率 f1 不必精确调整。f1 对 f0 进行调制，所以从“与非”门 4 输出的波形是断续的载波，这也是经红外发光二极管传送的波形。当 A 点波形为高电平时，红外发光二极管发射载波；当 A 点波形为低电平时，红外发光二极管不发射载波。这一停一发的频率就是低频振荡器频率 f1。 在红外发射电路中为什么不采用价格低廉的低速 CMOS 四重二输入“与非”门 CD4011，而采用价格较高的 74HC00 呢？主要是由于电源电压的限制。红外发射器的外壳有多种多样，但电源一般都设计成 3V，使用两节 5 号或 7 号电池作电源。虽然 CD4011 的标称工作电压为 3～18V，但却是对处理数字信号而言的。因为这里 CMOS“与非”门是用作振荡产生方波信号的，即模拟应用，所以它的工作电压至少要 4.5V 才行，否则不易起振，影响使用。而 74HC 系列的 CMOS 数字集成电路最低工作电压为 2V，所以使用 3V 电源便“得心应手”了。

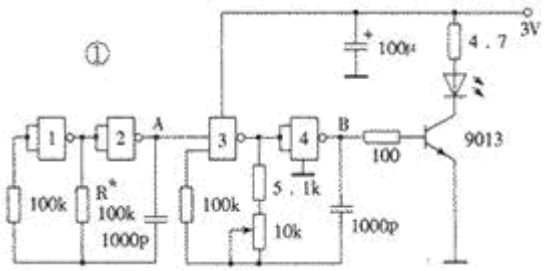


图 6-1 单通道红外遥控发射电路

2、15 通道红外遥控电路

下面介绍一种利用模拟电子开关进行编码和译码的红外遥控电路，15 个编码控制键与

译码输出一一对应，操作简便、响应速度快，可广泛用于电风扇、电动窗帘、装饰彩灯、视听设备及家用电器等的遥控操作。

编码发射电路由时钟振荡器、模拟电子开关、编码控制键、双稳态触发器、红外载频振荡器、与非门调制器、输出驱动器及红外发射管等组成，电路如图 6-2 所示。CD4060 是 14 位二进制串行异步计数器，内含振荡器及 14 级计数单元，但只有 10 个计数输出信号被引出。时钟振荡频率 f=1/2.2RP1・C1，要求 C1≥100pF，RP1≥1kΩ，R1≥10RP1，否则不易起振。本文根据电路需求，通过调整 RP1，使 f=16384Hz。由二进制计数分频公式 fQn=f/2n 不难算得：Q5 端输出频率 fQ5=512Hz 作其准同步脉冲信号使用；Q10 端输出频率 fQ10=16Hz 作计数闸门，决定 Q9-Q6 按 0000～1111 循环变化一周最多允许通过的脉冲个数。

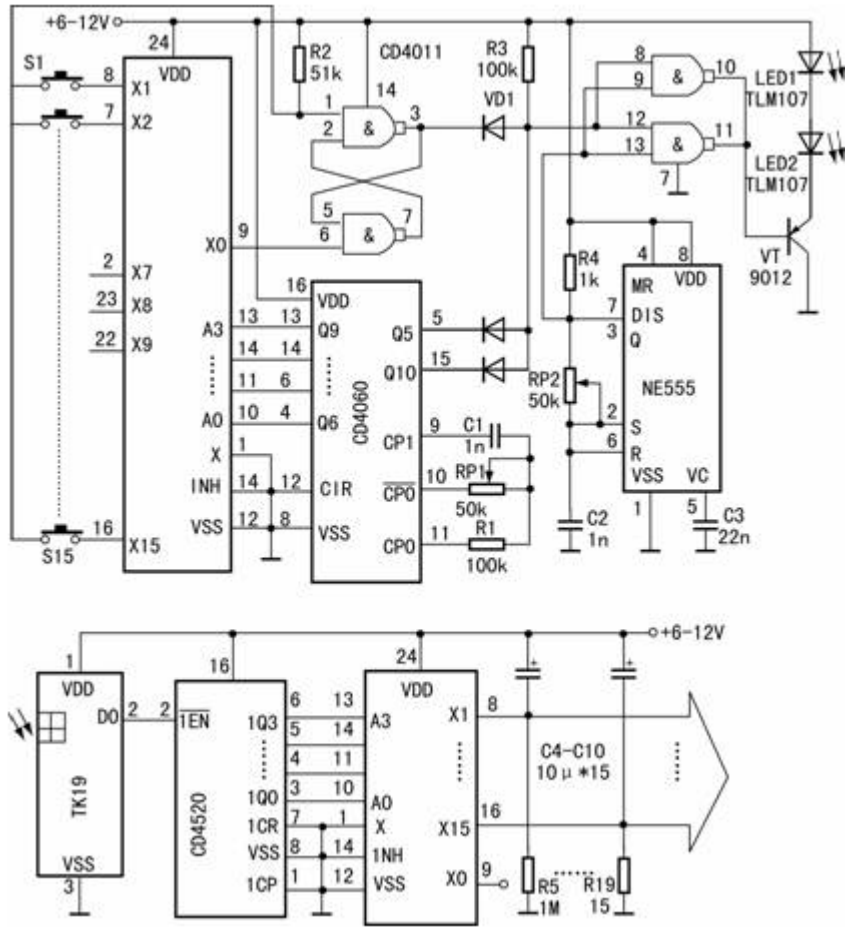


图 6-2 15 通道红外遥控电路

CD4067 是 16 选 1 模拟电子开关，因公共端 X=0，故当 A3～A0 受 CD4060 的 Q9～Q6 控制、按 0000～1111 循环变化时，X0～X15 将依次输出低电平。CD4011 是二输入端器与非门，共有两个与非门组成双稳态输出脉冲负跳变沿触发 R-S 触发器，另两个与非门并联后作调制器。调制器的一个输入端送入 38kHz 红外载频信号，另一端被 R3、VD1～VD3 构成的二极管与门扩展为三个输入端，这三个信号的作用是：R-S 触发器输出信号由

编码开关 S1～S15 控制，决定编码脉冲个数；CD4060 的 Q10 输出 16Hz 信号决定脉冲重复频率；Q10 输出信号作同步脉冲。每当 CD4060 计数到 Q10～Q6=10000 时，CD4067 的 X0=0R-S 触发器被置位。在未编码时，送入调制器的脉冲个数最多，为 16 个；按 S1～S15 进行编码控制时，送入调制器的脉冲个数与闭合的键号相同，比如按 S8 时，有 8 个脉冲送入等。更详细的脉冲时序关系可自行分析。

单时基电路 NE555 构成无稳态多谐振荡器，输出的载波振荡频率可按 f=1.44/R4+RP2C2 估算，调 RP2 使频率为 38kHz 左右。此信号经编码脉冲调制和 VT 放大后，驱动两只红外发光管向外发射红外遥控信息。

接收电路如图所示，一体化红外接收头 TK19（或 BA5302 等）从红外载波信号中解调（去载波）出编码信号，送至 CD4520 进行计数，选通模拟电子开关 CD4067 的 X1～X15 端，输出与编码一一对应的控制信号。CD4520 是双位二进制计数器，这里只用其中一个计数器，并采用脉冲下降沿触发方式。C4～C10 和 R5～R19 组成充放电延时电路，由于电容两端的电压不能突变，可防止按 Kn 编码键时，在 Xn 输出以前小于 Xn 的端子输出控制信号，避免干扰执行电器或电路正常工作。这里因 X=0，所以 X1～X15 输出的是低电平控制信号，若想使用高电平控制信号，可将①脚接正电源，使 X1=1 即可，但此时 C4～C10 和 R5～R19 的位置应互换。

3、编码解码芯片 PT2262/PT2272

PT2262/2272 是台湾普城公司生产的一种 CMOS 工艺制造的低功耗低价位通用编解码电路，PT2262/2272 最多可有 12 位(A0-A11)三态地址端管脚(悬空,接高电平,接低电平),任意组合可提供 531441 地址码,PT2262 最多可有 6 位(D0-D5)数据端管脚,设定的地址码和数据码从 17 脚串行输出，可用于无线遥控发射电路。

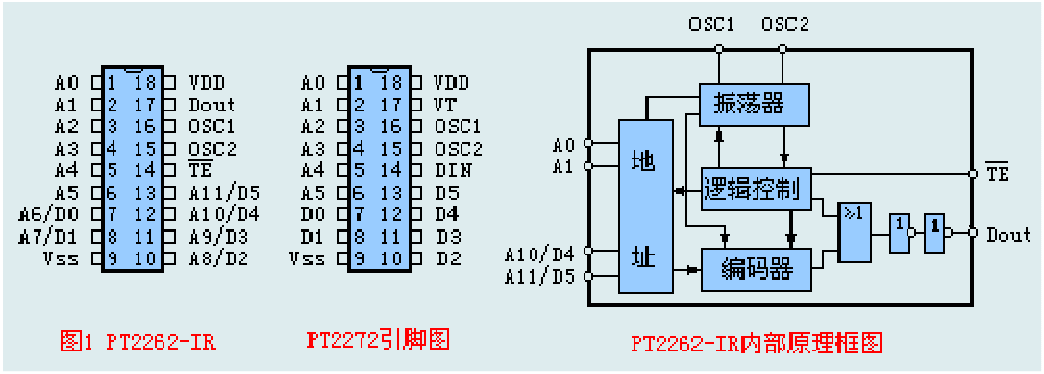


图 6-3 PT2262/2272 引脚及原理

编码芯片 PT2262 发出的编码信号由：地址码、数据码、同步码组成一个完整的码字，解码芯片 PT2272 接收到信号后，其地址码经过两次比较核对后，VT 脚才输出高电平，与此同时相应的数据脚也输出高电平，如果发送端一直按住按键，编码芯片也会连续发射。当发射机没有按键按下时，PT2262 不接通电源，其 17 脚为低电平，所以 315MHz 的高频发射电路不工作，当有按键按下时，PT2262 得电工作，其第 17 脚输出经调制的串行数据信号，当 17 脚为高电平期间 315MHz 的高频发射电路起振并发射等幅高频信号，当 17 脚

为低平期间 315MHz 的高频发射电路停止振荡，所以高频发射电路完全收控于 PT2262 的 17 脚输出的数字信号，从而对高频电路完成幅度键控（ASK 调制）相当于调制度为 100％的调幅。

PT2262/2272 特点：CMOS 工艺制造，低功耗，外部元器件少，RC 振荡电阻，工作电压范围宽：2.6～15v ，数据最多可达 6 位，地址码最多可达 531441 种。应用范围：车辆防盗系统、家庭防盗系统、遥控玩具、其他电器遥控。

PT2262 引脚功能如下：

名称	管脚	说 明
A0-A11	1-8、10-13	地址管脚,用于进行地址编码,可置为“0”,“1”,“f”(悬空),
D0-D5	7-8、10-13	数据输入端，有一个为“1”即有编码发出，内部下拉
Vcc	18	电源正端（＋）
Vss	9	电源负端（－）
TE	14	编码启动端，用于多数据的编码发射，低电平有效；
OSC1	16	振荡电阻输入端，与 OSC2 所接电阻决定振荡频率；
OSC2	15	振荡电阻振荡器输出端；
Dout	17	编码输出端（正常时为低电平）

在具体的应用中，外接振荡电阻可根据需要进行适当的调节，阻值越大振荡频率越慢，编码的宽度越大，发码一帧的时间越长。网站上大部分产品都是用 2262/1.2M＝2272/200K 组合的，少量产品用 2262/4.7M＝2272/820K。

地址码和数据码都用宽度不同的脉冲来表示，两个窄脉冲表示“0”；两个宽脉冲表示“1”；一个窄脉冲和一个宽脉冲表示“F”也就是地址码的“悬空”。

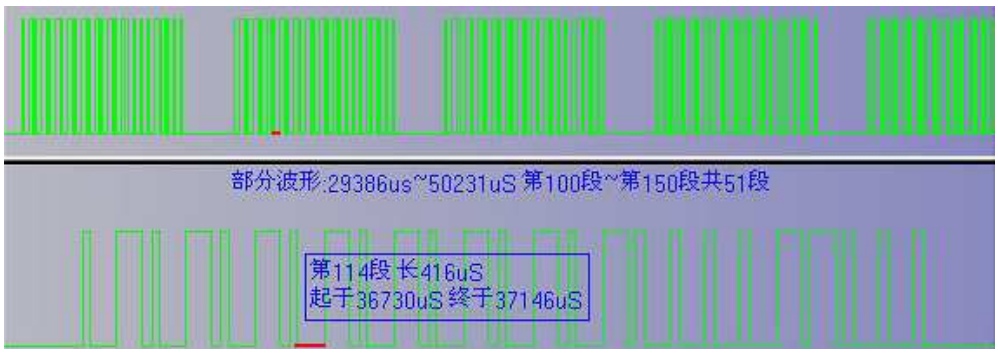


图 6-4 超再生接收模块信号输出波形

上面是我们从超再生接收模块信号输出脚上截获的一段波形，可以明显看到，图上半部分是一组一组的字码，每组字码之间有同步码隔开，所以我们如果用单片机软件解码时，程序只要判断出同步码，然后对后面的字码进行脉冲宽度识别即可。图下部分是放大的一组字码：一个字码由 12 位 AD 码（地址码加数据码，比如 8 位地址码加 4 位数据码）组成，每个 AD 位用两个脉冲来代表：两个窄脉冲表示“0”；两个宽脉冲表示“1”；一个窄脉冲和一个宽脉冲表示“F”也就是地址码的“悬空”。

2262 每次发射时至少发射 4 组字码，2272 只有在连续两次检测到相同的地址码加数据码才会把数据码中的“1”驱动相应的数据输出端为高电平和驱动 VT 端同步为高电平。因

为无线发射的特点，第一组字码非常容易受零电平干扰，往往会产生误码，所以程序可以丢弃处理。

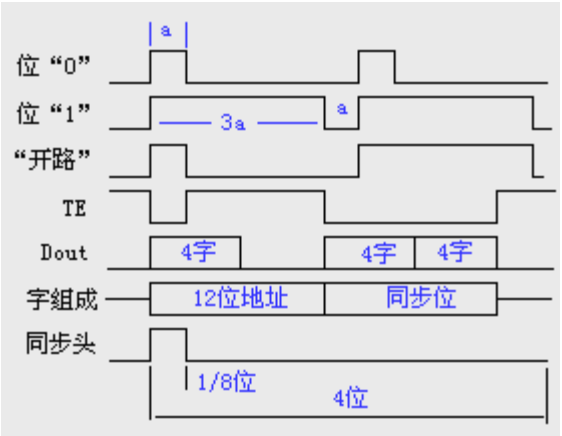


图 6-5 编码波形

PT2272 解码芯片有不同的后缀，表示不同的功能，有 L4/M4/L6/M6 之分，其中 L 表示锁存输出，数据只要成功接收就能一直保持对应的电平状态，直到下次遥控数据发生变化时改变。M 表示非锁存输出，数据脚输出的电平是瞬时的而且和发射端是否发射相对应，可以用于类似点动的控制。后缀的 6 和 4 表示有几路并行的控制通道，当采用 4 路并行数据时(PT2272-M4)，对应的地址编码应该是 8 位，如果采用 6 路的并行数据时(PT2272-M6)，对应的地址编码应该是 6 位。

PT2262/2272 芯片的地址编码设定和修改：

在通常使用中，我们一般采用 8 位地址码和 4 位数据码，这时编码电路 PT2262 和解码 PT2272 的第 1～8 脚为地址设定脚，有三种状态可供选择：悬空、接正电源、接地三种状态，3 的 8 次方为 6561，所以地址编码不重复度为 6561 组，只有发射端 PT2262 和接收端 PT2272 的地址编码完全相同，才能配对使用，遥控模块的生产厂家为了便于生产管理，出厂时遥控模块的 PT2262 和 PT2272 的八位地址编码端全部悬空，这样用户可以很方便选择各种编码状态，用户如果想改变地址编码，只要将 PT2262 和 PT2272 的 1～8 脚设置相同即可，例如将发射机的 PT2262 的第 1 脚接地第 5 脚接正电源，其它引脚悬空，那么接收机的 PT2272 只要也第 1 脚接地第 5 脚接正电源，其它引脚悬空就能实现配对接收。当两者地址编码完全一致时，接收机对应的 D1～D4 端输出约 4V 互锁高电平控制信号，同时 VT 端也输出解码有效高电平信号。用户可将这些信号加一级放大，便可驱动继电器、功率三极管等进行负载遥控开关操纵。

设置地址码的原则是：同一个系统地址码必须一致；不同的系统可以依靠不同的地址码加以区分。至于设置什么样的地址码完全随客户喜欢。

PT2262 和 PT2272 除地址编码必须完全一致外，振荡电阻还必须匹配，否则接收距离会变近甚至无法接收，随着技术的发展市场上出现一批兼容芯片，在实际使用中只要对振荡电阻稍做改动就能配套使用，根据我们的实际使用经验，下面的参数匹配效果较好：

编码发射芯片					编码接收芯片
PT2262	PT2260	SC2260	SC2262	CS5211	PT2272/SC2272/CS5212
1.2M	无	3.3M	1.1M	1.3M	200K
1.5M	无	4.3M	1.4M	1.6M	270K
2.2M	无	6.2M	2M	2.4M	390K
3.3M	无	9.1M	3M	3.6M	680K
4.7M	1.2M	12M	4.3M	5.1M	820K

2262 IR 是 2262 系列用于红外遥控的专用芯片，可以按照下面的图纸进行接线，可以通过调整发射端 Rosc 电阻的大小使接收距离最远，发射端电阻的调整范围 390～420K。

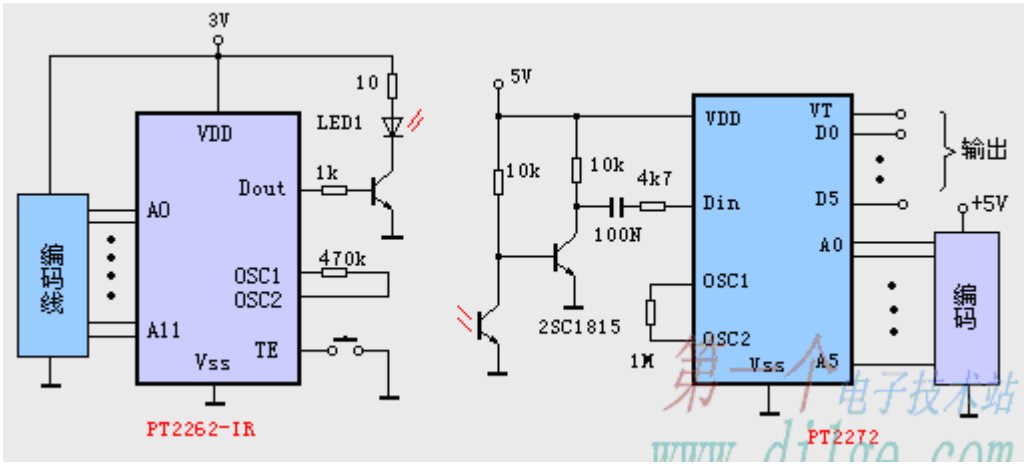


图 6-6 红外发射接收典型电路

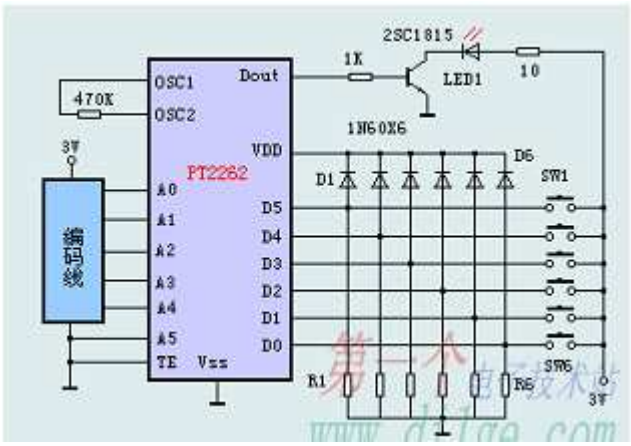


图 6-7 6 位发射应用电路

6.2 无线遥控

无线遥控就是利用电磁波在远距离上，按照人们的意志实现对物体对象的无线操纵和控制，这种无线控制的方式就叫做无线遥控。

无线遥控一般都包括遥控发射机和遥控接收机。发射机主要包括编码电路和发射电路。编码电路由操纵器(操纵开关或电位器等)控制，操纵者通过操纵器；使编码电路产生所需要的控制指令。这些控制指令是具有某些特征的、相互间易于区分的电信号，例如：用频率为 270hz 的正弦信号作为控制左舵的指令，用频率为 350hz 的正弦信号作为控制右舵的指令，即不同频率的正弦信号代表不同的控制指令。除了可利用频率特征外，还可用正弦信号的幅度及相位特征、脉冲信号的幅度、宽度及相位特征以及码组特征等表示各种指令。编码电路产生的指令信号都是频率较低的电信号，无法直接传送到遥控目标上去，还要将指令信号送到发射电路，使它载在高频信号(载波)上，才能由发射天线发送出去。就如同用火车、飞机等运载工具运送货物一样，指令信号相当于货物，载波相当于运载工具。我们把指令信号载到载波上去的过程叫调制，调制作用由发射电路的调制器完成。发射电路的主要作用是产生载波，并由调制器将指令信号调制在载波上，经天线将已调载波发送出去。

接收机由接收电路及译码电路组成。接收电路又包括高频部分及解调器部分。由接收天线送来的微弱信号经接收机高频部分的选择和放大后，送到解调器。就象火车、飞机等运载工具到站后，把货物卸下来的情况一样，解调器的作用是从载波上“卸”下指令信号。由于“卸”下来的各种指令信号是混杂在一起的、还要送到译码电路译码。译码电路的工作就象把卸下来的货物鉴别分类，再分别送到使用场地一样，它对各种指令信号进行签别，送到相应的执行放大电路。执行放大电路把指令信号放大到具有一定的功率，用以驱动执行机构。执行机构将电能转变为机械动作，例如电机的转动、电磁铁的吸动等，带动被控的调节机构(例如舵面)，从而实现对被控目标的控制。

多通道无线遥控电路

本例介绍的多通道无线电遥控电路是采用 P36-F36-J 组成的多通道无线电遥控电路。在发射电路中采用 DTMF 编译码电路将发射按键输入信号转变为 4 位二进制码输人 F36-F 并向外发射。在接收电路中，F36-J 将接收到的遥控信号通过解调译码还原为 4 位二进制码，将 4 位二进制码通过一个 4 一 16 线译码器转变为一个确定的通道控制信号输出，去控制一个相应的电路。

多通道无线电遥控电路组成如图 6-8 和 6-9 所示，包括遥控发射电路和接收译码。

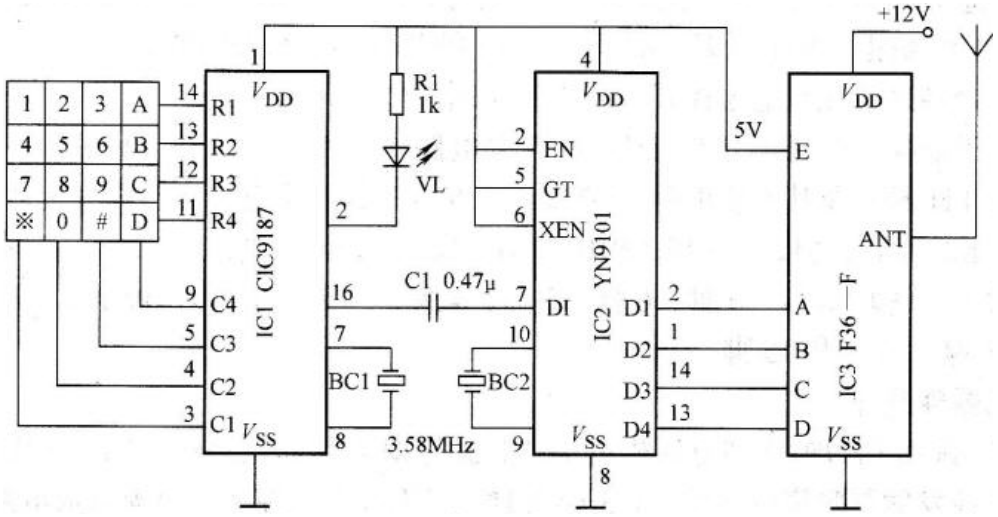


图 6-8 遥控发射电路

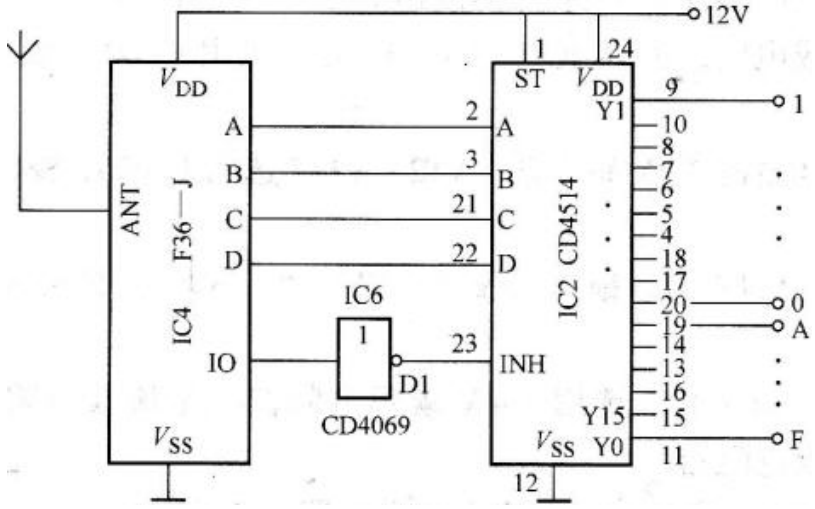


图 6-9 接收译码电路

遥控发射电路包括发射通道输人键盘、CIC9187 DTMF 编码电路、YN9101 DTMF 译码电路和 F36-F 发射电路。

本电路中 CIC9187 DTMF 编码器的按键输人端 R1 一 R4 与 Cl 一 C4 和键盘组成一个按键输人编码电路。当按下键盘中的某一按键时，CIC9187 编码器的输出端 16 脚就会串行输出一组与按键对应的编码脉冲。由于发射电路 F36-F 的输人端需要的是 4 位二进制码，而不是串行脉冲，因此通过 YN9101 DTMF 译码器将串行脉冲转变为 4 位二进制码输出。当 F36-F 接收到由 YN9101 输出的 4 位二进制码时，通过对内部载频进行调制后向外发射。CIC9187 和 YN9101 的工作电源由发射电路矜 6-F 的 E 端输出的 5V 电源提供。由于 4 位二进制码只能组成，l6 个十进制数，因此，该遥控发射器最多能组成 16 个发射通道。

无线电接收译码电路如图所示，由无线电接收组件 F36-J 和 4 一 16 线译码器电路

CD4514 组成。当接收电路 F36-J 接收到由发射电路发出的遥控信号时，通过内部电路解调译码后，由输出端 A 一 D 输出 4 位二进制码。

CD4514 为 4 一 16 线译码器，它能将输入的 4 位二进制码转换为对应的 16 个高电平输出。CD4514 为 24 脚扁平塑料封装结构，它有 4 个输人端 A、B、C, D,输人范围为 0000 一 1111;有 16 个输出端 YO 一 Y15,输出高电平有效；一个禁止端 INH，高电平时禁止输入低电平时允许输人；一个选通控制端 ST，高电平时，输出端输出状态决定于输人端，低电平时，输出端保持原输出状态。在本电路中，ST 端接电源，F36-J 的 10 端输出的高电平经反相器 D1 反相后变为低电平加至 INH 端。在译码输出端 A 一 D 输出译码的同时，10 端输出高电平，经 D1 变为低电平加至 INH 端，使 CD4514 允许输人。当 CD4514 的输人端 A 一 D 输人由 F36-F 输出的二进制码后，它的输出端 YO-Y15 输出对应的高电平。

CD4514 的输出端根据需要可连接双向晶闸管、光祸合器，固态继电器或经晶体管驱动电路接直流继电器。

IC1 选用 CIC9187 DTMF 拨号编码集成电路；IC2 选用 YN9001 DTMF 译码集成电路；IC3 选用 F36-F 无线遥控发射集成电路；IC4 选用 F36-J 无线遥控接收集
成电路；IC5 选用 CD4514 或 MC14514 4 一 16 线译码器集成电路；IC6 选用 CD4069 六反相器数字集成电路。

CD4514 也可用 CD4515 代用，它们的工作原理、引脚功能完全相同，只是 CD4515 的输出端为输出低电平有效。在发射电路中，CIC9187 DTMF 编码电路可用 52559、MK5087、MK5089、以及 9188 等代用。YN9101 DTMF 译码电路可用 YN9102、MT8870 等代用。

其他元器件均无特殊要求，可按图所标型号及参数进行选用。

第二部分 Protel DXP

Protel DXP 是 Alium 公司最新推出的一款绘制原理图与 PCB 图的 EDA 工具，它功能强大、使用方便、操作灵活，是国内最流行的 EDA 工具之一。

第 1 章 芯片封装形式的特点和优点

为了适应不同场合，集成电路芯片采用了多种封装形式，表 1-1 列出了常用的芯片封装形式。

表 1-1　封装形式与形状对照表

通孔封装		表面安装封装	
	DIP (双列直插式封装)		SO 或 SOP (小外形封装)
	SH - DIP (收缩双列直插式封装)		QFP (四边扁平封装)
	SK - DIP, SL - DIP (膜状双列直插式，细长双列直插式)		LCC (无引线芯片载体)
	SIP (单列直插式封装)		PLCC, SOJ (有粗端引线塑料芯片载体)
	ZIP (Z 字形直插式封装)		BGA 球栅阵列
	PGA (针栅阵列或柱形封装)		TCP (带载封装)
			CSP (芯片规模封装)

一、DIP 双列直插式封装

DIP(Dual－line Package)是指采用双列直插形式封装的集成电路芯片，绝大多数中小规模集成电路(IC)均采用这种封装形式，其引脚数一般不超过 100 个。采用 DIP 封装的 CPU 芯片有两排引脚，需要插入到具有 DIP 结构的芯片插座上。当然，也可以直接插在有相同焊孔数和几何排列的电路板上进行焊接。DIP 封装的芯片在从芯片插座上插拔时应特别小

心，以免损坏引脚。

DIP 封装具有以下特点：

1.适合在 PCB(印刷电路板)上穿孔焊接，操作方便。

2.芯片面积与封装面积之间的比值较大，故体积也较大。

Intel 系列 CPU 中 8088 就采用这种封装形式，缓存(Cache)和早期的内存芯片也是这种封装形式。

二、QFP 塑料方型扁平式封装和 PFP 塑料扁平组件式封装

QFP（Plastic Quad Flat Package）封装的芯片引脚之间距离很小，管脚很细，一般大规模或超大型集成电路都采用这种封装形式，其引脚数一般在 100 个以上。用这种形式封装的芯片必须采用 SMD（表面安装设备技术）将芯片与主板焊接起来。采用 SMD 安装的芯片不必在主板上打孔，一般在主板表面上有设计好的相应管脚的焊点。将芯片各脚对准相应的焊点，即可实现与主板的焊接。用这种方法焊上去的芯片，如果不用专用工具是很难拆卸下来的。

PFP（Plastic Flat Package）方式封装的芯片与 QFP 方式基本相同。唯一的区别是 QFP 一般为正方形，而 PFP 既可以是正方形，也可以是长方形。

QFP/PFP 封装具有以下特点：

1.适用于 SMD 表面安装技术在 PCB 电路板上安装布线。

2.适合高频使用。

3.操作方便，可靠性高。

4.芯片面积与封装面积之间的比值较小。

Intel 系列 CPU 中 80286、80386 和某些 486 主板采用这种封装形式。

三、PGA 插针网格阵列封装

PGA(Pin Grid Array Package)芯片封装形式在芯片的内外有多个方阵形的插针，每个方阵形插针沿芯片的四周间隔一定距离排列。根据引脚数目的多少，可以围成 2-5 圈。安装时，将芯片插入专门的 PGA 插座。为使 CPU 能够更方便地安装和拆卸，从 486 芯片开始，出现一种名为 ZIF 的 CPU 插座，专门用来满足 PGA 封装的 CPU 在安装和拆卸上的要求。

ZIF(Zero Insertion Force Socket)是指零插拔力的插座。把这种插座上的扳手轻轻抬起，CPU 就可很容易、轻松地插入插座中。然后将扳手压回原处，利用插座本身的特殊结构生成的挤压力，将 CPU 的引脚与插座牢牢地接触，绝对不存在接触不良的问题。而拆卸 CPU 芯片只需将插座的扳手轻轻抬起，则压力解除，CPU 芯片即可轻松取出。

PGA 封装具有以下特点：

1.插拔操作更方便，可靠性高。

2.可适应更高的频率。

Intel 系列 CPU 中，80486 和 Pentium、Pentium Pro 均采用这种封装形式。

四、BGA 球栅阵列封装

随着集成电路技术的发展，对集成电路的封装要求更加严格。这是因为封装技术关系到产品的功能性，当 IC 的频率超过 100MHz 时，传统封装方式可能会产生所谓的“CrossTalk”现象，而且当 IC 的管脚数大于 208 Pin 时，传统的封装方式有其困难度。因此，除使用 QFP 封装方式外，现今大多数的高脚数芯片（如图形芯片与芯片组等）皆转而

使用 BGA(Ball Grid Array Package)封装技术。BGA 一出现便成为 CPU、主板上南/北桥芯片等高密度、高性能、多引脚封装的最佳选择。

BGA 封装技术又可详分为五大类：

1.PBGA（Plasric BGA）基板：一般为 2-4 层有机材料构成的多层板。Intel 系列 CPU 中，Pentium II、III、IV 处理器均采用这种封装形式。

2.CBGA（CeramicBGA）基板：即陶瓷基板，芯片与基板间的电气连接通常采用倒装芯片（FlipChip，简称 FC）的安装方式。Intel 系列 CPU 中，Pentium I、II、Pentium Pro 处理器均采用过这种封装形式。

3.FCBGA（FilpChipBGA）基板：硬质多层基板。

4.TBGA（TapeBGA）基板：基板为带状软质的 1-2 层 PCB 电路板。

5.CDPBGA（Carity Down PBGA）基板：指封装中央有方型低陷的芯片区（又称空腔区）。

BGA 封装具有以下特点：

1.I/O 引脚数虽然增多，但引脚之间的距离远大于 QFP 封装方式，提高了成品率。

2.虽然 BGA 的功耗增加，但由于采用的是可控塌陷芯片法焊接，从而可以改善电热性能。

3.信号传输延迟小，适应频率大大提高。

4.组装可用共面焊接，可靠性大大提高。

BGA 封装方式经过十多年的发展已经进入实用化阶段。1987 年，日本西铁城（Citizen）公司开始着手研制塑封球栅面阵列封装的芯片（即 BGA）。而后，摩托罗拉、康柏等公司也随即加入到开发 BGA 的行列。1993 年，摩托罗拉率先将 BGA 应用于移动电话。同年，康柏公司也在工作站、PC 电脑上加以应用。直到五六年前，Intel 公司在电脑 CPU 中（即奔腾 II、奔腾 III、奔腾 IV 等），以及芯片组（如 i850）中开始使用 BGA，这对 BGA 应用领域扩展发挥了推波助澜的作用。目前，BGA 已成为极其热门的 IC 封装技术，其全球市场规模在 2000 年为 12 亿块，预计 2005 年市场需求将比 2000 年有 70%以上幅度的增长。

五、CSP 芯片尺寸封装

随着全球电子产品个性化、轻巧化的需求蔚为风潮，封装技术已进步到 CSP(Chip Size Package)。它减小了芯片封装外形的尺寸，做到裸芯片尺寸有多大，封装尺寸就有多大。即封装后的 IC 尺寸边长不大于芯片的 1.2 倍，IC 面积只比晶粒（Die）大不超过 1.4 倍。

CSP 封装又可分为四类：

1.Lead Frame Type(传统导线架形式)，代表厂商有富士通、日立、Rohm、高士达（Goldstar）等等。

2.Rigid Interposer Type(硬质内插板型)，代表厂商有摩托罗拉、索尼、东芝、松下等等。

3.Flexible Interposer Type(软质内插板型)，其中最有名的是 Tessera 公司的 microBGA，CTS 的 sim-BGA 也采用相同的原理。其他代表厂商包括通用电气（GE）和 NEC。

4.Wafer Level Package(晶圆尺寸封装)：有别于传统的单一芯片封装方式，WLCSP 是将整片晶圆切割为一颗颗的单一芯片，它号称是封装技术的未来主流，已投入研发的厂商包括 FCT、Aptos、卡西欧、EPIC、富士通、三菱电子等。

CSP 封装具有以下特点：

1.满足了芯片 I/O 引脚不断增加的需要。

2.芯片面积与封装面积之间的比值很小。
3.极大地缩短延迟时间。

CSP 封装适用于脚数少的 IC，如内存条和便携电子产品。未来则将大量应用在信息家电（IA）、数字电视（DTV）、电子书（E-Book）、无线网络 WLAN／GigabitEthemet、ADSL／手机芯片、蓝芽（Bluetooth）等新兴产品中。

六、MCM 多芯片模块

为解决单一芯片集成度低和功能不够完善的问题，把多个高集成度、高性能、高可靠性的芯片，在高密度多层互联基板上用 SMD 技术组成多种多样的电子模块系统，从而出现 MCM(Multi Chip Model)多芯片模块系统。

MCM 具有以下特点：

1.封装延迟时间缩小，易于实现模块高速化。
2.缩小整机/模块的封装尺寸和重量。
3.系统可靠性大大提高。

第 2 章 绘制单片机试验板

本章通过单片机试验板实例，介绍一个双面 PCB 的完整设计过程。对一些具体设计技巧和操作进行了重点介绍，如网络名称的使用、二层板的规划、全局修改、敷铜、补泪滴等。

2.1 原理图设计

本例欲绘制的原理图如图 2-1A 和 2-1B 所示。下面分别介绍各部分原理图的绘制方法。

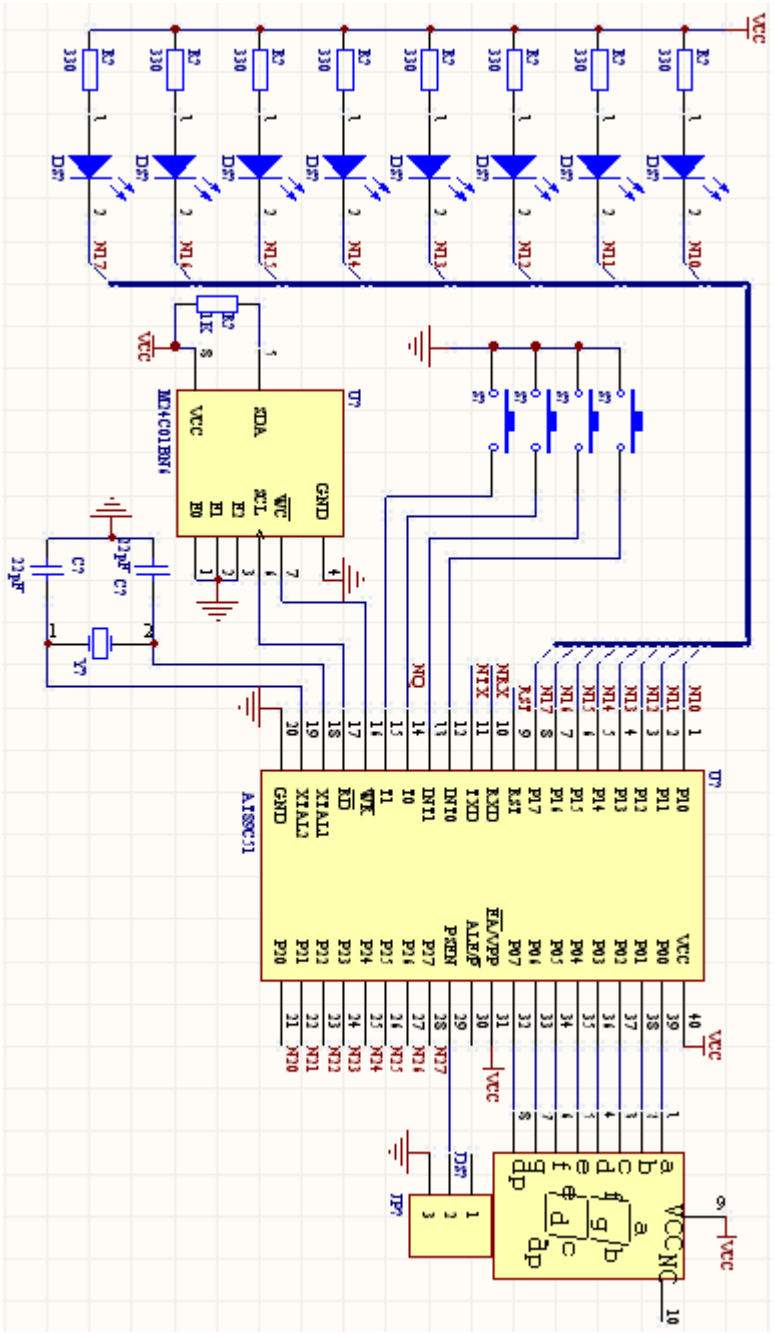


图 2-1A　单片机系统原理图

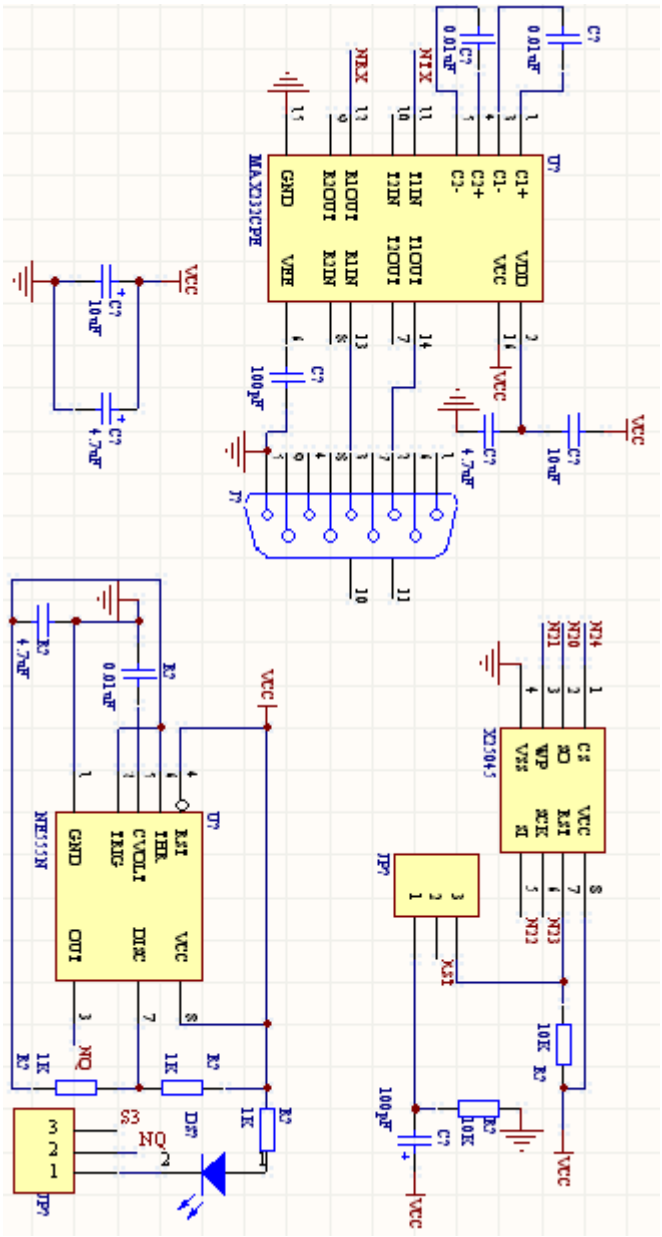


图 2-1B　其它部分原理图

2.1.1 新建 PCB 工程

首先打开 Protel DXP 软件，然后按照图 2-2 的方式新建空白 PCB 工程。再按照图 2-3 的方式保存该 PCB 工程，在弹出的对话框中选好合适的路径，并命名为“单片机试验板.PRJPCB”，如图 2-4 所示。

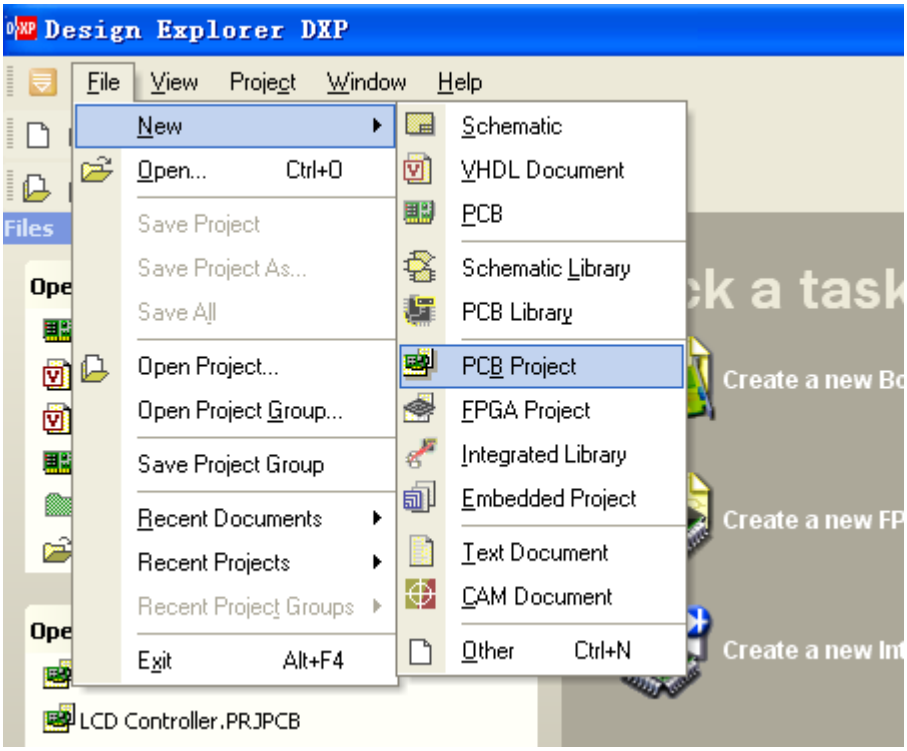

Design Explorer DXP
File View Project Window Help
New
Open... Ctrl+O
Save Project
Save Project As...
Save All
Open Project...
Open Project Group...
Save Project Group
Recent Documents
Recent Projects
Recent Project Groups
Exit Alt+F4
Schematic
VHDL Document
PCB
Schematic Library
PCB Library
PCB Project
FPGA Project
Integrated Library
Embedded Project
Text Document
CAM Document
Other Ctrl+N
Files
LCD Controller.PRJPCB

图 2-2


Design Explorer DXP
File View Project Window Help
New
Open... Ctrl+O
Save Project
Save Project As...
Save All
Open Project...
Open Project Group...
Save Project Group
Recent Documents
Recent Projects
Recent Project Groups
Exit Alt+F4
Group
Project
Pick a task
Create a new Board Lev
Create a new FPGA Desi
Create a new Integrated

图 2-3

图 2-4

2.1.2 新建原理图文件

右键单击“单片机试验板.PRJPCB”，如图 2-5 所示，新建原理图文件，右侧出现的白色空白区域即为原理图绘制区。保存该原理图文件，与 PCB 工程路径一致，重命名为“单片机试验板.SchDoc”。

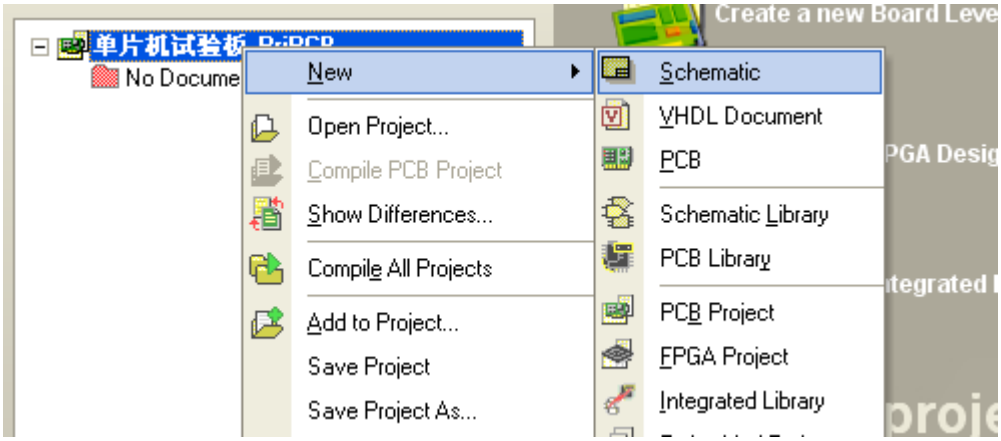


图 2-5

2.1.3 设置原理图纸张大小

单击“Design”菜单，选择“Options”命令，就会弹出“Document Option”对话框。

在对话框的“Standard Style”区通过下拉菜单设定图纸大小为 A4。实际中应根据需要设定图纸大小。

2.1.4 放置元件

下面向原理图中放置元件，先只放置主要的 6 个芯片元件。对于这 6 个元件，哪些在 Protel DXP 自带的元件库中有，哪些没有，暂时都不清楚。先利用 Protel DXP 的元件查找功能来搜索元件。

1、 搜索元件

鼠标移到 Protel DXP 右上角的【Libraries】标签上，系统会自动弹出【Libraries】面板，如图 2-6 所示。然后点击该面板上的【Search】按钮，弹出图 2-7 所示的对话框。

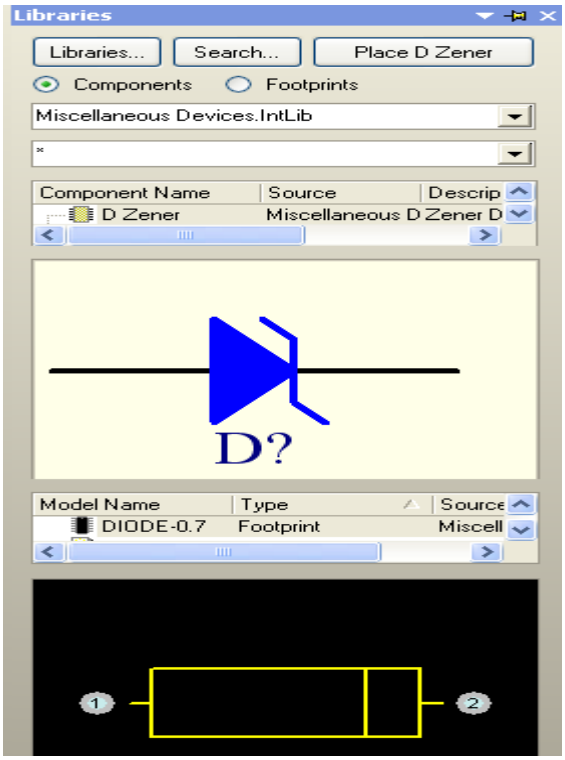


图 2-6

将对话框中库文件所在路径设置为 Protel DXP 安装文件夹中的库文件所在位置，这里的设置如图 2-7 所示。【Search Crietria】区域中【Name】栏中填入的是要搜寻的元件名称。

（1）首先填入“*89C51*”，然后点击【Search Libraries】对话框中的【Search】按钮，进行搜索。这时，对话框会自动切换到“Results”标签页下，并在对话框下方显示正在搜索的库文件。

注意：在元件名称前后最好都加上通配符“*”，因为很难确保元件库中该元件的名称正好和输入的搜索名称一字不差，否则会出现搜索不利的结果。

搜索完毕，对话框显示空白，说明 Protel DXP 自带的集成元件库中没有该元件，需要自己绘制它的原理图符号。这放在后面介绍，下面继续搜索其它元件。

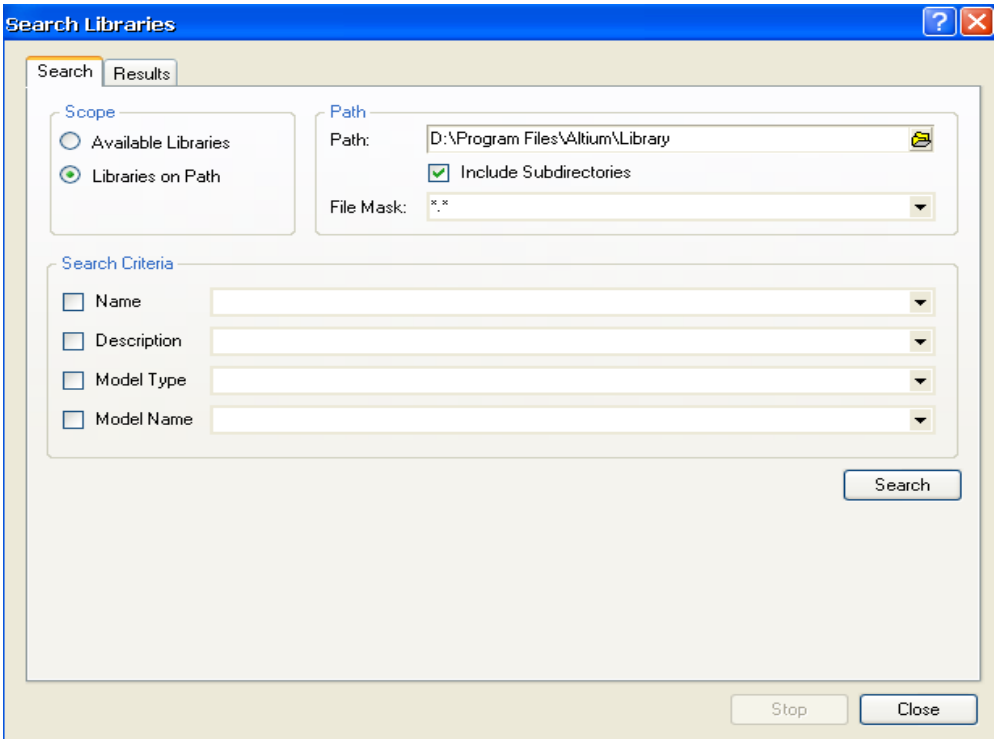


图 2-7 【Search Libraries】对话框

（2）搜索“*NE555*”，结果如图 2-8 所示。

其中，NE555N 就是本例所需要的，当然除了“NE555D”和“NE555FK”的 PCB 封装不同外，其它几个元件的原理图好 PCB 封装均相同，可以换用。

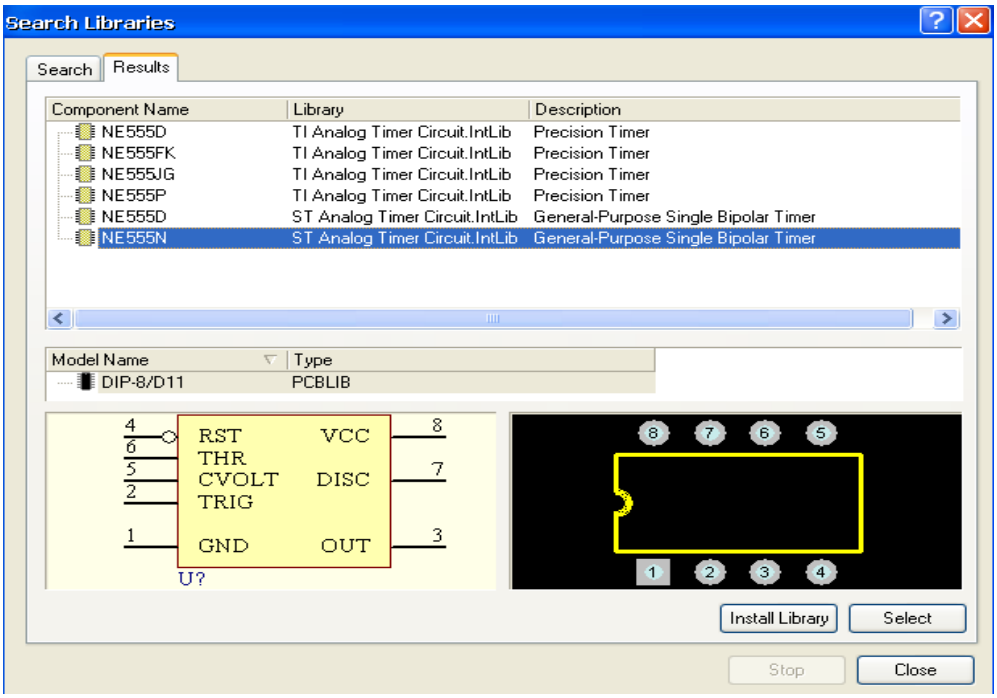


图 2-8　搜索“*NE555*”的结果显示

（3）点击“Select”按钮，则会载入该库文件并直接自动切换到【Libraries】面板。

（4）双击【Libraries】面板中的元件名“NE555N”，将其放置在原理图中。

（5）参照上面的步骤和操作，搜寻“MAX232”元件，并将其放入原理图中。

（6）下一个是“24C01”芯片，搜寻结果如图 2-9 所示。

在元件列表栏中出现这么多元件，是不是难以选择？其实只要通过对话框下面的原理图和 PCB 封装的示意图，就可以发现这些元件的原理图符号好 PCB 封装其实只有两种：一种是贴片式，一种是直插式。本例采用直插式，因此选择第一种器件。

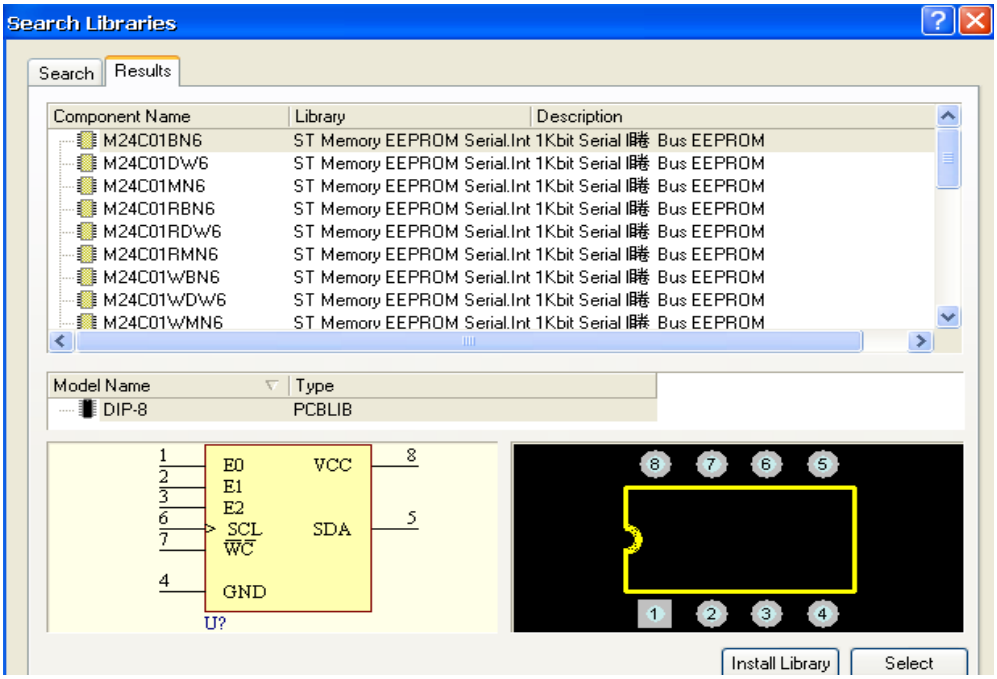


图 2-9　搜寻“24C01”的结果显示

（7）“X25045”芯片也没有搜索到，需要自己来创建。

此时可关闭【Libraries】面板，因为其它元件可直接在“Miscellaneous Devices.IntLib”中直接通过观察原理图符号示意图找到，如（8）中所述。

（8）寻找共阳极数码管。

如图 2-10 所示，单击工具栏上的  图标，会弹出图 2-11 所示的对话框。

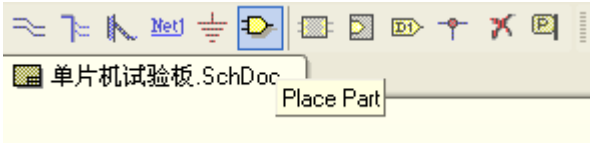


图 2-10　工具栏

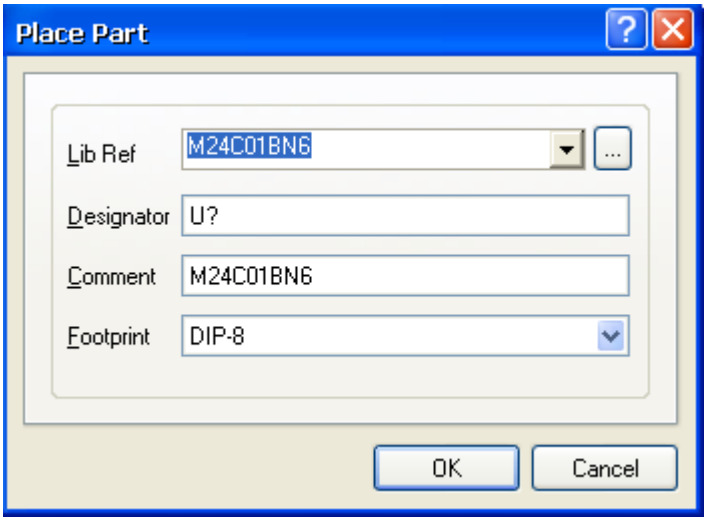


图 2-11　放置元件对话框

单击图 2-11 中的  按钮，就会弹出图 2-12 所示的浏览库对话框，在其中选择“Miscellaneous Devices.IntLib”库，在下面就可以找到所需的共阳极数码管，即“Dpy Red-CA”，左键单击，使其变蓝。注意，后面两个字符是“CC”的，为共阴极数码管。

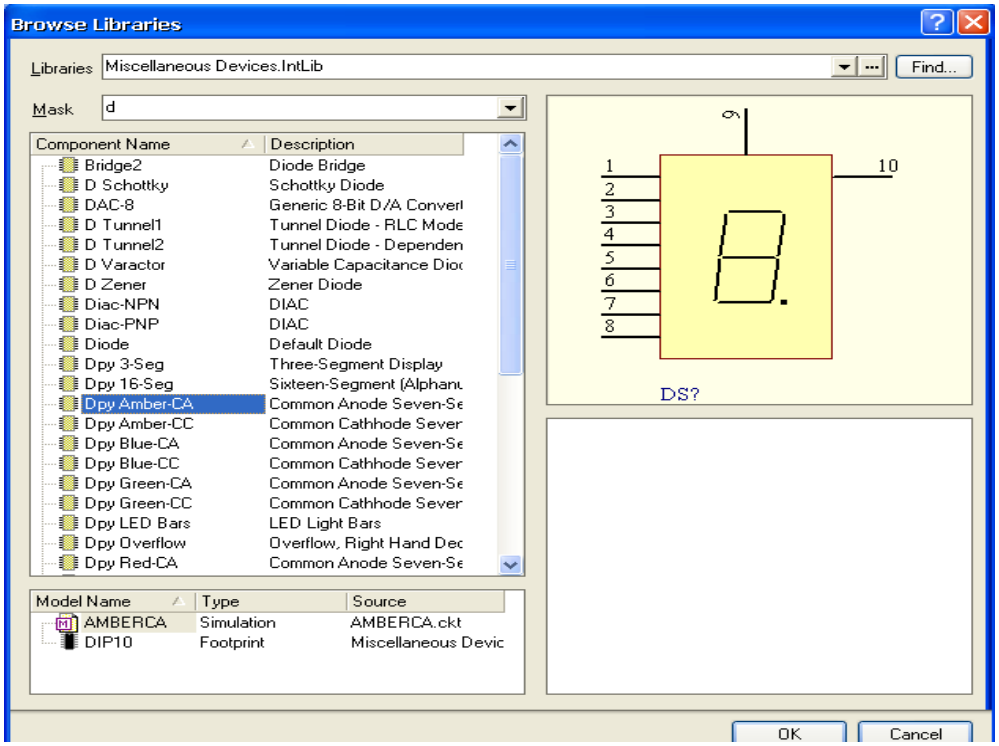


图 2-12　浏览库对话框

然后单击 OK 按钮，再接着出现的对话框中也单击 OK 按钮，该共阳极数码管原理图就跟随着鼠标了，在合适的位置单击左键，即可放置该元件，此方式可连续放置多个同样

的元件，如放置完毕可单击右键，再次区选择别的器件，也可单击“Cancel”按钮结束元件的选择。

2、 创建原理图库

并不是所有元件在自带库中都能找到，找不到的元件就只能自己去建立其原理图库，同样也必须建立其 PCB 库。

（1） 创建单片机原理图库

右键单击“单片机试验板.PRJPCB”，在弹出的菜单中执行【New】/【Schematic Library】，创建一个原理图库文件，并保存为“单片机试验板.SCHLIB”。注意不同文件的后缀。

单击面板下方的“Library Editor”标签，切换到【Library Editor】面板，如图 2-13 所示。

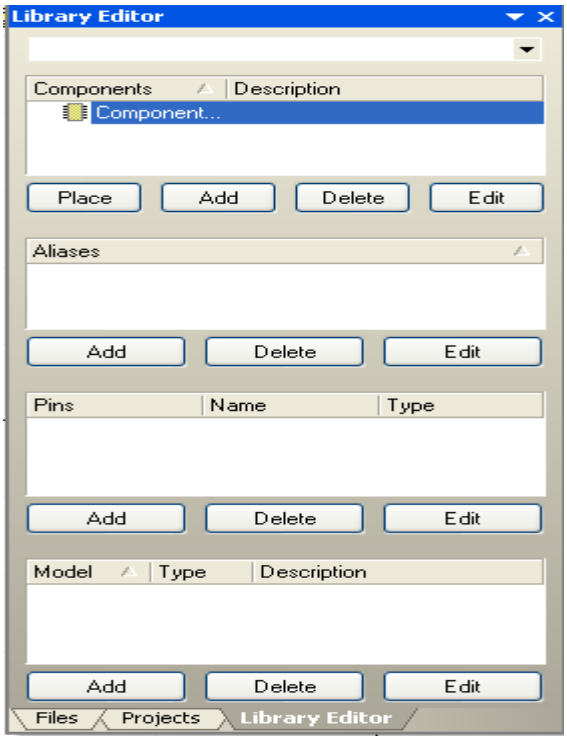


图 2-13 【Library Editor】面板

单击元件列表栏下的“Edit”按钮（最上面的），在弹出的【Library Component Properties】对话框中，将【Designator】项设为“U?”，【Library Ref】项设为“AT89C51”，如图 2-14 所示。确定后，就可以在绘图区开始绘制该元件的原理图符号了。

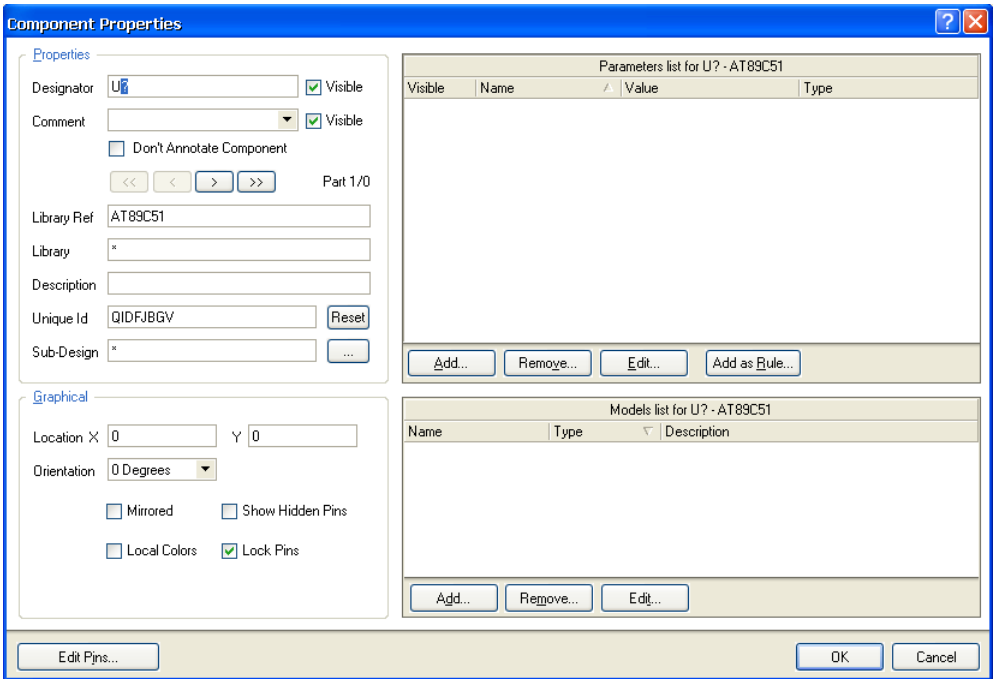


图 2-14 【Library Component Properties】对话框

单击工具栏上的图标，绘制元件轮廓。在绘图区单击左键，可确定绘图的起始点，然后拖动鼠标，画出一个合适大小的矩形区域，然后在单击左键即可。矩形大小还可以用鼠标拖动边框来调整。

单击工具栏上的图标，可以放置元件各管脚。如图 2-15 所示。

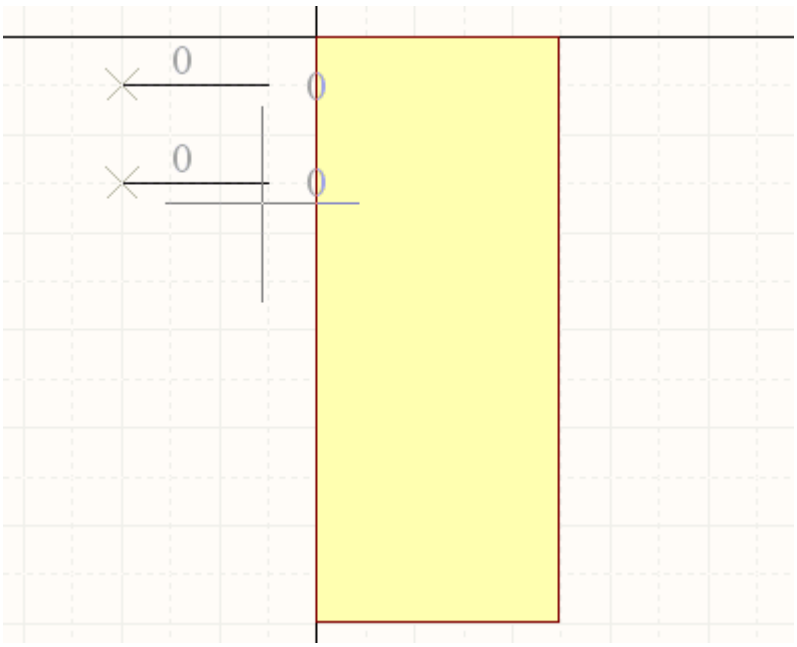


图 2-15　放置管脚

在管脚放置前，可按空格键进行管脚的旋转。注意，引脚带有叉号的为电气连接端，应朝外。在管脚放置前，还要更改管脚名称，按“Tab”键，会弹出图 2-16 所示的对话框。

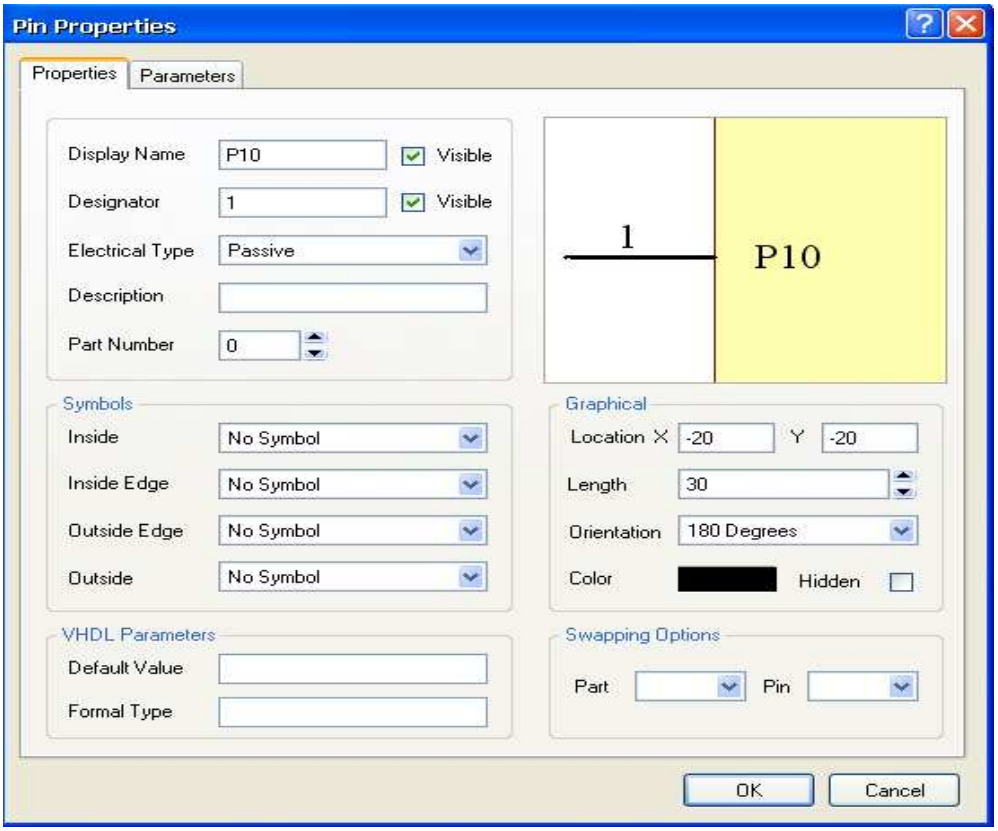


图 2-16　引脚属性对话框

更改对话框中的“Display Name”为合适的引脚名称，更改“Designator”为合适的管脚序号，然后单击 OK 按钮。每个管脚都要进行以上操作，引脚名称与序号都递增的除外。创建好的原理图见图 2-17。

注意：管脚 16 的写法为“W\R\”，其在图上就显示为字符上加横线。注意保存。

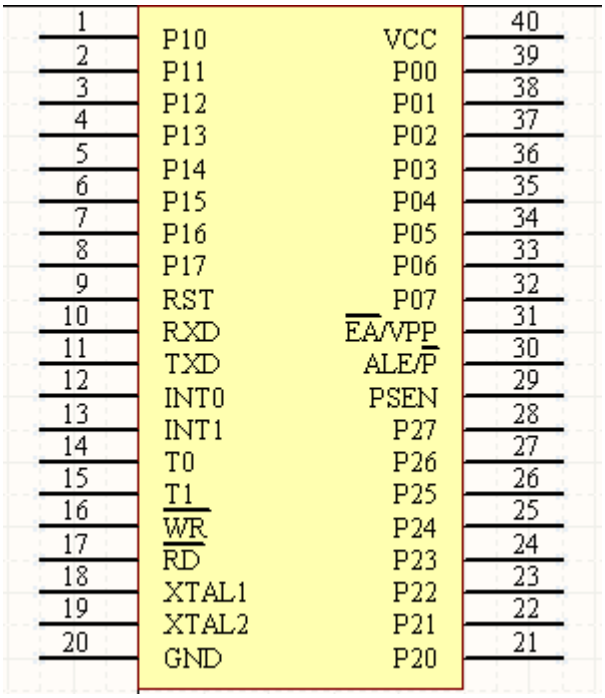


图 2-17　单片机原理图符号

（2）　创建 X25045 原理图库

单击元件下面的“Add”按钮，在弹出的对话框中输入“X25045”，然后单击 OK，在原理图库中就新添加了该元件。在右边空白区以上述方法绘制该元件的原理图。最后结果如图 2-18 所示。

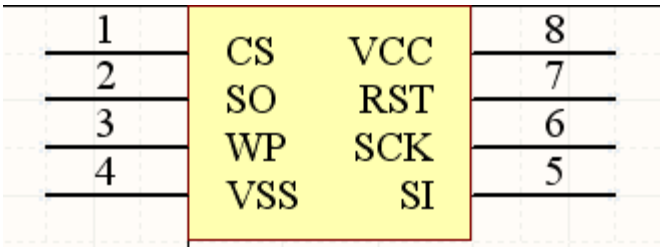


图 2-18　X25045 原理图符号

保存库文件，然后将两个元件托放入原理图中，拖放方法与拖放数码管类似。最后，原理图上放置了 6 个主要元件。

技巧：单击不放某个元件，按“空格”键可以旋转；按“X”键可以左右对调旋转；按“Y”键，可以使元件上下对调旋转。

2.1.5 AT89C51 的电路连接

1、阵列粘贴

在元件库“Miscellaneous Device.IntLib”中找到元件“LED3”和“Res2”，将其放入原理图纸中。本例只用到它们的原理图符号，而它们自带的封装将在后面的设计过程中进

行调换。

两元件的放置如图 2-19 所示。

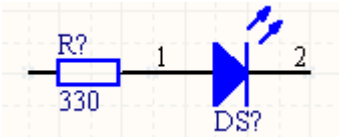


图 2-19 电阻与 LED 的放置

注意：双击任一元件可打开其属性对话框，可在其中更改元件名称、属性、参数等。将“Res2”的“Value”设置为 330。

下面进行阵列粘贴工作。

（1）用鼠标光标圈住图 2-19 所示的区域，选定两个元件。然后按下“Ctrl”和“C”键进行复制操作，出现十字光标后，在光标点击位置选取复制参考点。

（2）点击工具栏上的按钮，执行阵列粘贴命令。在弹出的对话框中，参数设置如图 2-20 所示。

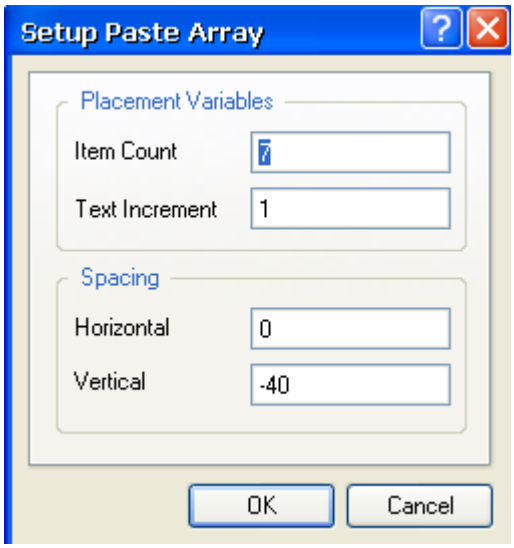


图 2-20 阵列粘贴对话框

【Item Count】：项目个数，即要重复粘贴复制的元件个数，设为 7；

【Text Increment】：粘贴中元器件序号的增量，取默认值“1”；

【Horizontal】：参考点的水平间距，设为“0”；

【Vertical】：参考点的垂直间距，设为“-40”，由上往下排列。

（3）单击 OK 按钮，光标变为十字后，在绘图区单击鼠标左键选择阵列粘贴的其实位置。阵列从单击鼠标处开始粘贴，如图 2-21 所示。

然后点击工具栏上的按钮，添加导线；点击按钮，按“Tab”键，在弹出的对话框中可以选择电源的形式以及符号，单击 OK 后即可添加电源符号。发光二极管电路连接结果如图 2-22 所示。

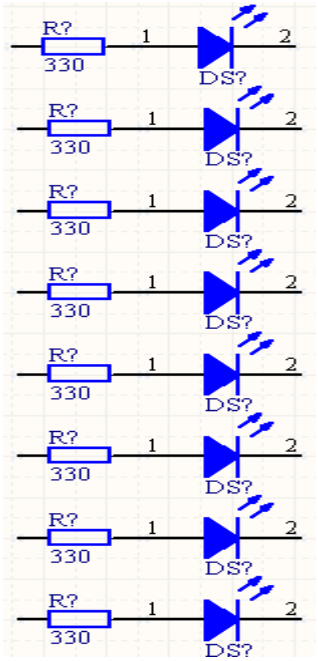


图 2-21 阵列粘贴结果

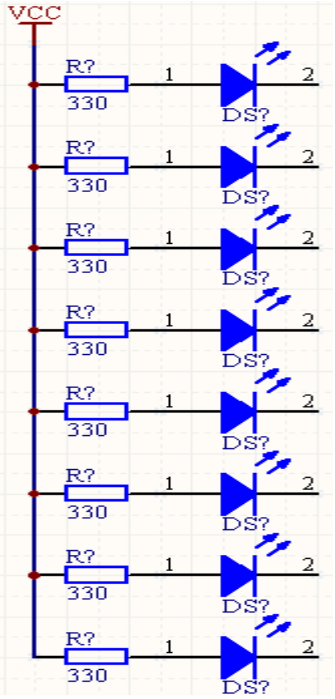


图 2-22 电路连接结果

2、绘制总线和添加网络名称

为了便于读图，看清不同元件间的电气连接关系，可以绘制总线。所谓总线就是代表数条并行导线的一条线。总线本身没有任何电气连接意义，电气连接关系还是要靠网络名称来定义。使用总线代替一组导线时，通常需要与总线分支线来配合。

操作步骤如下：

（1）单击工具栏中 按钮。

（2）出现十字光标后，可以绘制总线。绘制方法与绘制导线一样。在每一个转折点单击鼠标左键确认绘制的这一段总线，在终点单击鼠标左键确认。单击右键可结束总线的绘制。绘制好的总线如图 2-23 所示。

（3）绘制总线分支线。单击工具栏上的 按钮，十字光标出现，并带有总线分支线。按空格键可改变总线分支线的方向。将十字光标移到合适的位置，单击左键，即将总线分支线放置在光标当前位置。只要不单击右键取消操作，下一个分支线会自动附着在光标处，可以继续放置。绘制好的总线及分支线如图 2-23 所示。

（4）单击 按钮，将各元件管脚与总线分支线用导线相连。

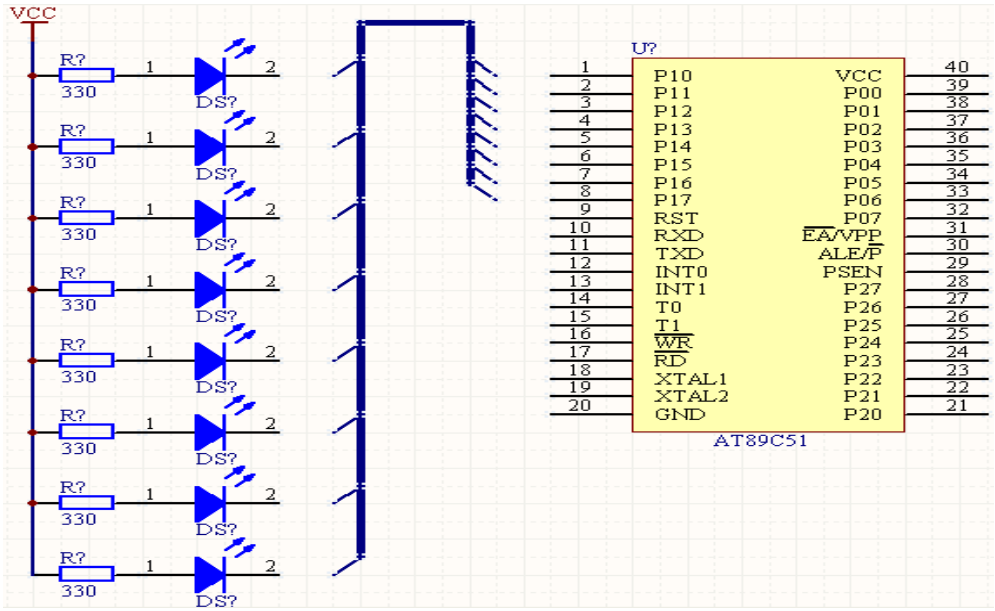


图 2-23　绘制好的总线及分支线

3、添加网络名称

电路原理图中除了通过导线连接来定义元件间的电气联系外，也可以通过网络名称来实现元件间的电气连接。所谓网络名称，就是一个电气节点。具有相同网络名称的 电源符号、接地符号、元件管脚、导线等在电气关系上是连接在一起的。在一些复杂的原理图中，直接画导线会使图样杂乱无章，而使用网络名称，可以使整张图样清晰易懂。总线不代表实际的电气连接关系，所以在绘制完总线后仍需要添加网络名称。

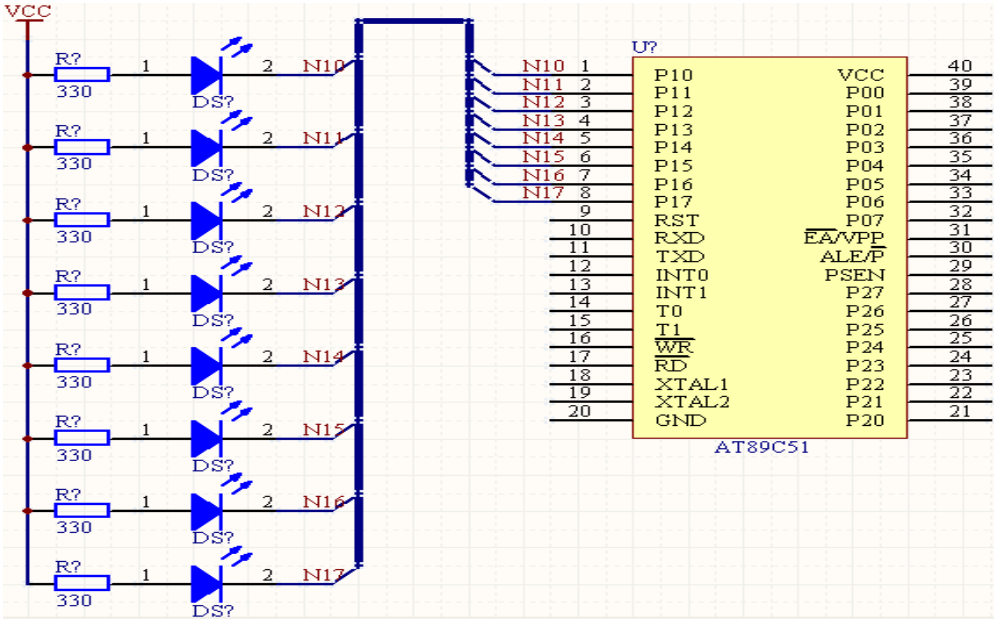


图 9-24　放置网络后的发光二极管电路

（1）单击工具栏上的 Net 按钮，光标变为十字形，并出现一个随光标移动的带虚线框的网络名称。

（2）单击“Tab”键，弹出属性对话框，将网络名称改为需要的符号，还可以设置网络名称的颜色及字体等。

（3）将网络名称对应地放在单片机的管脚及 LED 管脚上。当红色米字形电气捕捉标志出现时，再点击左键确定。最后结果如图 2-24 所示。

4、开关连接

在库文件中选择元件“SW-PB”（按键），如图 2-25 所示连接。

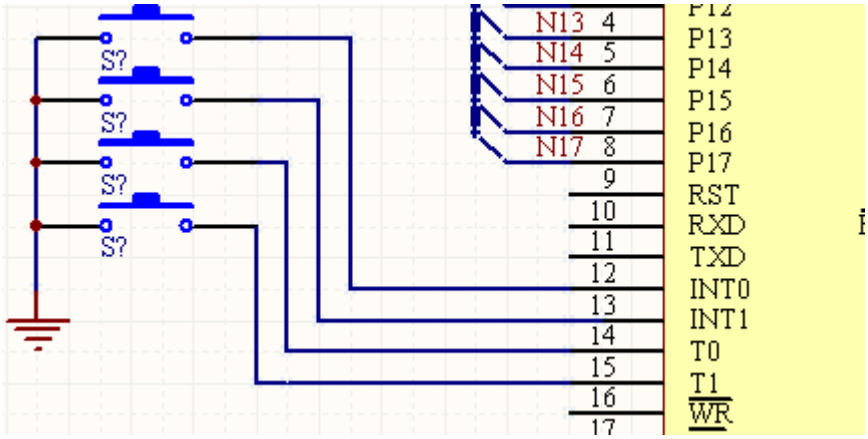


图 2-25　4 个开关的连接电路

5、晶振电路

在库中取一个晶振 XTAL 和两个电容 C，组成晶振电路。见图 2-26。

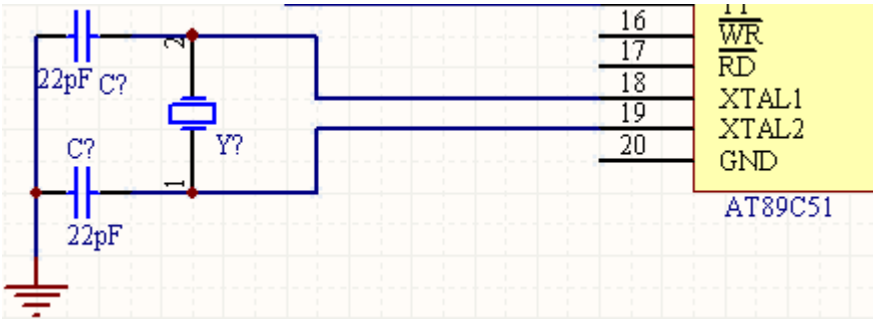


图 2-26　晶振电路

6、24C01 的连接

见图 2-27。

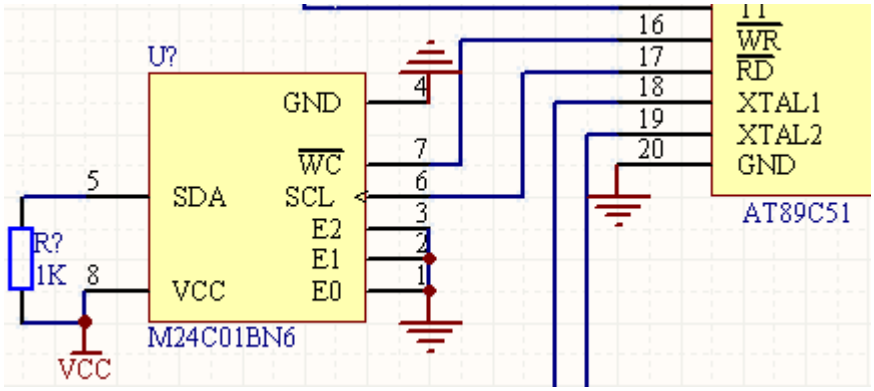


图 2-27　24C01 连接电路

7、跳线连接

管脚 29 连接一个跳线（在“Miscellaneous Connectors.IntLib”库中）。如图 2-28 所示。

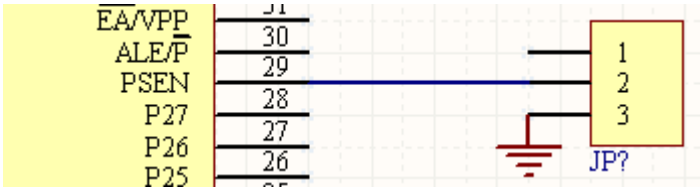


图 2-28　跳线连接电路

8、数码管连接

见图 2-29。

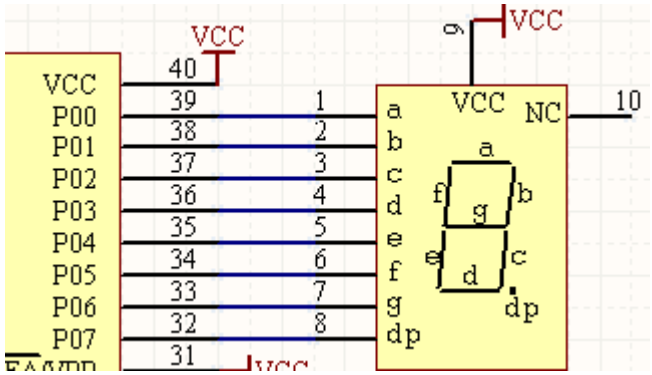


图 2-29　数码管连接电路

9、单片机其它管脚处理

40 脚连电源，20 脚接地，10、11 脚分别连网络标志 NRX 和 NTX，30 脚悬空，9 脚连网络标志 RST，21~28 脚连网络标志 N20~N27，31 脚接 VCC。

2.1.6 555 连接电路

见图 2-30。

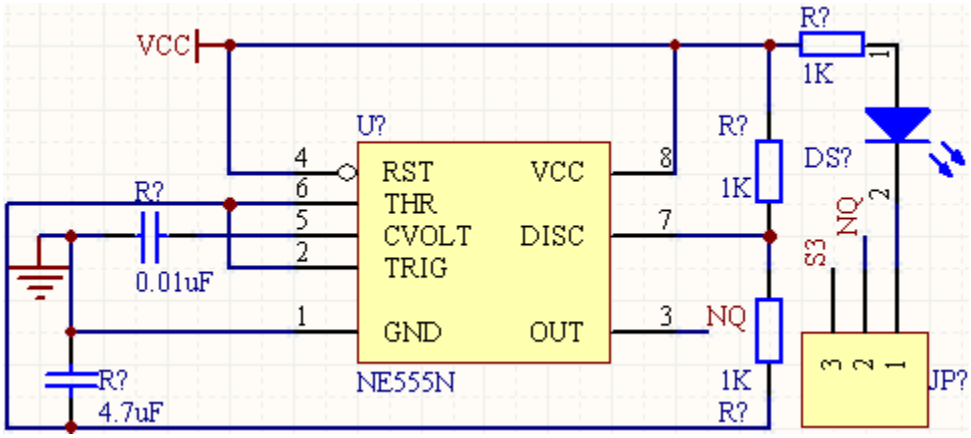


图 2-30　555 连接电路

2.1.7 复位电路

X25045 是一款看门狗电路，可在系统程序出现死锁时自动复位单片机。复位电路见图 2-31。

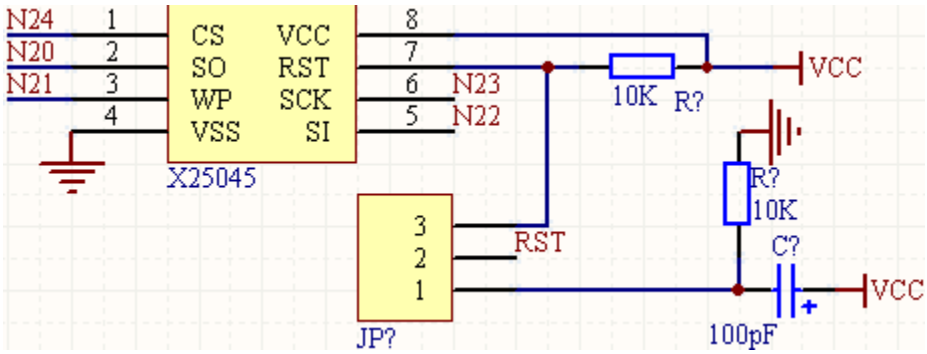


图 2-31　复位电路

2.1.8 串行接口电路

见图 2-32 。

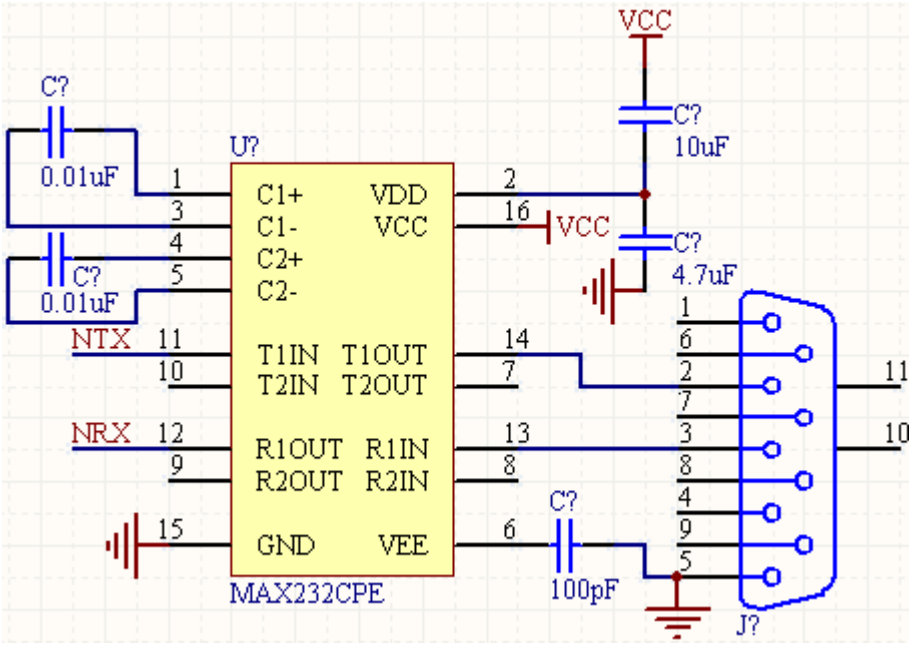


图 2-32　串行接口电路

2.1.9 重新编排元件序号和 ERC 检查

重新编排原理图中所有元件的序号，执行菜单命令【Tools】/【Annotate】后，弹出图 2-33 所示的对话框，选择第 4 种排序方式。

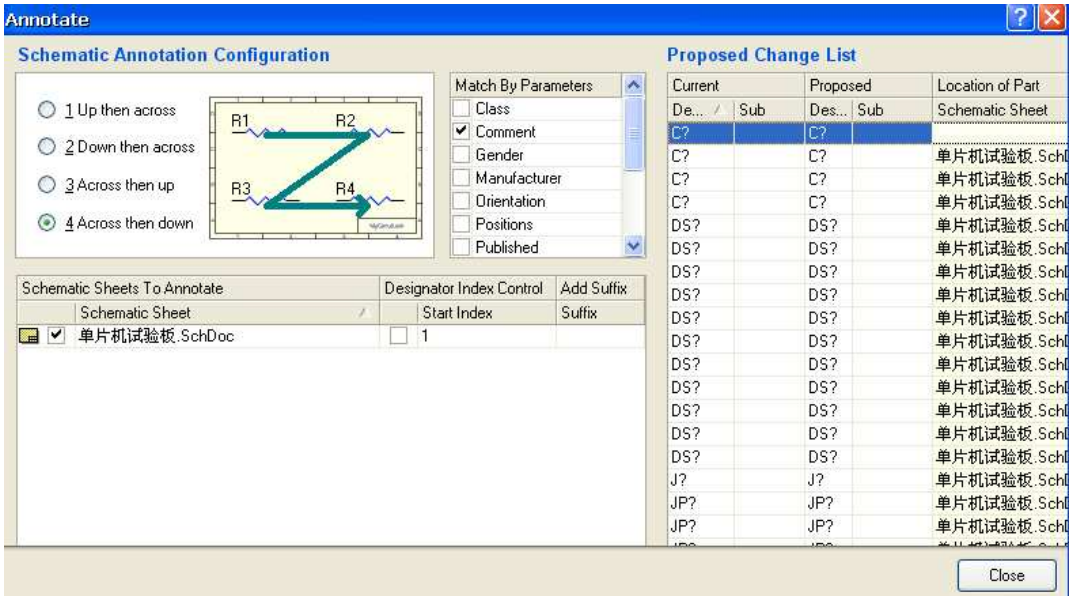


图 2-33　【Annotate】对话框

单击 Update Changes List 按钮后，再单击弹出的小对话框的 OK 按钮，然后单击 Accept Changes (Create ECO) 按钮，就会弹出图 2-34 所示的对话框。单击该对话框上的 Execute Changes 按钮，然后单击 Close 按钮，即完成了重新编排元件序号。

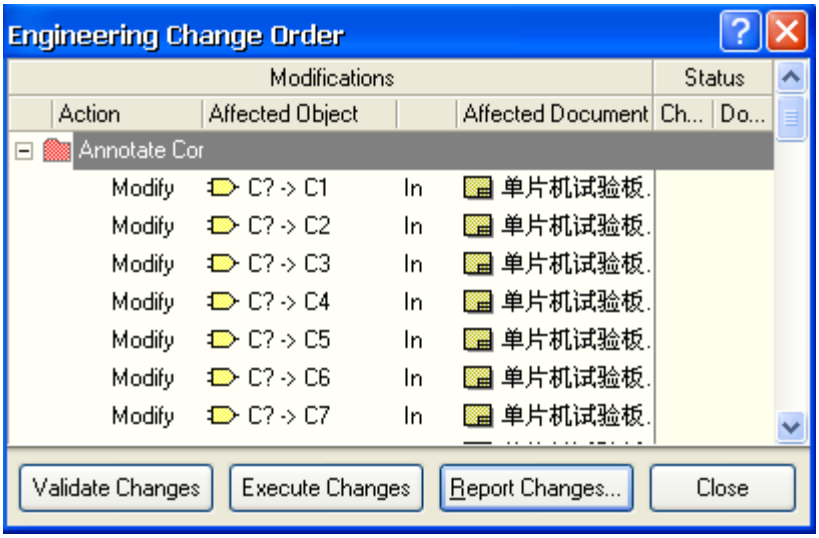


图 2-34　重新编排元件序号对话框

执行菜单命令【Project】/【Compiler All Projects】，进行 ERC 校验，弹出图 2-35 所示的对话框。如果编译原理图后，没有自动弹出【Message】对话框，可点击绘图区下方的【Message】标签。

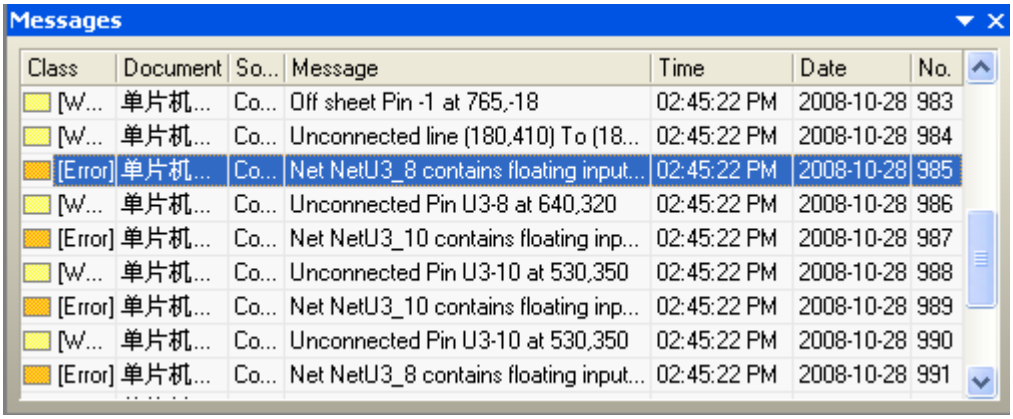


图 2-35 【Message】对话框

其中，图上灰色阴影所示的 Error 是指 MAX232 的第 8 引脚没有连接，没有关系，其它的警告等也没关系，读者请自己查阅。

2.2 PCB 设计

2.2.1 元件的 PCB 封装准备

本例中，元件的 PCB 封装准备工作包括两部分。一是创建自己的 PCB 元件库，即绘制 Protel DXP 自带元件库中没有的 PCB 封装；二是在原理图中，对没有 PCB 封装和需要调整 PCB 封装的元件指定 PCB 封装。

1、 绘制元件的 PCB 封装

首先创建 PCB 库和原理图中电阻使用的“AXIAL-0.2”PCB 封装。

（1)新建PCB库文件，右键单击“单片机试验板.PRJPCB”，在弹出的菜单中执行【New】/【PCB Library】，创建一个 PCB 库文件，并保存为“单片机试验板.PCBLIB”。

（2）单击工程区下方的【PCB Library】标签，会弹出图 2-36 所示的面板。

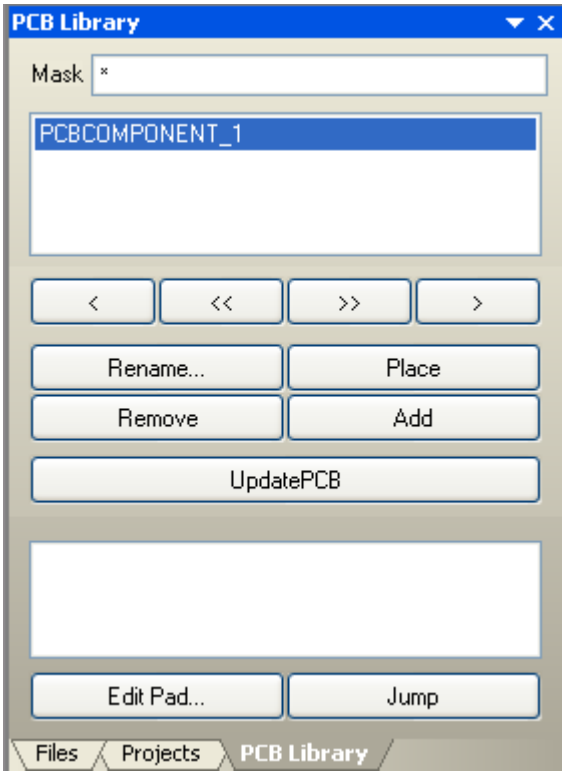


图 2-36 【PCB Library】面板

系统会自动创建元件“PCBCOMPONENT_1”，单击 Rename... 按钮，将元件命名为“AXIAL-0.2”。

（3）执行菜单命令【Edit】/【Set Reference】/【Location】，在图上选择合适位置放置原点。

（4）单击绘图区下方的【Top Overlay】标签，切换到 Top Overlay 层。

（5）绘制元件轮廓：点击工具栏上的图标，绘制 120mil×60mil 的矩形。注意：当鼠标划线时，左下方状态栏会显示当前坐标。

（6）然后在矩形左右两端绘制两个 40mil 长的线段。

（7）切换到【Multi-Layer】层，点击工具栏上的图标，放置焊盘。按 Tab 键可以改变焊盘属性，这里取默认值。放置完焊盘后，完成的 PCB 元件封装“AXIAL-0.2”如图 2-37 所示。切记及时保存。

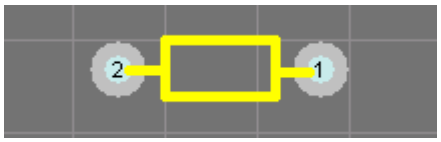


图 2-37 完成的 PCB 元件封装“AXIAL-0.2”

接下来，在创建的“单片机试验板.PCBLIB”库中，使用 PCB 元件封装向导添加发光二极管的封装。

（1）单击【PCB Library】面板中的 Add 按钮，系统会弹出 PCB 元件封装向导欢迎对话框。

（2）单击 Next 按钮，在出现的对话框中选择“Capacitors”（电容）作为要创建的元件封装类型，如图 2-38 所示。

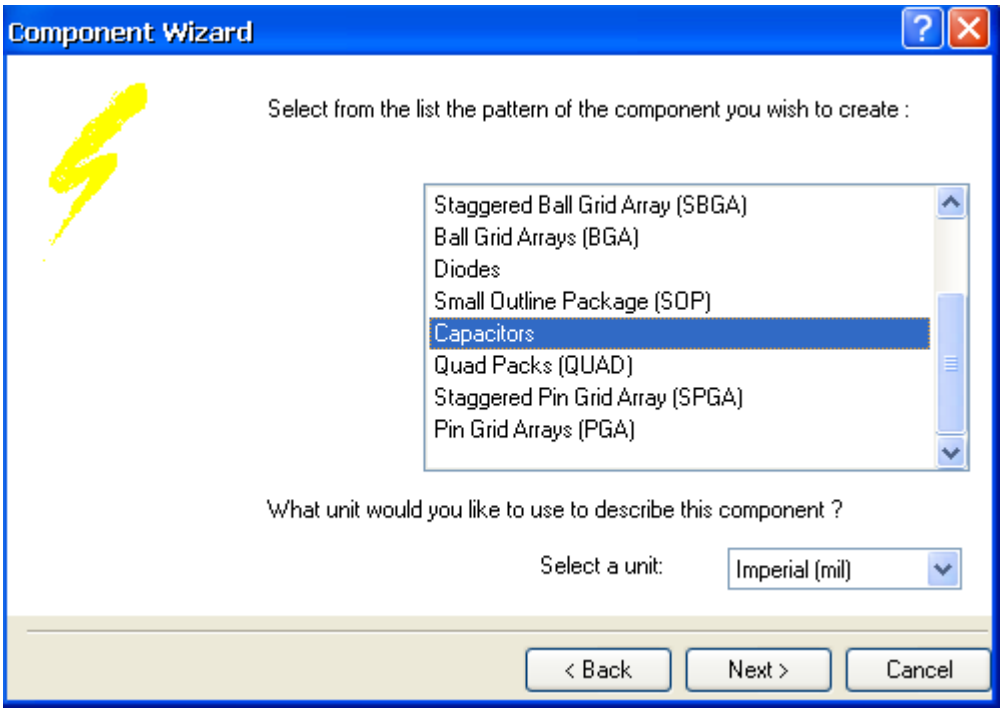


图 2-38　选择电容作为要创建的元件封装类型

（3）单击 Next 按钮，本步是选择电容的类型为直插式。

（4）单击 Next 按钮，本步设置孔径和焊盘的尺寸，将孔径设置为 30mil，焊盘各层均为 70mil，如图 2-39 所示。

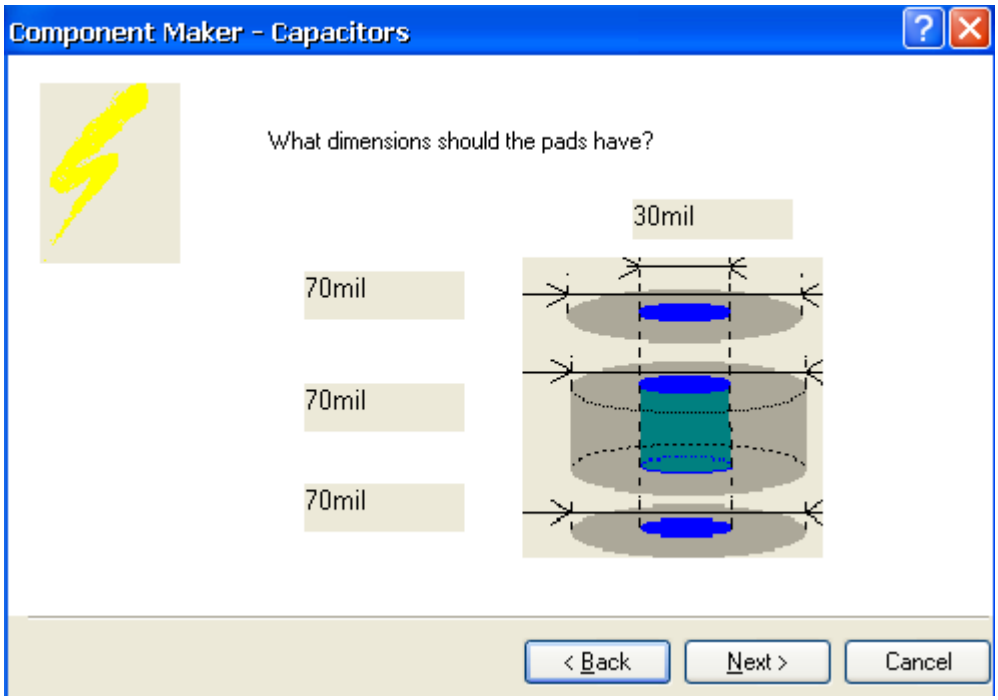


图 2-39　设置孔径和焊盘尺寸

（5）单击 Next 按钮，本步设置焊盘间距，将其设置为 100mil。

（6）单击 Next 按钮，本步选择电容元件封装的类型。见图 2-40。

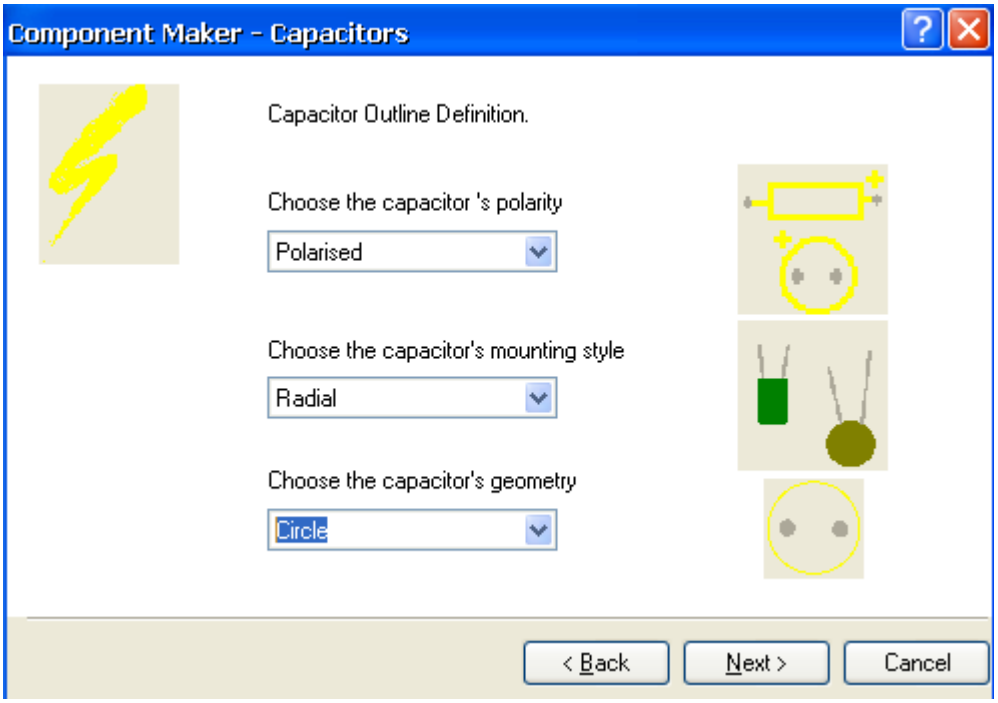


图 2-40　设置电容类型与形状

该对话框中，最上面一项是选择电容是否有极性，选择“Polarised”（有极性）。中间选项决定电容装配型号，选择“Radial”，这时会出现第三项，选择电容的几何形状，选定“Circle”（圆形）。

（7）单击 Next 按钮，设置元件轮廓的线宽和圆形轮廓的半径，线宽取默认值 10mil，半径取 130mil。

（8）单击 Next 按钮，将封装的名称改为“YLED”。

（9）单击 Next 按钮，出现封装结束对话框，单击 Finishi 按钮结束。生成的 PCB 元件封装如图 2-41 所示。

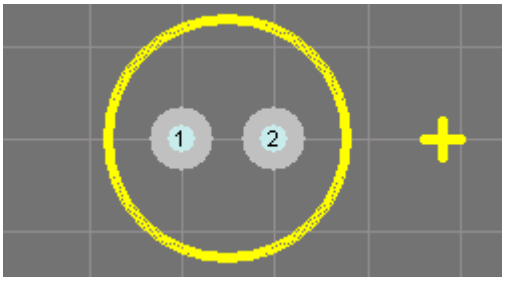


图 2-41　向导生成的发光二极管的 PCB 封装

（10）最后需要对该封装进行调整，将正极符号移到合适的位置，从而减少元件封装所占的面积，如图 2-42 所示。

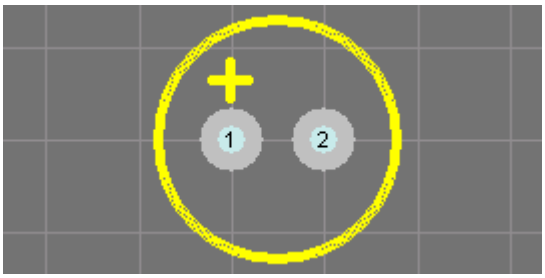


图 2-42　完成的 YLED 封装

再接下来，添加一个开关的 PCB 封装。

（1）单击【PCB Library】面板中的 Add 按钮添加一个新的元件。

在出现的 PCB 向导对话框中单击 Cancel 按钮，器件区就会出现一个新的元件“PCBCOMPONENT_1”。单击 Rename... 按钮，将名称改为“SW-PB”。

（2）首先在【Top Overlay】层绘制一个 320mil×240mil 的元件轮廓。点击工具栏上的焊盘按钮添加焊盘。此时，系统会自动切换到【multi-layer】层。在四个顶角处添加 4 个焊盘，如图 2-43 所示。焊盘属性取默认值。

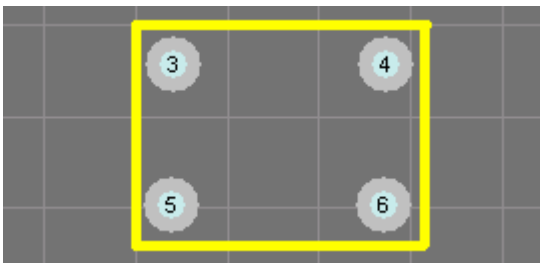


图 2-43 添加了 4 个焊盘的“SW-PB”封装

（3）切换到【Top Layers】层，点击工具栏上的按钮，添加一个完整的圆周，代表重启键的按钮。先在矩形轮廓的中心处点击鼠标，确定圆心。然后光标由圆心向外移动，将圆周拖到合适大小。在点击鼠标确定前，圆周只是虚线。完成的“SW-PB”封装如图 2-44 所示。

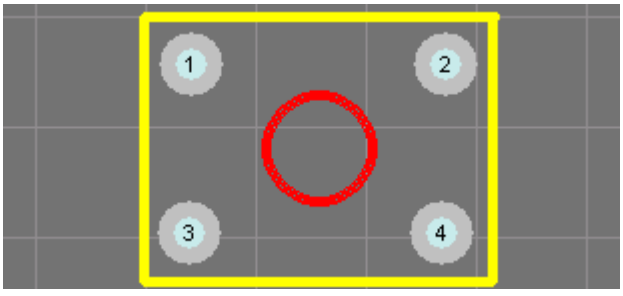


图 2-44 完成的“SW-PB”封装

添加电容使用的“RB.1.2”封装。这次，使用手工绘制。

（1）在创建一个空白 PCB 封装后，重命名为“RB.1.2”。切换到【Top Overlay】层，点击按钮，以原点为圆心，绘制一个半径为 110mil 的元件轮廓。在点击圆心后，光标顺着纵轴向下，以屏幕左下角的坐标为参照，当坐标显示为“X:0mil Y:110mil”时，点击确定。完成的圆周如图 2-45 所示。

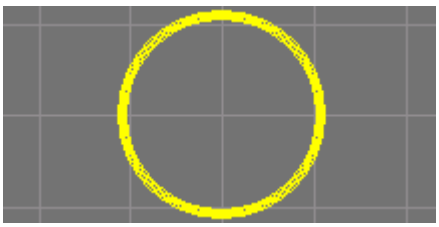


图 2-45 确定后的圆周

（2）切换到【multi-layer】层，点击工具栏上的按钮，为封装添加焊盘。按 Tab 键，将焊盘属性设置为：孔径为 28mil，x 和 y 轴的大小均为 70mil。

将两个焊盘分别添加在（-50mil，0）(50mil，0)处。

然后在管脚 2 的上方绘制一个正极标志十字形，如图 2-46 所示。

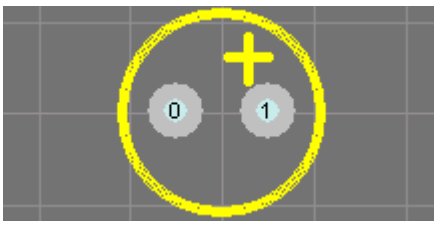


图 2-46 完成的“RB.1.2”封装

还余下两个芯片“AT89C51”和“X25045”的封装没有指定，但它们都是 DIP 标准封装，可以在库中搜索到，不必自己创建。

2、 指定元件的 PCB 封装

接下来在原理图中指定或修改各元件的 PCB 封装。

（1）先处理原理图中的电阻，所有电阻的封装均重新设定为“AXIAL.2”封装。可使用全局修改的功能来一次性修改所有电阻的 PCB 封装。

在选中一个电阻后，单击鼠标右键，在弹出的菜单中执行“Find Similar Objects…”菜单命令。

在弹出的【Find Similar Objects】对话框中，如图 2-47 所示，将“Description”设为“Same”，选中“Selected Matching”复选框。其它选项保持默认值，然后点击 OK 键，运行全局查找功能。

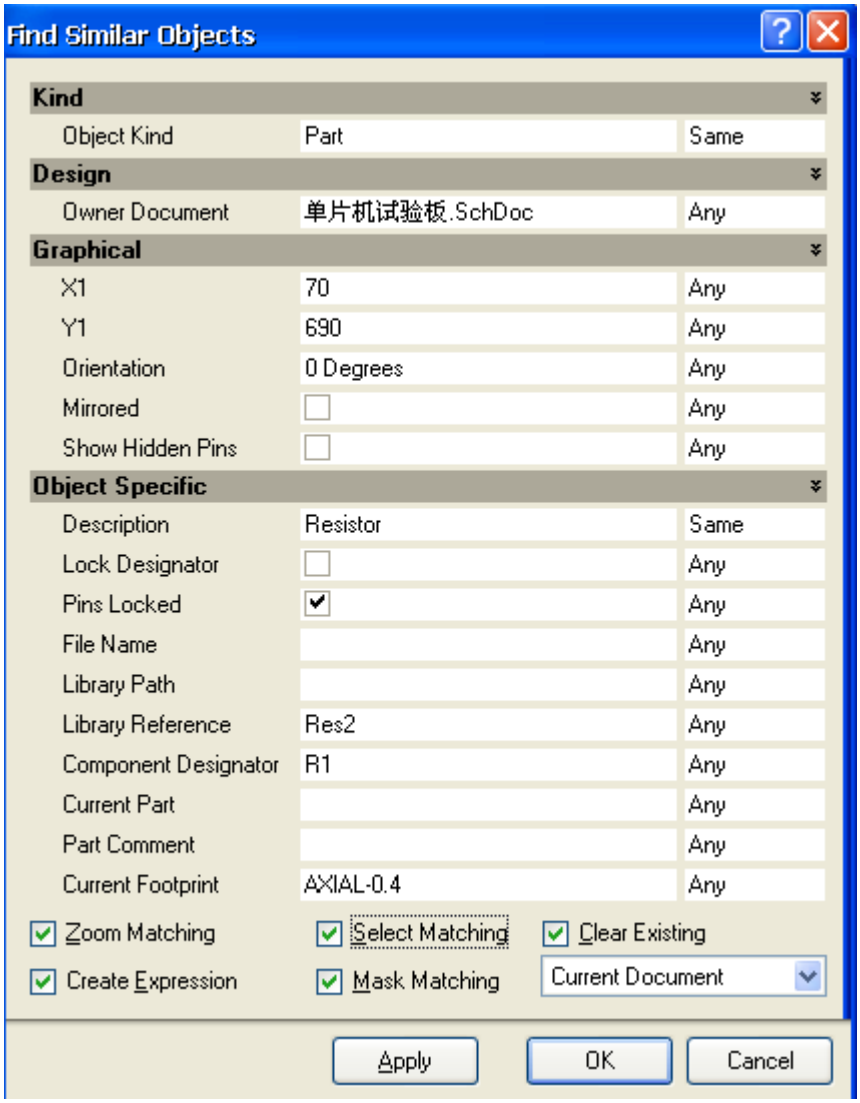


图 2-47 查找电阻的【Find Similar Objects】对话框

运行全局查找后，弹出【Inspector】对话框，如图 2-48 所示。如果没有弹出该对话框，可以单击下方的“Inspect”标签。所有的电阻元件被突出显示，其它元件被蒙板覆盖。

在【Inspector】对话框中将元件封装修改为“AXIAL-0.2”。按回车键，进行修改操作。

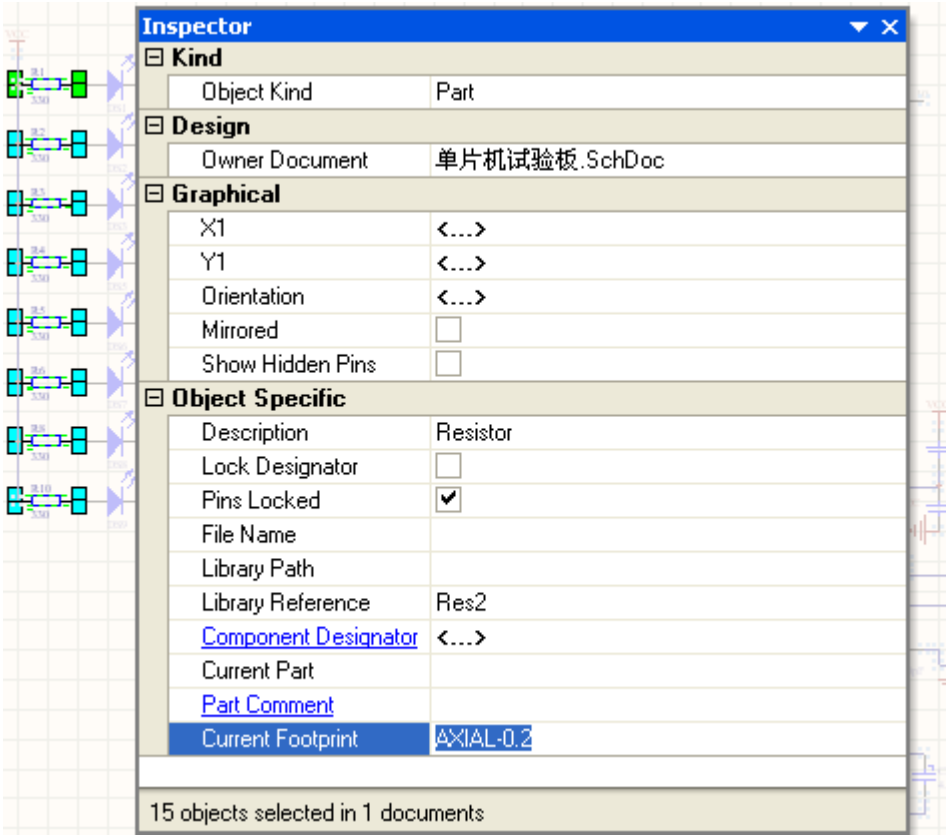


图 2-48 元件封装修改为“AXIAL-0.2”

这样，所有电阻元件的 PCB 封装均变为“AXIAL-0.2”。最后点击绘图区右下方的【Clear】标签，清除元件选中状态，恢复正常绘图过程。

（2）同样的操作步骤，修改发光二极管的 PCB 封装为“YLED”。

（3）修改 4 个开关的封装为“SW-PB”。

修改后，可以双击元件，在打开的元件属性对话框中检查元件的封装是否已经更改为指定的 PCB 封装，如图 2-49 所示。也可在元件属性对话框中修改元件的封装。

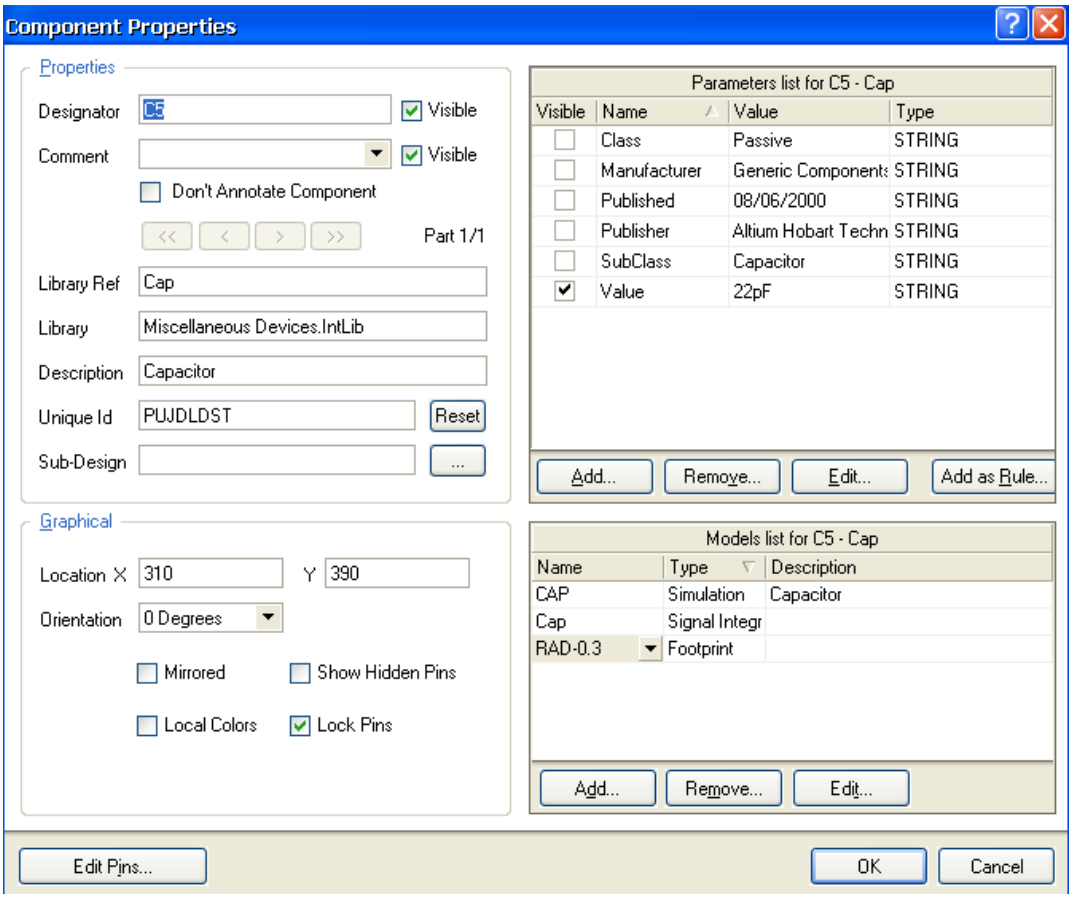


图 2-49 在元件属性对话框中检查封装

（4）修改电容封装。

由于电容有很多种，大小不一，所以本例中电容需要多种封装。将晶振电路中的两个电容封装改为“RAD-0.1”，其它无极性电容封装修改为“RAD-0.2”，其它有极性电容封装改为“RB.1.2”。

双击某个电容，弹出图 2-49 所示的属性对话框。选中对话框右下角的封装项，单击下方的 Edit... 按钮，重新指定电容的封装。在弹出的【PCB Model】对话框中，选中“Library Name”单选框，并单击 Browse... 按钮，弹出图 2-50 所示的对话框，在该对话框中重新设定封装。选择“Miscellaneous Devices.IntLib”库中的“RAD-0.1”封装。

以同样的方式修改所有电容的封装。

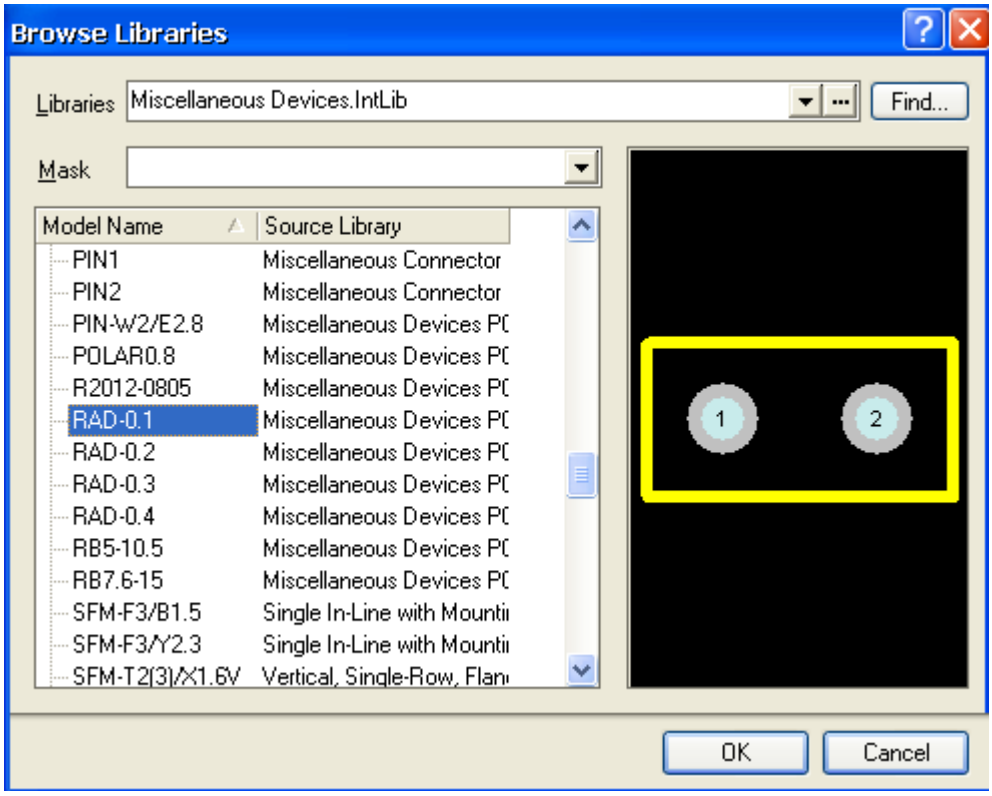


图 2-50 重新设定所选器件的封装

（5）晶振的封装修改为“RAD-0.2”。

（6）为“AT89C51”添加 PCB 封装。双击“AT89C51”，在弹出的元件属性对话框右下角的【Model】栏中，点击编辑按钮 Edit... 。

系统弹出【PCB Model】对话框，单击 Browse... 按钮，在弹出的【Browse Libraries】对话框中点击 Find... 按钮。在弹出的【Search Libraries】对话框中查找 DIP-40 封装，结果如图 2-51 所示。可以看出，有很多 DIP-40 封装，它们都是标准的，完全相同，任一选择一个，单击 Select 按钮即可。

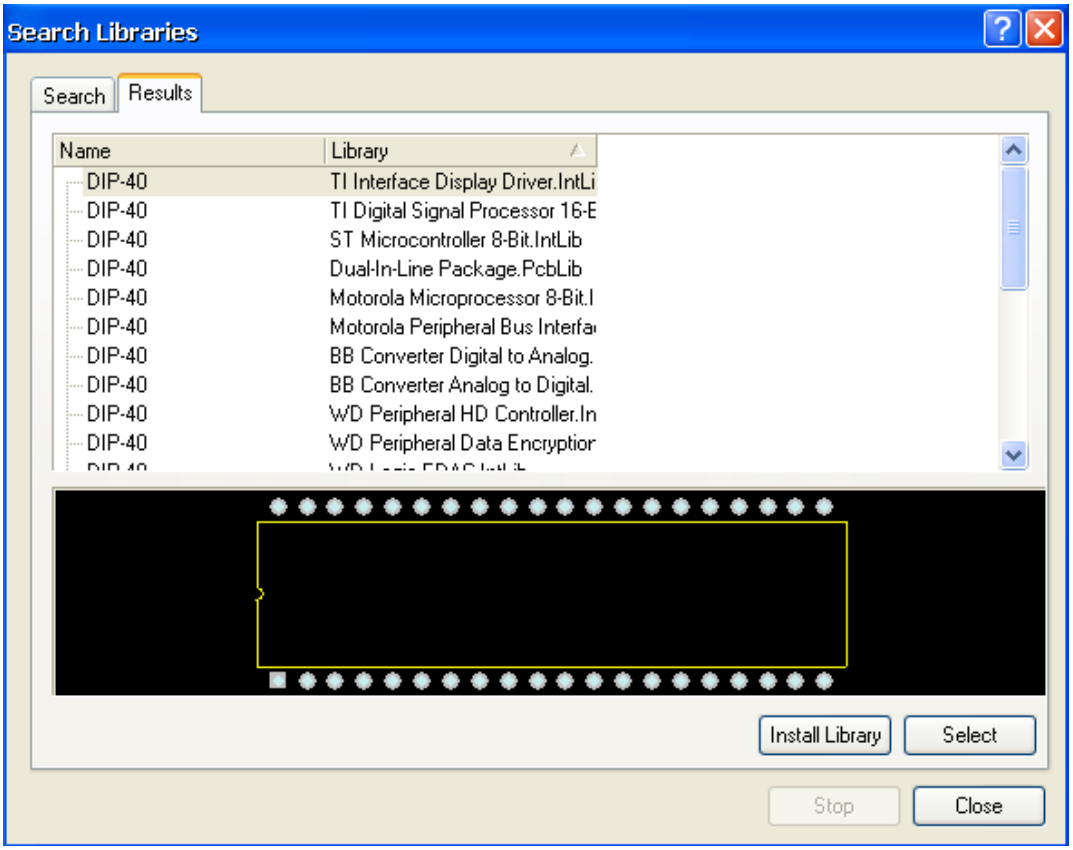


图 2-51 DIP-40 的搜索结果

（7）仿照第 4 步的方式将“X25045”的封装指定为 DIP8。

这样，原理图中所有元件的 PCB 封装设置完毕。

2.2.2 PCB 生成向导

完成原理图的设计和封装的修改，就进入 PCB 图的设计。下面利用 PCB 文件生成向导来完成空白 PCB 文件的创建。

（1）右键单击“单片机试验板.PRJPCB”，在弹出的菜单中执行【New】/【Other】，左侧会出现图 2-52 所示的面板。单击面板中的 PCB Board Wizard... 命令，进入 PCB 文件生成向导。系统会弹出【PCB Board Wizard】欢迎对话框。

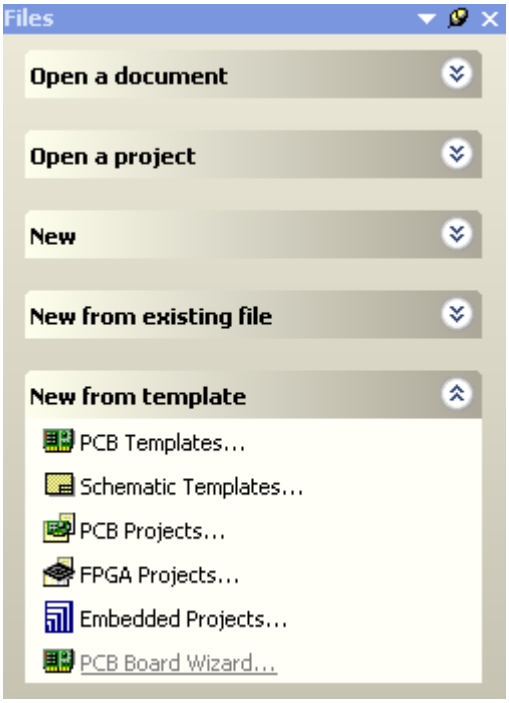


图 2-52 文件面板

（2）单击“Next”按钮继续，下一步对话框是设置 PCB 的尺寸单位。选择“Metric”单选框，将尺寸单位设为公制“mm”。

（3）单击“Next”按钮继续，设置 PCB 的类型。选择“Custom”选项，根据需要自己来规划 PCB。

（4）单击“Next”按钮继续，设定 PCB 的尺寸参数，如图 2-53 所示。将图纸的宽度设为 180mm，高度设为 150mm，系统会自动将其变为 mil 的单位。注意：此处设置的是图纸大小，不是印制电路板的大小。

将所有的复选框均设为非选中状态，将“Dimension Layer”设为 None，不添加尺寸标注层。

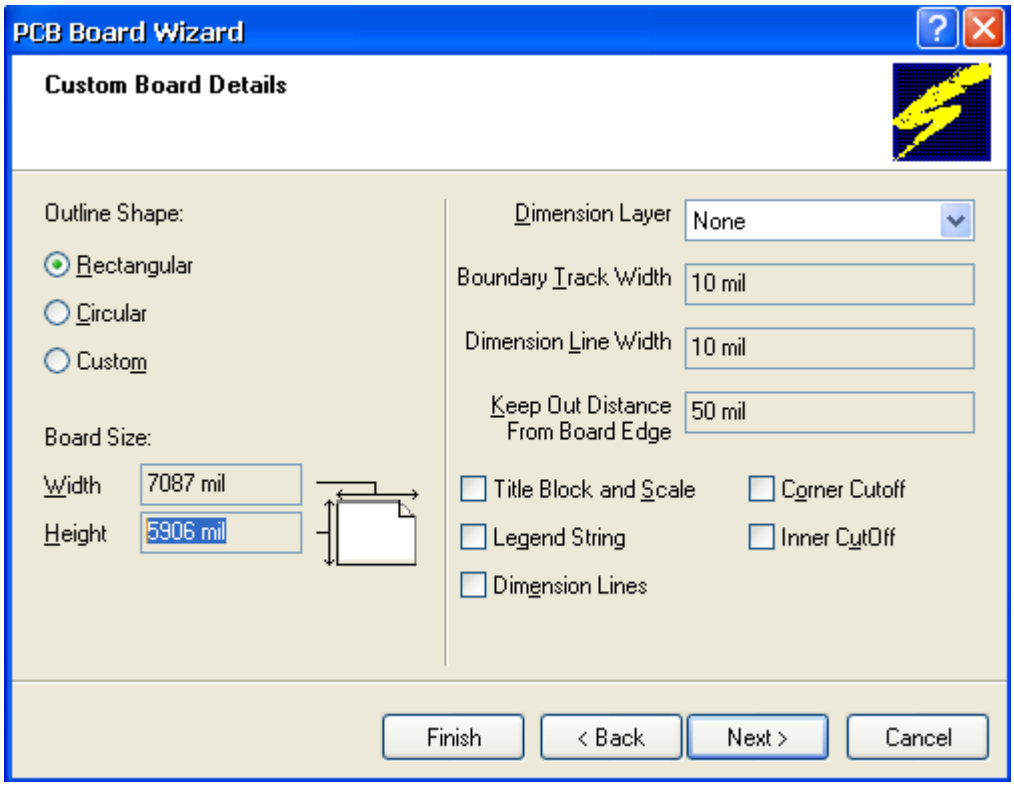


图 2-53　设定 PCB 的尺寸参数

（5）单击“Next”按钮继续，本步是板层设置对话框。由于本例是双层板，所以选择 2 个信号层（Signal Layer），将内电层（Power Plane）设为 0。

（6）单击“Next”按钮继续，设置过孔的样式，选择通孔（Thruhole Vias only）。另一个选项为盲孔和过孔。

（7）单击“Next”按钮继续，设置元件/布线策略。根据器件的选型，看使用的器件是直插式元件还是表贴元件，还要看元件的安装方式是单面或双面。本例取默认，即直插元件，并将元件只布置在一面。

（8）单击“Next”按钮继续，设置导线和过孔的尺寸以及安全间距等参数。取默认值即可。

（9）单击“Finishi”按钮，完成 PCB 板向导的设置。

保存文件，命名为“单片机试验板.PcbDoc”，注意要保存在工程路径中。

可以看出，PCB 板区的栅格很大，不利于布线，因此需对其进行设置。在黑色 PCB 板区单击鼠标右键，执行【Options】/【Grids】命令，则会弹出栅格参数设置对话框。将其中的三种栅格选项均设为最小值，如图 2-54 所示。单击 OK 按钮后可以看出 PCB 板区栅格变小，已看不见了，只有放大后才可见。

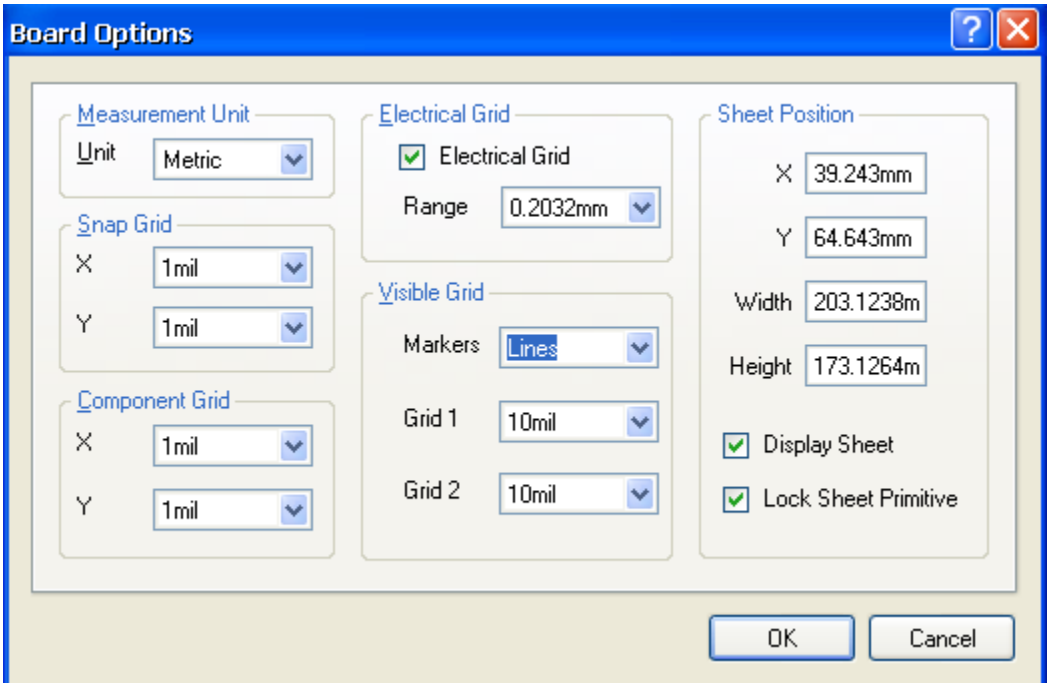


图 2-54 栅格参数设置对话框

2.2.3 PCB 元件布局

首先，在 PCB 文件中同步导入网络表和元件封装。

（1）确认在 PCB 绘图区，执行菜单命令【Design】/【Import Changes From】。

注意：要保证“NE555”、“MAX232”和“24C01”三个芯片所在的库文件也加载到了工程里。否则，它们的元件封装无法载入到 PCB 图中。

（2）在弹出的【Engineering Change Order】对话框中，点击 Validate Changes 按钮。如图 2-55 所示。右侧“Check”列就会出现绿色的对号或红色的叉号，如果是红色的叉号表明该元件的引脚封装不正确，应该回到原理图修改，直到全部都为绿色的对号。

（3）核实无误后，点击 Execute Changes 按钮，执行变更。然后单击 Close 按钮。图 2-56 为载入元件封装好网络表后的 PCB 电路图。

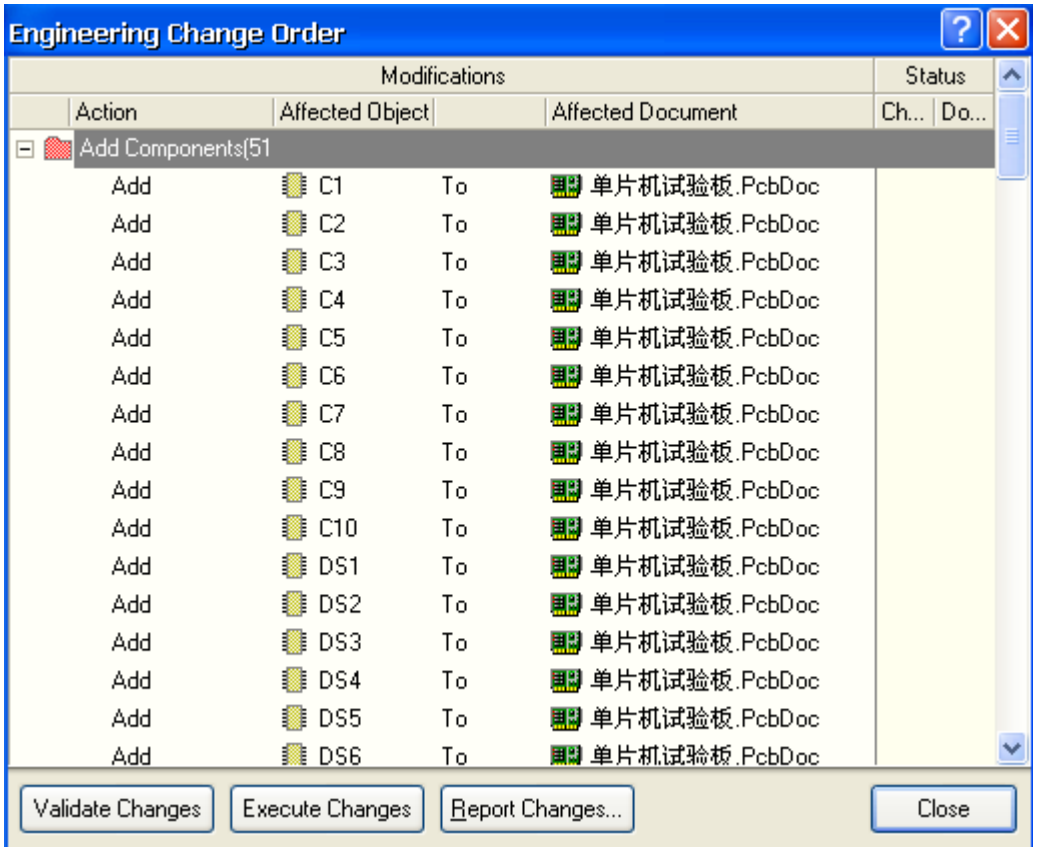


图 2-55 载入元件封装的对话框

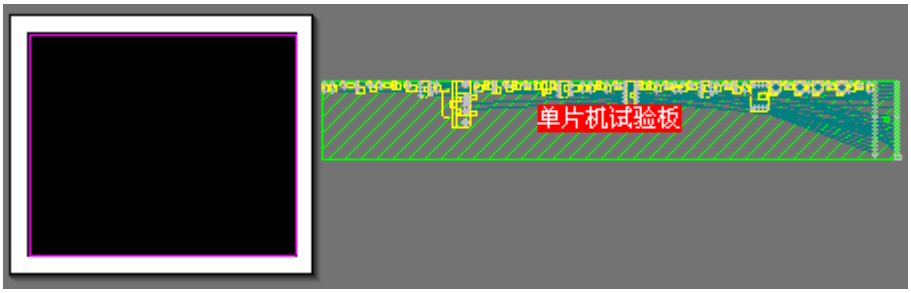


图 2-56 载入网络表和元件封装的 PCB 图

1、元件预定位

首先单击绿色的斜纹框，点击键盘的“Delete”键，将其删除。然后手工方式将各元件拖入 PCB 图区，根据需要排好位置。

在手工调整位置时，可以利用查找功能，根据元件的序号在原理图和 PCB 图上快速查找元件，提高操作效率。举例如下：

原理图中，同时按下“Ctrl”和“F”键，弹出【Find Text】对话框。在其中的“Text Find”栏中输入要查找的元件序号“C1”，点击 OK 键后，系统自动查找元件，并将窗口显示为适当的大小、移动匹配元件到窗口中心。

PCB 图中查找元件，需要在工具栏上的查找框中输入查找元件的序号，如图 2-57 所示。输入“C3”后回车，系统会放大显示搜索到的器件，并屏蔽其它元件。

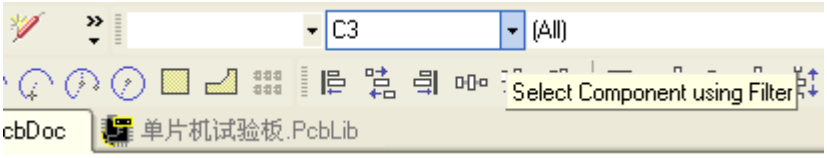


图 2-57 查找工具栏

可按“Page Down”或“Page Up”键缩小或放大视图，选择合适的视图大小，以便拖动元件。可以点击右下方的【Clear】标签或按“Shift”+“C”键清除屏蔽。

在布局调整过程中，会用到元件布局自动排列调整功能。自动排列工具栏如图 2-58 所示，共有 15 项。

图 2-58 自动排列工具栏

举例如下：图 2-59 中几个电阻不整齐，而且间距不均匀。鼠标选中全部 6 个电阻，然后点击工具栏上的（上对齐）按钮，则 6 个电阻水平对齐。再点击工具栏上的（水平等间距）按钮，操作结果如图 2-60 所示。

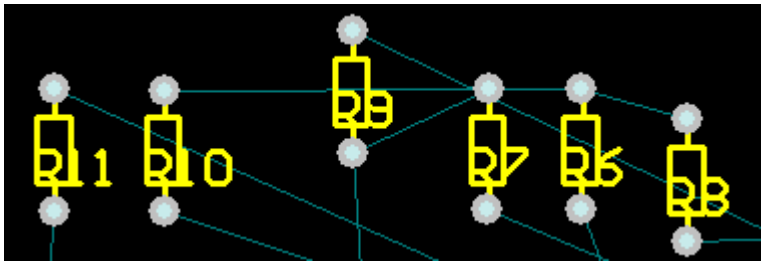


图 2-59 不整齐的 6 个电阻

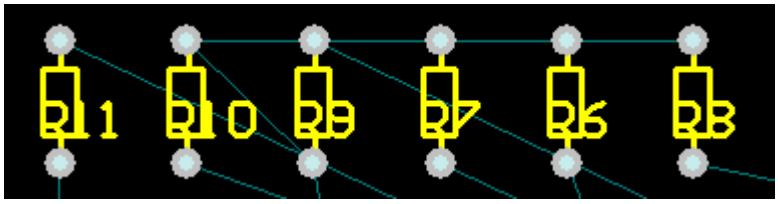


图 2-60 自动排列操作后的结果

2、调整元件标注

大部分元件标注，只需要被点击选中后，直接拖动并在放置前按空格键旋转到合适的角度即可。但有时某些元件的标注因为空间狭小，无法安排，就需要调整标注的字体。双击标注，弹出【Designator】对话框，将字体的高度修改为合适的值，则标注字体就会变小。

2.2.4 布线

因为本例是 2 层板，使用自动布线与手工调整的方式就可以达到比较好的效果。

在自动布线参数设置中，先对线宽进行设置。

（1）执行【Design】菜单下的【Rules】命令，设置第一个线宽规则，规定所有网络的导线基本宽度是 0.3mm，如图 2-61 所示。

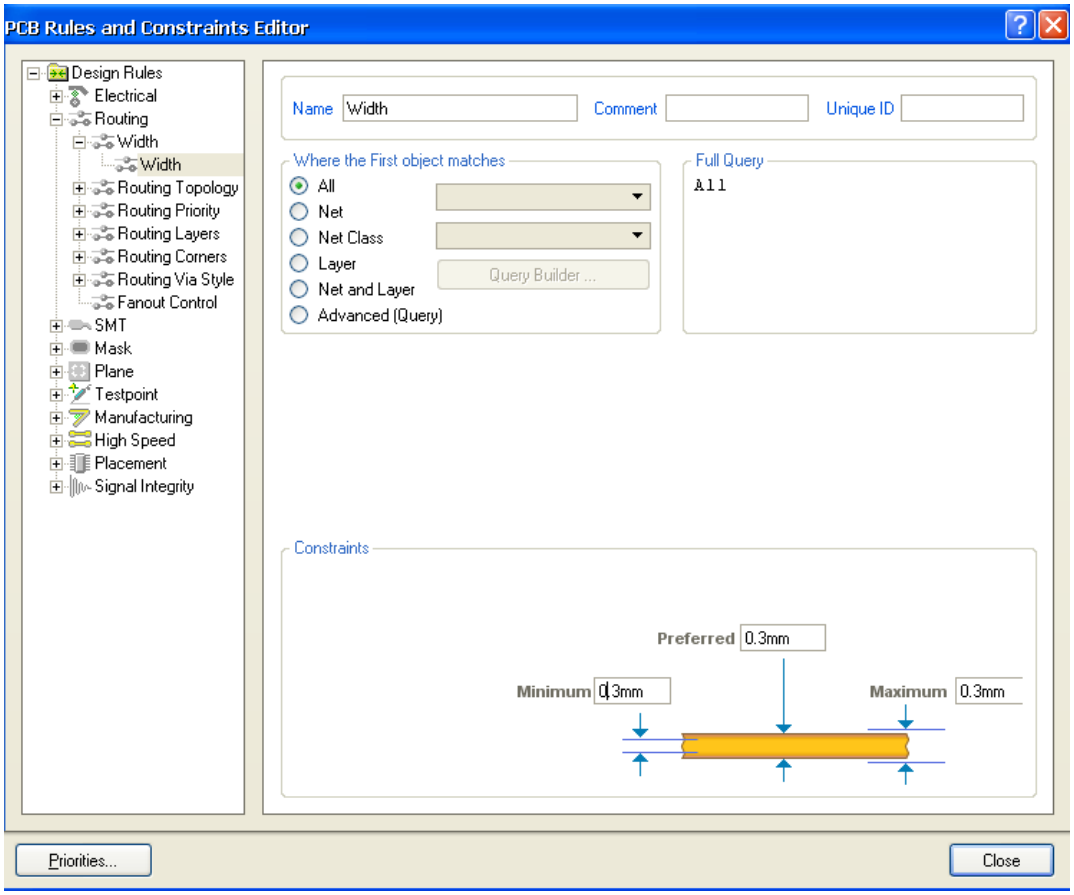


图 2-61 设定第一个线宽规则

（2）添加一个线宽规则，将网络分别设为“GND”和“VCC”，推荐线宽设为 1mm，最小线宽设为 0.6mm，最大线宽设为 1mm。在设定完数值后，需要鼠标点击对话框中的其它数值，才能使设定数值生效。注意：必须首先设定最大线宽，才能设定推荐线宽。

执行菜单命令【Auto Route】/【All…】。在弹出的对话框中点击 Route All 按钮，进行自动布线。

自动布线后，还有很多不尽人意的地方，需要手工调整。

若想取消布线，可以执行菜单命令【Tools】/【Un-Route】/【All】。

2.2.5 补泪滴

在导线和焊盘或者导孔的连接处添加一段过渡，其形状呈泪滴状，形象地叫做泪滴。泪滴的作用是在钻孔时，避免导线与焊盘的接触点因为应力集中而断裂。添加泪滴的操作就叫做补泪滴。

执行菜单命令【Tools】/【Teardrops…】，弹出图 2-62 所示的泪滴项对话框。

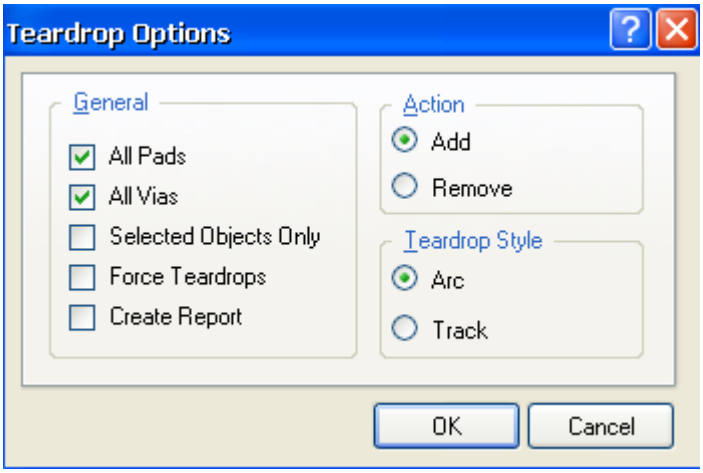


图 2-62　补泪滴选项对话框

在该对话框中有 3 个区域设置。

（1）【General】区

“All Pads”：设置是否所有焊盘都补泪滴；

“All Vias”：设置是否所有通孔都补泪滴；

“Selected Objects Only”：设置是否将被选取的组件补泪滴；

“Force Teardrops”：设置是否强制补泪滴；

“Create Report”：设置是否生成补泪滴的报告文件。

（2）【Action】区

“Add”：添加泪滴；

“Remove”：删除泪滴。

（3）【Teardrop Style】区

“Arc”：圆弧形泪滴；

“Track”：导线形泪滴。

在本例中，不需要为所有的焊盘都添加泪滴，所有选中“Selected Objects Only”选项，只为选中的焊盘补泪滴。另外需要注意的是，“All Pads”选项也必须被选中，否而执行补泪滴操作并不能给选择的焊盘添加上泪滴。

举例如下，对串口的 3 个连接有导线的焊盘补泪滴。

（1）按下“Shift”键的同时，点击 3 个焊盘。这样可以同时选中它们，如图 2-63 所示。

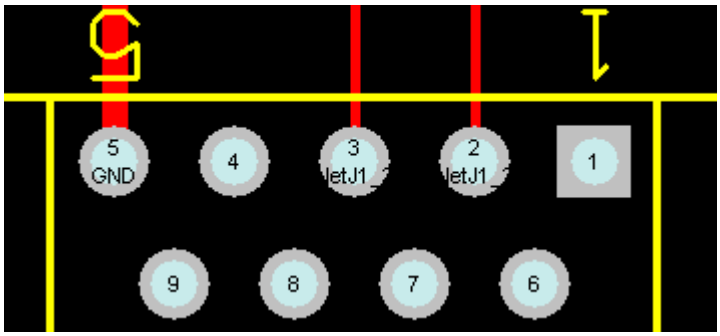


图 2-63 同时选中串口的 3 个焊盘

（2）启动菜单命令【Tools】/【Teardrops…】，设置只为选中的对象补泪滴和 Track 形泪滴。

（3）点击 OK 按钮，执行补泪滴操作，结果如图 2-64 所示。

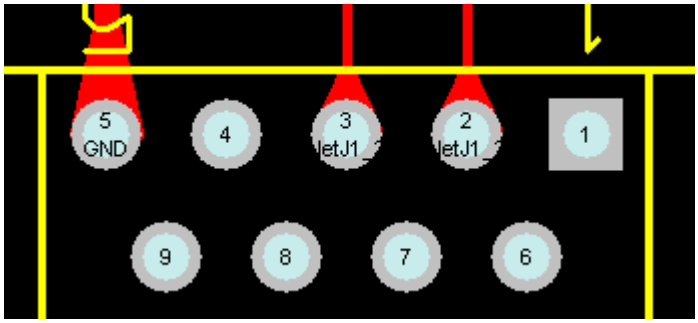


图 2-64 补泪滴后的效果

2.2.6 敷铜

敷铜就是将电路板空白的地方铺满铜膜，注意目的是提高电路板的抗干扰能力。通常将铜膜接地，这样电路板中空白的地方就铺满了接地的铜膜，电路板的抗干扰能力会得到显著提高。

执行【Design】菜单下的【Rules】命令，进入布线规则和线宽设置对话框。双击左侧的【Plane】项，下面有 3 项与敷铜相关的设计规则。

（1）【Power Plane Connect Styles】设计规则

这是当敷铜与导孔或焊盘网络相同时，敷铜与焊盘间的连接。取默认值即可。

（2）【Power Plane Clearance】故则

设置敷铜与导线或焊盘间的安全间距，推荐采用 0.6mm 以上。

（3）【Polygon Connect Style】规则

连接方式选择。Protel DXP 提供了 3 种连接方式：

【Relief Connect】：导线连接；

【Direct Connect】：直接连接；

【None Connect】：没有连接。

本例使用的是系统默认的 4 个导线/90° 连接方式。

下面进行敷铜操作。

（1）点击工具栏上的按钮，弹出图 2-65 所示的对话框。并按该图所示方式设置参数。

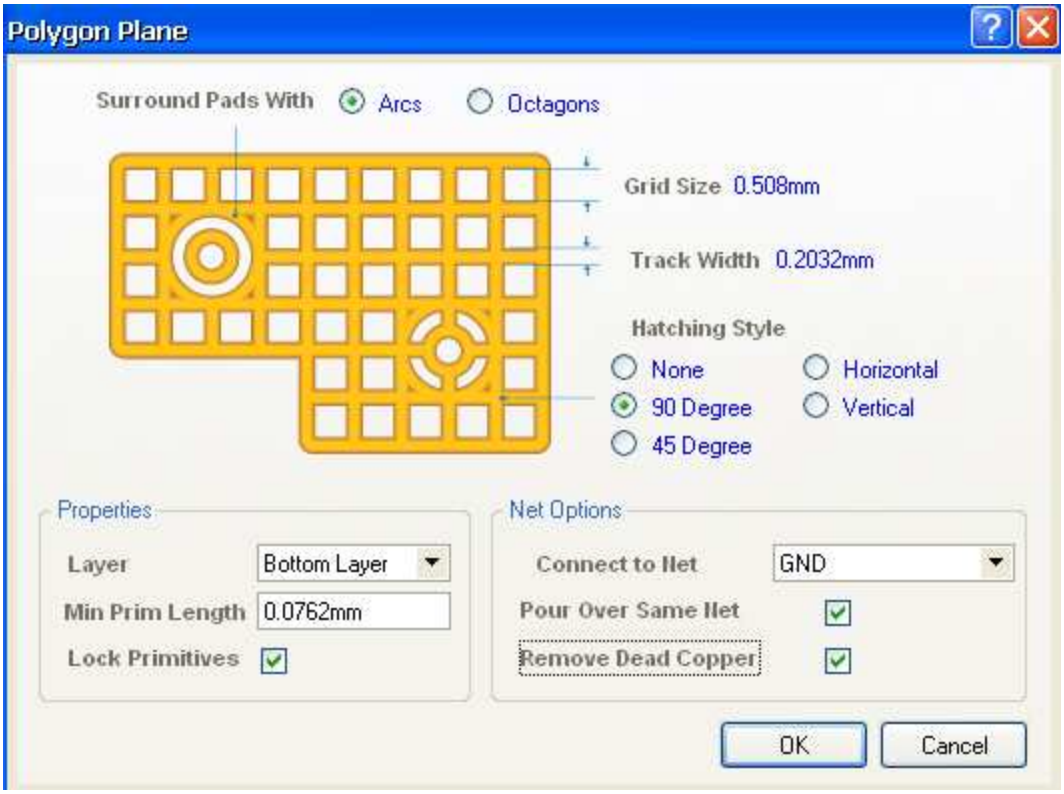


图 2-65 敷铜参数设置

（2）点击 OK 按钮，然后将变为十字的光标在板的左下侧点击，依次放置下确定敷铜范围的 4 个矩形顶点。放置下第 3 个顶点后，可以看到出现一个三角形的范围，重新点击第 1 个顶点，从而形成一个封闭的矩形区域。这时，系统会自动开始在这个区域内按前面设定的敷铜规则完成敷铜。

2.2.7 电路 DRC 检验

执行菜单命令【Tools】/【Design Rule Check】，检查 PCB 图是否有错误。【Messages】对话框会显示当前电路板的问题，如果有问题根据提示修改。注意，并不是所有提示的错误都需要修改，有些不影响正确性，请在练习中体会。

至此，本例的“单片机试验板”电路板就完成了，如果想看一下电路板的效果图，可执行菜单命令【View】/【Board in 3D】。

第 3 章 高级实例

- ▲ 四层板的概念
- ▲ 层次原理图
- ▲ FPGA 等芯片介绍
- ▲ 创建贴片式元件的 PCB 封装
- ▲ 原件集成库的建立
- ▲ 对系统自带的库文件进行修改
- ▲ SOP、QUAD、PLCC 封装
- ▲ 自动布局的详细介绍
- ▲ 内层分割

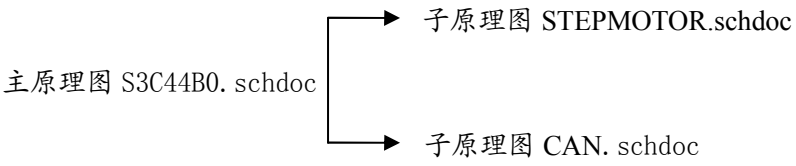
3.1 总体方案介绍

本例介绍一个 ARM7 试验板的设计。核心处理器采用三星公司的 S3C44B0，外加一个步进电机控制电路和一个 CAN 总线通信电路。

3.2 层次原理图设计

层次原理图也称为多张原理图，实际上是一种模块化的设计方法。对于复杂庞大的电路原理图，不可能把它画在一张图纸上，也不可能一个人完成。通常，将设计系统划分为多个子系统，子系统下面又可划分为若干功能模块。这样，设计好基本模块，定义好模块间的连接关系，即可完成整个设计过程。层次原理图可以将一个设计任务分割为几个相对独立的部分，并同时进行设计，大大加快设计进程。

本例的整个原理图设计方案的结构如下：



其中，S3C44B0. schdoc 作为根图纸，是该设计方案的纲领性图纸。其它 3 个图纸是该图纸的子图纸。

层次式结构是指在一张原理图上存在一些图纸符号，它们各自代表着另一张子原理图。而每一张子图纸还可以有自己的子图纸。原理图的层次是没有限制的。图之间的连通方式取决于选择的网络标识符。

网络标识符是用来标注和识别网络的实体，从而实现原理图上网络之间的电气连接。使用网络标识符可以不用导线就将两个电气连接点直接连接起来。只需给两个电气连接点

赋予相同的网络标识符，就可以在原理图之间或者原理图内部建立同一网络间的电气连接。

网络标识符包括网络名称、端口、图纸符号入口、电源端口等。

3.3 主原理图设计

首先新建一个工程项目“高级实例.prjdoc”和原理图“S3C44B0.schdoc”。

3.3.1 元件集成库的创建

Protel DXP 自带库中的电阻、电容元件不符合本实例的使用要求。因此可以采用上一章的方法，分别绘制原理图库和 PCB 库，然后使用全局修改功能来指定它们的 PCB 封装。也可以直接建立自己的元件集成库，在放置元件的原理图符号的同时，就确定了它的 PCB 封装，使用起来非常方便。创建元件集成库首先需要分别创建原理图库和 PCB 库。

1、创建原理图库

首先创建“高级实例 SchLib”原理图库文件。

（1）在【Library Editor】面板的元件列表栏中，选择系统自动创建的“Component”元件，然后单击下方的 Edit 按钮，弹出【Component Property】对话框，将“Designator”区改为“R？”，“Library Ref”区改为“Res”，表示电阻。

（2）执行菜单命令【Tools】/【Document Options…】，在弹出的对话框中将“Grids”栏中的“Snap”和“Visible”数值设定为“3mil”。

（3）单击工具栏中的放置线条按钮，画图 3-1 所示的折线。注意：默认状态下，线段总是水平或者垂直的。需要 45° 折线时，需要按一下“Shift”+“空格键”。

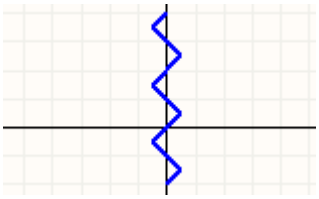


图 3-1 绘制电阻的原理图符号

（4）添加电阻的两个管脚。点击工具栏上的图标，按下“Tab”键，在弹出的【Pine Properties】对话框中做两处修改。“Display Name”设为空，或者把“Visible”属性前的对勾去掉；将“Designator”设为不可视。点击 OK 键，回到绘图区。

（5）按空格键，调整管脚方向，将带有叉号的一端向外。

（6）然后在元件列表中点击 Edit 按钮，在库元件属性对话框中为“Res”元件添加数值显示。在【Parameters for R?-Res】栏中，点击 Add... 按钮，添加一个规则。

（7）在弹出的【Parameter Property】对话框中，“Name”栏填入“Value”，“Value”

栏填入“100K”。 关闭对话框，单击图标，保存原理图库文件。完成的电阻原理图符号如图 3-2 所示。

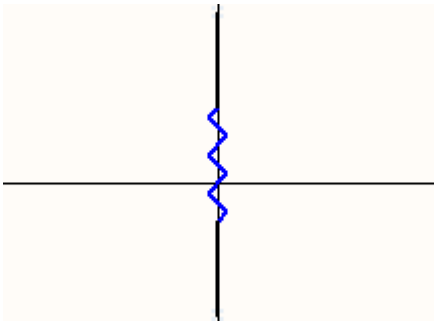


图 3-2 完成的电阻原理图符号

下面添加一种电容，本实例中 10uF 和 22uF 的电容使用该符号。步骤如下：

（1）在【Library Editor】面板的元件列表栏中，单击 Add 按钮，添加新元件“CAP1”。

（2）在绘图区中，点击鼠标右键，执行【Document Options】命令。在弹出的【Library Editor Workspace】对话框中，设置【Snap】项为“5”。

（3）点击工具栏上的按钮，绘制如图 3-3 所示的图形。横线长 20，竖线长 5。

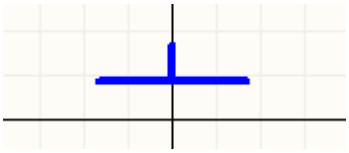


图 3-3 上半部分

（4）然后点击工具栏中的图标，绘制椭圆弧线。

（5）添加两个管脚以及一个十字标注表示正极。

（6）单击【Library Editor】面板的 Edit 按钮，在弹出的对话框中设置“Designator”为“C？”，并在【Parameters for CAP1】栏中，添加元件的 Value 值，设为 10uF。

最后元件“CAP1”的原理图符号如图 3-4 所示。

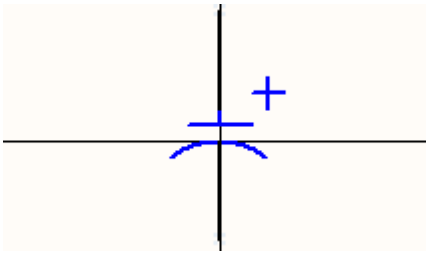


图 3-4 完成的元件“CAP1”

下面添加一个 LED 元件，因为它的原理图符号和自带库中的 LED 元件完全相同，所有通过下面的方法，直接将其复制到“高级实例.SCHLIB”库中，不必自己绘制。

（1）点击【Library Editor】面板的 Add 按钮，添加新元件“LED”。

（2）点击工具栏上的按钮，打开“Miscellaneous Devices.IntLib”库文件。

（3）双击打开该库文件后，系统弹出提示对话框，询问是否将集成库打开。单击 Yes 键，打开库文件。可以看到工程面板已经添加了该库文件。

（4）切换到“Miscellaneous Devices.IntLib”库的【Library Editor】面板，在元件列表中选择“LED0”，可以看出元件原理图符号编辑区。如图 3-5 所示。

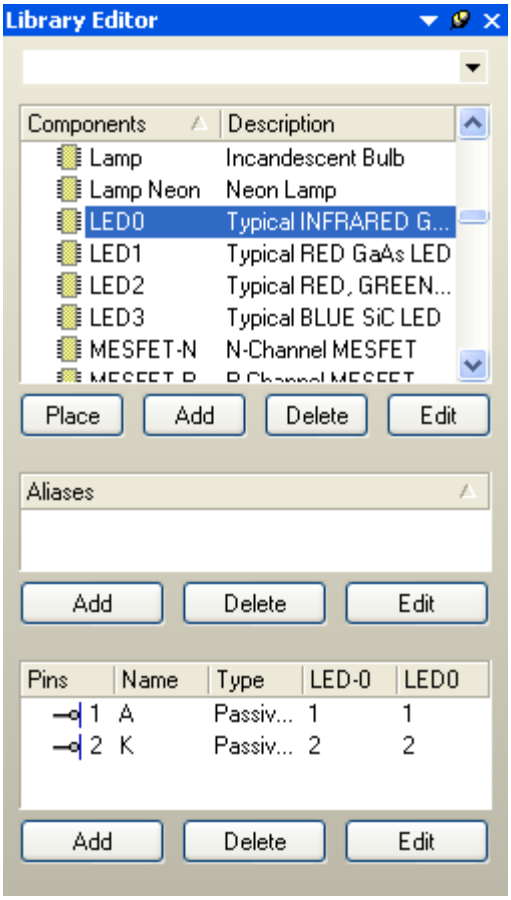


图 3-5 “Miscellaneous Devices.IntLib”库的元件列表

（5）光标选中完整的原理图符号，然后按下“Ctrl”+“C”键复制。

（6）切换到“高级实例.SCHLIB”中的 LED 原理图符号编辑区，按“Ctrl”+“V”键粘贴。

（7）双击元件的管脚，取消“Designator”的显示。注意保存。

参照上面的步骤，添加 1 个电容元件“CAP2”，如图 3-6 所示。

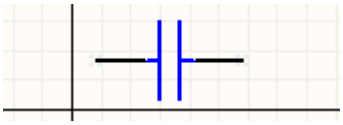


图 3-6 元件 CAP2”

2、创建“高级实例.PCBLIB”PCB 元件库

（1）在 PCB 库的元件列表中，选择系统自动生成的“PCBCOMPONENT_1”元件。单击下方的 Rename... 按钮，在弹出的对话框中将元件命名为“0603”，然后值接在绘图区域中绘制该封装。

（2）在绘图区单击右键，执行【Library Options…】菜单命令。在【Board Options】对话框中做如下修改：

【Unit】设为“Metric”;【Grid1】设为“0.025mm”;【Grid2】设为“0.025mm”。

注意：0603 封装，就是长 1.6mm，宽 0.8mm。

（3）选择“Top Layer”层，在该层绘制贴片电阻的焊盘。

（4）设置坐标原点。注意，可以方便地利用状态栏确定尺寸、位置等信息。执行菜单命令【Edit】/【Set Reference】/【Location】,光标变为十字，选择适当的位置点击鼠标左键， 则光标所在位置的坐标变为（0，0）。

（5）放置焊盘。单击 PCB 元件库放置工具栏中的 按钮，光标变为十字形状。按<Tab>键，在【Pab】（焊盘）对话框中做如下修改。

【Hole Size】：0mm;

【Size and Shape】区域中【X-Size】：0.6mm;【Y-Size】：0.8mm;【Shape】：Rectangle。

(6) 在（0，0）处点击鼠标左键放置焊盘。

（7）坐标（1，0）处放置第二个焊盘。

（8）绘制元件外框。选择【Top Overlay】层，单击工具栏中的 按钮，执行画线命令。按<Tab>键，在弹出的对话框中，将线宽设为“0.2mm”。

（9）在焊盘的边界 0.4mm 处，画边框。最后结果如图 3-7 所示。

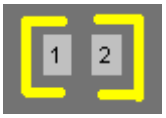


图 3-7 完成的“0603”封装

（10）保存库文件，完成“0603”封装的创建。

接下来采用本节添加 LED 用到的方法，直接复制第 2 章中创建了的“RB.1/.2”封装到“高级实例.PCBLIB”库文件中。

原理图库和 PCB 库中的元件，暂时只创建这么多，下面生成元件集成库。

3、创建元件集成库

我们希望调用器件时可以象使用 Protel DXP 本身所集成库一样，同时调用原理图符号和 PCB 封装。下面就建立自己的集成库，将本实例建立的元件的各种信息放在这个库中。

（1）执行菜单命令【File】/【New】/【Integrated Library】,或者右键点击 Paoject，在弹出的菜单中选择新建集成库。

执行命令后，可以看到在【Projects】（工程）面板中出现一个名为“Integrated Library1.LibPkg”文件。

系统同时会在该目录下生成“No Source Libraries Added”文件夹，存放以后生成的“IntLib”文件。

集成库文件的扩展名是“IntLib”，这里却是“LibPkg”，此为集成库文件包。下面经

过特定操作，便可以生成“IntLib”的集成库文件供画图时使用。

（2）在“Integrated Library1.LibPkg”点击右键，执行命令【Save Project As】保存集成库文件包为“高级实例.LibPkg”，【Project】面板中的“Integrated Library1.LibPkg”变为“高级实例.LibPKG”。

（3）直接在【Project】面板中，将“高级实例.SCHLIB”和“高级实例.PCBLIB”两个文件拖入到“高级实例.LibPkg”里。

（4）双击“高级实例.SchLib”，打开原理图文件，切换到【Library Editor】面板下，如图 3-8 所示。

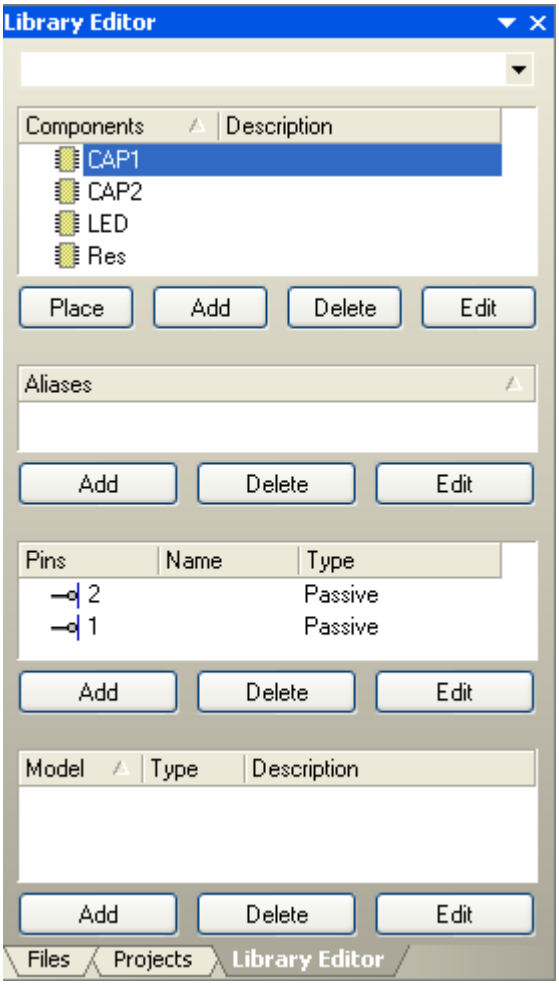


图 3-8 【Library Editor】面板

在元器件列表框中选中要编辑的元器件“CAP1”，单击【Model】栏中的 Add 按钮，添加 PCB 封装。

（5）系统弹出模型类型选择对话框，如图 3-9 所示。选择“Footprint”。

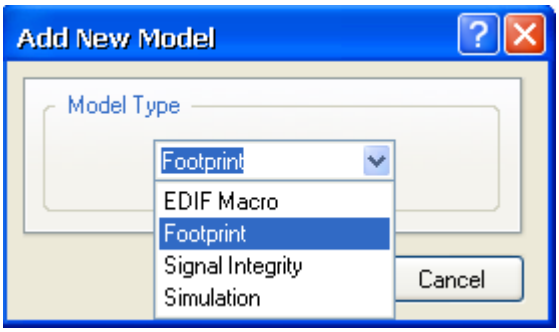


图 3-9 【Add New Model】对话框

（6）然后系统弹出选择 PCB 封装对话框。

（7）单击对话框中的 Browse... 按钮，弹出 PCB 库浏览对话框。在该对话框中选择“高级实例.PCBLIB”库的“RB.1/.2”PCB 封装。

（8）同样，指定“LED”的封装为“RB.1/.2”，“CAP2”和“Res”的封装为“0603”。

（9）编辑集成库文件。执行菜单命令【Project】/【Compile Integrated Library】。

注意：要想编译集成库，必须保证当前编辑页面补是集成库所在工程中的原理图 PCB 图。否则【Project】菜单下没有【Compile Integrated Library】项。

（10）完成编译后，系统自动弹出【Libraries】面板。在其中，可以看到“高级实例 IntLib”集成库中元件的原理图符号和 PCB 封装。

这样，当绘制原理图调用该元件时，也同时调用 PCB 封装，使用起来非常方便。

3.3.2 S3C44B0 核心板的原理图设计

S3C44B0 是三星公司的一款带有 ARM7 核的芯片，采用 160 脚的 QFP 封装。

S3C44B0 核心板包括电源、S3C44B0 芯片、复位电路、晶振电路等。

1、S3C44B0 原理图库建立

Protel DXP 自带库中没有 S3C44B0 的原理图，所以需要按照第 2 章的方式建立 S3C44B0 的原理图库。注意，在第 1 管脚和 160 管脚之间加一个圆，表示起始管脚。建立好的原理图如图 3-10 所示。各引脚说明如下表：

引脚	说 明	引脚	说 明	引脚	说 明
1~4	ADDR3~ADDR0	49	CLKOUY	93	VLINE
5~6	CAS0~CAS1	50	RESET	94	VCLK
7	SCAS	51~54	OM0~OM3	95~98	VD3~VD0
8	SRAS	55	CODECLK	99	RXD0
9	VDDIO	56	SIOCLK	100	TXD0
10	VSSIO	57	SIORXD	101~102	GPC15~GPC14
11~14	DQM0~DQM4	58	SIORDY	103	RXD1
15	OE	59	SIOTXD	104	TXD1

16	WE	60	IICSDA	105	CTS1
17~20	GCS0~GCS3	61	IICSCL	106	RTS1
21	VDD	62	VDD	107	XDREQ1
22	VSS	63	VSS	108	XDACK1
23~24	GCS4~GCS5	64	XTAL0	109	VDD
25~26	SCS0~ SCS1	65	EXTAL0	110	VSS
27	SCKE	66	PLLCAP	111~114	VD4~VD7
28	SCLK	67	EXTCLK	115	IISCLK
29	WAIT	68~72	TOUT0~TOUT4	116	IISDI
30	XDREQ0	73	VSSIO	117	IISDO
31	SDACK0	74	VSSADC	118	IISLRCK
32~33	EXINT0~EXINT1	75~82	AIN0~AIN7	119~124	DATA15~DATA10
34	VDD	83	AREFT	125	VDDIO
35	VSS	84	AREFB	126	VSSIO
36~41	EXINT2~EXINT7	85	AVCOM	127~136	DATA9~DATA0
42	TRST	86	VDDADC	137	GPA9
43	TCK	87	XTAL1	138	VDD
44	TMS	88	EXTAL1	139	VSS
45	TDI	89	VDDRTC	140~151	ADDR23~ADDR12
46	TDO	90	VSSIO	152	VSSIO
47	VDDIO	91	VFRAME	153~160	ADDR11~ADDR4
48	VSSIO	92	VM		

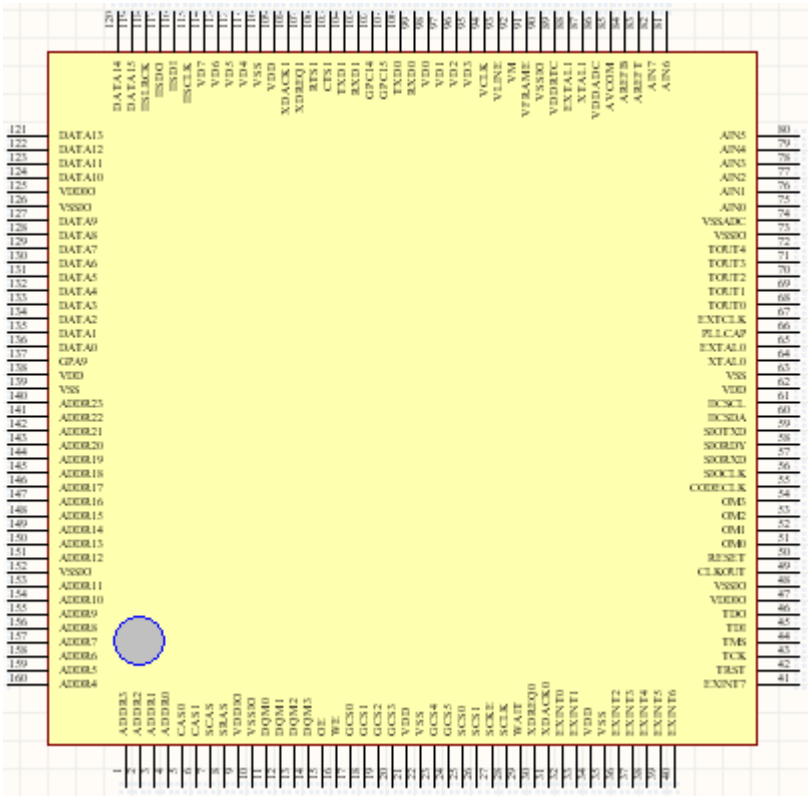


图 3-10 S3C44B0 原理图

2、电源电路

ARM 供电电路设计是系统设计的一个重要组成部分。一般 ARM 会要求有独立的内核电源和 I/O 电源，本实例中，S3C44B0 的内核电压是 2.5，I/O 电压是 3.3V。由于 S3C44B0 在系统中承担大量的实时数据计算，其内部频繁的部件开关转换会使系统功耗大大增加，所以降低其内部供电的核心电压，无疑是降低系统功耗的最有效的办法之一。

电源设计采用 5 V 输入主板，经电压稳压，提供 I/O 端口需要的电压 3.3V，在核心板上采用稳压块供 CPU 内核电压 2.5V 或 1.8V。电源电路如图 3-11 和 3-12 所示。

REG1117 芯片是一款稳压芯片，输出有 5V、3.3V、3V 和 2.8V 等级别，最大过 1A 电流，可通过搜索方式找到该元件，贴片封装。注意：图中的电阻、电容、二极管都是自建的集成库【高级实例.SchLib】中的元件。

RT9167-25CB 是另外一款 2.5V 的稳压芯片，RT9167 系列稳压器输出电压范围很广，从 1.5V 到 5V 范围内每隔 0.1V 就有一款稳压芯片。该芯片在 Protel DXP 的自带库中没有，需要自行建立其原理图库及 PCB 库。原理图库参照图 3-12 所示去建立，也存放在【高级实例.SchLib】中。下面简述其 PCB 库的建立过程。

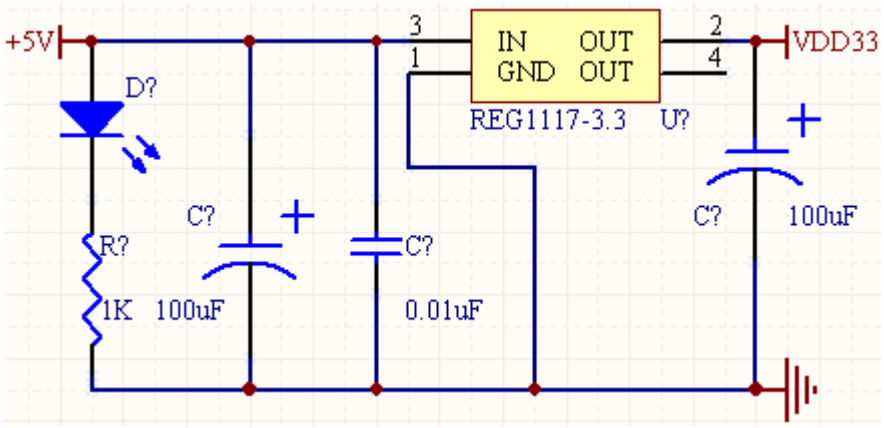


图 3-11　3.3V 电源电路

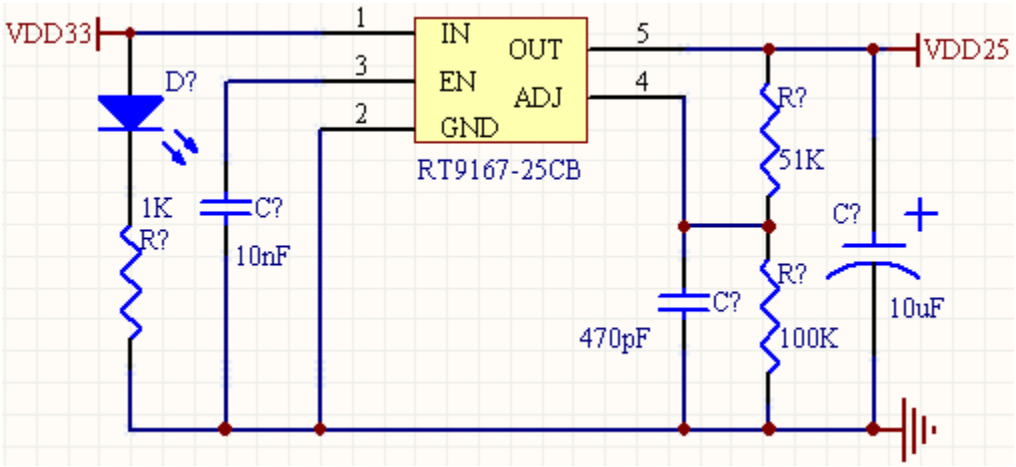


图 3-12　2.5V 电源电路

（1）打开“高级实例.PCBLIB”库文件，单击 Add 按钮，执行【Component Wizard】命令（元件封装生成向导）。

（2）弹出元件 PCB 封装向导对话框。单击<Next>，弹出的对话框用于设定元件的封装类型。在对话框中一共罗列了 12 种标准封装类型，选择“Small Outline Package(SOP)”封装。对话框中的“Select a unit”中的长度单位，选择“Metric(mm)”。

（3）单击<Next>按钮进入如图 3-13 所示的对话框，设定焊盘尺寸。执行尺寸直观地标注在示意图上，只需要将鼠标移至相应的尺寸上，单击就能重新设定焊盘尺寸。焊盘的宽度设为“0.45mm”，长度为“1mm”。

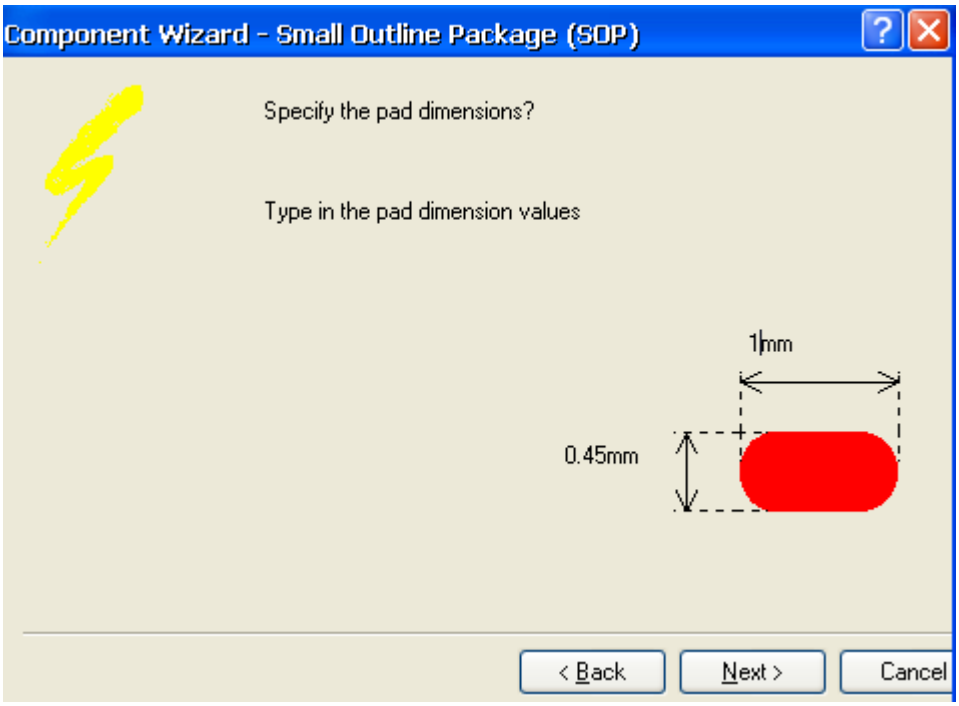


图 3-13 设置焊盘尺寸

（4）下一步设置元件管脚的相对位置与间距。根据元件说明书上的封装，分别取相邻管脚间距为“0.92mm”，列间距为“2.2mm”，如图 3-14 所示。

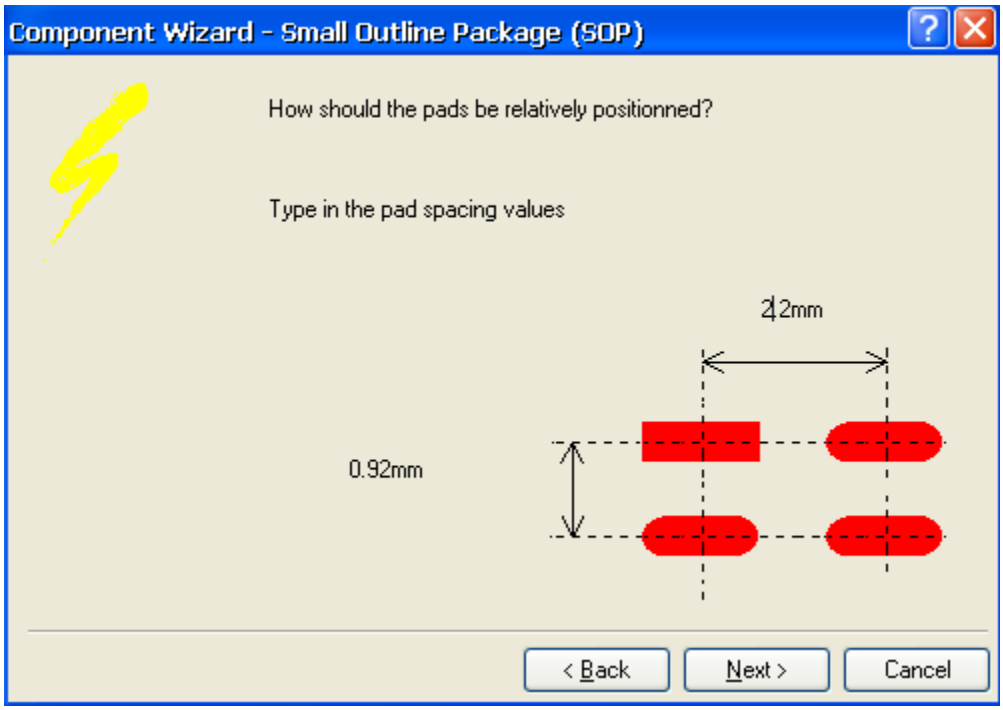


图 3-14 设置引脚间距

（5）单击<Next>，设定元件边框线宽。保持缺省值 0.2mm 不变。

（6）接下来的对话框是设定管脚数目。单击文本框右边的按钮就能改变管脚数目，设为“6”。

（7）设定元件名称。将元件名称设为“RT9167-25CB”。

（8）所有设定工作元件完成，进入向导结束对话框，单击<Finish>确认所有设置，系统自动生成了 PCB 文件。

由于 RT9167-25CB 芯片只有 5 个管脚，所以去掉（选中该管脚，点击键盘的 Delete 键）右侧中间的 5 号管脚，并将 6 号管脚改为 5 号，即完成了 PCB 库的建立，如图 3-15 所示。

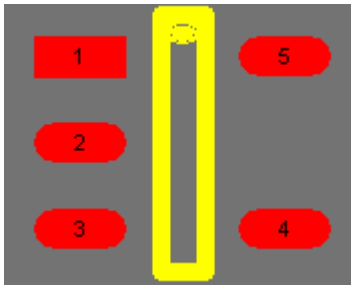


图 3-15 完成的 RT9167-25CB 封装

下面将该芯片的 PCB 封装指定为此封装：

在“高级实例.SCHLIB”库【Library Editor】面板中的【Model】栏，点击 Add 按钮，添加“Footprint”封装。然后参照 3.3.1 节讲述的方法，选择元件的 PCB 封装。

3、复位电路

复位电路用来在 ARM 工作不正常时发出复位信号。为了提供性能优越的电源监视性能，这里选用专门的系统监视复位芯片 MAX811，该芯片性能优良，可通过手动控制系统复位，还可实时监视系统的电源，一旦系统的电源低于系统复位的阈值（2.9V），MAX811 将会其作用，对系统进行复位。

先选择【Miscellaneose Devices.IntLib】库中的“SW-DPST”（按钮）元件，然后在“单击实例.IntLib”库中选择电阻。在库中搜索 MAX811 芯片，最后用导线连接线路，添加接地、电源和网络标识，构成整个复位电路，如图 3-16 所示。

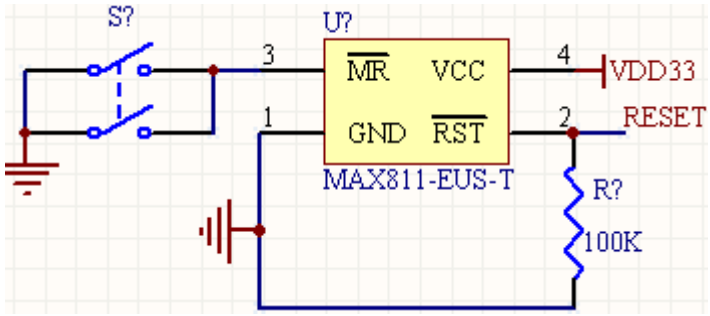


图 3-16 复位电路

4、系统时钟电路

晶振电路设计是电路应用设计中的重要环节之一，因此有必要了解晶振电路的特点、

组成以及如何选用相关电子元件。晶振的英文是 Cryatal，也叫石英晶体振动器，简称晶振。对于一个高可靠行性的系统设计来说，晶体的选择非常重要。晶体的选择至少必须考虑谐振频点、负载电容、激励功率、温度特性、长期稳定性。

晶振根据不同的使用要求及特点，通常分为以下几类：普通晶振、温补晶振、压控晶振、温控晶振等。安装晶振时，应根据其引脚功能标识与相应电路相连接，避免电源引线与输出管脚相接。

（1）普通晶振 PXO：一种没有采取温度补措施的晶体振动器。在整个温度范围内，晶振的频率稳定取决于其内部所用晶体的性能。一般用于普通场所作为本振源或中间信号，是晶振中最廉价的产品。

（2）温补晶振 TCXO:在晶振内部采取了补偿晶体频率温度特性的措施，以达到在比较大的范围内满足稳定度要求的晶体振动器。一般模拟式温补晶振采用热敏补偿网络。由于良好的开机特性、优越的性能价格比、功耗低、体积小、环境适应性较强等多方面优点，温补晶振获得了广泛应用。

（3）压控晶振 VCXO：一种可通过调整外加电压使晶振输出频率随之改变的晶体振动器，主要用于锁相环路或频率微调。压控晶振的频率控制范围及线性度主要取决于电路所用的变容二极管及晶体参数两者的组合。

（4）恒温晶振 OCXO：采用精密控温，使电路元件及晶体工作在晶体的零温度系数点上的晶体振动器。主要用作频率源或标准信号。

应选用优质的晶振来保证高精度的时钟频率。并且在线路的设计上应使晶振尽可能接近主芯片，以缩短信号线的长度，增加传输的稳定性。一般四管脚的卧式有源晶振的工作电压是 3.3V。

本例中系统时钟源直接采用外部晶振，内部 PLL 电路，可以调整系统时钟，使系统运行速度更快。

下面绘制电路板上的晶振电路。

（1）在“高级实例.IntLib”库文件中，参照前述方法添加新元件“XTAL_OSC”。晶振原理图符号如图 3-17 所示。

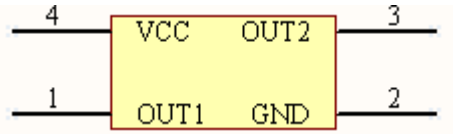


图 3-17 四脚晶振原理图

（2）手工绘制晶振的 PCB 封装。首先设定位置原点。焊盘设置中，将焊盘 1 的形状设为“Round”，它的属性对话框如图 3-18 所示，注意焊盘在“Top Layer”层。其他 3 个焊盘设为“Rectangle”。

（3）焊盘 4 的坐标为（0，0），焊盘 2 位于（190，—180）。其他两个焊盘位置与之对称。最后，晶振的 PCB 封装如图 3-19 所示。

（4）按前面 3.3.1 节讲述的方法，指定晶振封装，编译集成库。

（5）将元件拖入原理图中，完成晶振部分的原理图设计，如图 3-20 上部分所示。

（6）为实时时钟（RTC）绘制晶振电路，RTC 需要 32.768KHz 频率的时钟输入，晶

振电路如图 3-20 下部分所示。

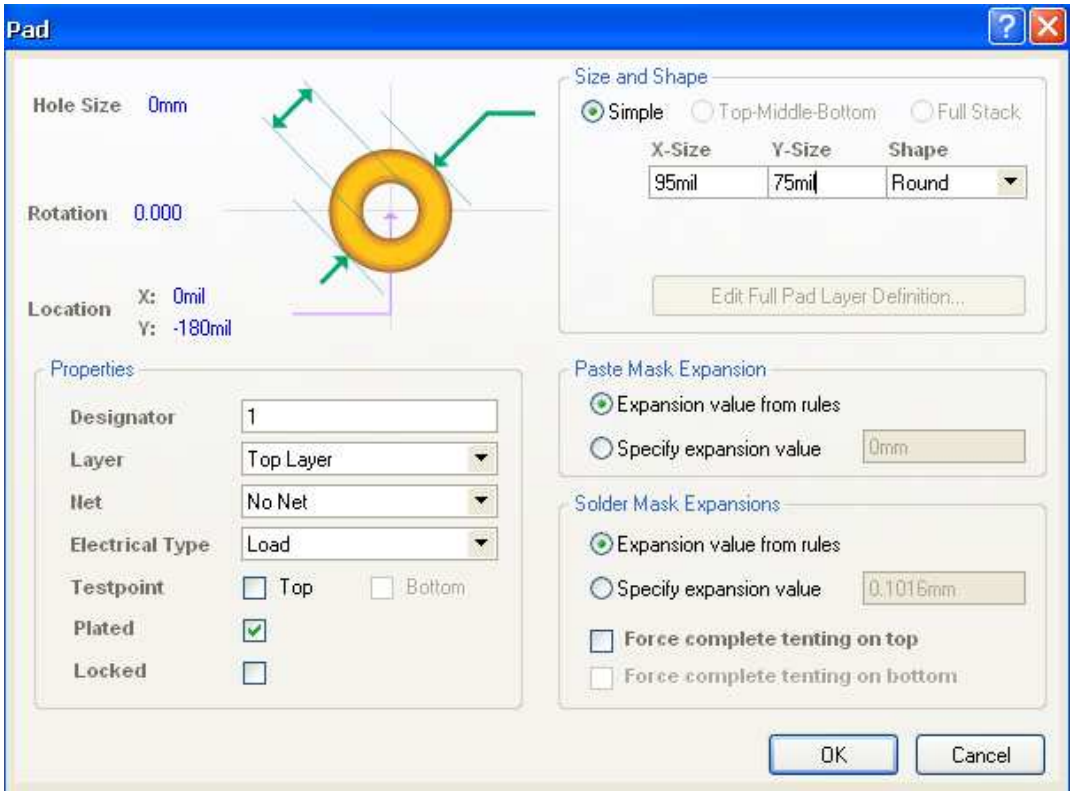


图 3-18 焊盘 1 的属性对话框

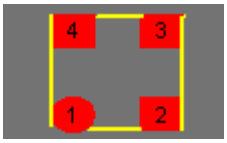


图 3-19 晶振的 PCB 封装

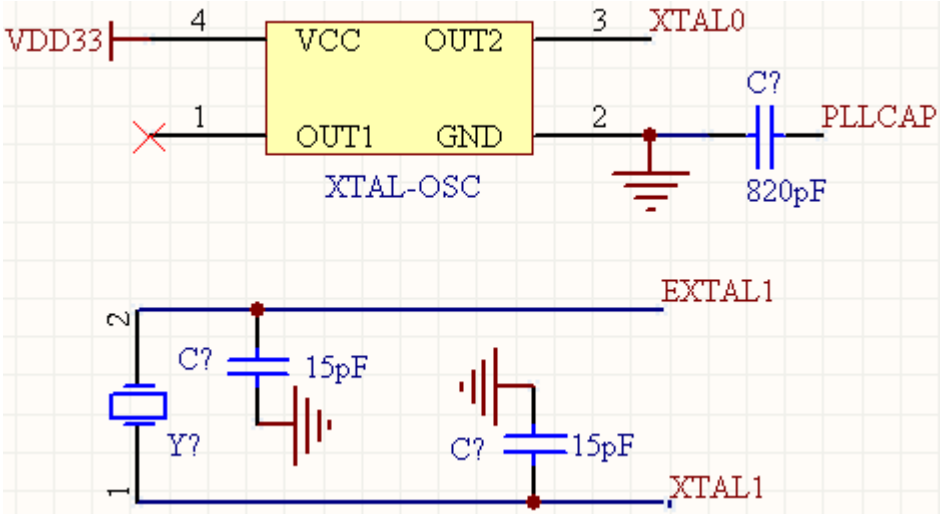


图 3-20　ARM 晶振电路

5、设置 S3C44B0 的电源管脚、接地管脚、时钟管脚和复位管脚

（1)将 S3C44B0 所有的 VSS、VSSIO、VSSADC 管脚接地；将所有的 VDDIO、VDDADC 管脚接“VDD33”，供 3.3V 电源；将所有的 VDD 管脚接“VDD25”，供 2.5V 电源。

（2）将 87、88 脚添加网络标识“XTAL1”和“EXTAL1”，与 RTC 的时钟（32.767KHz）相连；将 64 脚添加网络标识“XTAL0”，与晶振电路相连。

（3）将 50 引脚添加网络标识“RESET”，与复位电路相连。

（4）最后，要加上一些去耦电容并联在电源和地之间，目的是增强电路的电磁兼容性能 EMC,提高电源抗干扰性，如图 3-21 所示。当数字电路受到跳变电流作用时，也将产生阻抗噪声，解决问题的有效措施是设置合适的去耦电容。

去耦电容的一般配制原则是：

1）电源输入端跨接 10～100ｕF 的电解电容。如有可能，100μF 以上的更好。

2）原则上每个集成电路芯片都应布置一个 0.01uF 的瓷片电容。如遇印制板空隙不够，可每 4～8 个芯片布置一个 1～10uF 的钽电容。

3）对于抗噪能力弱、关断时电源变化大的器件，如 RAM、ROM 存储器件，应在芯片的电源线的地线之间接入去耦电容。

4）电容引线不能太长，尤其是高频旁路电容不能有引线。

5）在印制板中有接触器、继电器、按钮等元器件时，操作它们均会产生较大火花放电，必须采用 RC 电路来吸收放电电流。一般 R 取 1～2KΩ，C 取 2.2～4.7μF。

6）CMOS 的输入阻抗很高且易受感应，因此在使用时不用的管脚要接地或接正电源。

7）使用逻辑电路的建议：凡能不用高速逻辑电路的就不用；在电源与地之间加去耦电容；注意长线传输中的波形畸变；用 R-S 触发器作按钮与电子线路之间配合的缓冲电路。

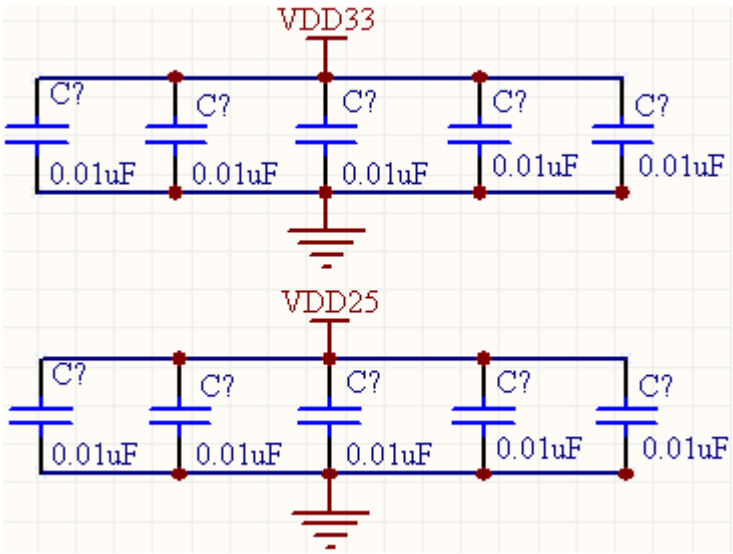


图 3-21 去耦电容

原理图上，一般将去耦电容绘制在一起。但在实际的 PCB 上，这些去耦电容并不是如此规则、靠近地并联在一起，而是放置在 ARM 的各个电源管脚附近。

注意：这里的 0.01μF 的电容，是“高级实例.IntLib”库中的电容“CAP2”。

6、子图标识的绘制

下面的设计，就将用到本章 3.2 节所讲述的层次原理图设计。在主原理图“S3C44B0.SchDoc”中绘制 3 个子图标识，从而展开三个子图。

（1）确认在“S3C44B0.SchDoc”界面，点击工具栏中的 按钮，绘制子图标识。

（2）将鼠标移至原理图的绘图区上，这时光标变为十字形状，十字的右下角有一个虚线子图标志识。子图标识会随着光标的移动而移动。

（3）单击鼠标左键，确定子图标识的左上角顶点。随后在适当位置再次单击鼠标左键，即可确定子图标识的右下角顶点。

（4）鼠标双击子图标识，弹出如图 3-22 所示的【Sheet Symbol】对话框。

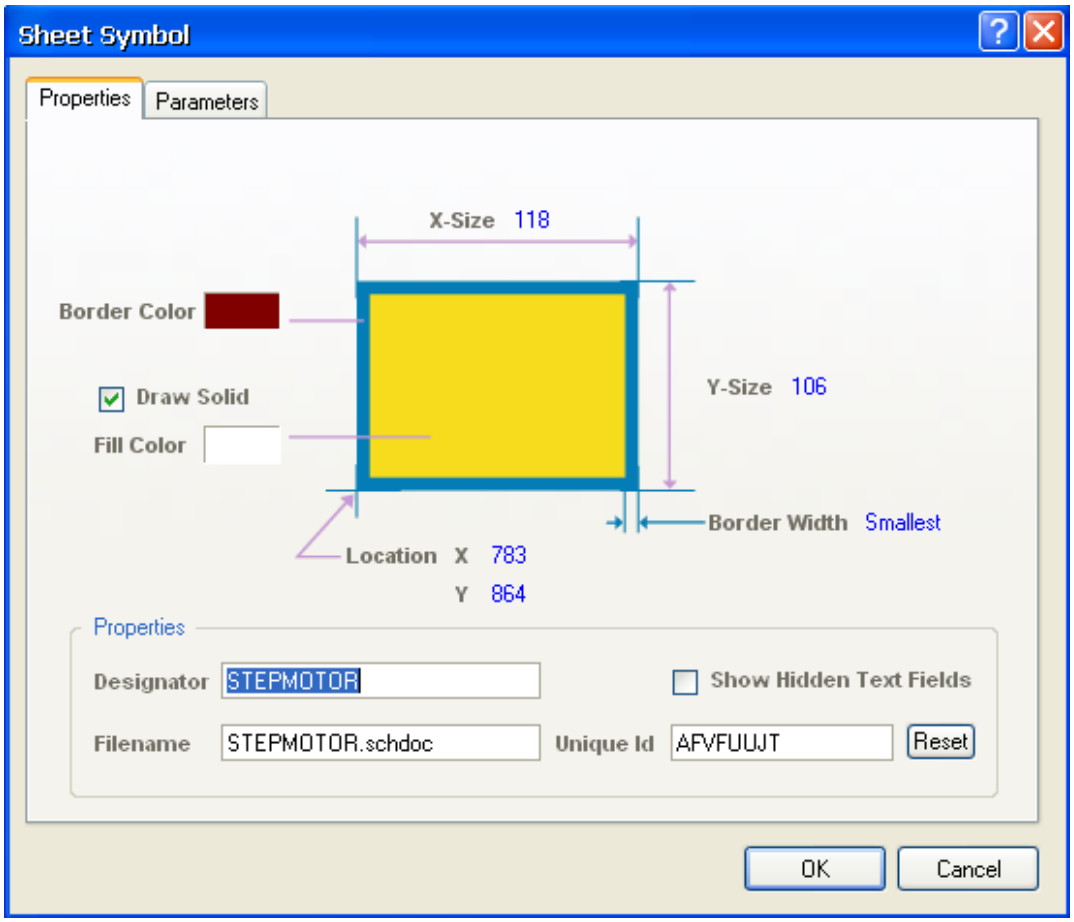


图 3-22 【Sheet Symbol】对话框

①【Location X】、【Locatio Y】：确定方块电路符号的位置；

②【X-Size】、【Y-Size】：精确确定方块电路符号的位置；

③【Border Color】：边界颜色；

④【Border Solid】：边界宽度；

⑤【Draw Solid】：是否填实；

⑥【Fill Color】：填充颜色；

⑦【Filename】：该栏中需要填写该方块电路符号所代表的子原理图的文件名，输入“STEPMOTOR.schdoc”；

⑧【Designator】：用于填写该方块电路的标志，输入“STEPMOTOR”；

⑨【Unique ID】：系统的区别码，一般不用修改。

修改名称后的子图标识如图 3-23 所示。

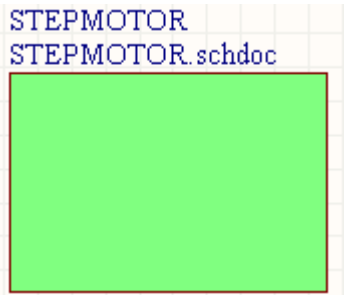


图 3-23 修改名称后的子图标识

（5）可以对子图符号的文字标注大小、颜色（改为白色）以及文字、角度进行修改。

（6）单击布线工具栏中的，执行放置子图电路端口的命令。这时鼠标光标变为十字形状。

（7）将光标移入子图标识中，单击鼠标左键，十字光标将叠加一个子图电路端口，该端口会随着光标的移动在子图符号的边缘移动。

（8）在此状态下按<TAB>键，弹出子图端口属性设置对话框，如图 3-24 所示。

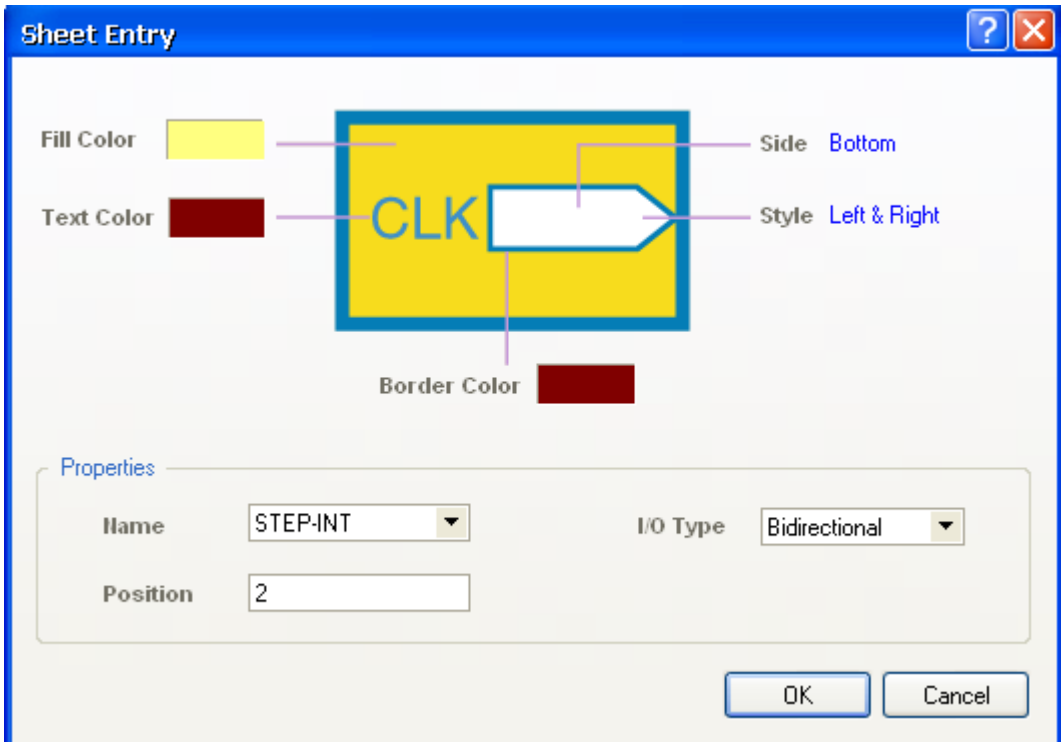


图 3-24 子图端口属性对话框

①【Name】：代表了子图电路端口的名称，在此将其改为“STEP-INT”；

②【I/O Type】：子图电路端口的输入/输出类型，单击该项右侧的下拉按钮，在下拉列表中有“Unspecified Type”（不指定）、“Output Type”（输出类型）、“Input Type”（输入类型）、“Bidirection Type”（双向类型）。本例选择为双向类型；

③【Fill Color】：设置端口符号的填充颜色；

④【Text Color】：设置文字颜色；

⑤【Border Color】：设置端口边框的颜色；

⑥【Side】：设定端口在子图电路中的放置位置；

⑦【Style】：设置端口符号的外观样式，这里就取其默认的样式。

（9）确认设置。下一个端口符号会自动附着在光标上，并且序号自动加 1.

（10）重复上面的操作步骤，添加完毕所有的端口。

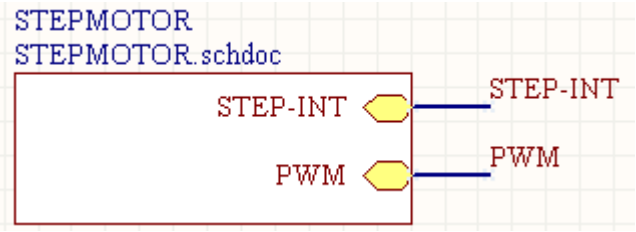


图 3-25　完成的子图 STEPMOTOR 符号

（11）然后，为管脚连接导线，注明网络名称。**同时也在 S3C44B0 的相应管脚处引出导线，标注网络名**，其中，STEP-INT 与 S3C44B0 的 39 脚相连，PWM 与 68 脚相连。如图 3-25 所示。

同样，建立其另外一个子图标识，如图 3-26 所示。并为管脚连接导线，注明网络名

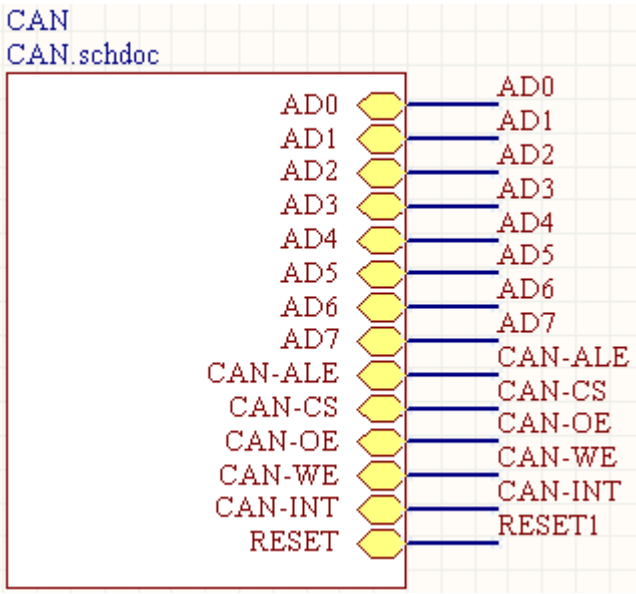


图 3-26　CAN 总线接口子图标识

称。同时也在 S3C44B0 的相应管脚处引出导线，标注网络名，其中 AD0~AD7 与 S3C44B0 的 136~129 脚相连，CAN-ALE 与 13 脚相连，CAN-CS 与 24 脚相连，CAN-OE 与 15 脚相

连，CAN-WE 与 16 脚相连，CAN-INT 与 40 脚相连，RESET1 与 38 脚相连。

这样就完成了整个“S3C44B0.Schdoc”原理图的设计。

3.4 子原理图设计

下面开始进行 2 个子原理图的绘制。

执行菜单命令【Design】/【Create Sheet From Symbol】，光标变为十字形，将其移到子图标识“STEPMOTOR”上。单击鼠标左键，系统自动弹出是否转换输入/输出方向的确认对话框。

单击 Yes 按钮，Protel DXP 自动生成一个与子图电路的【Filename】同名的原理图文件“STEPMOTOR.SchDoc”。这个新的原理图已经布好了与子图电路相对应的输/输出端口，并且有着相同的名称和对应的输入/输出方向。注意保存。

接下来就是在系统自动生成的子原理图中添加元件，完成电路绘制。

3.4.1 “STEPMOTOR.schdoc”子原理图

本例中步进电机驱动电路采用 TA8435 芯片。 TA8435 是东芝公司生产的单片正弦细分二相步进电机驱动专用芯片，该芯片具有以下特点：

1）工作电压范围宽（10－40V）；

2）输出电流可达 1.5A（平均）和 2.5A（峰值）；

3）具有整步、半步、1/4 细分、1/8 细分运行方式可供选择；

4）采用脉宽调试式斩波驱动方式；

5）具有正/反转控制功能；

6）带有复位和使能引脚；

7）可选择使用单时钟输入或双时钟输入

TA8435 芯片共有 25 个引脚，采用 ZIP 封装，其引脚如图 3-27 所示。

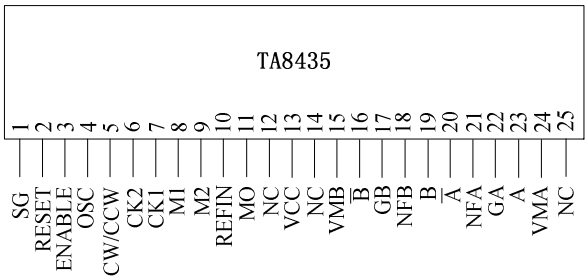


图 3-27　TA8435 引脚图

Protel DXP 自带库中没有该芯片的原理图及 PCB 封装，所以需要自己建立其原理图库及 PCB 库。

为方便绘制步进电机电路图，TA8435 的原理图在绘制时引脚可不按顺序排列，而是按功能排列，这样容易走线。

1、建立原理图库

采用以前讲述的方法在“高级实例.SchLib”中添加并绘制 TA8435 的原理图，如图 3-28 所示。

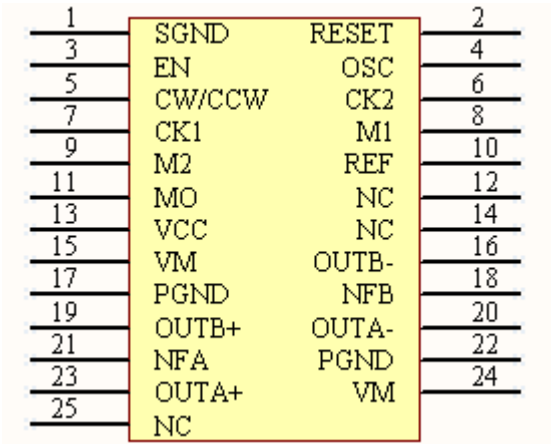


图 3-28 TA8435 的原理图

2、添加 TA8435 封装

TA8435 采用 25 脚 ZIP 封装，在建立其 PCB 库时可选择 DIP 类型，并设成 26 脚，最后去掉一个管脚即可。步骤如下：

（1）选择“高级实例.PcbLib”文件，打开【PCB Library】面板，单击 Add 按钮，弹出 PCB 库设计向导“Component Wizard”。

（2）单击 Next 按钮，在出现的对话框中选择“DIP”类型，并选择公制“Metric”。如图 3-29 所示。

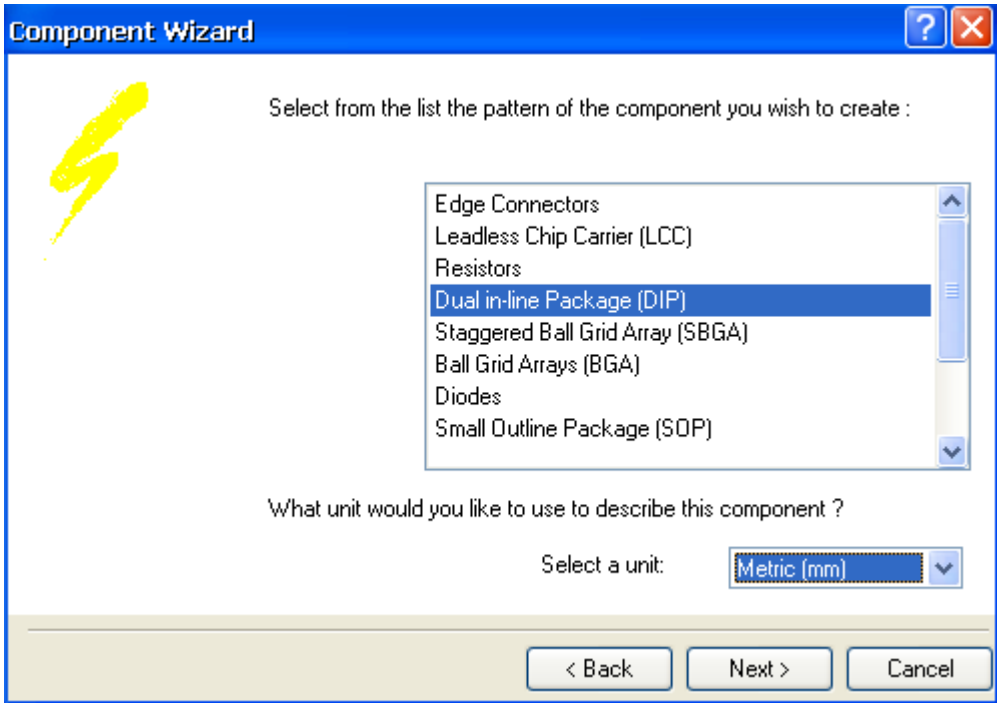


图 3-29 设置封装类型

（3）单击 Next 按钮，设置焊盘尺寸，将焊盘孔设为 1.2mm，焊盘长设为 2mm，宽设为 1.8mm。如图 3-30 所示。

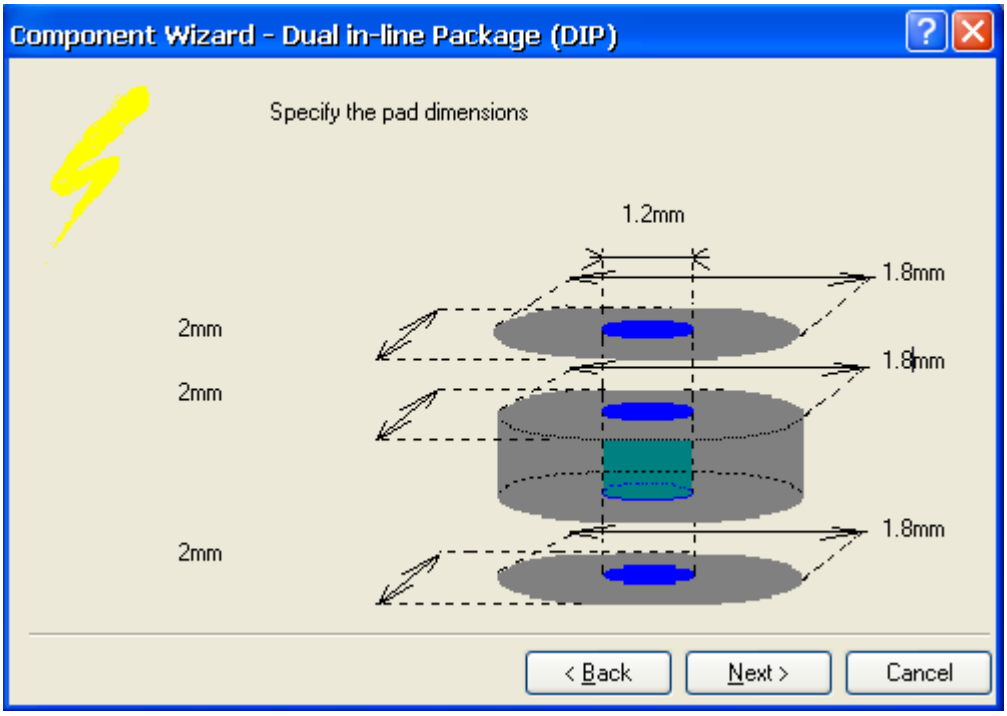


图 3-30 设置焊盘尺寸

（4）单击 Next 按钮，将焊盘纵向间距设为 2.54mm，横向间距设为 2mm。如图 3-31 所示。

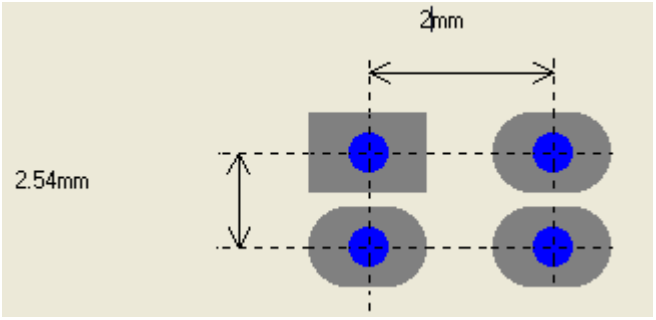


图 3-31 设置焊盘间距

（5）单击 Next 按钮，取默认值。

（6）单击 Next 按钮，将含盘数设为 26。

（7）单击 Next 按钮，将该 PCB 封装命名为 TA8435。

（8）单击 Next 按钮，PCB 向导完成，单击 Finish 按钮结束。生成的 PCB 封装如图 3-32 所示（图中将焊盘旋转成了横向）。

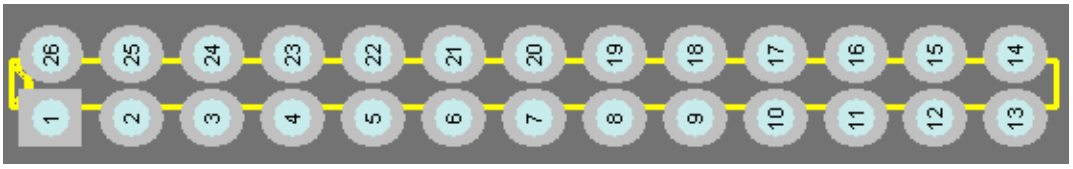


图 3-32 生成的 PCB 封装

从图 3-32 中可以看出，向导生成的 PCB 封装与我们需要的并不一致，所以需要手工调整。

（1）由于 TA8435 只有 15 个引脚，所以删掉图 3-32 中的第 26 号管脚。

（2）TA8435 采用左 13 脚、右 12 脚的间隔封装，所以选中第 14~25 脚，将其向上拖移 1.27mm（注意观察屏幕左下方的状态栏中 Y 坐标的变化，保持 X 坐标不变）。

（3）双击各引脚焊盘更改引脚号，使其左侧从上到下为 1、3、5….25，右侧从上到下为 2、4、6…24。

（4）删掉原有的黄色轮廓线，自行绘制其轮廓（注意切换到 Top Overlay 层），上下轮廓与焊盘间隙均为 4mm，左右与焊盘间隙各为 2mm。

完成的 PCB 封装如图 3-33 所示。

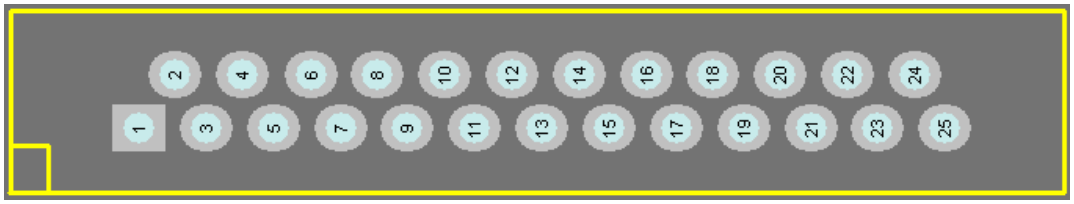


图 3-33 完成的 TA8435 的 PCB 封装

按照以前的方式，为 TA8435 的原理图符号指定其封装，这样自建的集成库中就有了 TA8435 元件及其封装。

3、绘制步进电机电路图

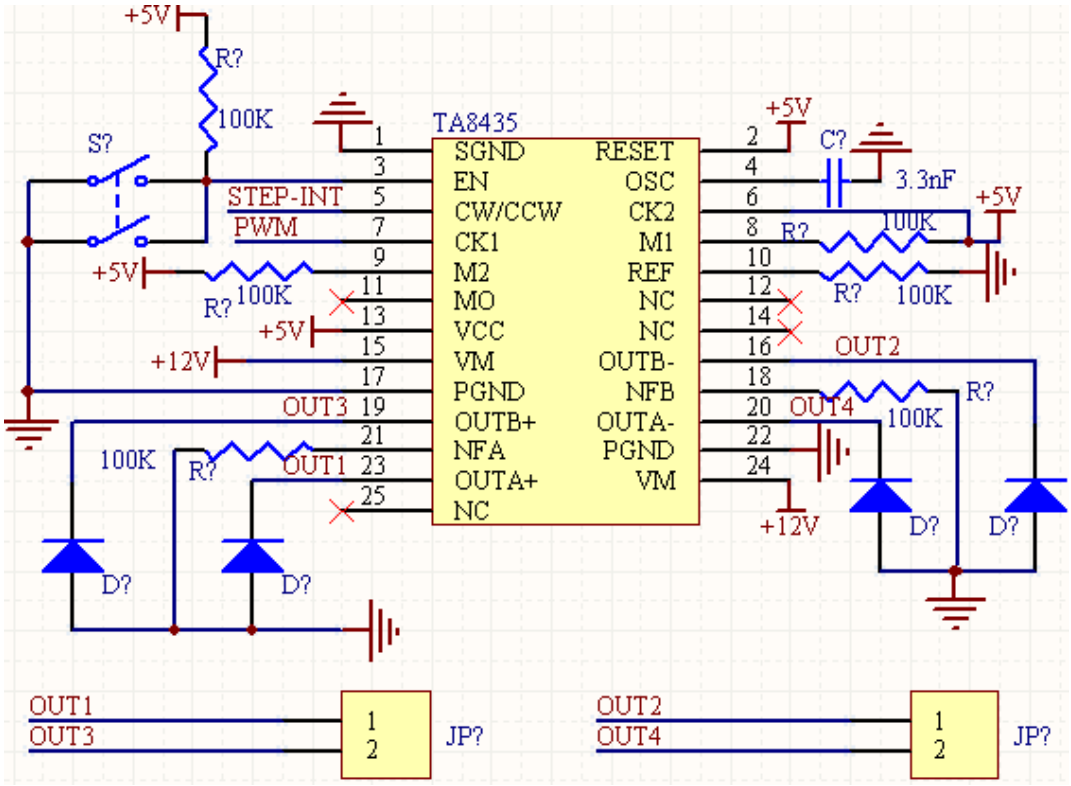


图 3-34 步进电机驱动原理图

从“高级实例.SchLib”中选出合适数量的电阻、电容以及刚建立好的 TA8435，并从 Protel DXP 自带的器件库中选出按键和二极管，从 Protel DXP 自带的接插件库中选出两个 2 针的接插件，绘制如图 3-34 所示的步进电机驱动原理图。

两个接插件引出 4 条线路，用来连接步进电机的 4 个接线端子。

3.4.2 CAN 总线接口子原理图绘制

本例中 CAN 总线控制器选用 SJA1000 芯片，并配有 TJA1040 总线收发器。SJA1000 是一种独立控制器，用于移动目标和一般工业环境中的区域网络控制。

SJA1000 有 DIP28 和 SO28 两种封装，本例选用 SO28 封装。Protel DXP 自带库中没有 SJA1000 和 TJA1040 的原理图及引脚封装，所以需要自行建立。

1、SJA1000 原理图库建立

按照以前的方式在“高级实例.SchLib”中添加并绘制 SJA1000 的原理图符号。为了连线方便，引脚没有按顺序排列，而是按功能排列。建立好的原理图符号如图 3-35 所示。

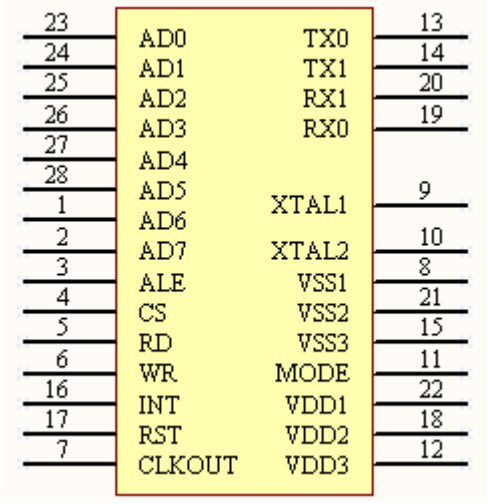


图 3-35　SJA1000 原理图符号

2、TJA1040 原理图库建立

在“高级实例.SchLib”中添加并绘制 TJA1040 的原理图符号。建立好的原理图符号如图 3-36 所示。

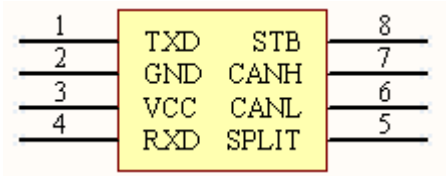


图 3-35　TJA1040 原理图符号

3、SJA1000 的 PCB 库建立

SJA1000 采用 SO28 封装，可以根据手册中的引脚参数来建立其 PCB 库。

（1）选择“高级实例.PcbLib”文件，打开【PCB Library】面板，单击 Add 按钮，弹出 PCB 库设计向导“Component Wizard”。

（2）单击 Next 按钮，在出现的对话框中选择“SOP”类型，并选择公制“Metric”。如图 3-36 所示。

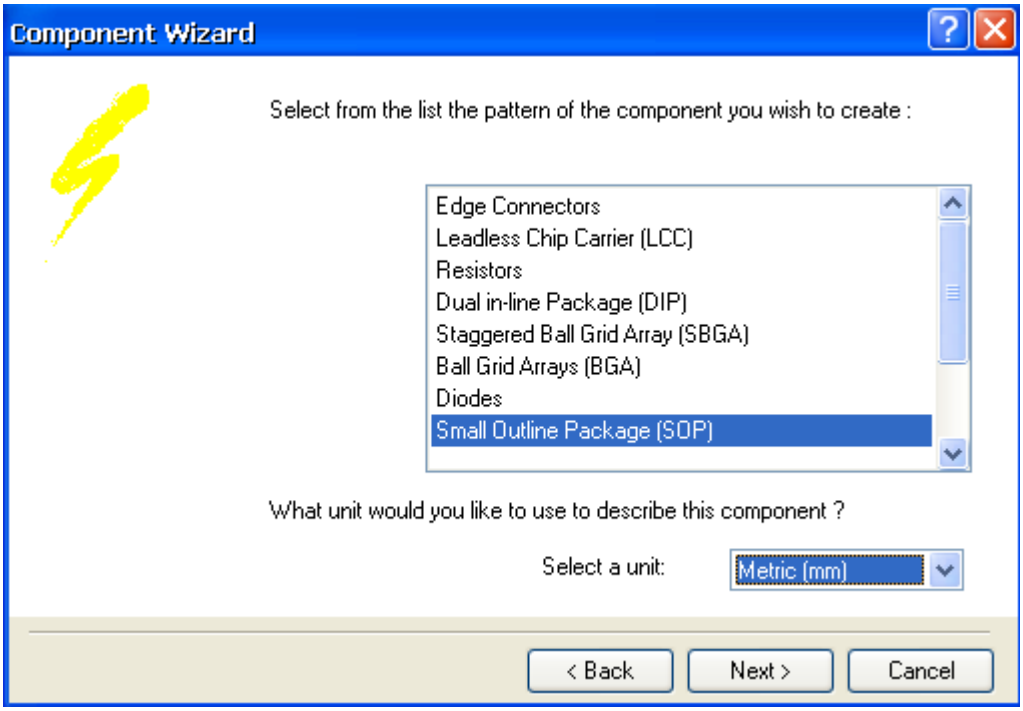


图 3-36 设置封装类型

（3）单击 Next 按钮，设置焊盘尺寸，焊盘长设为 1.4mm，宽设为 0.6mm。如图 3-37 所示。

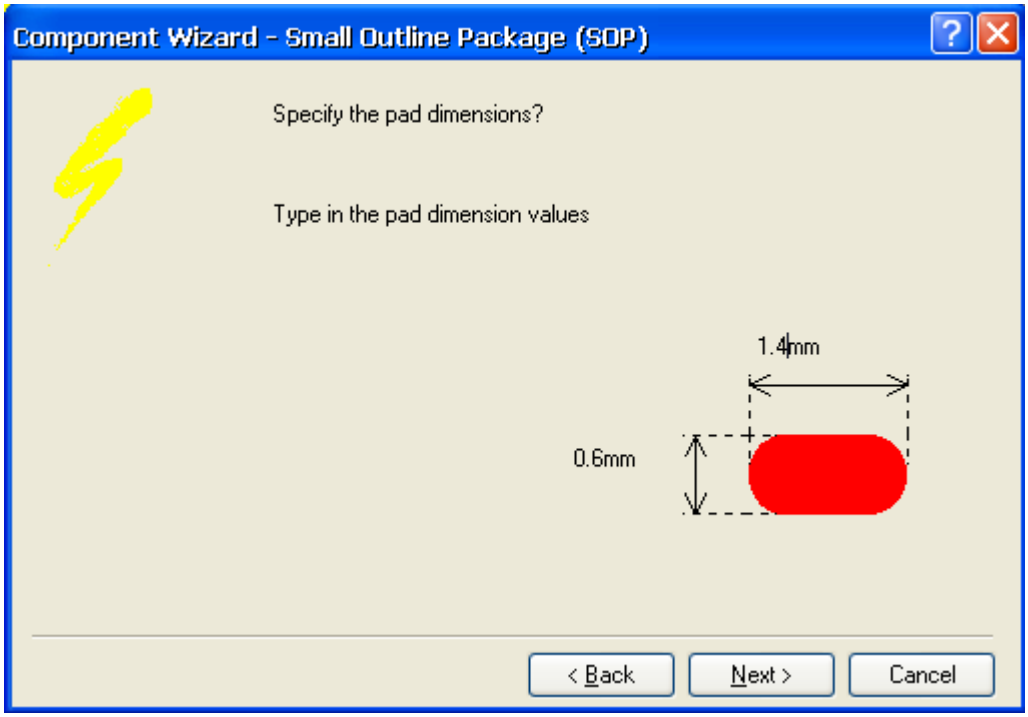


图 3-37 设置焊盘尺寸

（4）单击 Next 按钮，将焊盘纵向间距设为 1.27mm，横向间距设为 9.6mm。如图 3-38 所示。

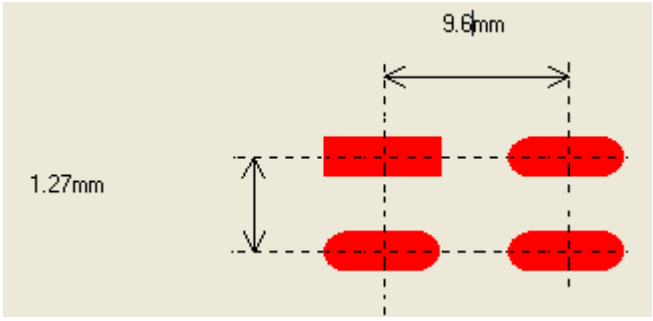


图 3-38 设置焊盘间距

（5）单击 Next 按钮，取默认值。

（6）单击 Next 按钮，将焊盘数设为 28。

（7）单击 Next 按钮，将该 PCB 封装命名为 SJA1000。

（8）单击 Next 按钮，PCB 向导完成，单击 Finish 按钮结束。生成的 PCB 封装如图 3-39 所示。

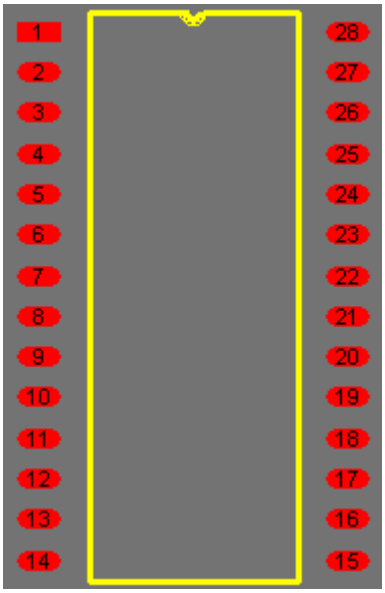


图 3-39 生成的 SJA1000 芯片的 PCB 封装

按照以前的方式，为 SJA1000 的原理图符号指定其封装，这样自建的集成库中就有了 SJA1000 元件及其封装。

4、绘制 CAN 总线接口原理图

确定在“S3C44B0.SchDoc”界面，执行菜单命令【Design】/【Create Sheet From Symbol】，光标变为十字形，将其移到子图标识“CAN”上。单击鼠标左键，系统自动弹出是否转换输入/输出方向的确认对话框。

单击 Yes 按钮，Protel DXP 自动生成一个与子图电路的【Filename】同名的原理

图文件“CAN.SchDoc”。这个新的原理图已经布好了与子图电路相对应的输/输出端口，并且有着相同的名称和对应的输入/输出方向。注意保存。

接下来就是在系统自动生成的子原理图中添加元件，完成电路绘制。完成的 CAN 接口电路原理图如图 3-40 所示。

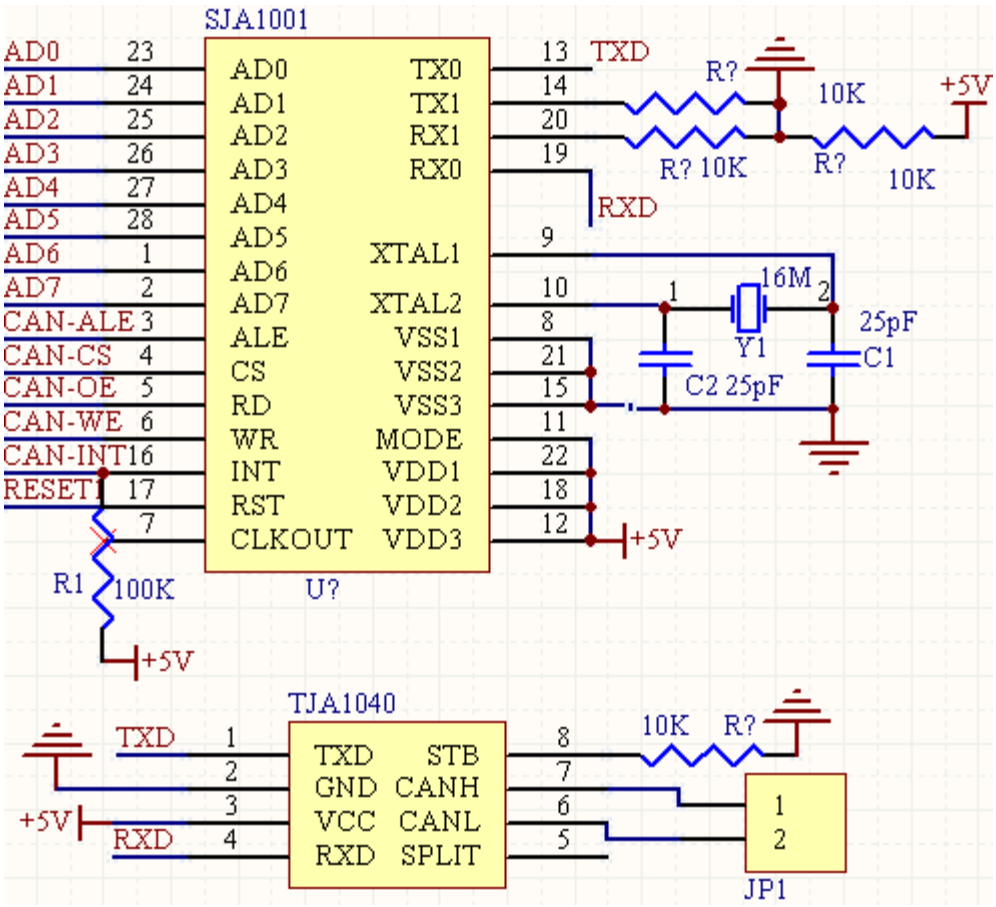


图 3-40　完成的 CAN 接口电路原理图

5、指定 TJA1040 的 PCB 封装

Protel DXP 自带库中虽然没有 TJA1040 专门的封装，但是该芯片的封装是标准的 SOP8 封装，所以只需要指定其封装即可。

（1）在“CAN.SchDoc”原理图中，双击“TJA1040”。

（2）在弹出的对话框右下角的【Model】栏中，点击添加按钮 Add 。在弹出的对话框中选择“Footprint”，单击 OK 按钮。

（3）系统弹出【PCB Model】对话框，单击 Browse... 按钮，在弹出的【Browse Libraries】对话框中点击 Find... 按钮。

（4）在弹出的【Search Libraries】对话框的 Name 栏输入 **SO-G8**，查找 SO-8 封装，结果如图 3-41 所示。可以看出，有很多 SO-G8 封装，它们都是标准的，完全相同，任一

选择一个，单击 Select 按钮即可。

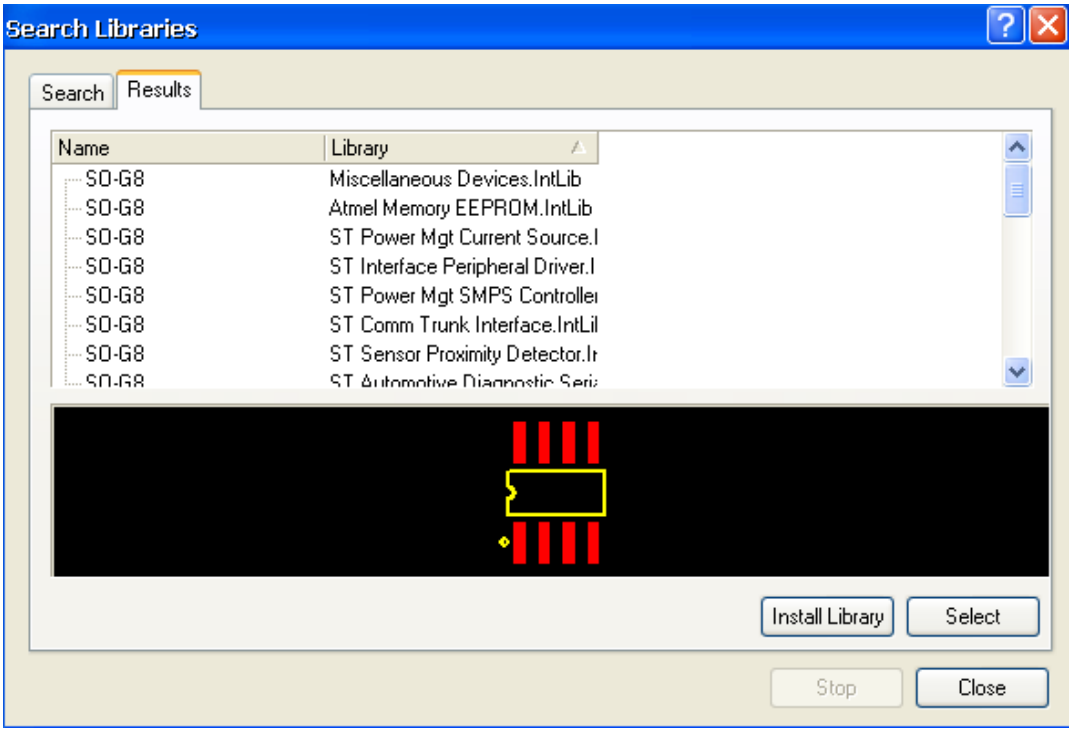


图 3-41　SO-G8 封装的搜索结果

6、编排元件序号

在上述绘制原理图的过程中，没有去特意地编排元件序号，所以比较混乱。执行菜单命令【Tools】/【Annotate】后，弹出【Annotate】对话框，选择第 4 种排序方式。单击 Update Changes List 按钮后，再单击弹出的小对话框的 OK 按钮，然后单击 Accept Changes (Create ECO) 按钮，就会弹出另一个对话框。单击该对话框上的 Execute Changes 按钮，然后单击 Close 按钮，即完成了重新编排元件序号。

7、ERC 校验

执行菜单命令【Project】/【Compile All Projects】，进行 ERC 校验，弹出【Message】对话框。从中可以看出当前原理图的问题。如果编译原理图后，没有自动弹出【Message】对话框，可点击绘图区下方的【Message】标签。

3.5 PCB 设计

3.5.1 绘制 S3C44B0 芯片的 PCB 封装

原理图中，ARM 芯片 S3C44B0 的封装还没有建立，在转 PCB 图之前需要建立其 PCB 封装。该芯片采用 16 引脚的 LQFP 封装。下面介绍 S3C44B0 的 PCB 封装建立方法。

（1）选择“高级实例.PcbLib”文件，打开【PCB Library】面板，单击 Add 按钮，弹出 PCB 库设计向导“Component Wizard”。

（2）单击 Next 按钮，在出现的对话框中选择“QUAD”类型，并选择公制“Metric”。如图 3-42 所示。

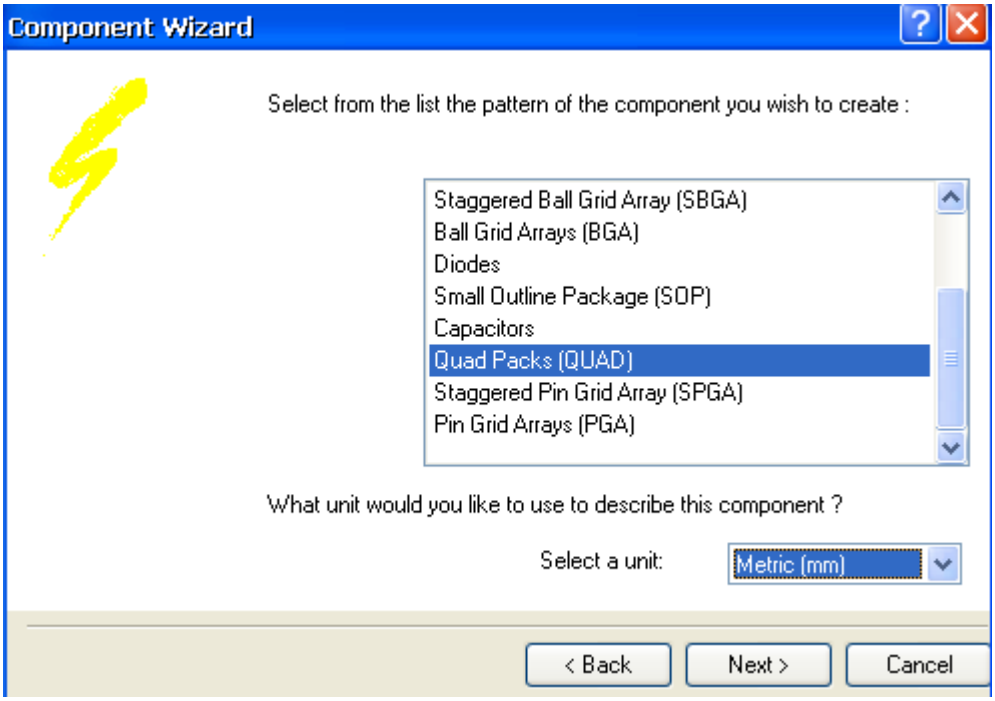


图 3-42 设置封装类型

（3）单击 Next 按钮，设置焊盘尺寸，焊盘长设为 1.2mm，宽设为 0.3mm。如图 3-43 所示。

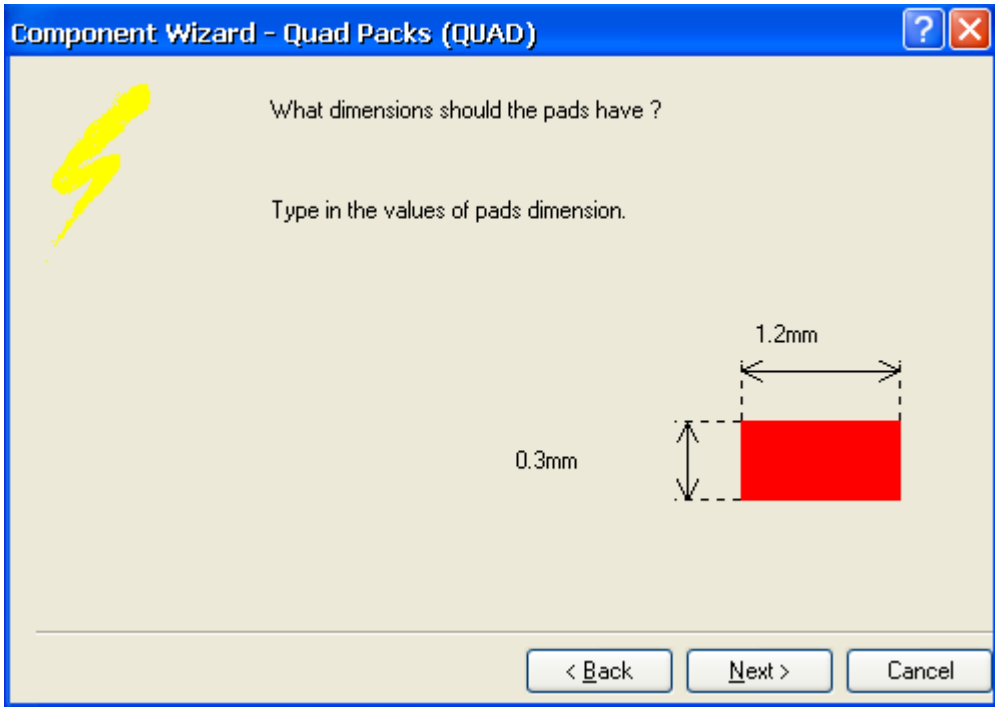


图 3-43 设置焊盘尺寸

（4）单击 Next 按钮，取默认值。将第 1 个焊盘设为圆角，其它焊盘设为矩形。

（5）单击 Next 按钮，取默认值。

（6）单击 Next 按钮，将焊盘纵向间距和横向间距都设为 0.5mm，相邻面间距设为设为 2.75mm。如图 3-44 所示。

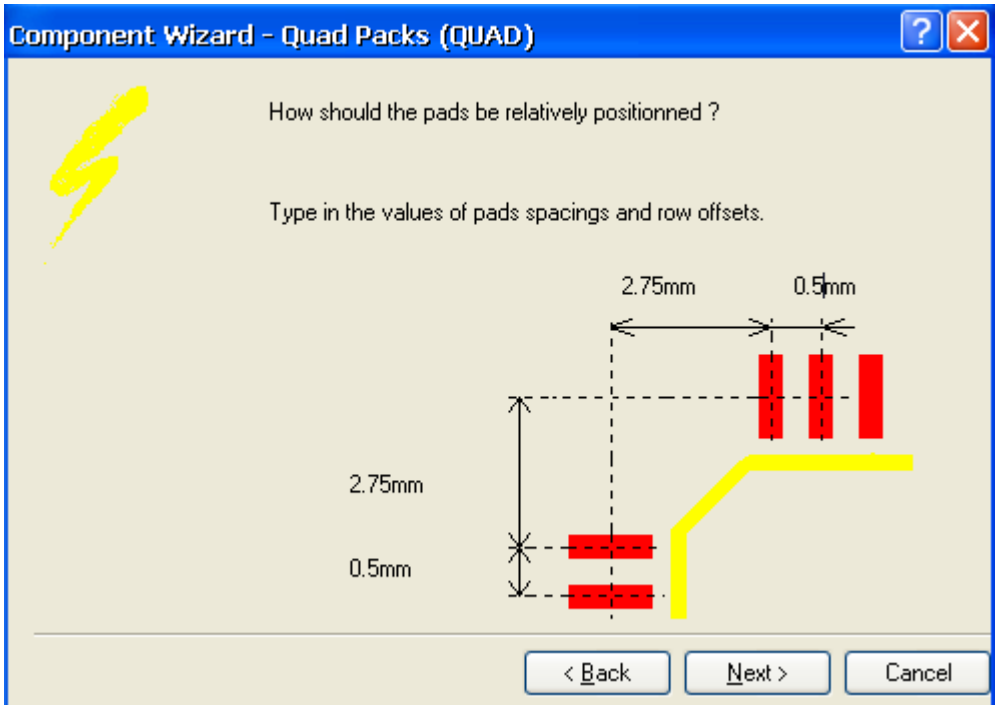


图 3-44 设置焊盘间距

（7）单击 Next 按钮，设定第 1 引脚位置及引脚号递增方向，取默认值。

（8）单击 Next 按钮，将横向及纵向焊盘数都设为 40。

（9）单击 Next 按钮，将该 PCB 封装命名为 S3C44B0。

（10）单击 Next 按钮，PCB 向导完成，单击 Finish 按钮结束。生成的 PCB 封装如图 3-45 所示。

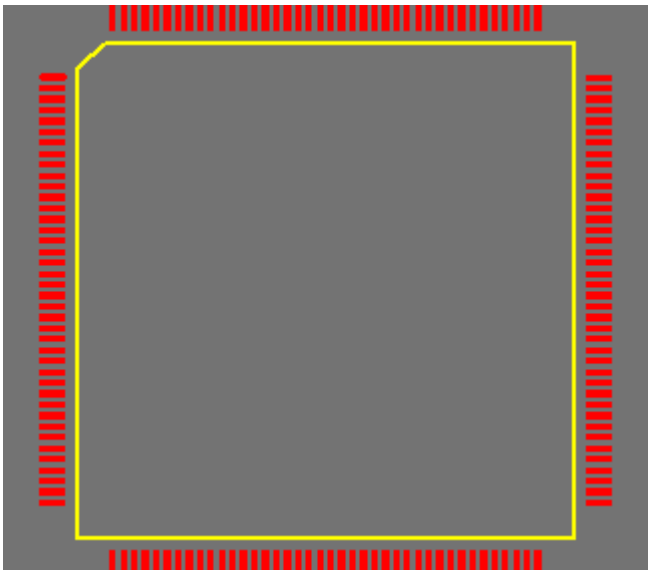


图 3-45 生成的 S3C44B0 芯片的 PCB 封装

按照以前的方式，在原理图文件中双击 S3C44B0 的原理图符号，为其指定封装。

3.5.2 PCB 生成向导

完成原理图设计后，就进入 PCB 图的设计。在将设计从原理图编辑器切换到 PCB 编辑器前，需要创建一个空白的 PCB 文件。下面利用 Protel DXP 的 PCB 文件生成向导，来完成这一步：

（1）右键单击“高级实例.PRJPCB”，在弹出的菜单中执行【New】/【Other】，单击左侧面板中最下方的 PCB Board Wizard... 命令，进入 PCB 文件生成向导。系统会弹出【PCB Board Wizard】欢迎对话框。

（2）系统自动弹出【PCB Board Wizard】欢迎对话框。

（3）单击<Next>按钮继续，下一步对话框是设置 PCB 的尺寸单位。选择“Metric”的单选框，将尺寸单位设为公制“mm”。

（4）设置 PCB 的类型。选择“Custom”选项，根据需要自己来规划 PCB。

（5）设定 PCB 的尺寸参数，图纸的宽度（Width）：200mm，高度（height）:150mm.

①将“Title Block and Scale”、“Legend String”、“Dimgension Lines”三个选项设为非选中状态；

②将“Dimgension Layer”设为 None，不添加尺寸标注层。

（6）接下来是板层设置对话框。因为本例是 4 层板，所以选择 2 个信号层（Signal Layer）”和 2 个内电层（Power Plane）。

（7）设置过孔样式，选择通孔（Thruhole Vias only）。其他选项为盲孔和过孔（Blind and Buried Vias onlyu）。

（8）设置元件/布线策略。根据器件的选型，看使用的器件多是直插式元件（Through-hole components）还是表贴式元件（Surface-mount components），还要看元件的安装方式是单面或是双面（Both sides）。本例中选择“Surface-mount components”和将元件只布置一面。

（9）设置导线和过孔的尺寸以及安全间距等参数。取默认值。

（10）单击<Finish>按钮，完成 PCB 板向导的设置。

PCB 板生成向导创建的 PCB 文件名为“PCB1.pcbdoc”，将其保存为“高级实例.pcbdoc”。下一步，设定工作层面。

3.5.3 工作层面的说明和设置

Protel DXP 提供的工作层面主要有以下几种类型。

（1）信号层（Signal Layers）

前面提到过 Protel DXP 中共有 32 个信号层，主要包括【Top Layer】、【Bottom Layer】、【Mid Layer】1～30。信号层是主要用来放置元件和布线的工作层。【Top Layer】、【Bottom Layer】是顶层、底层敷铜布线层面，可以放置元件和布线。【Mid Layer】为中间布线层，布置信号线。

（2）内部电源/接地层（Internal Planes）。

Protel DXP 提供了 16 个内部电源/接地层【Plane1】～【Plane16】，用于布置电源线和地线。

（3）机械层（Mechanical Layers）

Protel DXP 提供了 16 个机械层。机械层一般用来绘制印制电路板的边框（边界），通常只需使用一个机械层。

（4）防护层（Mask layers）

主要用于防止电路板上不希望镀锡的地方被镀上锡，包括阻焊层【Solder Mask】和锡膏防护层【Paste Mask】。Protel DXP 提供了顶层(Top solder)和底层（Bottom mask）两个阻焊层；同样提供了顶层(Top Paste)和底层（Bottom Paste）两个锡膏防护层。

（5）丝印层（Silkscreen）

Protel DXP 提供了(Top Overlay)和底层（Bottom Overlay）两个丝印层。丝印层主要用于绘制元件的外形轮廓。

（6）其他工作层（Other）

①【Keep Out Layer】：禁止布线层；

②【Multi Layer】：设置多层面；

③【Drill Guide】：钻孔位置；

④【Drill Drawing】：钻孔。

尽管 Protei DXP 提供了多达 72 层的工作层面，但在设计工作中，常用到的工作层有顶层信号层、底层信号层、丝印层和禁止布线层。

可以使用 Protel DXP 的图层堆栈管理器，在管理器内添加、删除工作层面，更改各个工作层面的顺序。

执行菜单命令【Design】/【Layer Stack Manager】，在弹出的如图 3-46 所示的【Layers Stack Manager】对话框（图层堆栈管理器）中，选择设定工作层面。

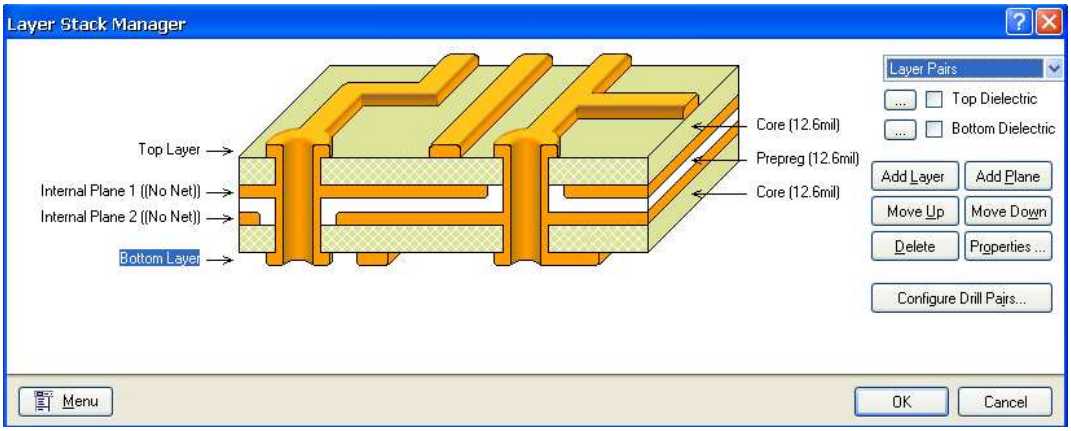


图 3-46 【Layers Stack Manager】对话框

可以通过单击右侧的各功能按钮添加、删除、上移、下移层面，也可设置层面属性。

执行菜单命令【Design】/【Board Layers】，会弹出的如图 3-47 所示的【Board Layers】对话框（图层设置）。每个工作层面后面都有一个复选框，单击复选框，出现勾号，即打开该工作层面。否则，该工作层面处于关闭状态。

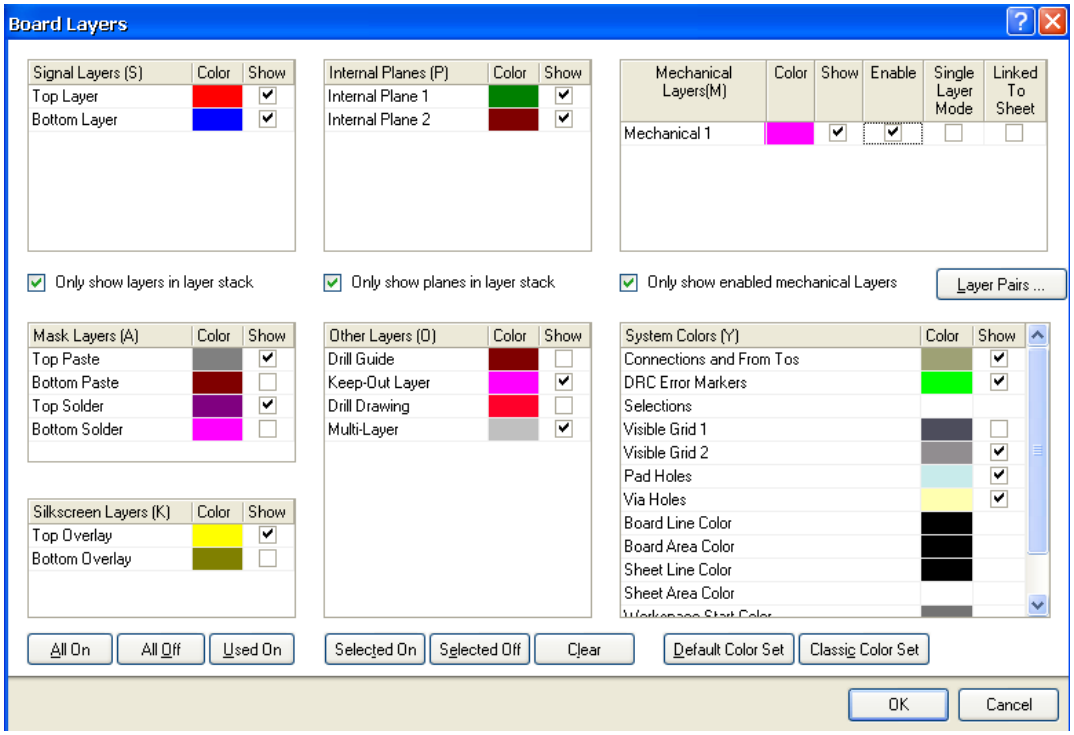


图 3-47 【Board Layers】对话框

本例设置如下：保留信号层和电源/接地层，另外取“Top Paste”、“Top Solder”层、“Keep-Out Layer”、“Multi-Layer”层和 1 个机械层。

工作层面的颜色也可以由用户定义。

设置环境参数。执行菜单命令【Design】/【Option】，或者在绘图区点击鼠标右键，在【Options】的弹出菜单中执行【Board Options】命令，弹出【Board Options】对话框。

①【Measurement Unit】：单位设为“mil”。

②【Snap Grid】：捕获栅格，指的是光标捕获图件时跳跃的最小间隔。X、Y 方向均设为 5mil。

③【Comonent Grid】：元件放置捕获栅格。X、Y 方向均设为 5mil。

④【Electrcal Grid】：电气栅格，取默认。

⑤【Visible Grid】：可视栅格。选择栅格的类型为“Lines”，第一可视栅格 Grid1 和第二可视栅格 Grid2 均设为“100mil”。

技巧：

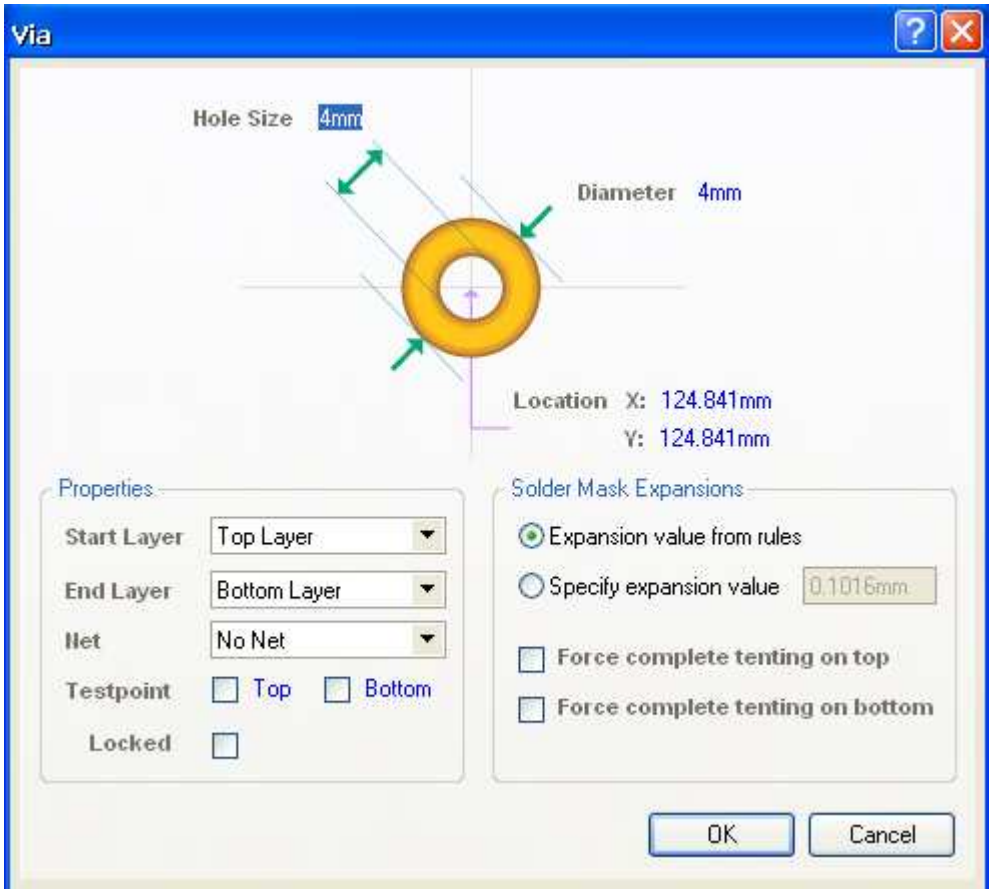
●捕获栅格和电气栅格应该相近。相差过大，在连线时光标会很难捕捉到用户需要的电气连接点。

●电气栅格和捕获栅格不能大于元件封装的管脚间距，否则会给用户连线带来不必要的麻烦。

●第一可视栅格和第二可视栅格建议设为相同的栅距，有助于用户掌握元件。图纸和线间距的大小。

根据 PCB 板的安装要求，需要在固定安装孔的位置上放置适当大小的通孔。对于 3mm 的螺钉，一般采用内外径均为 4mm 的通孔。先选择【Board Option】将单位换为 Metric，在画完通孔后，再换回英制单位。

单击工具栏上的  按钮，按<TAB>键，会弹出图 3-48 所示的过孔设置对话框。将内外径均设为 4mm 的通孔，然后在 PCB 板的 4 个顶角处放置 4 个通孔，如图 3-49 所示。 注意：放置通孔时，要选择是在“Multi-Layer”层。



3-48 过孔设置对话框

图 3-49 放置 4 个固定安装孔

设置完毕后，在 PCB 文件中同步导入网络表和元件封装。

（1）确认在 PCB 绘图区，执行菜单命令【Design】/【Import Changes From】。

注意：要保证“高级实例.IntLib”库文件也加载到了工程里。否则，其中的元件图封

装无法载入到 PCB 图中。

（2）在弹出的【Engineering Change Order】对话框中，点击 Validate Changes 按钮。右侧“Check”列就会出现绿色的对号或红色的叉号，如果是红色的叉号表明该元件的引脚封装或连接不正确，应该回到原理图修改，直到全部都为绿色的对号。

（3）核实无误后，点击 Execute Changes 按钮，执行变更。然后单击 Close 按钮。

布线及后期处理请听专业教师讲解，这里不再给出。

第 4 章 常用操作

4.1 原理图打印

在原理图绘制完毕后，经常需要打印出来以便备份与查看，下面介绍原理图的打印方法。

首先设置打印机，执行菜单命令【File】/【Page Setup】，弹出打印机设置对话框，如图 4-1 所示。

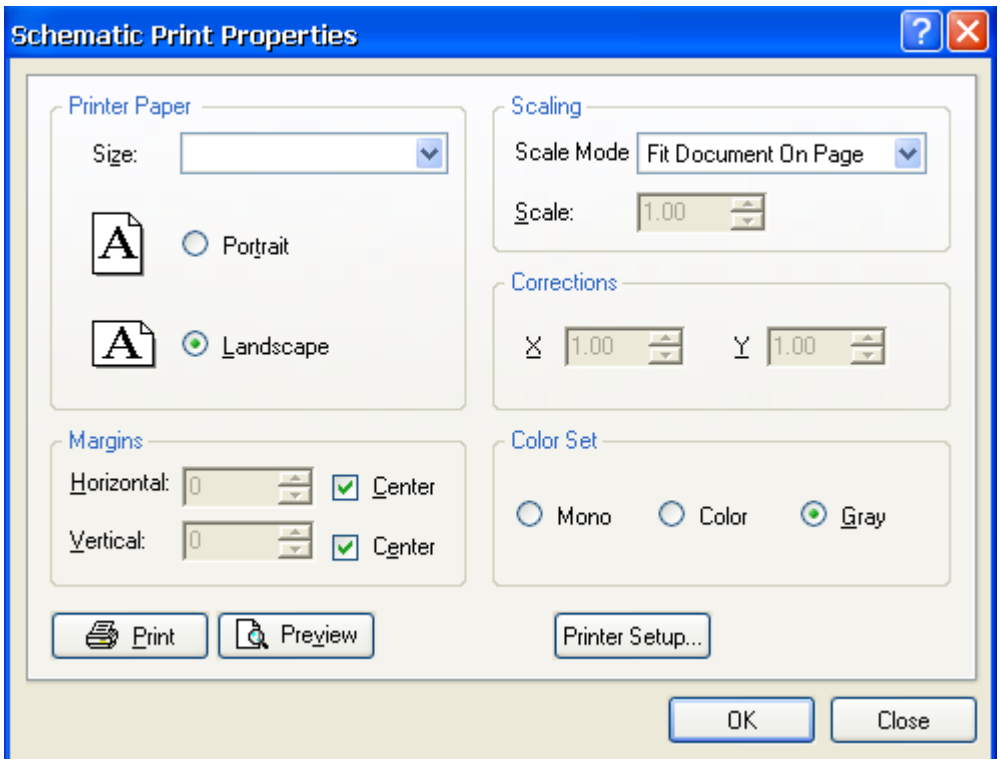


图 4-1 原理图打印属性对话框

说明如下：

【Printer Paper】分组框：打印机的纸张设置。

① “Size”：打印机纸张的大小，一般打印机是 A4，如果是绘图仪，纸张可以更大；

② “Portrait”：纵向打印；

③ “Landscape”：横向打印。

【Scaling】分组框：打印时的缩放设置，可以选择两种模式。

① “Fit Document On Page”：图纸缩放到适合当前纸张的大小；

② “Scaled Point”：按指定的所放率打印，如果指定的所放率使得原理图不能打印在一张图纸上，系统会自动把图纸分成几张。从打印预览窗口可以看到图纸的切分情况。

【Margins】分组框：原理图边框和纸张边沿的距离。

【Colorado Set】色彩设置，选择黑白还是彩色打印。“Mono”是单色，只有一种深度的黑。而灰色则有深浅之分。

设置好以上参数后，单击 OK 按钮保存，单击 Preview 按钮，可以对打印效果进行预览。

打印普通电路图（指产品说明书、插图等，它不需图框和标题栏）时，在原理图中执行右键菜单命令【Document Options】，会弹出图 4-2 所示的对话框。将图中的“Show Reference Zone”（显示分区坐标）、“Show Border”（显示图框）、“Title Block”（显示标题栏）项全部取消。

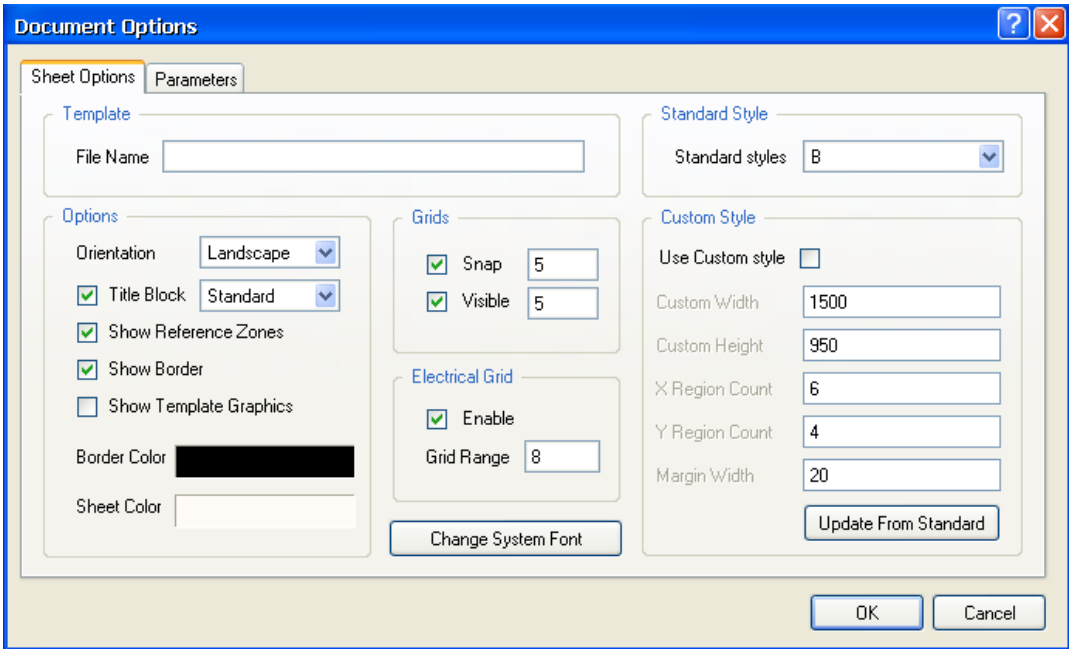


图 4-2 【Document Options】对话框

若要提高打印图纸的清晰度，可以取消栅格和底色的显示。取消底色的方法是在【Document Options】对话框中将图纸的底色设为白色，去掉栅格的方法是将对话框中【Grids】分组框中的“Visible”项取消。这时就可以打印出清晰并且带有图框、分区和标题栏的标准图纸。

4.2 自动更新功能

在原理图设计过程中或者完成后，发现某一元件的原理图库需要重新修改。那么在完成修改后，在原理图库界面中执行菜单命令【Tools】/【Update Schematics】来更新原理图。

执行操作时，要确认需要更新的原理图已经被打开。执行更新命令后，会弹出图 4-3 所示的【Design Explorer Info】窗口，显示有几个元件被更新以及更新在哪个原理图中。

图 4-3 【Design Explorer Info】窗口

4.3 PCB 图的打印

执行菜单命令【File】/【Page Setup】，弹出图 4-4 所示的对话框，与原理图中打印设置对话框不同，该图过了一个 Advanced 按钮。

设置完纸张尺寸、打印方向、缩放模式、页边距和色彩模式后，点击 Advanced 按钮，弹出高级设置对话框，设置打印 PCB 图的哪些层和哪些内容，如图 4-5 所示。

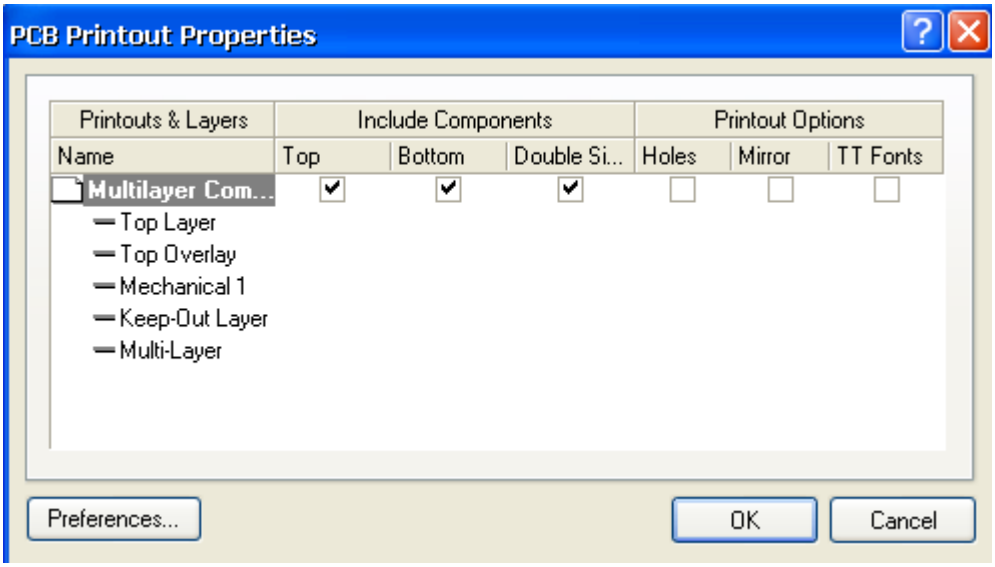


图 4-5 打印高级设置对话框

双击对话框中Multilayer Composite Print图标，弹出如图 4-6 所示的打印层面设置对话框。

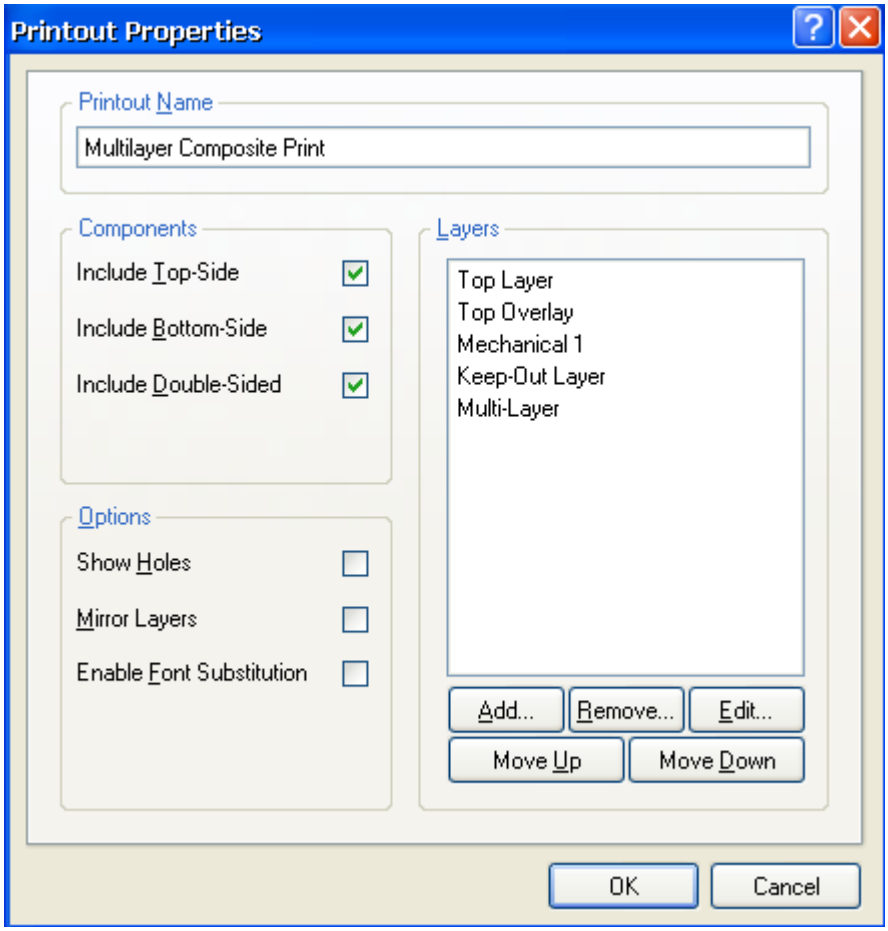


图 4-6 打印层面设置对话框

如图希望添加某一层，可以单击该对话框中的 Add... 按钮，在弹出的对话框中选择该层面，如图 4-7 所示。

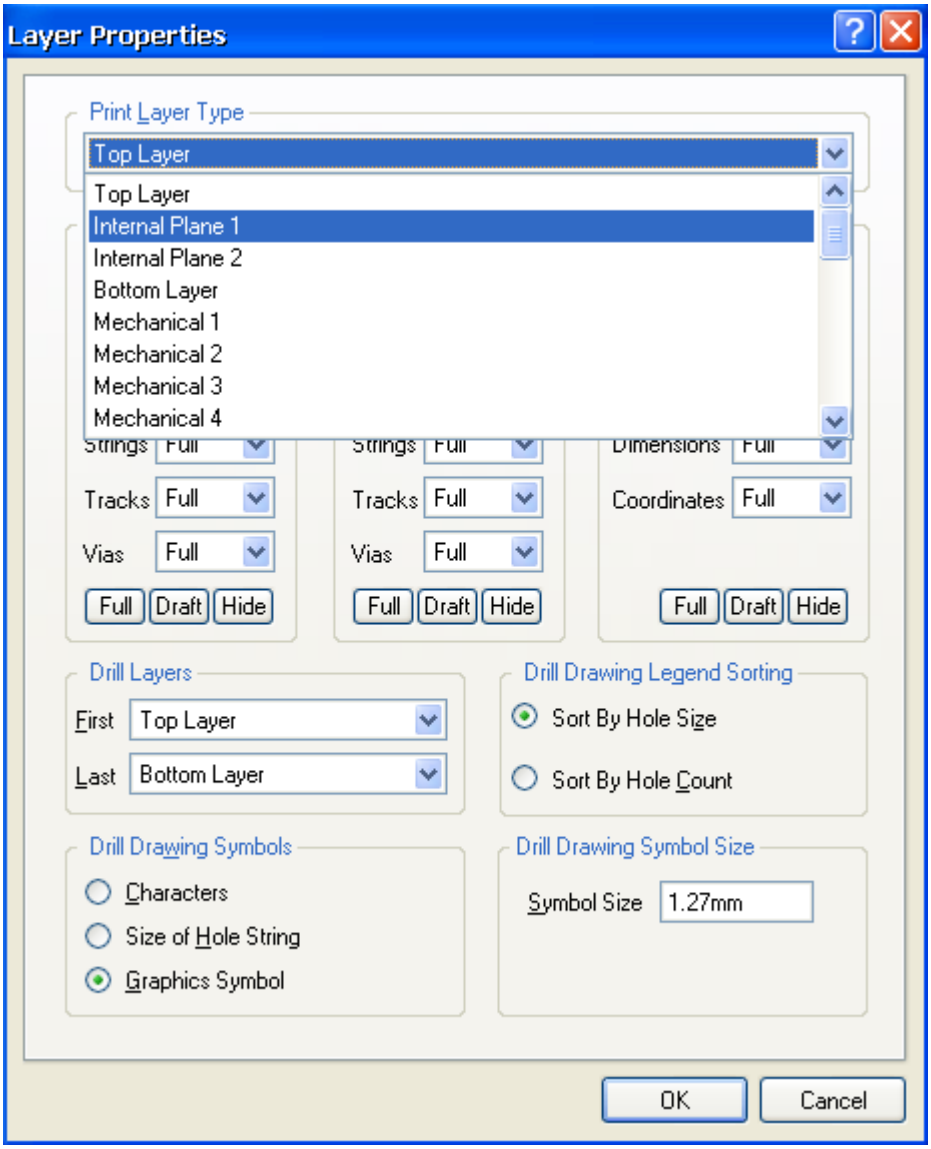


图 4-7 打印中添加【Internal Plane1】层

4.4 生成元件清单

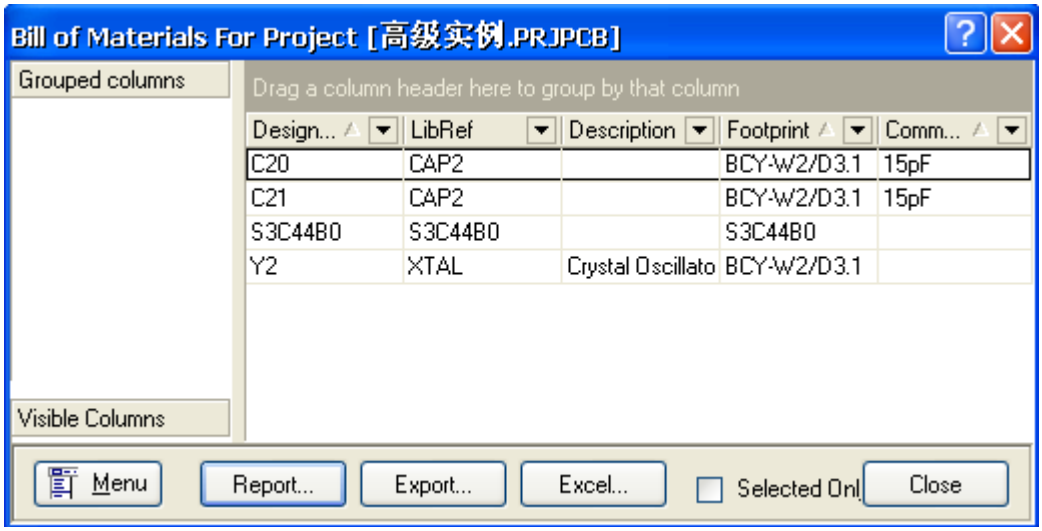
可以利用 Protel DXP 自带的生成元件清单功能来生成元件清单列表，以便查看或购买等。

执行菜单命令【Reports】/【Bill of Materials】，系统会弹出图 4-8 所示的 PCB 元件清单对话框。

对话框左侧是设置在右边区域要显示的项目，右侧是元件清单的项目和内容。在对话框中还可以设置文件输出的格式和模板等内容。

设置完成后，单击 Report... 按钮，则显示元件清单的打印预览对话框，在其中单击

Print... 按钮即可对元件清单进行打印。



4-8　PCB 元件清单对话框

第 5 章 数字电路的抗干扰方法

在电子系统设计中，为了少走弯路和节省时间，应充分考虑并满足抗干扰性的要求，避免在设计完成后再去进行抗干扰的补救措施。

5.1 形成干扰的基本要素

形成干扰的基本要素有三个：

（1）干扰源，指产生干扰的元件、设备或信号，用数学语言描述如下：du/dt， di/dt 大的地方就是干扰源。如：雷电、继电器、可控硅、电机、高频时钟等都可 能成为干扰源。

（2）传播路径，指干扰从干扰源传播到敏感器件的通路或媒介。典型的干扰传 播路径是通过导线的传导和空间的辐射。

（3）敏感器件，指容易被干扰的对象。如：A/D、D/A 变换器，单片机，数字 IC， 弱信号放大器等。

5.2 抗干扰设计的基本原则

抗干扰设计的基本方法是抑制干扰源，切断干扰传播路径，提高敏感器件的抗干扰性能（类似于传染病的预防）。

5.2.1 抑制干扰源

抑制干扰源就是尽可能的减小干扰源的 du/dt，di/dt。这是抗干扰设计中最优 先考虑和最重要的原则，常常会起到事半功倍的效果。 减小干扰源的 du/dt 主要是通过在干扰源两端并联电容来实现。减小干扰源的 di/dt 则是在干扰源回路串联电感或电阻以及增加续流二极管来实现。

抑制干扰源的常用措施如下:

（1）继电器线圈增加续流二极管，消除断开线圈时产生的反电动势干扰。仅加 续流二极管会使继电器的断开时间滞后，增加稳压二极管后继电器在单位时间内可 动作更多的次数。

（2）在继电器接点两端并接火花抑制电路（一般是 RC 串联电路，电阻一般选几 K 到几十 K，电容选 0.01uF），减小电火花影响。

（3）给电机加滤波电路，注意电容、电感引线要尽量短。

（4）电路板上每个 IC 要并接一个 0.01μF～0.1μF 高频电容，以减小 IC 对电源的 影响。注意高频电容的布线，连线应靠近电源端并尽量粗短，否则，等于增大了电 容的等

效串联电阻，会影响滤波效果。

（5）布线时避免 90 度折线，减少高频噪声发射。

（6）可控硅两端并接 RC 抑制电路，减小可控硅产生的噪声（这个噪声严重时可能 会把可控硅击穿的）。

5.2.2 切断干扰传播路径

按干扰的传播路径可分为传导干扰和辐射干扰两类。

所谓传导干扰是指通过导线传播到敏感器件的干扰。高频干扰噪声和 有用信号的频带不同，可以通过在导线上增加滤波器的方法切断高频干扰 噪声的传播，有时也可加隔离光耦来解决。电源噪声的危害最大，要特别 注意处理。所谓辐射干扰是指通过空间辐射传播到敏感器件的干扰。一般 的解决方法是增加干扰源与敏感器件的距离，用地线把它们隔离和在敏感 器件上加 蔽罩。

切断干扰传播路径的常用措施如下：

（1）充分考虑电源对单片机的影响。电源做得好，整个电路的抗干扰就 解决了一大半。许多单片机对电源噪声很敏感,要给单片机电源加滤波电路 或稳压器，以减小电源噪声对单片机的干扰。比如，可以利用磁珠和电容 组成 π 形滤波电路，当然条件要求不高时也可用 100Ω 电阻代替磁珠。

（2）如果单片机的 I/O 口用来控制电机等噪声器件，在 I/O 口与噪声源之 间应加隔离（增加 π 形滤波电路）。 控制电机等噪声器件，在 I/O 口与噪声源之 间应加隔离（增加 π 形滤波电路）。

（3）注意晶振布线。晶振与单片机引脚尽量靠近，用地线把时钟区隔离 起来，晶振外壳接地并固定。此措施可解决许多疑难问题。

（4）电路板合理分区，如强、弱信号，数字、模拟信号。尽可能把干扰源（如电机，继电器）与敏感元件（如单片机）远离。

（5）用地线把数字区与模拟区隔离，数字地与模拟地要分离，最后在一 点接于电源地。A/D、D/A 芯片布线也以此为原则，厂家分配 A/D、D/A 芯片 引脚排列时已考虑此要求。

（6）单片机和大功率器件的地线要单独接地，以减小相互干扰。 大功率 器件尽可能放在电路板边缘。

（7）在单片机 I/O 口，电源线，电路板连接线等关键地方使用抗干扰元件 如磁珠、磁环、电源滤波器，屏蔽罩，可显著提高电路的抗干扰性能。

5.2.3 提高敏感器件的抗干扰性能

提高敏感器件的抗干扰性能是指从敏感器件这边考虑尽量减少对干扰噪声 的拾取，以及从不正常状态尽快恢复的方法。

提高敏感器件抗干扰性能的常用措施如下：

（1）布线时尽量减少回路环的面积，以降低感应噪声。

（2）布线时，电源线和地线要尽量粗。除减小压降外，更重要的是降低耦 合噪声。

（3）对于单片机闲置的 I/O 口，不要悬空，要接地或接电源。其它 IC 的闲置 端在不改变系统逻辑的情况下接地或接电源。

（4）对单片机使用电源监控及看门狗电路，如：IMP809，IMP706，IMP813， X25043，X25045 等，可大幅度提高整个电路的抗干扰性能。

（5）在速度能满足要求的前提下，尽量降低单片机的晶振和选用低速数字 电路。

（6）IC 器件尽量直接焊在电路板上，少用 IC 座。

印制电路板设计原则和抗干扰措施

印制电路板(PCB)是电子产品中电路元件和器件的支撑件，它提供电路元件和器件之间的电气连接。随着电于技术的飞速发展，PGB的密度越来越高。PCB设计的好坏对抗干扰能力影响很大。因此，在进行PCB设计时．必须遵守PCB设计的一般原则，并应符合抗干扰设计的要求。

5.3 PCB 设计的一般原则

要使电子电路获得最佳性能，元器件的布且及导线的布设是很重要的。为了设计质量好、造价低的PCB，应遵循以下一般原则：

1.布局

首先，要考虑PCB尺寸大小。PCB尺寸过大时，印制线条长，阻抗增加，抗噪声能力下降，成本也增加；过小，则散热不好，且邻近线条易受干扰。在确定PCB尺寸后，再确定特殊元件的位置。最后，根据电路的功能单元，对电路的全部元器件进行布局。在确定特殊元件的位置时要遵守以下原则：

(1)尽可能缩短高频元器件之间的连线，设法减少它们的分布参数和相互间的电磁干扰。易受干扰的元器件不能相互挨得太近，输入和输出元件应尽量远离。

(2)某些元器件或导线之间可能有较高的电位差，应加大它们之间的距离，以免放电引出意外短路。带高电压的元器件应尽量布置在调试时手不易触及的地方。

(3)重量超过15g 的元器件、应当用支架加以固定，然后焊接。那些又大又重、发热量多的元器件，不宜装在印制板上，而应装在整机的机箱底板上，且应考虑散热问题。热敏元件应远离发热元件。

(4)对于电位器、可调电感线圈、可变电容器、微动开关等可调元件的布局应考虑整机的结构要求。若是机内调节，应放在印制板上方便于调节的地方；若是机外调节，其位置要与调节旋钮在机箱面板上的位置相适应。

(5)应留出印制扳定位孔及固定支架所占用的位置。

根据电路的功能单元．对电路的全部元器件进行布局时，要符合以下原则：

(1)按照电路的流程安排各个功能电路单元的位置，使布局便于信号流通，并使信号尽可能保持一致的方向。

(2)以每个功能电路的核心元件为中心，围绕它来进行布局。元器件应均匀、整齐、紧凑地排列在PCB上．尽量减少和缩短各元器件之间的引线和连接。

(3)在高频下工作的电路，要考虑元器件之间的分布参数。一般电路应尽可能使元器件

平行排列。这样，不但美观，而且装焊容易，易于批量生产。

(4)位于电路板边缘的元器件，离电路板边缘一般不小于2mm。电路板的最佳形状为矩形。长宽比为3：2 成4：3。电路板面尺寸大于200x150mm 时，应考虑电路板所受的机械强度。

2．布线

(1) 输入输出端用的导线应尽量避免相邻平行。最好加线间地线，以免发生反馈藕合。

(2) 印制摄导线的最小宽度主要由导线与绝缘基扳间的粘附强度和流过它们的电流值决定。 当铜箔厚度为0.05mm、宽度为1~15mm 时．通过2A的电流，温度不会高于3℃，因此导线宽度为1.5mm 可满足要求。对于集成电路，尤其是数字电路，通常选0.02~0.3mm导线宽度。当然，只要允许，还是尽可能用宽线．尤其是电源线和地线。导线的最小间距主要由最坏情况下的线间绝缘电阻和击穿电压决定。对于集成电路，尤其是数字电路，只要工艺允许，可使间距小至5~8mm。

(3) 印制导线拐弯处一般取圆弧形，而直角或夹角在高频电路中会影响电气性能。此外，尽量避免使用大面积铜箔，否则．长时间受热时，易发生铜箔膨胀和脱落现象。必须用大面积铜箔时，最好用栅格状.这样有利于排除铜箔与基板间粘合剂受热产生的挥发性气体。

3.焊盘

焊盘中心孔要比器件引线直径稍大一些。焊盘太大易形成虚焊。焊盘外径D 一般不小于(d+1.2)mm，其中d 为引线孔径。对高密度的数字电路，焊盘最小直径可取(d+1.0)mm。

5.4 PCB 及电路抗干扰措施

印制电路板的抗干扰设计与具体电路有着密切的关系，这里仅就PCB抗干扰设计的几项常用措施做一些说明。

1.电源线设计

根据印制线路板电流的大小，尽量加租电源线宽度，减少环路电阻。同时、使电源线、地线的走向和数据传递的方向一致，这样有助于增强抗噪声能力。

2.地段设计

地线设计的原则是：

(1)数字地与模拟地分开。若线路板上既有逻辑电路又有线性电路，应使它们尽量分开。低频电路的地应尽量采用单点并联接地，实际布线有困难时可部分串联后再并联接地。高频电路宜采用多点串联接地，地线应短而租，高频元件周围尽量用栅格状大面积地箔。

(2)接地线应尽量加粗。若接地线用很纫的线条，则接地电位随电流的变化而变化，使抗噪性能降低。因此应将接地线加粗，使它能通过三倍于印制板上的允许电流。如有可能，接地线应在2~3mm 以上。

(3)接地线构成闭环路。只由数字电路组成的印制板，其接地电路布成团环路大多能提高抗噪声能力。

3.退藕电容配置

PCB设计的常规做法之一是在印制板的各个关键部位配置适当的退藕电容。退藕电容

的一般配置原则是：

(1)电源输入端跨接10~100uf 的电解电容器。如有可能，接100uF 以上的更好。

(2)原则上每个集成电路芯片都应布置一个0.01uF 的瓷片电容，如遇印制板空隙不够，可每4~8 个芯片布置一个1~10uF 的但电容。

(3)对于抗噪能力弱、关断时电源变化大的器件，如RAM、ROM 存储器件，应在芯片的电源线和地线之间直接接入退藕电容。

(4)电容引线不能太长，尤其是高频旁路电容不能有引线。

此外，还应注意以下两点：

(1) 在印制板中有接触器、继电器、按钮等元件时．操作它们时均会产生较大火花放电，必须采用附图所示的RC 电路来吸收放电电流。一般R 取1~2K，C 取.2~47UF。

(2) CMOS 的输入阻抗很高，且易受感应，因此在使用时对不用端要接地或接正电源。

第三部分 FPGA/CPLD 技术

第 1 章 基本概念

数字通信和自动化控制等领域的高速度发展和世界观范围的高技术竞争对数字系统提出了越来越高的要求，特别是需发设计具有实时信号处理能力的专用集成电路，要求把包括多个 CPU 内核的整个电子系统综合到一个芯片（SOC）上。设计并验证这样复杂的电路及系统已不再是简单的个人劳动，而需要综合许多专家的经验和知识才能够完成。近两年 10 年来，电路制造工艺技术进步非常迅速，目前国际上 0.13μm 的制造工艺已达到工业化生产的规模，而电路设计能力却远远落后于制造技术的进步。在数字逻辑领域，迫切需要一种共同的工业标准来统一对数字逻辑电路及系统的描述，这样就能把系设计工作分解为逻辑设计（前端）和电路实现(后端)两个互相独立而又相关的部分。由于逻辑设计的相对独立性，因此，就可以把专家们设计的各种常用数字逻辑电路和系统部件(如 FFT 算法、DCT 算法部件）建成宏单元(megcell）或软核(soft-core IP)库供设计者引用，以减少重复劳动，提高工作效率。电路和实现电则可借助于 IP 和重复利用和综合工具，以及布局线工具（与具体工艺技术有关）来自动完成。

由于 Verilog HDL 和 VGDL 这 2 种工业标准的产生顺应了历史的潮流，因南昌得到了迅速的发展。因为美国、日本等国的高级设计工程师人力资源成本远高于中国，所以近年来把许多设计工作都转移到中国大陆，以降低设计成本，作为新世纪的中国大学生和年轻电子工程师，应该尽早掌握这种新的设计方法，使我国在复杂数字电路用系统的设计竞争中逐步缩小与美国等先进的工业发达国家的差距，成为我国在新世纪深亚微米千万门级的复杂系统设计的技术骨干。

1.1 Verilog HDL 的基本知识

硬件描述语言 HDL（Hardware Description language）是一种用形式化方法来描术数字电路和系统的语言。数电路系统的设计者利用这种语言可以从上层到下层（从抽象到具体）逐层描术自己的设计思想，用一系列分层次的模块来表示极其复杂的数字系统；然后利用电子设计自动化（双下简称为 EDA）工具逐层进行仿真验证；再把其中需要变为具体物理电路的模块组合经自动综合工具思换到门级电路网表；最后再用专用集成电路（ASIC）或场可编程门阵列（FPGA）自动布局布线工具，把网表转换为具体电路布线结构的实现。在制成物理器件之前，还可以用 Verliog HDL 的门极模型（元语元件或 UDP）来代替具体基本元件。因其逻辑功能和延时特性与真实的物理元件完全一致，所以在仿真工具的支持下能能验证复杂数字系统物理结构的正确性，使投片的成功率达到期 100%。目前这种称之为高层次设计（high-level-desing）的方法已被广泛采应。据统计，目前在美

国硅谷有 90%以上的 ASIC 和 FPGA 已采用硬件描述语言方法进行设计。

硬件描述语言的发殿已有 20 多年的历史，并成功地应用于设计的各个阶段；建模、仿真、验证和综合等。到达 20 世纪律性 80 年代，已出现了上百种硬件语言，对设计自动化曾起到极大的促进和推动作用；但是，这些语言一般各自面向特定的设计领域与层次，而且众多的语言使用户无所适从，因此急需一种面向设计和多领域、多层次、并得到普遍认同的标准硬件描述述语言。进入场 20 世纪律性 80 年代后期，硬件描述语言向标准化的方向发展。最终，VHDL 和 Verilog HDL 语言适应了这种趋势的要求，先后成为 IEEE 标准。把硬件描述语言用于自动综合还只有近两年 10 年的历史。最近几处来，用综合工具把可综合风格的 HDL 模块自动转换为具体电路发展非常迅速，大大地提高了复杂数字系统和设计生产率。在美国和日本等抚进电子工业国，Verilog HDL 语言已成为设计数字系统的基础。本章将通过设计实例由浅入深地帮助大家学习和掌握以下内容；

- 如何来编写各种层次可综合风格的 Verilog HDL 模块；
- 如何用可综合的 Verilog HDL 模块构成一个可靠的复杂 IP 软核和固核模块。
- 如何借助于 Verilog HDL 语言，并利用已有的虚拟模块对所设计的系统模块（由可综合和自主和商业 IP 模块组成）进行全面可靠的测试。

1.2 Verilog HDL 的历史

1、什么是 Verilog HDL

Verilog HDL 是硬件描述语言的一种，用于数字电子系统设计。它允许设计者用其来进行各种级别的逻辑设计，以及数字逻辑系统的仿真验证、时序分析和逻辑综合。Verilog HDL 是目前应用于最广泛的一种硬件描述语言。据有关文献报道，目前在美国使用 Verilog HDL 进行设计的工程师有 10 万多人，全美国有 200 多所大学教授有 Verilog 硬件描述语言的设计方法。在我国台湾地区，几乎所有著名大学的电子和计机工程系授与 Verilog 有关的课程。

2、Verilog HDL 产生和发展

Verilog HDL 是在职 1983 年由 GDA（）公司的 Phil Moorby 首的。Phil Moorby 后来成为 Verilog—XL 主要设计者和 Cadence 公司（Cadence Desin System）的第一的合伙人。1984-1985 年，Phil Moorby 设计出了第一个名为 Verilog—XL 的仿真器。1986 年，他对 Verilog HDL 了另一个巨大贡献；提出了用于快速门极仿真的 XL 算法。

Verilog—XL 算法的成功，使 Verilog HDL 语言得到迅速发展。1989 年，Cadence 公司收购了 GDA 公司，Verilog HDL 语言成为 Cadence 公司的私有财产。1990 年，Cadence 公司决定公开 Verilog HDL 语言，于是成立了 OVI（Open Verilog Interantiona）组织来负责促进 Verilog HDL 语言的发展。基于 Verilog HDL 的优越性，IEEE 于是 1995 年制定了 Verilog HDL 的 IEEE 标准，Verilog HDL 1364-1995，2001 年发布了 Verilog HDL 解情况 364-2001 标准。

3、Verilog HDL 与 VHDL 的比较

Verilog HDL 和 VHDL 都是用于逻辑设计的硬件描述语言，并且都已成为 IEEE 标准，VHDL 是在 1987 年成为 IEEE 标准的，Verilog HDL 则在 1995 年才正式成为 IEEE 标准。之所以VHDL比Verilog HDL早面为IEEE标准,是因为VHDL是美国军方组织开发的,而Verilog HDL 则是从一个普通的民间公司的私有财产转化而来，基于 Verilog HDL 的优越性才成为 IEEE 标准。因而有更强的一命力。

VHDL 其英文全名为，而 VHDL 慢是 Very 的缩写词，意为甚高速集成电路，故 VHDL 其准确的中文译旬为甚高速集成电路的硬件描述语言。

Verilog HDL 和 VHDL 作为描述硬件电路设计的语言，其共同的特点在于；能形式化地抽象表示电路的行为和结构，支持逻辑设计中层次与范围的描述，可借用高级语言的精巧结构来简化电路行为描述，具有电路仿真与验证机制以保证设计的正确性，支持电路描述由高层到低层的综合转换，硬件描述与实现工艺无关（有关工艺参数可通过语言提供的属性包括进去），便于文档管理，易于理解和设计重用等。

Verilog HDL 和 VHDL 又各有其特点。由于 Verilog HDL 早在 1983 年就已推出，至今已有近 20 年的应用历史，因而 Verilog HDL 拥有更广泛的设计群体，成熟的资源也远比 VHDL 丰富。与 VHDL 相比，Verilog HDL 的最大优点是；它是一种非常容易掌握的硬件描述语言，只要有 C 语言的编程基础，通过 20 学时的学习，再加上一时实际操作，一般右在 2-3 个月内掌握这种设计技术，而掌握 VHDL 技术就比较困难。这是因为 VHDL 不很直观，需要有 Ade 编程基础，一般需要至少半年以上的专业培训，才能掌握 VHDL 的基础设计技术。目前版本的 Verilog HDL 和 VHDL 在行为级抽象建模的覆盖范围方面也有所不同。Verilog HDL 在系统抽象方面比 VHDL 略差一些，而在门级开关电路描述方面比 VHDL 强的多。

这 2 种语言在不断地完善。由于新公布的 IEEE Verilog HDL 2001 标准使 Verilog HDL 在系统极和可综合性能方面都有大幅度的提高，因此，Verilog HDL 作为学习 HDL 设计方法的入门和基础是比较合适的,。学习掌握 Verilog HDL 建模、仿真和综合技术不仅可以使大家对数字电路设计有更进一步的了解，而且可以为以后学习高级的行为综合和物理综合打下坚实的基础。

4、Verilog HDL 目前的应用情况和适用的设计

近 10 年以来，EDA 界一直对在数字逻辑设计究竟采用哪一种硬件描述语言争论不休，目前的情况是两者各有千秋，在美国，高层次数字系统领域 Verilog HDL 和 VHDL 的应用比率是 80%和 20%；日本和我国台湾省与美国相同，而在欧洲 VHDL 的发展比较好。在中国大陆，由于 Verilog HDL 和 VHDL 的使用才开始 2-3 年，应用比率还没有统计。根据作者了解，国内大多数集成电路设计公司都采用 Verilog HDL。Verilog HDL 专门为复杂数字系统的设计仿真而开发的，本身就非常适合复杂数字逻辑真路用系统的仿真和综合。由于Verilog HDL 在其门极描述的底层，也就是在晶体管开关的描述方面比 VHDL 有强得多的功能，所以使在 VHDL 的设计环境，在底层实质上也是库存 Verilog HDL 描述的器件库所支持的。1998 年通过 Verilog HDL 新标准，把 Verilog HDL-A 并入 Verilog HDL 的新标准，使其不仅支持数字逻辑电路的描述，还支持模拟电路的描述；因些在混合信号的电路系统的设计中它会有更广泛的应用。在深亚微米 ASIC 和向密度 FPGA 已成为电子设计主流的今天，Verilog HDL 的发展前景地非常远大的。2001 年 3 月，Verilog 有（EE1364-2001 标

准的公布，使得 Verilog HDL 语言在综合和仿真性能方面都有很大的提高，更加证明了 Verilog HDL 的发展前景。作者本人的意见是；若要推广采用硬件描述的设计的设计方法，则首先从推广 Verilog HDL 开始，然后再推广 VHDL。

Verilog HDL 较为适全系统极（）、算法极、寄存器传输极（RTL）逻辑（）、门极和电路开发极设计，而对特大性（千万门极以上）的系统极设计，则 VHDL 更为适合。由于两种 HDL 语言还在不断地发展过程中，它们都会逐步地完善自己。

5、Verilog HDL 设计复数字电路的优点

(1)传统设计方法——电路原理图输入法

几十年前，当时所做的复杂数字逻辑电路及系统的设计规模比较小，也比较简单，其中所用到的 FGAT 或 ASIC 设计工作往往只能采用厂家提供的专用电路图输入工具来进行。为了满足设计性能指标，工程师往往需要花好几天或更长的时间进行艰苦的手工布线。工程师还得非常熟悉所选器件的内部结构和外部引线特点，才能达到设计要求。这种低水平的设计方法大大延长的设计周期

近年来。FPGAT 和 ASIC 的设计在规模和复杂度方面不断取得进展，而对逻辑电路及系统设计时间要求却越来越短。这些因素促进使设计人员采用高水平准的设计工具，如硬件描述语言（Verilog HDL 和 VHDL）来进行设计。

（2）Verilog HDL 设计法与传统的电路原理图输入法的比较

采用电路原理图输入法进行设计，具有周期长、需要专门的设计工具以及需手工布线等缺陷。采用 Verilog HDL 输入法时，由于 Verilog HDL 的标准化，可以很容易地把完成的设计移植到不同厂家的不同芯片中去，并在不同规模应用时可以较容易地进行修改。这不仅是因为用 Verilog HDL 所完成的设计，其信号位数是很容易改变的，可以很容易地对它进行修改来适应不同规模的应用；在仿真验证时，仿真测试矢量还可以用同一种描述语言来完成；而且还因为采用 Verilog HDL 综合器生成的数字逻辑是一种标准的电子设计互换格式（EDIF）文件，它独立于所采用的实现工艺。有关工艺参数的描述可以通过 Verilog HDL 提供的属性包括进去，然后利用不同厂家的布局线工具，在不同工艺的芯片上实现（绝大多数 FPGA 厂商也都支持 Verilog HDL 设方法）。

采用 Verilog HDL 输入法最大的优点是与工艺无关。这使得工程师在功能设计、逻辑验证阶段，可以不必过多考虑门级及工艺实现的具体细节，只需要利用系统设计时对芯片的要求，施加不同的约束条件，即可设计出实际电路。实际上，这是利用了计算机的巨大能力在 EDA 工具的帮助下，把逻辑验证与具 工艺库匹配、布线及时延计算分成不同的阶段来实现。从而减轻了人们的繁琐劳动。

（3）Verilog HDL 的标准化和软核的重用

Verilog HDL 是在 1983 年由 GATEWAY 公司首先开发成功的，经地诸多改进，于 1995 年 11 月正式被批准为 Verilog IEEE 1364—1995 标准。2001 年 3 月在原标准的基础上经过改进和补充又推出 Verilog IEEE 1364—2001 新标准。

Verilog HDL 的标准化大大加快了 Verilog HDL 的推广和发展。由于 Verilog HDL 的设计方法与工艺无关，因而大大提高了 Verilog HDL 模型的可重用性。我们把功能经过验证的、综合的、实现后电路结构总门数在 500 门以上的 Verilog HDL 模型称之为“软核”（softcore），而把软核构成的器件称为虚拟器件。在新电路的研制过程中，软核和虚拟

器件可以很容易地借助 EDA 综合工具与其他外部逻辑结合为一体。这样，软核和虚拟器件的重用性就可以大大缩短设计周期，加快了复杂电路的设计。目前，国际上有一个叫作虚拟接口联盟的组织 VSIA（Virtual Socket Interface Alliance）来协调这方面的工作。

（4）软核、固核、硬核的概念及它们的重用

前面已介绍了软核的 Verilog HDL，下面再介绍一下固核（firm core）和硬核（hard core）的概念。把在某一种现场可编程门阵列（FPGA）器件上实现的、验证是正确的、总门数在 5000 门以上的电路结构编码文件，称之为固核。把在某一种专用集成电路工艺的（ASIC）器件上实现的、验证是正确的、总门数在 5000 门以上的电路结构版图掩膜，称这为硬核。

显而易见，在具体实现手段和工艺技术尚未确定的逻辑设计阶段，软核具有最大的灵活性。它可以很容易地借助 EDA 综合工具与其他外部逻辑结合为一体。当然，由于实现技术的不确定性，有可能要做一些改动以适应相应的工艺。相比之下，固核和硬核与其他外部逻辑结合为一体的灵活性要差的很，特别是电路实现工艺技术改变时更是如此。近年来，电路实现工艺技术的发展是相当迅速的，为了逻辑电路设计成果的积累，更快、更好地设计更大规模的电路，发展软核的设计和推广软核的重用技术是非常必要的。新一代的数字逻辑电路设计师必须掌握这方面的知识和技术。

6、采用硬件描述语言（Verilog HDL）的设计流程

（1） 自顶向下（top—down）设计的基本概念

现代集成电路制造工艺技术的改进，使得在一个芯片上集成数十万乃至数千万个器件成为可能。很难设想仅由一个设计师独立设计如此大规模的电路而不出现错误。利用层次化、结构化的设计方法，一个完整的硬件设计任务首先总设计师划分为若干个可操作的模块，编制出相应的模型（行为的或结构的）；通过仿真加以验证后，再把这些模块分配给下一层的设计师。这就允许多个设计者同时设计一个硬件系统中的不同模块，其中每个设计者负责自己所承担的部分，而由上一层的设计师对其下层的设计者完成的设计用行为级上层模块对其所做的设计进行验证。为了提高设计质量，其中有一部分模块可由商业渠道得到，可以购买它们的知识产权的使用权（IP 核的重用），以节省时间和开发经费。

自顶向下的设计是从系统级开始，把系统划分为基本单元，然后再把每个基本单元划分为下一层次的基本单元，一直这样做下去，直到可以直接用 EDA 技术元件库中的基本元件来实现为止。

对于设计开发整机电子产品的单位和个人来说，新产品的开发总是从系统设计入手，先进行方案的总体论证、功能描述、任务和指标的分配。随着系统变得复杂和庞大，特别需要在样机问世之前，对产品的全貌有一定的预见性。目前，EDA 技术的发展使得设计师有可通用实现真正的自顶向下的设计。

（2） 层次管理的基本概念

复杂数字逻辑电路和系统的层次化、结构化设计隐含着对系统硬件设计方案的逐次分解。在设计过程中的任意层次，至少得有一种形式来描述硬件。硬件的技术（特别是行为描述）通常称为建模。在集成电路设计的每一层次，硬件可以分为一些模块。该层次的硬件结构由这些模块的互连描述，该层次的硬件的行为由这些模块的行为描述。这些模块称为该层次的基本单元，而该层次的基本单元又由下一层次的基本单元互连而成。如此下去，

完整的硬件设计就可以由图所示的设计的树描述，树枝对应着基本单元的结构分解。在不同层次者可以进行仿相真以对设计思想进行验证。EDA 工具提供了有效的手段来管理错综复杂的层次，即可以很方便地查看某一层次某一模块的源代码或电路图，以改正仿真时发现的错误。

（3） 具体模块的设计编译和仿真的过程

在不同的层次做具体模块的设计所有的方法也有所不同，在高层次上往往编写一些行为级的模块通过仿真加以验证，其主要目的是系统性能的总体考虑和各模块的指标分配，并非具电路的实现，因而综合及其以后的步骤往往不需要进行；而当设计的层次比较接近底层时，行这描述往往需要用电路逻辑来实现。这时的模块不仅需要通过仿真加以验证，而且还需要进行综合、优化、布线和后仿真。总之，具体电路是从底向上逐步实现的。EDA 工具往往不仅支持 HDL 描述，也支持电路图输入。有效地处用这 2 种方是提高效率的办法之一。

- 设计开发；即从编写设计文件→综合到布局布线→投片生成等一系列步骤。
- 设计验证；即进行各种仿真的一系列步骤。如果在仿真过程中发现问题，就返回设计输入进行修改。

（4） 对应具体工艺器件的优化、映象和布局线

由于各种 ASIC 和 FPFA 器件的工艺各不相同，因而当用不同厂家的不同器件来实现已验证的逻辑网表（EDIF 文件）时，就需要不同的基本单元库与布线延迟模型与之对应，才能进行准确的优化、映象和布局布线。基本单元库延迟模型由熟悉本厂工艺的工程师提供，再由 EDA 厂商的工程师编入相应的处理程序，而逻辑电路设计师只需要用一个文件说明所用的工艺器件和约束条件，EDA 工具就会自动地根据这一文件选择相应的库和模型并进行准确的处理，从而大提高设计效率。

1.3 总结

采用 Verilog HD 设计方法比采用电路图输入的方法更有优越性。这就是为什么美国等先进工业国家在进入 20 世纪 90 年代以后纷纷采用 HDL 设计方法的原因。在 2 种符合 IEEE 标准的硬件描述语言中，Verilog HDL 与 HDL 相比更加基础更易学习。掌握 HDL 设计方法应从学习 Verilog HDL 设计方法开始。Verilog HDL 可用一复杂数字逻辑电路和系统的总体仿真、子系统仿真和具体电路综合等各个设计阶段。

自顶向下的设计方法是首先从系统设计入手，从顶层进行功能划分和结构设计。系统的总体仿真是顶层进行功能划分的重要环节，这时的设计是与工艺无关的。由于设计的主要仿真和调试过程是在高层次完成的，所以能够早期妇现结构设计上的错误，避免设计工作的浪费，同时也减少逻辑仿真的工作量，自顶向下的设计方法方便了从系统级划分和管理整个项目，使得几十万门甚至几千万门规模的复杂数字电路的设计成为可能，并可减少设计人员，避免不必要的重复设计，提高了设计的一次成功率。

从底向上的设计在某种意义上讲可以看作上述自顶处下设计的过程。虽然设计也是从系统能开始，即从设计树的树根开始对设计进行逐次划分，但划分时首先要考虑单元是否存在，即设计划分过程必须存在的基本单元出发，设计树最末枝上的单元要么是已经制造

出的单元，要么是其他项目自己开发好的单元，或者是可外购得到的单元。

自顶向下的设计过程中在每一层次划分时都要对某些目标作优化，自顶向下的设计过程是理想的设计过程，它的缺点是得到的最小单元不标准，制造成本可能很高，从底向上的设计过程全采用标准基本单元，通常比较经济，但有时可能不能满足一些特定的指标要求。复杂数字逻辑电路和系统的设计过程通常是这 2 种设计方法的结合，设计时需要考虑多个目标的综合平衡。

Verilog HDL 是一种用于数字系统设计的语言。用 Verilog HDL 描述的电路设计就是该电路的 Verilog HDL 模型。也称为模块。Verilog HDL 既一种行为描述的语言，也是一种结构描述的语言。也就是说，无论描述电路功能行为的模块或描述元器件或较大部件互连的模块，都可以用 Verilog HDL 来建立电路模型。如果按照一定的规矩编写，功能行为模块可以通过工具自动地转换为门级互连模块。Verilog HDL 的模型可以是实际电路的不同级别的抽象。这些抽象的级别和它们对应的模型类型共有以下 5 种：

- 系统级（system）：用语言提供的高级结构实际设计模块外部性能的模型。
- 算法级（algorithm）用语言提供的高级结构实际算法运行的模型。
- RTL 级（Register Transfer Level）：描述数据在寄存器之间流动和如何处理及控制这些数据流动的模型（以上 3 种都属于行为描述，只有 RTL 级才与逻辑电路有明确的对应关系）。
- 门级（gate-level）描述逻辑门之间连接的模型（与逻辑电路有确定的连接关系，以上 4 种数字系统设计工程师必须掌握）。
- 开关级（switch-level）：描述器件中三极管和储存节点以及它们之间连接的模型（与具体的物理电路有对应关系，工艺库元件和宏部件设计人员必须掌握，本章中将不予以介绍）。一个复杂电路系统的完整 Verilog HDL 模型是由若干个 Verilog HDL 模块构成的，每一个模块又可以由若干个子模块构成。其中有一些模块需要综合成具体电路，而有些模块只是与用户所设计的模块有交互联系的现存电路或激励信号源。利用 Verilog HDL 语言结构所提供的这种功能就可以构造一个模块间的清晰层次结构来描述极其复杂的大型设计，并对所做设计的逻辑电路进行严格的验证。

Verilog HDL 行为描述语言作为一种结构化和过程性的语言，其语法结构非常适合于算法级和 RTL 级的模型设计。这种行为描述语言具有以下功能：

- 可描述序顺执行表达式来明确地控制过程的启动时间。
- 通过命名的事件来触发其他过程时的激活行为或停止行为。
- 提供了条件语句如（如 if-else、case）和循环程序结构。
- 提供了可带参数且非零延续时间的任务（task）程序结构。
- 提供了可定义新的操作符的函数结构（function）。
- 提供了用于建立表达式的算术运算符、逻辑运算符和位运算符。
- Verilog HDL 语言作为一种结构化的语言，也非常适合于门级和开关级的模型设计。因其结构化的特点，又使它具有以下功能：

- 提供一套完整的表示组合逻辑基本元件的原语（primitive）；
- 建立了双向通路（总线）和电阻器件的原语；

- 可建产 MOS 器件的电荷分享和电荷衰减动态模型。

因为在 Verilog HDL 中，提供了延迟和输出强度的原语来建立精确程度很高的信号模型，所以 Verilog HDL 的构造性语句可以精确地建立信号的模型。信号值可以有不同的强度，可以通过设定宽范围的模糊值来降低不确定条件的影响。

Verilog HDL 作为一种高级的硬件描述编程语言，它与 C 语言的风格有许多类似之处，其中有许多语句如：if 语句、case 语句等与 C 语言中的对自学成才语句十分相似。如果读者已经掌握 C 语言编程的基础，那么学习 Verilog HDL 并不难。只要对 Verilog HDL 某些语句的特殊方面着重理解，并加强上机练习就能很好地掌握它，就能利用它的强大功能来设计复杂的数字逻辑电路系统。下面将对 Verilog HDL 中的基本语法通过实例逐一加以介绍。

第 2 章 HDL 指南

2.1 模块

模块是Verilog 的基本描述单位，用于描述某个设计的功能或结构及其与其他模块通信的外部端口。一个设计的结构可使用开关级原语、门级原语和用户定义的原语方式描述；设计的数据流行为使用连续赋值语句进行描述；时序行为使用过程结构描述。一个模块可以在另一个模块中使用。

一个模块的基本语法如下：

```
Module module_name(port_list) ;
Declarations:
reg, wire, parameter,
input, output, inout,
function, task, . . .
Statements:
Initial statement
Always statement
Module instantiation
Gate instantiation
UDP instantiation
Continuous assignment
endmodule
```

说明部分用于定义不同的项，例如模块描述中使用的寄存器和参数。语句定义设计的功能和结构。说明部分和语句可以散布在模块中的任何地方；但是变量、寄存器、线网和参数等的说明部分必须在使用前出现。为了使模块描述清晰和具有良好的可读性，最好将所有的说明部分放在语句前。本书中的所有实例都遵守这一规范。图2-1为建模一个半加器电路的模块的简单实例。

```
module HalfAdder(A, B, Sum, Carry) ;
input A, B;
output Sum, Carry;
assign #2 Sum = A ^ B;
assign #5 Carry = A & B;
endmodule
```

模块的名字是*HalfAdder*。模块有4个端口：两个输入端口*A*和*B*，两个输出端口*Sum*和*Carry*。由于没有定义端口的位数，所有端口大小都为1位；同时，由于没有各端口的数据类型说明，这四个端口都是线网数据类型。

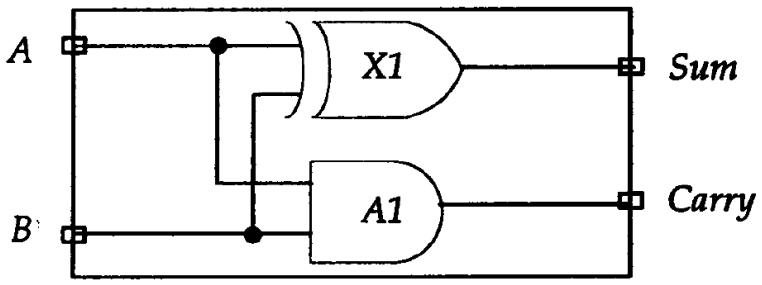


图2-1 半加器电路

模块包含两条描述半加器数据流行为的连续赋值语句。从这种意义上讲，这些语句在模块中出现的顺序无关紧要，这些语句是并发的。每条语句的执行顺序依赖于发生在变量A和B上的事件。在模块中，可用下述方式描述一个设计：

1) 数据流方式;

2) 行为方式;

3) 结构方式;

4) 上述描述方式的混合。

下面几节通过实例讲述这些设计描述方式。不过有必要首先对Verilog HDL的时延作简要介绍。

2.2 时延

Verilog HDL模型中的所有时延都根据时间单位定义。下面是带时延的连续赋值语句实**assign** #2 *Sum* = A ^ B;# 2指2个时间单位。使用编译指令将时间单位与物理时间相关联。这样的编译器指令需在模块描述前定义，如下所示:

` **timescale** 1ns /100ps

此语句说明时延时间单位为1 n s并且时间精度为100ps (时间精度是指所有的时延必须被限定在0 . 1 n s内)。如果此编译器指令所在的模块包含上面的连续赋值语句, #2 代表2 n s。如果没有这样的编译器指令, Verilog HDL 模拟器会指定一个缺省时间单位。IEEE Ve r i l o gHDL 标准中没有规定缺省时间单位。

2.3 数据流描述方式

用数据流描述方式对一个设计建模的最基本的机制就是使用连续赋值语句。在连续赋值语句中，某个值被指派给线网变量。连续赋值语句的语法为:

assign [delay] LHS_net = RHS_ expression;

右边表达式使用的操作数无论何时发生变化, 右边表达式都重新计算, 并且在指定的时延后变化值被赋予左边表达式的线网变量。时延定义了右边表达式操作数变化与赋值给左边表达式之间的持续时间。如果没有定义时延值, 缺省时延为0。

图2 - 2显示了使用数据流描述方式对2 - 4解码器电路的建模的实例模型。

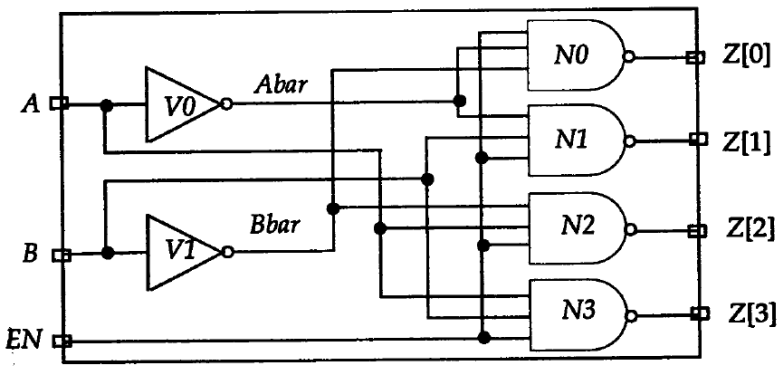


图2-2　2-4解码器电路

```
`timescale 1ns/1ns
module Decoder 2x4 (A, B, EN, Z) ;
input A, B, EN;
output [ 0 :3]Z;
wire Abar, Bbar;
assign #1 Abar = ~ A; / / 语句1。
assign #1 Bbar = ~ B; / / 语句2。
assign #2 Z[0] = ~ (Abar & Bbar & EN ) ;   // 语句3。
assign #2 Z[1] = ~ (Abar & B & EN) ;       // 语句4。
assign #2 Z[2] = ~ (A & Bbar & EN) ;       // 语句5。
assign #2 Z[3] = ~ ( A & B & EN) ;         // 语句6。
Endmodule
```

以反引号“`”开始的第一条语句是编译器指令，编译器指令`timescale 将模块中所有时延的单位设置为1ns，时间精度为1ns。例如，在连续赋值语句中时延值#1和#2分别对应时延1ns和2ns。模块Decoder2x4有3个输入端口和1个4位输出端口。线网类型说明了两个连线型变量Abar和Bbar(连线类型是线网类型的一种)。此外，模块包含6个连续赋值语句。参见图2 - 3中的波形图。当E N在第5ns变化时,语句3、4、5和6执行。这是因为EN是这些连续赋值语句中右边表达式的操作数。Z[0]在第7 ns时被赋予新值0。当A在第15ns变化时,语句1、5和6执行。执行语句5和6不影响Z[0]和Z[1]的取值。执行语句5导致Z[2]值在第17 ns变为0。执行语句1导致A b a r在第16 ns被重新赋值。由于A b a r的改变，反过来又导致Z[0]值在第18ns变为1。请注意连续赋值语句是如何对电路的数据流行为建模的；这种建模方式是隐式而非显式的建模方式。此外,连续赋值语句是并发执行的，也就是说各语句的执行顺序与其在描述中出现的顺序无关。

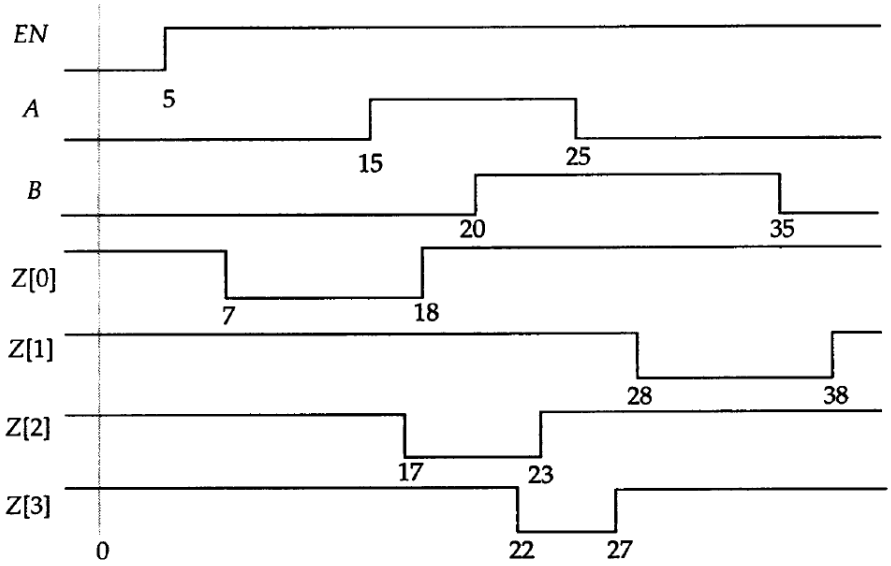


图2-3 连续赋值语句实例

2.4 行为描述方式

设计的行为功能使用下述过程语句结构描述：

1) initial语句：此语句只执行一次。

2) always语句：此语句总是循环执行，或者说此语句重复执行。

只有寄存器类型数据能够在这两种语句中被赋值。寄存器类型数据在被赋新值前保持原有值不变。所有的初始化语句和always语句在0时刻并发执行。下例为always语句对1位全加器电路建模的示例，如图（2-4）。

```
Module FA_Seq (A, B, Cin, Sum, Cout ) ;
Input A, B, Cin ;
output Sum, Cout;
reg Sum, Cout;
reg T1, T2, T3;
always
@(AorBorCin) begin
Sum = (A ^ B) ^ Cin ;
T1 = A & Cin;
T2 = B & Cin;
T3 = A & B;
Cout = (T 1| T 2) | T 3;
end
endmodule
```

模块FA_Seq有三个输入和两个输出。由于Sum、Cout、T1、T2和T3在always 语句中被

赋值,它们被说明为reg 类型(reg 是寄存器数据类型的一种)。always 语句中有一个与事件控制(紧跟在字符@ 后面的表达式)。相关联的顺序过程(begin-end对)。这意味着只要A、B或Cin上发生事件，即A、B或Cin之一的值发生变化，顺序过程就执行。在顺序过程中的语句顺序执行，并且在顺序过程执行结束后被挂起。顺序过程执行完成后，always语句再次等待A、B或Cin上发生的事件。

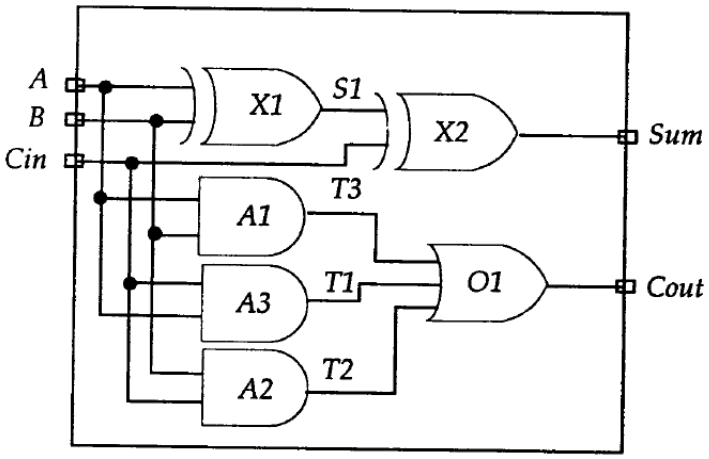


图2-4 1位全加器电路

在顺序过程中出现的语句是过程赋值模块化的实例。模块化过程赋值在下一条语句执行前完成执行。过程赋值可以有一个可选的时延。时延可以细分为两种类型:

1) 语句间时延: 这是时延语句执行的时延。

2) 语句内时延: 这是右边表达式数值计算与左边表达式赋值间的时延。

下面是语句间时延的示例：

```
Sum = (A^B)^Cin;
#4 T1 = A&Cin;
```

在第二条语句中的时延规定赋值延迟4个时间单位执行。就是说，在第一条语句执行后等待4个时间单位，然后执行第二条语句。下面是语句内时延的示例。

```
Sum = #3 (A^B)^Cin;
```

这个赋值中的时延意味着首先计算右边表达式的值, 等待3个时间单位，然后赋值给Sum。如果在过程赋值中未定义时延，缺省值为0时延，也就是说，赋值立即发生。这种形式以及在always 语句中指定语句的其他形式将在第8章中详细讨论。

下面是initial语句的示例：

```
`timescale 1ns/1ns
module Test (Pop, Pid ) ;
output Pop, Pid;
reg Pop, Pid;
initial
begin
Pop = 0;     // 语句1。
```

```
    Pid = 0;     // 语句2。
    Pop = #5 1; // 语句3。
    Pid = #3 1; // 语句4。
    Pop = #6 0; // 语句5。
    Pid = #2 0; // 语句6。
    end
    endmodule
```

这一模块产生如图2-5 所示的波形。initial语句包含一个顺序过程。这一顺序过程在0ns时开始执行，并且在顺序过程中所有语句全部执行完毕后，initial语句永远挂起。这一顺序过程包含带有定义语句内时延的分组过程赋值的实例。语句1和2在0ns时执行。第三条语句也在0时刻执行,导致Pop 在第5ns时被赋值。语句4在第5ns 执行,并且Pid 在第8ns被赋值。同样,Pop在14ns被赋值0,Pid在第16ns被赋值0。第6条语句执行后，initial语句永远被挂起。在后面的章节将更详细地讲解initial语句。

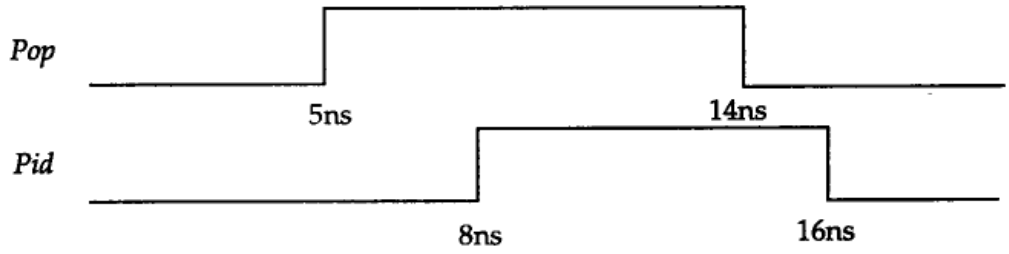


图2-5 *Test* 模块的输出波形

2.5 结构化描述形式

在Verilog HDL中可使用如下方式描述结构:

1) 内置门原语(在门级);

2) 开关级原语(在晶体管级);

3) 用户定义的原语(在门级);

4) 模块实例(创建层次结构)。

通过使用线网来相互连接。下面的结构描述形式使用内置门原语描述的全加器电路实例。该实例基于图2 - 4所示的逻辑图。

```
module FA_Str(A, B, Cin, Sum, Cout ) ;
input A, B, Cin ;
output Sum, Cout;
wire S1, T1, T2, T3;
xor
X1 (S1, A, B) ,
X2 (Sum, S1, Cin) ;
and
```

```
A1 (T3, A, B ) ,
A2 (T2, B, Cin) ,
A3 (T1, A, Cin) ,
or
O1 (Cout, T1, T2, T3 ) ;
endmodule
```

在这一实例中，模块包含门的实例语句，也就是说包含内置门xor、and和or的实例语句。门实例由线网类型变量S1、T1、T2和T3互连。由于没有指定的顺序，门实例语句可以以任何顺序出现；图中显示了纯结构； xor、and和or是内置门原语；X1、X2、A1等是实例名称。紧跟在每个门后的信号列表是它的互连；列表中的第一个是门输出，余下的是输入。例如， S1与xor门实例X 1的输出连接，而A和B与实例X 1的输入连接。4位全加器可以使用4个1位全加器模块描述，描述的逻辑图如图2-6所示。下面是4位全加器的结构描述形式。

```
module FourBitFA (FA, FB, FCin, FSum, FCout ) ;
parameter SIZE = 4;
input [SIZE:1] FA, FB;
output [SIZE:1] FSum
input FCin;
input FCout;
wire [1:SIZE－1] FTemp;
FA_Str
FA1(.A(FA[1]), .B(FB[1]), .Cin(FCin) ,
.Sum(FSum[1]), .Cout(FTemp[2])) ,
FA2(.A(FA[2]), .B(FB[2]), .Cin(FTemp[1]) ,
.Sum(FSum[2]), .Cout(FTemp[2])),
FA3(FA[3],FB[3], FTemp[2], FSum[3],FTemp[3] ,
FA4(FA[4],FB[4], FTemp[3],FSum[4],FCout) ;
Endmodule
```

在这一实例中，模块实例用于建模4位全加器。在模块实例语句中，端口可以与名称或位置关联。前两个实例FA1和FA2使用命名关联方式，也就是说，端口的名称和它连接的线网被显式描述（每一个的形式都为“ .port_name(net_name)）。最后两个实例语句，实例FA3和FA4使用位置关联方式将端口与线网关联。这里关联的顺序很重要，例如，在实例FA4中，第一个FA[4]与FA_Str 的端口A连接，第二个FB[4]与FA_Str的端口B连接，余下的由此类推。

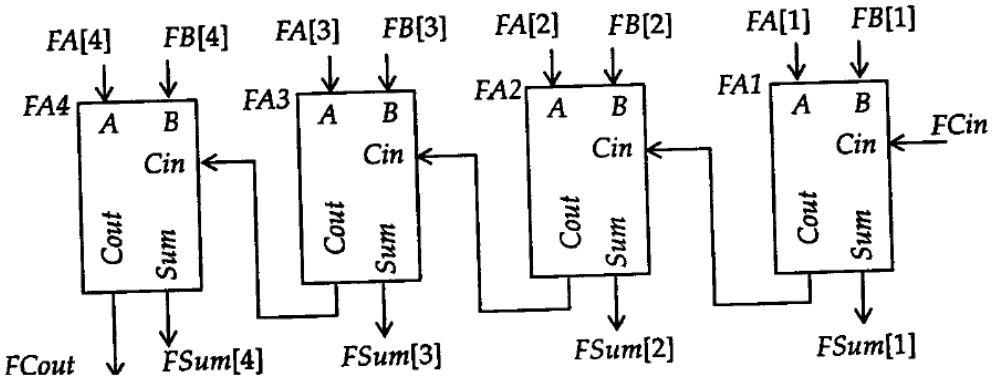


图2-6 4位全加器

2.6 混合设计描述方式

在模块中，结构的和行为的结构可以自由混合。也就是说，模块描述中可以包含实例化的门、模块实例化语句、连续赋值语句以及always语句和initial语句的混合。它们之间可以相互包含。来自always语句和initial语句（切记只有寄存器类型数据可以在这两种语句中赋值）的值能够驱动门或开关，而来自于门或连续赋值语句（只能驱动线网）的值能够反过来用于触发always语句和initial语句。下面是混合设计方式的1位全加器实例。

```
module FA_Mix (A, B, Cin, Sum, Cout ) ;
input A,B, Cin;
output Sum, Cout;
reg Cout;
reg T1, T2, T3;
wire S1;
xor X1(S1, A, B); // 门实例语句。
always
@(A or B or Cin ) begin    // always 语句。
T1 = A & Cin;
T2 = B & Cin;
T3 = A & B;
Cout = (T1 | T2) | T3;
end
assign Sum = S1^Cin; // 连续赋值语句。
endmodule
```

只要A或B上有事件发生，门实例语句即被执行。只要A、B或C i n上有事件发生，就执行always 语句，并且只要S1或Cin上有事件发生，就执行连续赋值语句。

2.7 设计模拟

Verilog HDL不仅提供描述设计的能力，而且提供对激励、控制、存储响应和设计验证的建模能力。激励和控制可用初始化语句产生。验证运行过程中的响应可以作为“变化时保存”或作为选通的数据存储。最后，设计验证可以通过在初始化语句中写入相应的语句自动与期望的响应值比较完成。下面是测试模块Top的例子。该例子测试2．3节中讲到的FA_Seq模块。

```
‘timescale 1ns/1ns
Module Top;                    // 一个模块可以有一个空的端口列表。
reg PA, PB, PCi;
wire PCo, PSum;
// 正在测试的实例化模块：
FA_Seq F1(PA, PB, PCi, PSum, PCo ); // 定位。
initial
begin: ONLY_ONCE
reg [3:0] Pal;
//需要4位, Pal才能取值8。
for (Pal = 0; Pal < 8; Pal = Pal + 1)
begin
{PA, PB, PCi} = Pal;
#5 $display (“PA, PB, PCi = %b%b%b ”, PA, PB, PCi,
“:::PCo,PSum=%b%b ”, PCo, PSum) ;
end
end
endmodule
```

在测试模块描述中使用位置关联方式将模块实例语句中的信号与模块中的端口相连接。也就是说，PA连接到模块FA_Seq的端口A，PB连接到模块FA_Seq的端口B，依此类推。注意初始化语句中使用了一个for循环语句，在PA、PB和PCi上产生波形。for 循环中的第一条赋值语句用于表示合并的目标。自右向左，右端各相应的位赋给左端的参数。初始化语句还包含有一个预先定义好的系统任务。系统任务$display将输入以特定的格式打印输出。系统任务$display调用中的时延控制规定$display任务在5个时间单位后执行。这5个时间单位基本上代表了逻辑处理时间。即是输入向量的加载至观察到模块在测试条件下输出之间的延迟时间。

这一模型中还有另外一个细微差别。Pal在初始化语句内被局部定义。为完成这一功能，初始化语句中的顺序过程（begin-end）必须标记。在这种情况下，ONLY_ONCE是顺序过程标记。如果在顺序过程内没有局部声明的变量，就不需要该标记。测试模块产生的波形如图2 - 7显示。下面是测试模块产生的输出。

```
PA, PB, PCi = 000 ::: PCo, PSum = 00
PA, PB, PCi = 001 ::: PCo, PSum = 01
PA, PB, PCi = 010 ::: PCo, PSum = 01
PA, PB, PCi = 011 ::: PCo, PSum = 10
PA, PB, PCi = 100 ::: PCo, PSum = 01
PA, PB, PCi = 101 ::: PCo, PSum = 10
PA, PB, PCi = 110 ::: PCo, PSum = 10
PA, PB, PCi = 111 ::: PCo, PSum = 11
```

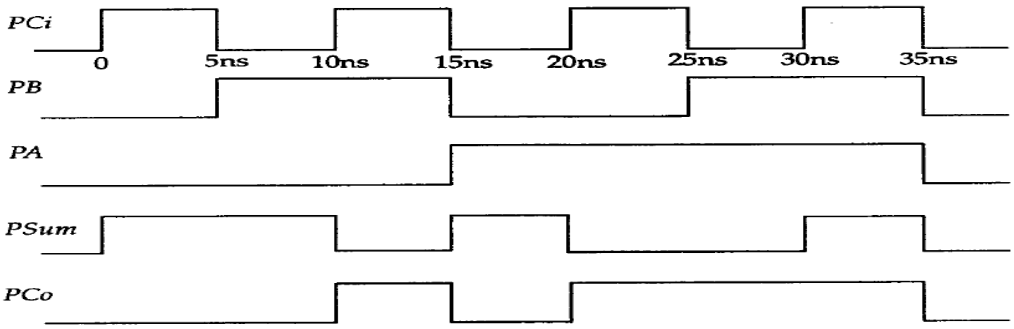
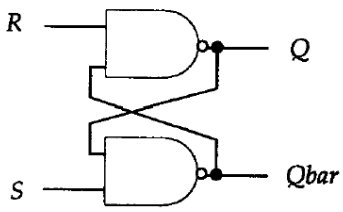


图2-7 测试模块Top执行产生的波形

验证与非门交叉连接构成的R S _ F F模块的测试模块如图2 - 8所示。

```
` timescale 10ns/1ns
module RS_FF(Q, Qbar, R, S) ;
output Q, Qbar;
input R, S;
nand #1 (Q, R, Qbar) ;
nand #1 (Qbar, S, Q,) ;
//在门实例语句中，实例名称是可选的。
endmodule
module Test;
reg TS, TR;
wire TQ, TQb;
//测试模块的实例语句：
RS_FF NSTA ( .Q(TQ), .S(TS), .R(TR), .Qbar(TQb));
//采用端口名相关联的连接方式。
// 加载激励：
initial
begin:
TR = 0;
TS = 0;
#5 TS = 1;
```



```
#5 TS = 0;
```

图2-8　交叉连接的与非门

```
TR = 1;
#5 TS = 1;
TR = 0;
#5 TS = 0;
#5 TR = 1;
end
//输出显示：
initial
$monitor ("At time %t ," , $ t i m e,
"TR = %b, TS=%b, TQ=%b, TQb= %b", TR, TS, TQ, TQb) ;
Endmodule
```

RS_FF模块描述了设计的结构。在门实例语句中使用门时延；例如，第一个实例语句中的门时延为1个时间单位。该门时延意味着如果R或Qbar假定在T时刻变化，Q将在T+1时刻获得计算结果值。模块Test是一个测试模块。测试模块中的RS_FF用实例语句说明其端口用端口名关联方式连接。在这一模块中有两条初始化语句。第一个初始化语句只简单地产生TS和TR上的波形。这一初始化语句包含带有语句间时延的程序块过程赋值语句。第二条初始化语句调用系统任务$monitor。这一系统任务调用的功能是只要参数表中指定的变量值发生变化就打印指定的字符串。产生的相应波形如图2 - 9所示。下面是测试模块产生的输出。请注意`timescale指令在时延上的影响。

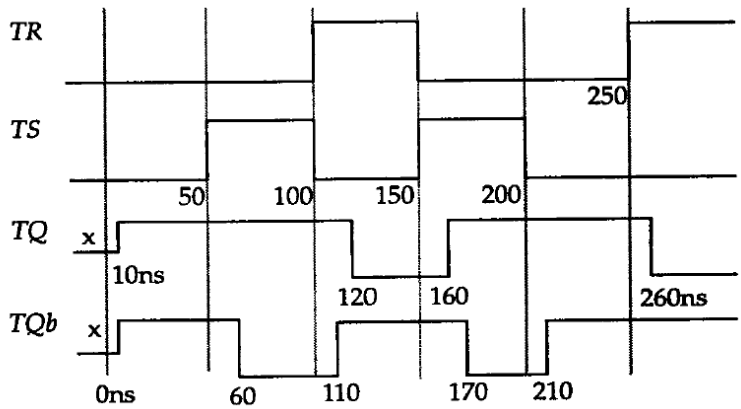


图2-9 Test模块产生的波形

```
At time 0, TR=0, TS=0, TQ=x, TQb= x
At time 10, TR=0, TS=0, TQ=1, TQb= 1
At time 50, TR=0, TS=1, TQ=1, TQb= 1
At time 60, TR=0, TS=1, TQ=1, TQb= 0
At time 100, TR=1, TS=0, TQ=1, TQb= 0
```

```
At time 110, TR=1, TS=0, TQ=1, TQb= 1
At time 120, TR=1, TS=0, TQ=0, TQb= 1
At time 150, TR=0, TS=1, TQ=0, TQb= 1
At time 160, TR=0, TS=1, TQ=1, TQb= 1
At time 170, TR=0, TS=1, TQ=1, TQb= 0
At time 200, TR=0, TS=0, TQ=1, TQb= 0
At time 210, TR=0, TS=0, TQ=1, TQb= 1
At time 250, TR=1, TS=0, TQ=1, TQb= 1
At time 260, TR=1, TS=0, TQ=0, TQb= 1
```

后面的章节将更详细地讲述这些主题。

习题

1. 在数据流描述方式中使用什么语句描述一个设计？
2. 使用`timescale 编译器指令的目的是什么？举出一个实例。
3. 在过程赋值语句中可以定义哪两种时延？请举例详细说明。
4. 采用数据流描述方式描述图2 - 4中所示的1位全加器。
5. initial语句与always 语句的关键区别是什么？
6. 写出产生图2-10所示波形的变量BullsEye的初始化语句。

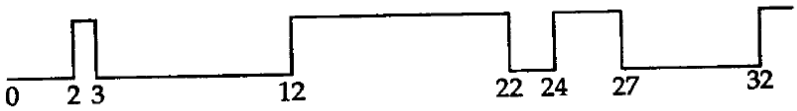


图2-10 变量BullsEye 的波形

7. 采用结构描述方式描写图2-2中所示的2-4译码器。
8. 为2.3节中描述的模块Decode2x4编写一个测试验证程序。
9. 列出你在Verilog HDL模型中使用的两类赋值语句。
10. 在顺序过程中何时需要定义标记？
11. 使用数据流描述方式编写图2-11所示的异或逻辑的Verilog HDL描述，并使用规定的时延。

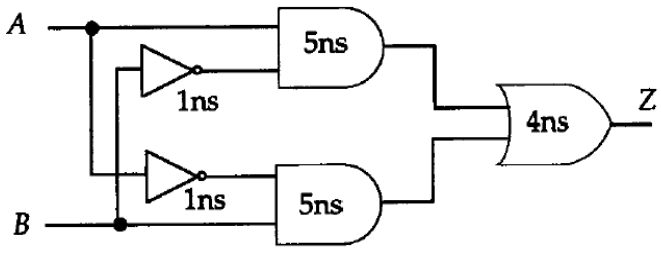


图2-11 异或逻辑

12. 找出下面连续赋值语句的错误。

```
assign Reset = #2^WriteBus;
```

第 3 章　Verilog 语言要素

本章介绍Verilog HDL的基本要素，包括标识符、注释、数值、编译程序指令、系统任务和系统函数。另外，本章还介绍了Verilog硬件描述语言中的两种数据类型。

3.1 标识符

Verilog HDL中的标识符(identifier)可以是任意一组字母、数字、$符号和_(下划线)符号的组合，但标识符的第一个字符必须是字母或者下划线。另外，标识符是区分大小写的。以下是标识符的几个例子：

```
Count
COUNT //与Count不同。
_R1_D2
R56_68
FIVE$
```

转义标识符(escaped identifier)可以在一条标识符中包含任何可打印字符。转义标识符以\(反斜线)符号开头，以空白结尾（空白可以是一个空格、一个制表字符或换行符）。下面例举了几个转义标识符：

```
\7400
\.*.$
\{******}
\~Q
\OutGate与OutGate相同。
```

最后这个例子解释了在一条转义标识符中，反斜线和结束空格并不是转义标识符的一部分。也就是说，标识符\OutGate和标识符OutGate恒等。Verilog HDL定义了一系列保留字，叫做关键词，它仅用于某些上下文中。注意只有小写的关键词才是保留字。例如，标识符always (这是个关键词)与标识符ALWAYS(非关键词)是不同的。另外，转义标识符与关键词并不完全相同。标识符\initial 与标识符initial（这是个关键词）不同。注意这一约定与那些转义标识符不同。

3.2 注释

在Verilog HDL中有两种形式的注释。
/*第一种形式:可以扩展至
多行*/
//第二种形式:在本行结束。

3.3 格式

Verilog HDL区分大小写。也就是说大小写不同的标识符是不同的。此外，Verilog HDL是自由格式的，即结构可以跨越多行编写，也可以在一行内编写。白空（新行、制表符和空格）没有特殊意义。下面通过实例解释说明。

```
initial begin Top = 3'b001; #2 Top = 3'b011; end
```

和下面的指令一样:

```
initial
begin
Top = 3'b001;
#2 Top = 3'b011;
end
```

3.4 系统任务和函数

以$字符开始的标识符表示系统任务或系统函数。任务提供了一种封装行为的机制。这种机制可在设计的不同部分被调用。任务可以返回0个或多个值。函数除只能返回一个值以外与任务相同。此外，函数在0时刻执行，即不允许延迟，而任务可以带有延迟。

```
$display ("Hi, you have reached LT today");
/* $d i s p l a y 系统任务在新的一行中显示。* /
$time
```

//该系统任务返回当前的模拟时间。系统任务和系统函数在第1 0章中详细讲解。

3.5 编译指令

以`反引号）开始的某些标识符是编译器指令。在Verilog 语言编译时，特定的编译器指令在整个编译过程中有效（编译过程可跨越多个文件），直到遇到其它的不同编译程序指令。

完整的标准编译器指令如下:

- `define, `undef

- `ifdef, `else, `endif
- `default_nettype
- `include
- `resetall
- `timescale
- `unconnected_drive, `nounconnected_drive
- `celldefine, `endcelldefine

1、define 和`undef

`define指令用于文本替换，它很像C语言中的#define 指令，如:

```
`define MAX_BUS_SIZE32
. . .
Reg [`MAX_BUS_SIZE-1:0] AddReg;
```

一旦`define指令被编译，其在整个编译过程中都有效。例如，通过另一个文件中的`define指令，MAX_BUS_SIZE能被多个文件使用。

`undef 指令取消前面定义的宏。例如:

```
`define WORD 16 //建立一个文本宏替代。
. . .
wire [`WORD:1] Bus;
. . .
`undef WORD
//在`undef编译指令后, WORD的宏定义不再有效.
```

2、 `ifdef、`else 和`endif

这些编译指令用于条件编译，如下所示：

```
`ifdef WINDOWS
parameter WORD_SIZE = 16
`else
parameter WORD_SIZE = 32
`endif
```

在编译过程中，如果已定义了名字为WINDOWS的文本宏，就选择第一种参数声明，否则选择第二种参数说明。`else 程序指令对于`ifdef 指令是可选的。

3、 `default_nettype

该指令用于为隐式线网指定线网类型。也就是将那些没有被说明的连线定义线网类型。`default_nettype wand该实例定义的缺省的线网为线与类型。因此，如果在此指令后面的任何模块中没有说明的连线，那么该线网被假定为线与类型。

4、 `include

`include 编译器指令用于嵌入内嵌文件的内容。文件既可以用相对路径名定义，也可以用全路径名定义, 例如:

`include"../../primitives.v"

编译时，这一行由文件“../../primitives.v”的内容替代。

5、 `resetall

该编译器指令将所有的编译指令重新设置为缺省值。

`resetall

例如，该指令使得缺省连线类型为线网类型。

6、 `timescale

在Verilog HDL 模型中，所有时延都用单位时间表述。使用`timescale编译器指令将时间单位与实际时间相关联。该指令用于定义时延的单位和时延精度。`timescale编译器指令格式为：

`timescale time_unit/time_precision

time_unit和time_precision由值1、10、和100以及单位s、ms、us、ns、ps和fs组成。例如：

`timescale 1ns/100ps

表示时延单位为1ns, 时延精度为100ps。`timescale 编译器指令在模块说明外部出现, 并且影响后面所有的时延值。例如:`timescale 1ns/100ps

```
Module AndFunc (Z, A, B) ;
outpu  Z;
input A, B;
and #(5.22, 6.17 ) A 1 (Z, A, B);
//规定了上升及下降时延值。
endmodule
```

编译器指令定义时延以ns为单位，并且时延精度为1/10 ns（100 ps）。因此，时延值5.22对应5.2ns,时延6.17对应6.2ns。如果用如下的`timescale程序指令代替上例中的编译器指令,`timescale 10ns/1ns那么5.22对应52ns,6.7对应62ns。在编译过程中，`timescale指令影响这一编译器指令后面所有模块中的时延值，直至遇到另一个`timescale指令或`resetall指令。当一个设计中的多个模块带有自身的`timescale编译指令时将发生什么？在这种情况下，模拟器总是定位在所有模块的最小时延精度上，并且所有时延都相应地换算为最小时延精度。例如，

```
`timescale 1ns/100ps
Module AndFunc (Z, A, B) ;
output Z;
input A, B;
and #(5.22, 6.17 ) A1 (Z, A, B) ;
endmodule
`timescale 10ns/1ns
module TB;
reg PutA, PutB;
```

```
wire GetO;
initial
begin
PutA = 0;
PutB = 0;
#5.21 PutB = 1;
#10.4 PutA = 1;
#15 PutB = 0;
end
AndFuncAF1(GetO, PutA, PutB) ;
endmodule
```

在这个例子中，每个模块都有自身的`timescale编译器指令。`timescale编译器指令第一次应用于时延。因此，在第一个模块中， 5.22对应5.2ns, 6.17对应6.2 ns; 在第二个模块中5.21对应52ns,10.4对应104ns, 15对应150ns。如果仿真模块TB，设计中的所有模块最小时间精度为100ps。因此，所有延迟（特别是模块TB中的延迟）将换算成精度为100ps。延迟52ns现在对应520*100ps，104对应104 0*100 ps，150对应1500*100 ps。更重要的是，仿真使用100ps为时间精度。如果仿真模块AndFunc，由于模块TB不是模块AddFunc的子模块，模块TB中的`timescale程序指令将不再有效。

7、 `unconnected_drive和`nounconnected_drive

在模块实例化中，出现在这两个编译器指令间的任何未连接的输入端口或者为正偏电路状态或者为反偏电路状态。`unconnected_drive pull1

```
. . .
/*在这两个程序指令间的所有未连接的输入端口为正偏电路状态（连接到高电平）*
/
`nounconnected_drive
`unconnected_drive pull0
. . .
/*在这两个程序指令间的所有未连接的输入端口为反偏电路状态（连接到低电平）*
/
`nounconnected_drive
```

8、 `celldefine 和`endcelldefine

这两个程序指令用于将模块标记为单元模块。它们表示包含模块定义，如下例所示。

```
`celldefine
Module FD1S3AX(D,CK,Z) ;
. . .
endmodule
`endcelldefine
```

某些PLI例程使用单元模块。

3.6 值集合

Verilog HDL有下列四种基本的值：

1）0：逻辑0或“假”

2）1：逻辑1或“真”

3）x：未知

4）z：高阻

注意这四种值的解释都内置于语言中。如一个为z的值总是意味着高阻抗，一个为0的值通常是指逻辑0。在门的输入或一个表达式中的为“ z”的值通常解释成“ x”。此外，x值和z值都是不分大小写的，也就是说，值0x1z与值0X1Z相同。Verilog HDL中的常量是由以上这四类基本值组成的。

Verilog HDL中有三类常量：

1）整型

2）实数型

3）字符串型

下划线符号（_）可以随意用在整数或实数中，它们就数量本身没有意义。它们能用来提高易读性；唯一的限制是下划线符号不能用作为首字符。

1、整型数

整型数可以按如下两种方式书写：

- 简单的十进制数格式
- 基数格式

1） 简单的十进制格式

这种形式的整数定义为带有一个可选的“+”（一元）或“－”（一元）操作符的数字序列。下面是这种简易十进制形式整数的例子。

32 十进制数3 2

－15 十进制数－1 5

这种形式的整数值代表一个有符号的数。负数可使用两种补码形式表示。因此3 2在5位的二进制形式中为1 0 0 0 0，在6位二进制形式中为11 0 0 0 1；－1 5在5位二进制形式中为1 0 0 0 1，在6位二进制形式中为11 0 0 0 1。

2） 基数表示法

这种形式的整数格式为：

[size]’base value

size定义以位计的常量的位长；base为o或O（表示八进制），b或B（表示二进制），d或D（表示十进制），h或H（表示十六进制）之一；value是基于base的值的数字序列。值x和z以及十六进制中的a到f不区分大小写。

下面是一些具体实例：

5’O37 5位八进制数

4’D2 4位十进制数

4'B1x_01 4位二进制数

7'Hx 7位x(扩展的x), 即x x x x x x x

4'hZ 4位z(扩展的z) , 即z z z z

4'd-4 非法：数值不能为负

8'h 2 A 在位长和字符之间,以及基数和数值之间允许出现空格

3'b001 非法: ` 和基数b之间不允许出现空格

(2+3)'b10 非法:位长不能够为表达式

注意，x（或z）在十六进制值中代表4位x（或z），在八进制中代表3位x（或z），在二进制中代表1位x（或z）。基数格式计数形式的数通常为无符号数。这种形式的整型数的长度定义是可选的。如果没有定义一个整数型的长度，数的长度为相应值中定义的位数。下面是两个例子：

'o721 9位八进制数

'hAF 8位十六进制数

如果定义的长度比为常量指定的长度长，通常在左边填0补位。但是如果数最左边一位为x或z，就相应地用x或z在左边补位。例如：

10'b10 左边添0占位, 0000000010

10'bx0x1 左边添x占位, x x x x x x x 0 x 1

如果长度定义得更小，那么最左边的位相应地被截断。例如：

3'b1001_0011与3'b011 相等

5'H0FFF 与5'H1F 相等

? 字符在数中可以代替值z在值z被解释为不分大小写的情况下提高可读性

2、 实数

实数可以用下列两种形式定义：

1) 十进制计数法；例如

2.0

5.678

11572.12

0.1

2. // 非法：小数点两侧必须有1位数字

2) 科学计数法； 这种形式的实数举例如下：

23_5.1e2 其值为23510.0; 忽略下划线

3.6E2360.0 (e与E相同)

5E－40.0005

Verilog语言定义了实数如何隐式地转换为整数。实数通过四舍五入被转换为最相近的整数。

42.446，42.45 转换为整数4 2

92.5, 92.699 转换为整数93

－15.62 转换为整数－1 6

－26.22 转换为整数－2 6

3、 字符串

字符串是双引号内的字符序列。字符串不能分成多行书写。例如:

″INTERNAL ERROR″

″REACHED－>HERE ″

用8位ASCII值表示的字符可看作是无符号整数。因此字符串是8位ASCII值的序列。为存储字符串“INTERNAL ERROR”，变量需要8*14位。

```
reg[1:8*14]Message;
. . .
Message = ″INTERNAL ERROR″
```

反斜线(\ ) 用于对确定的特殊字符转义。\n 换行符\t 制表符\\ 字符\本身\″ 字符″\206 八进制数2 0 6对应的字符

3.7 数据类型

Verilog HDL 有两大类数据类型。

1) 线网类型。nettype 表示Verilog结构化元件间的物理连线。它的值由驱动元件的值决定，例如连续赋值或门的输出。如果没有驱动元件连接到线网，线网的缺省值为z。

2) 寄存器类型。register type表示一个抽象的数据存储单元，它只能在a l w a y s语句和initial语句中被赋值，并且它的值从一个赋值到另一个赋值被保存下来。寄存器类型的变量具有x的缺省值。

3.7.1 线网类型

线网数据类型包含下述不同种类的线网子类型。

- wire
- tri
- wor
- trior
- wand
- triand
- trireg
- tri1
- tri0
- supply0
- supply1

简单的线网类型说明语法为：

```
net_kind[msb:lsb] net1, net2, . . . , netN;
```

net_kind 是上述线网类型的一种。msb和lsb是用于定义线网范围的常量表达式；范

围定义是可选的；如果没有定义范围，缺省的线网类型为1位。下面是线网类型说明实例。

```
wire Rdy, Start; //2个1位的连线。
wand [2:0] Addr; //A d d r是3位线与。
```

当一个线网有多个驱动器时，即对一个线网有多个赋值时，不同的线网产生不同的行为。

例如，

```
worRde;
...
assignRde = Blt&Wy l;
...
Assign Rde = Kbl|Kip;
```

本例中，Rde有两个驱动源，分别来自于两个连续赋值语句。由于它是线或线网，Rde的有效值由使用驱动源的值（右边表达式的值）的线或(wor)表（参见后面线或网的有关章节）决定。

1.wire和tri线网用于连接单元的连线是最常见的线网类型。连线与三态线（ t r i ）网语法和语义一致；三态线可以用于描述多个驱动源驱动同一根线的线网类型；并且没有其他特殊的意义。

```
wire Reset;
wire [3:2] Cla, Pla, Sla;
tri [MSB－1:LSB+1] Art;
```

如果多个驱动源驱动一个连线（或三态线网），线网的有效值由下表决定。

wire（或 tri）	0	1	x	z
0	0	x	x	0
1	x	1	x	1
x	x	x	x	x
z	0	1	x	z

下面是一个具体实例：

```
assign Cla = Pla & Sla;
...
assign Cla = Pla^Sla;
```

在这个实例中， Cla有两个驱动源。两个驱动源的值（右侧表达式的值）用于在上表中索引，以便决定Cla的有效值。由于Cla是一个向量，每位的计算是相关的。例如，如果第一个右侧表达式的值为01x, 并且第二个右测表达式的值为11z，那么Cla 的有效值是x1x(第一位0和1在表中索引到x, 第二位1和1在表中索引到1，第三位x 和z在表中索引到x)。

2. wor和trior线网

线或指如果某个驱动源为1，那么线网的值也为1。线或和三态线或(trior)在语法和功能上是一致的。

wor[MSB:LSB] Art;

trior [MAX－1:MIN－1] Rdx, Sdx, Bdx;

如果多个驱动源驱动这类网，网的有效值由下表决定。

wor(或 trior)	0	1	x	z
0	0	1	x	0
1	1	1	1	1
x	x	1	x	x
z	0	1	x	z

3. wand和triand线网线与(wand)网指如果某个驱动源为0，那么线网的值为0。线与和三态线与(triand)网在语法和功能上是一致的。

wand[-7:0] Dbus;

triand Reset, Clk;

如果这类线网存在多个驱动源，线网的有效值由下表决定。

wand(或 triand)	0	1	x	z
0	0	0	0	0
1	0	1	x	1
x	0	x	x	x
z	0	1	x	z

4. trireg线网

此线网存储数值(类似于寄存器)，并且用于电容节点的建模。当三态寄存器(trireg)的所有驱动源都处于高阻态，也就是说，值为z时，三态寄存器线网保存作用在线网上的最后一个值。此外，三态寄存器线网的缺省初始值为x。

trireg[1:8] Dbus, Abus;

5. tri0和tri1线网

这类线网可用于线逻辑的建模，即线网有多于一个驱动源。tri0（tri1）线网的特征是，若无驱动源驱动，它的值为0（tri1的值为1）。

tri0[－3:3] GndBus;

tri1[0:－5] OtBus, ItBus;

下表显示在多个驱动源情况下tri0或tri1网的有效值。

tri0(tri1)	0	1	x	z
0	0	x	x	0
1	x	1	x	1
x	x	x	x	x
z	0	1	x	0(1)

6. supply0和supply1线网supply0用于对“地”建模，即低电平0；supply1网用于对电源建模，即高电平1；例如:

```
supply 0 Gnd, ClkGnd;
supply 1 [2:0] Vcc;
```

3.7.2 未说明的线网

在Verilog HDL中，有可能不必声明某种线网类型。在这样的情况下，缺省线网类型为1位线网。可以使用`default_nettype编译器指令改变这一隐式线网说明方式。使用方法如下：

```
`default_nettypenet_kind
```

例如，带有下列编译器指令：

```
`default_nettype wand
```

任何未被说明的网缺省为1位线与网。

3.7.3 向量和标量线网

在定义向量线网时可选用关键词scalared 或vectored。如果一个线网定义时使用了关键词vectored, 那么就不允许位选择和部分选择该线网。换句话说，必须对线网整体赋值（位选择和部分选择在下一章中讲解）。例如:

```
wire vectored[3:1] Grb;
//不允许位选择Grb[2]和部分选择Grb[3:2]
wor scalared [4:0] Best;
//与wor [4:0] Best相同，允许位选择Best[2]和部分选择Best[3:1]。
```

如果没有定义关键词，缺省值为标量。

3.7.4 寄存器类型

有5种不同的寄存器类型。

- reg
- integer
- time
- real
- realtime

1. reg寄存器类型

寄存器数据类型reg是最常见的数据类型。reg类型使用保留字reg加以说明，形式如下：

reg [msb:lsb] reg1, reg2, . . . regN;

msb和lsb 定义了范围，并且均为常数值表达式。范围定义是可选的；如果没有定义范围，缺省值为1位寄存器。例如：

reg [3:0] Sat; //S a t为4 位寄存器。

regCnt; //1 位寄存器。

reg [1:32] Kisp, Pisp, Lisp;

寄存器可以取任意长度。寄存器中的值通常被解释为无符号数, 例如：

reg [1:4] Comb;

...

Comb = －2; //C o m b 的值为1 4（1 1 1 0），1 1 1 0是2的补码。

Comb = 5; //C o m b的值为1 5（0 1 0 1）。

2. 存储器

存储器是一个寄存器数组。存储器使用如下方式说明：

reg [msb:1sb] memory1 [upper 1:lower1] ,

memory 2 [upper 2: lower 2] ,... ;

例如：

reg [0:3] MyMem [0:63]

//MyMem为64个4位寄存器的数组。

reg Bog [1:5]

//Bog为5个1位寄存器的数组。

My Mem和Bog都是存储器。数组的维数不能大于2。注意存储器属于寄存器数组类型。线网数据类型没有相应的存储器类型。单个寄存器说明既能够用于说明寄存器类型，也可以用于说明存储器类型。parameterADDR_SIZE = 16 ,WORD_SIZE = 8;reg [1: WORD_SIZE]Ram Par [ADDR_SIZE－1:0], DataReg;

RamPar是存储器，是16个8位寄存器数组，而DataReg是8位寄存器。

在赋值语句中需要注意如下区别：存储器赋值不能在一条赋值语句中完成，但是寄存器可以。因此在存储器被赋值时，需要定义一个索引。下例说明它们之间的不同。

reg [1:5] Dig; //D i g为5位寄存器。

...

Dig = 5'b11011;

上述赋值都是正确的, 但下述赋值不正确：

reg BO g[1:5]; //B o g为5个1位寄存器的存储器。

...

Bog = 5'b11011;

有一种存储器赋值的方法是分别对存储器中的每个字赋值。例如：

reg [0:3] Xrom [1 : 4]

...

Xrom[1] = 4'hA;

Xrom[2] = 4'h8;

```
Xrom[3] = 4'hF;
Xrom[4] = 4'h2;
```

为存储器赋值的另一种方法是使用系统任务：

1) $readmemb （加载二进制值）

2) $readmemb （加载十六进制值）

这些系统任务从指定的文本文件中读取数据并加载到存储器。文本文件必须包含相应的二进制或者十六进制数。例如：

```
reg [1:4] RomB [7:1] ;
$readmemb ("ram.patt", RomB);
```

Romb是存储器。文件“ram.patt”必须包含二进制值。文件也可以包含空白空间和注释。下面是文件中可能内容的实例。

```
1 1 0 1
1 1 1 0
1 0 0 0
0 1 1 1
0 0 0 0
1 0 0 1
0 0 1 1
```

系统任务$readmemb促使从索引7即Romb最左边的字索引，开始读取值。如果只加载存储器的一部分，值域可以在$readmemb方法中显式定义。例如：

$readmemb ("ram.patt", RomB,5,3);在这种情况下只有Rom b[5] ,Rom b[4]和Rom b[3]这些字从文件头开始被读取。被读取的值为11 0 1、11 0 0和1 0 0 0。文件可以包含显式的地址形式。

```
@hex_address value
```

如下实例：

```
@5 11001
@2 11010
```

在这种情况下，值被读入存储器指定的地址。

当只定义开始值时，连续读取直至到达存储器右端索引边界。例如：

```
$readmemb ("rom.patt", RomB, 6);
//从地址6开始，并且持续到1。
$readmemb ( "rom.patt",RomB, 6, 4);
//从地址6读到地址4。
```

3. Integer寄存器类型

整数寄存器包含整数值。整数寄存器可以作为普通寄存器使用，典型应用为高层次行为建模。使用整数型说明形式如下：

```
integer integer1, integer2 ,... intergerN [msb:lsb] ;
```

msb和lsb是定义整数数组界限的常量表达式，数组界限的定义是可选的。注意容许无位界限的情况。一个整数最少容纳32位。但是具体实现可提供更多的位。下面是整数说明

的实例。

```
Integer A, B, C; //三个整数型寄存器。
integer Hist [3:6]; //一组四个寄存器。
```

一个整数型寄存器可存储有符号数，并且算术操作符提供2的补码运算结果。整数不能作为位向量访问。例如，对于上面的整数B的说明，B[6]和B[20:10]是非法的。一种截取位值的方法是将整数赋值给一般的reg类型变量，然后从中选取相应的位，如下所示：

```
reg [31:0] Breg;
integerBint;
...
//Bint[6]和Bint[20:10]是不允许的。
...
Breg = Bint;
```

/*现在，Breg[6]和Breg[20:10]是允许的，并且从整数Bint获取相应的位值。*/上例说明了如何通过简单的赋值将整数转换为位向量。类型转换自动完成，不必使用特定的函数。从位向量到整数的转换也可以通过赋值完成。例如:

```
integer J;
reg[3:0] Bcq;
J= 6; // J的值为3 2 ’ b 0 0 0 0 . . . 0 0 1 1 0。
Bcq = J; // B c q的值为4 ’ b 0 1 1 0。
Bcq = 4’b0101.
J= Bcq; //J的值为3 2 ’ b 0 0 0 0 . . . 0 0 1 0 1。
J = －6; //J 的值为3 2 ’ b 1 1 1 1 . . . 1 1 0 1 0。
Bcq = J; //B c q的值为4 ’ b 1 0 1 0。
```

注意赋值总是从最右端的位向最左边的位进行；任何多余的位被截断。如果你能够回忆起整数是作为2的补码位向量表示的，就很容易理解类型转换。

4. time类型

time类型的寄存器用于存储和处理时间。time类型的寄存器使用下述方式加以说明。

```
time time_id1,time_id2 , . . ., time_idN[msb:1sb] ;
```

msb和lsb是表明范围界限的常量表达式。如果未定义界限，每个标识符存储一个至少6 4位的时间值。时间类型的寄存器只存储无符号数。例如:

```
time Events [0:31]; //时间值数组。
timeCurrTime; //C u r r T i m e 存储一个时间值。
```

5. real和realtime类型

实数寄存器（或实数时间寄存器）使用如下方式说明：

```
//实数说明：
realreal_reg1,real_reg2, . . ., real_regN;
//实数时间说明：
Realtime realtime_reg1,realtime_reg2,..., realtime_regN;
```

realtime与real类型完全相同。例如:

```
real Swing, Top;
realtime CurrTime;
```

real说明的变量的缺省值为0。不允许对real声明值域、位界限或字节界限。

当将值x和z赋予real类型寄存器时，这些值作0处理。

```
Real RamCnt;
...
RamCnt = 'b01x1Z;
```

RamCn t在赋值后的值为' b 0 1 0 1 0。

3.8 参数

参数是一个常量。参数经常用于定义时延和变量的宽度。使用参数说明的参数只被赋值一次。参数说明形式如下：

```
Parameter param1 = const_expr1, param2 = const_expr2, ...,
paramN = const_exprN;
```

下面为具体实例：

```
Parameter LINELENGTH = 132, ALL_X_S = 16'bx;
parameter BIT = 1, BYTE = 8, PI = 3.14;
parameter STROBE_DELAY= (BYTE+BIT)/2;
parameter TQ_FILE = "/home/bhasker/TEST/add.tq" ;
```

参数值也可以在编译时被改变。改变参数值可以使用参数定义语句或通过在模块初始化语句中定义参数值（这两种机制将在第9章中详细讲解）。

习题

1. 下列标识符哪些合法，哪些非法？
   COunT,1_2Many, **1, R e a l?, \wait, Initial
2. 系统任务和系统函数的第一个字符标识符是什么？
3. 举例说明文本替换编译指令？
4. 在Verilog HDL中是否有布尔类型？
5. 下列表达式的位模式是什么？
   7'o44, 'Bx0, 5'bx110, 'hA0, 10'd2, 'hzF
6. 赋值后存储在Qpr中的位模式是什么？

```
reg [1:8*2] Qpr;
. . .
Qpr = "ME" ;
```

7. 如果线网类型变量说明后未赋值，其缺省值为多少？
8. Verilog HDL 允许没有显式说明的线网类型。如果是这样，怎样决定线网类型？

9. 下面的说明错在哪里？

```
integer [0:3] Ripple;
```

10. 编写一个系统任务从数据文件“memA.data”中加载32×64字存储器。
11. 写出在编译时覆盖参数值的两种方法。

第 4 章　表达式

本章讲述在Verilog HDL中编写表达式的基础。

表达式由操作数和操作符组成。表达式可以在出现数值的任何地方使用。

4.1 操作数

操作数可以是以下类型中的一种：

1) 常数
2) 参数
3) 线网
4) 寄存器
5) 位选择
6) 部分选择
7) 存储器单元
8) 函数调用

1、常数

前面的章节已讲述了如何书写常量。下面是一些实例。

```
256,7 // 非定长的十进制数。
4'b10_11, 8'h0A // 定长的整型常量。
'b1, 'hFBA // 非定长的整数常量。
90.00006 // 实数型常量。
"BOND" // 串常量；每个字符作为8位A S C I I值存储。
```

表达式中的整数值可被解释为有符号数或无符号数。如果表达式中是十进制整数，例如，1 2被解释为有符号数。如果整数是基数型整数（定长或非定长），那么该整数作为无符号数对待。下面举例说明。

12是0 1 1 0 0的5位向量形式（有符号）

-12是1 0 1 0 0的5位向量形式（有符号）

5'b 0 1 1 0 0是十进制数12（无符号）

5'b 1 0 1 0 0是十进制数20（无符号）

4'd12是十进制数12（无符号）

更为重要的是对基数表示或非基数表示的负整数处理方式不同。非基数表示形式的负整数作为有符号数处理，而基数表示形式的负整数值作为无符号数。因此-44和-6'o54（十进制的44等于八进制的54）在下例中处理不同。

```
integer Cone;
...
Cone = -44/4
Cone = -6'o54/4;
```

注意－44和－6 'o54以相同的位模式求值；但是－44作为有符号数处理，而－6 'o54作为无符号数处理。因此第一个字符中Cone的值为－11，而在第二个赋值中Cone的值为1073741813。（因为C o n e 为非定长整型变量，基数表示形式的负数在机内以补码形式出现。—译者注）

2、 参数

前一章中已对参数作了介绍。参数类似于常量，并且使用参数声明进行说明。下面是参数说明实例。

```
Parameter LOAD = 4'd12, STORE = 4'd10;
```

LOAD和STORE为参数的例子，值分别被声明为12和10。

3、线网

可在表达式中使用标量线网（1位）和向量线网（多位）。下面是线网说明实例。

```
wire[0:3] Prt; //Prt 为4位向量线网。
wire Bdq; //Bbq 是标量线网。
```

线网中的值被解释为无符号数。在连续赋值语句中，

```
Assign Prt = -3;
```

Prt被赋于位向量1101，实际上为十进制的13。在下面的连续赋值中，

```
assign Prt = 4'HA;
```

Prt被赋于位向量1010，即为十进制的10。

4、 寄存器

标量和向量寄存器可在表达式中使用。寄存器变量使用寄存器声明进行说明。例如:

```
integer TemA, TemB;
reg[1:5] State;
time Que[ 1 : 5 ] ;
```

整型寄存器中的值被解释为有符号的二进制补码数，而reg寄存器或时间寄存器中的值被解释为无符号数。实数和实数时间类型寄存器中的值被解释为有符号浮点数。

TemA = -10; //T e m A值为位向量1 0 1 1 0，是1 0的二进制补码。

TemA = 'b1011; // T e m A值为十进制数1 1。
State = -10; // S t a t e值为位向量1 0 1 1 0，即十进制数2 2。
State = 'b1011; // S t a t e值为位向量0 1 0 1 1，是十进制值1 1。

5、位选择

位选择从向量中抽取特定的位。形式如下：
net_or_reg_vector [bit_select_expr]
下面是表达式中应用位选择的例子。
State[1] && State[4] //寄存器位选择。
Prt[0]|Bbq // 线网位选择。
如果选择表达式的值为x、z，或越界，则位选择的值为x。例如State[x]值为x。

6、部分选择

在部分选择中，向量的连续序列被选择。形式如下：
net_or_reg_vector[msb_const_expr:1sb_const_expr]
其中范围表达式必须为常数表达式。例如。
State[1:4] //寄存器部分选择。
Prt[1:3] // 线网部分选择。
选择范围越界或为x、z时，部分选择的值为x。

7、存储器单元

存储器单元从存储器中选择一个字。形式如下：
memory[word_address]
例如：
reg[1:8] Ack, Dram[0:63];
. . .
Ack = Dram[60]; //存储器的第6 0个单元。
不允许对存储器变量值部分选择或位选择。例如，
Dram [60][2] 不允许。
Dram [60][2:4] 也不允许。
在存储器中读取一个位或部分选择一个字的方法如下：将存储器单元赋值给寄存器变量，然后对该寄存器变量采用部分选择或位选择操作。例如，Ack[2]和Ack[2:4]是合法的表达式。

8、 函数调用

表达式中可使用函数调用。函数调用可以是系统函数调用（以$字符开始）或用户定义的函数调用。例如：
$time + SumOfEvents(A, B)
/ * $time是系统函数，并且SumOfEvents是在别处定义的用户自定义函数。* /

4.2 操作符

Verilog HDL中的操作符可以分为下述类型：

1) 算术操作符
2) 关系操作符
3) 相等操作符
4) 逻辑操作符
5) 按位操作符
6) 归约操作符

（归约操作符为一元操作符，对操作数的各位进行逻辑操作，结果为二进制数。—译者）

7) 移位操作符
8) 条件操作符
9) 连接和复制操作符

下表显示了所有操作符的优先级和名称。操作符从最高优先级（顶行）到最低优先级（底行）排列。同一行中的操作符优先级相同。

+	一元加	≫	右移
-	一元减	<	小于
!	一元逻辑非	< =	小于等于
~	一元按位求反	>	大于
&	归约与	>=	大于等于
~&	归约与非	==	逻辑相等
^	归约异或	!=	逻辑不等
^~ 或 ~^	归约异或非	===	全等
|	归约或	!==	非全等
~|	归约或非	&	按位与
*	乘	^	按位异或
/	除	^~ or ~^	按位异或非
%	取模	|	按位或
+	二元加	&&	逻辑与
_	二元减	||	逻辑或
<<	左移	?:	条件操作符

除条件操作符从右向左关联外，其余所有操作符自左向右关联。下面的表达式：

A+B-C等价于：(A+B)-C //自左向右

而表达式：

A?B:C?D:F

等价于：A?B:(C?D:F) //从右向左

圆扩号能够用于改变优先级的顺序，如以下表达式：(A?B:C)?D:F

1、算术操作符

算术操作符有：

- +（一元加和二元加）
- —（一元减和二元减）
- *（乘）
- /（除）
- %（取模）

整数除法截断任何小数部分。例如：

7/4 结果为1

取模操作符求出与第一个操作符符号相同的余数。

7%4 结果为3

-7%4 结果为-3

如果算术操作符中的任意操作数是X或Z，那么整个结果为X。例如：

'b10x1 + 'b01111 结果为不确定数' bx x x x x

1）算术操作结果的长度

算术表达式结果的长度由最长的操作数决定。在赋值语句下，算术操作结果的长度由操作符左端目标长度决定。考虑如下实例：

```
reg [0:3] Arc, Bar, Crt;
reg [0:5] Frx;
. . .
Arc = Bar + Crt;
Frx = Bar + Crt;
```

第一个加的结果长度由Bar，Crt和Arc长度决定，长度为4位。第二个加法操作的长度同样由Frx的长度决定（Frx、B at和Crt中的最长长度），长度为6位。在第一个赋值中，加法操作的溢出部分被丢弃；而在第二个赋值中，任何溢出的位存储在结果位Frx[1]中。

在较大的表达式中，中间结果的长度如何确定？在Verilog HDL中定义了如下规则：表达式中的所有中间结果应取最大操作数的长度（赋值时，此规则也包括左端目标）。考虑另一个实例：

```
wire [4:1] Box, Drt;
wire [1:5] Cfg;
wire [1:6] Peg;
wire [1:8] Adt;
. . .
Assign Adt = (Box + Cfg) + (Drt + Peg) ;
```

表达式左端的操作数最长为6，但是将左端包含在内时，最大长度为8。所以所有的加操作使用8位进行。例如：Box和Cfg相加的结果长度为8位。

2） 无符号数和有符号数

执行算术操作和赋值时，注意哪些操作数为无符号数、哪些操作数为有符号数非常重要。

无符号数存储在：

- 线网
- 一般寄存器
- 基数格式表示形式的整数

有符号数存储在：

- 整数寄存器
- 十进制形式的整数

下面是一些赋值语句的实例：

```
reg [0:5] Bar;
integer Tab;
. . .
Bar = -4'd12; //寄存器变量B a r的十进制数为52，向量值为1 1 0 1 0 0。
Tab = -4'd12; //整数T a b的十进制数为-12，位形式为1 1 0 1 0 0。
-4'd12/4 //结果是1073741821。
-12/4 //结果是- 3
```

因为B a r是普通寄存器类型变量，只存储无符号数。右端表达式的值为' b 11 0 1 0 0（12的二进制补码）。因此在赋值后，Bar存储十进制值52。在第二个赋值中，右端表达式相同，值为' b 11 0 1 0 0，但此时被赋值为存储有符号数的整数寄存器。Tab存储十进制值－12（位向量为11 0 1 0 0）。注意在两种情况下，位向量存储内容都相同；但是在第一种情况下，向量被解释为无符号数，而在第二种情况下，向量被解释为有符号数。

下面为具体实例：

```
Bar = - 4'd12/4;
Tab = - 4'd12 /4;
Bar = - 12/4
Tab = - 12/4
```

在第一次赋值中，Bar被赋于十进制值61（位向量为1111 0 1）。而在第二个赋值中，Tab被赋于与十进制1073741821（位值为0 0 11 . . . 111 0 1）。Bar在第三个赋值中赋于与第一个赋值相同的值。这是因为Bar只存储无符号数。在第四个赋值中，Bar被赋于十进制值－3。

下面是另一些例子：

```
Bar = 4 - 6;
Tab = 4 - 6;
```

Bar被赋于十进制值6 2（－2的二进制补码），而Ta b被赋于十进制值－2（位向量为11111 0）。

下面为另一个实例：

```
Bar = -2 + (-4);
Tab = -2 + (-4);
```

Bar被赋于十进制值58（位向量为111 0 1 0），而Tab被赋于十进制值－6（位向量为111010）。

2、 关系操作符

关系操作符有：

- >（大于）
- <（小于）
- >=（不小于）
- <=（不大于）

关系操作符的结果为真（1）或假（0）。如果操作数中有一位为X或Z，那么结果为X。例如：

23 > 45结果为假（0），而：52< 8'hxFF

结果为x。如果操作数长度不同，长度较短的操作数在最重要的位方向（左方）添0补齐。例如：'b1000 > = 'b01110等价于：'b01000 > = 'b01110结果为假（ 0）。

3、相等关系操作符

相等关系操作符有:

- = =（逻辑相等）
- != （逻辑不等）
- = = =（全等）
- != =（非全等）

如果比较结果为假，则结果为0；否则结果为1。在全等比较中，值x和z严格按位比较。也就是说，不进行解释，并且结果一定可知。而在逻辑比较中，值x和z具有通常的意义，且结果可以不为x。也就是说，在逻辑比较中，如果两个操作数之一包含x或z，结果为未知的值（x）。如下例，

假定：Data = 'b11x0;

Addr = 'b11x0;

那么：Data = = Addr不定，也就是说值为x，但：Data === Addr为真，也就是说值为1。

如果操作数的长度不相等，长度较小的操作数在左侧添0补位，例如：

2'b10 = = 4'b0010

与下面的表达式相同：

4'b0010 = = 4'b0010

结果为真（1）。

4、 逻辑操作符

逻辑操作符有:

- && (逻辑与)
- || (逻辑或)
- ! (逻辑非)

这些操作符在逻辑值0或1上操作。逻辑操作的结构为0或1。例如, 假定:

Crd = 'b0; //0为假

Dgs = 'b1; //1为真

那么:

Crd && Dgs 结果为0（假）

Crd || Dgs 结果为1（真）

! Dgs 结果为0（假）

对于向量操作，非0向量作为1处理。例如，假定:

A_Bus = 'b0110;

B_Bus = 'b0100;

那么:

A_Bus||B_Bus 结果为1

A_Bus && B_Bus 结果为1

并且:!A_Bus 与!B_Bus的结果相同。

结果为0。

如果任意一个操作数包含x，结果也为x。!x 结果为x

5、 按位操作符

按位操作符有:

- ~（一元非）
- &（二元与）
- |（二元或）
- ^（二元异或）
- ~^, ^~（二元异或非）

这些操作符在输入操作数的对应位上按位操作，并产生向量结果。下表显示对于不同操作符按步操作的结果。

& 与	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

| 或	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

^ 异或	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

^~异或非	0	1	x	z
0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

~ 非	0	1	x	z
	1	0	x	x

例如，假定,

A = 'b0110;

B = 'b0100;

那么:

A | B 结果为0 1 1 0

A & B 结果为0 1 0 0

如果操作数长度不相等，长度较小的操作数在最左侧添0补位。例如,

'b0110 ^ 'b10000与如下式的操作相同:'b00110 ^ 'b10000结果为' b 1 0 11 0。

6、归约操作符

归约操作符在单一操作数的所有位上操作，并产生1位结果。归约操作符有:

• & (归约与)

如果存在位值为0，那么结果为0；若如果存在位值为x或z，结果为x；否则结果为1。

• ~& (归约与非)与归约操作符&相反。

• | (归约或)

如果存在位值为1，那么结果为1；如果存在位x或z，结果为x；否则结果为0。

• ~| (归约或非)

与归约操作符|相反。

• ^ (归约异或)

如果存在位值为x或z，那么结果为x；否则如果操作数中有偶数个1, 结果为0；否则结果为1。

• ~^ (归约异或非)

与归约操作符^正好相反。如下所示。假定,

A = 'b0110;

B = 'b0100;

那么:

|B 结果为1

& B 结果为0

~ A 结果为1

归约异或操作符用于决定向量中是否有位为x。假定,

MyReg = 4'b01x0；那么:^MyReg 结果为x

上述功能使用如下的if语句检测:

if (^MyReg === 1'bx)

$display ("There is an unknown in the vector MyReg !")

注意逻辑相等(==)操作符不能用于比较；逻辑相等操作符比较将只会产生结果x。全等操作符期望的结果为值1。

7、 移位操作符

移位操作符有:

• << (左移)

• >> (右移)

移位操作符左侧操作数移动右侧操作数表示的次数，它是一个逻辑移位。空闲位添0补位。如果右侧操作数的值为x或z, 移位操作的结果为x。假定:

reg [0：7] Qreg;

. . .

Qreg = 4'b0111;

那么:Qreg >> 2 是8 'b0000_0001

Verilog HDL中没有指数操作符。但是，移位操作符可用于支持部分指数操作。例如，如果要计算ZNumBits的值，可以使用移位操作实现，例如:

32'b1 << NumBits //NumBits必须小于3 2。

同理，可使用移位操作为2 - 4解码器建模，如

wire [0:3] DecodeOut = 4'b1 << Address[0:1] ;

Address[0:1] 可取值0,1,2和3。与之相应，DecodeOut可以取值4 ' b 0 0 0 1、4 ' b 0 0 1 0、4 ' b 0 1 0 0 和4 ' b 1 0 0 0，从而为解码器建模。

8、 条件操作符

条件操作符根据条件表达式的值选择表达式，形式如下:

cond_expr?expr1:expr2

如果cond_expr为真(即值为1)，选择expr1；如果cond_expr为假(值为0)，选择expr2。如果cond_expr为x或z，结果将是按以下逻辑expr1和expr2按位操作的值：0与0得0，1与1得1，其余情况为x。如下所示：

wire [0:2] Student = Marks > 18?Grade_A:Grade_C;

计算表达式Marks>18；如果真，Grade_A 赋值为Student；如果Marks < =18，Grade_C赋值为Student。下面为另一实例：

always

#5Ctr = (Ctr!=25)?(Ctr+1):5;

过程赋值中的表达式表明如果Ctr不等于25，则加1；否则如果Ctr值为25时，将Ctr值重新置为5。

9、 连接和复制操作

连接操作是将小表达式合并形成大表达式的操作。形式如下：

{expr1,expr2, . . ., exprN}

实例如下所示：

wire [7:0] Dbus;

wire [11:0] Abus;

assign Dbus [7:4] = {Dbus[0],Dbus[1],Dbus[2],Dbus[3]} ;

//以反转的顺序将低端4位赋给高端4位。

assign Dbus = {Dbus [3:0], Dbus[7:4]} ;//高4位与低4位交换。

由于非定长常数的长度未知，不允许连接非定长常数。例如，下列式子非法：

{D b u s,5} //不允许连接操作非定长常数。

复制通过指定重复次数来执行操作。形式如下：

{repetition_number {expr1, expr2, ...,exprN}}

以下是一些实例：

Abus = {3{4'b1011}}; //位向量12 'b1011_1011_1011)

Abus = {{4{Dbus[7]}},Dbus}; /*符号扩展*/

{3{1'b1}} 结果为111

{3{Ack}} 结果与{Ack,Ack,Ack}相同。

4.3 表达式种类

常量表达式是在编译时就计算出常数值的表达式。通常，常量表达式可由下列要素构成：

1) 表示常量文字，如'b10和326。

2) 参数名，如RED的参数表明：

parameter RED = 4'b1110;

标量表达式是计算结果为1位的表达式。如果希望产生标量结果，但是表达式产生的

结果为向量，则最终结果为向量最右侧的位值。

习题

1. 说明参数GATE_DELAY，参数值为5。

2. 假定长度为64个字的存储器，每个字8位，编写Verilog 代码，按逆序交换存储器的内容。即将第0个字与第63个字交换，第1个字与第62个字交换，依此类推。

3. 假定32位总线Address_Bus，编写一个表达式，计算从第11位到第20位的归约与非。

4. 假定一条总线Control_Bus[15:0],编写赋值语句将总线分为两条总线：Abus[0:9]和Bbus[6:1]。

5. 编写一个表达式，执行算术移位，将Qparity 中包含的8位有符号数算术移位。

6. 使用条件操作符，编写赋值语句选择NextState的值。如果CurrentState的值为RESE T,那么NextState的值为GO；如果CurrentState的值为GO，则NextState 的值为BUSY；如果CurrentState的值为BUSY；则NextState的值为RESET。

7. 使用单一连续赋值语句为图2-2所示的2-4解码器电路的行为建模。[提示：使用移位操作符、条件操作符和连接操作符。]

8. 如何从标量变量A，B，C和D中产生总线Bus Q[0:3]？如何从两条总线Bus A[0 : 3]和Bus Y[20:15]形成新的总线Bus R[10:1]?

第 5 章　门电平模型化

本章讲述Verilog HDL为门级电路建模的能力，包括可以使用的内置基本门和如何使用它们来进行硬件描述。

5.1 内置基本门

Verilog HDL中提供下列内置基本门：

1) 多输入门：

and,nand,or,nor,xor,xnor

2) 多输出门：

buf,not

3) 三态门：

bufif0, bufif1, notif0,notif1

4) 上拉、下拉电阻：

pullup, pulldown

5) MOS开关：

cmos, nmos, pmos, rcmos, rnmos, rpmos

6) 双向开关：

tran,tranif0, tranif1, rtran, rtranif0, rtranif1

门级逻辑设计描述中可使用具体的门实例语句。下面是简单的门实例语句的格式。

gate_type[instance_name](term1,term2, . . . ,termN) ;

注意，instance_name是可选的；gate_type为前面列出的某种门类型。各term用于表示与门的输入/输出端口相连的线网或寄存器。同一门类型的多个实例能够在一个结构形式中定义。语法如下:gate_type[instance_name1](term11,term12, . . .,term1N) ,

[instance_name2](term21, term22, . . .,term2N) ,

. . .

[instance_nameM](termM1, termM2, . . .,termMN

5.2 多输入门

内置的多输入门如下:

and nand nor or xor xnor

这些逻辑门只有单个输出， 1个或多个输入。多输入门实例语句的语法如下:

multiple_input_gate_type[instance_name](OutputA,Input1,Input2,...,InputN) ;

第一个端口是输出，其它端口是输入。如图5 - 1所示。

图5-1 多输入门

下面是几个具体实例。图5-2为对应的逻辑图。

```
and A1(Out1, In1, In2) ;
and RBX (Sty, Rib, Bro, Qit, Fix);
xor (Bar, Bud[0] ,Bud[1], Bud[2]),
(Car, Cut[0], Cut[1]) ,
(Sar, Sut[2], Sut[1], Sut[0], Sut[3]) ;
```

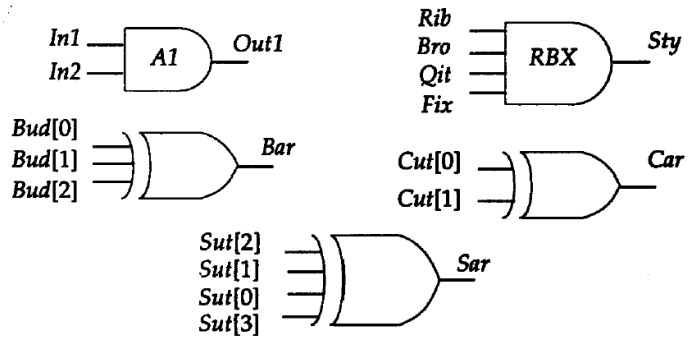


图5-2 多输入门实例

第一个门实例语句是单元名为A1、输出为Out1、并带有两个输入In1和In2的两输入与门。第二个门实例语句是四输入与门，单元名为RBX，输出为Sty，4个输入为Rib、Bro、Qit和Fix。第三个门实例语句是异或门的具体实例，没有单元名。它的输出是Bar，三个输入分别为Bud[0]、Bud[1]和Bud[2]。同时，这一个实例语句中还有两个相同类型的单元。

下面是这些门的真值表。注意在输入端的z与对x的处理方式相同；多输入门的输出决不能是z。

nand	0	1	x	z
0	1	1	1	1
1	1	0	x	x
x	1	x	x	x
z	1	x	x	x

and	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

or	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

nor	0	1	x	z
0	1	0	x	x
1	0	0	0	0
x	x	0	x	x
z	x	0	x	x

xor	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

xnor	0	1	x	z
0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

5.3 多输出门

多输出门有:buf not这些门都只有单个输入，一个或多个输出。如图5 - 3所示。这些门的实例语句的基本语法如下:

multiple_output_gate_type [instance_name](Out1, Out2, . . . OutN ,InputA);

最后的端口是输入端口，其余的所有端口为输出端口。

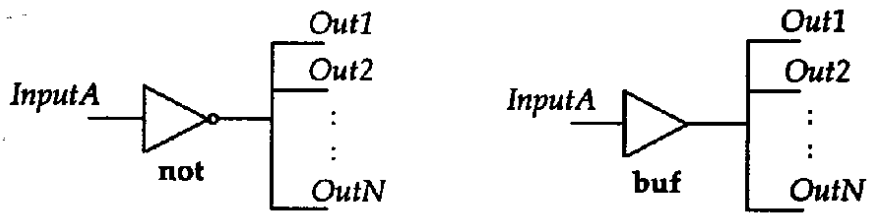


图5-3 多输出门

[instance_name](Out1,Out2, . . . OutN ,InputA) ;

最后的端口是输入端口，其余的所有端口为输出端口。例如:

bufB1（Fan[0]，Fan[1]，Fan[2]，Fan[3]，Clk）；

notN1（PhA，PhB，Ready）；

在第一个门实例语句中，Clk是缓冲门的输入。门B1有4个输出：Fan[0]到Fan[3]。在第二个门实例语句中，Ready是非门的唯一输入端口。门N1有两个输出：PhA和PhB。

这些门的真值表如下:

buf	0	1	x	z
(输出)	0	1	x	x

not	0	1	x	z
(输出)	1	0	x	x

5.4 三态门

三态门有:bufif0 bufif1 notif0 notif1

这些门用于对三态驱动器建模。这些门有一个输出、一个数据输入和一个控制输入。三态门实例语句的基本语法如下：

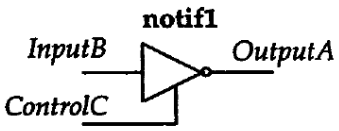
tristate_gate[instance_name](OutputA,InputB,ControlC) ；

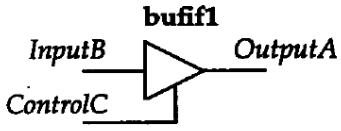
第一个端口Output A是输出端口，第二个端口Input B是数据输入，Control C是控制输入。参见图5-4。根据控制输入，输出可被驱动到高阻状态，即值z。对于bufif0，若通过控制输入为1，则输出为z；否则数据被传输至输出端。对于bufif1，若控制输入为0，则输出为z。对于notif0，如果控制输出为1，那么输出为z；否则输入数据值的非传输到输出端。对于notif1，若控制输入为0；则输出为z。例如:

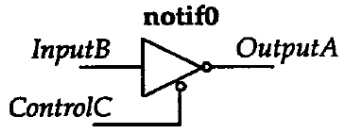
bufif1 BF1（Dbus，MemData，Strobe）；

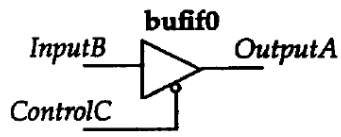
notif0NT2 (Addr, Abus, Probe) ；

当Strobe为0时，bufif1门BF1驱动输出Dbus为高阻；否则MemData被传输至Dbus。在第2个实例语句中，当Probe为1时，Addr为高阻；否则Abus的非传输到Addr。









下面是这些门的真值表。表中的某些项是可选项。例如，0/z表明输出根据数据的信号强度和控制值既可以为0也可以为z。

bufif0		控制			
		0	1	x	z
数据	0	0	z	0/z	0/z
	1	1	z	1/z	1/z
	x	x	z	x	x
	z	x	z	x	x

bufif1		控制			
		0	1	x	z
数据	0	z	0	0/z	0/z
	1	z	1	1/z	1/z
	x	z	x	x	x
	z	z	x	x	x

notif0		控制			
		0	1	x	z
数据	0	1	z	1/z	1/z
	1	0	z	0/z	0/z
	x	x	z	x	x
	z	x	z	x	x

notif1		控制			
		0	1	x	z
数据	0	z	1	1/z	1/z
	1	z	0	0/z	0/z
	x	z	x	x	x
	z	z	x	x	x

5.5 上拉、下拉电阻

上拉、下拉电阻有：pullup pulldown这类门设备没有输入只有输出。上拉电阻将输出置为1。下拉电阻将输出置为0。门实例语句形式如下：

pull_gate[instance_name](OutputA);门实例的端口表只包含1个输出。例如：

pullup PUP(Pwr) ;

此上拉电阻实例名为PUP，输出Pwr置为高电平1。

5.6 MOS 开关

MOS开关有：cmos pmos nmos rcmos rpmos rnmos这类门用来为单向开关建模。即数据从输入流向输出，并且可以通过设置合适的控制输入关闭数据流。pmos(p类型MOS管)、nmos(n类型MOS管)，rnmos(r代表电阻)和rpmos开关有一个输出、一个输入和一个控制输入。实例的基本语法如下：gate_type[instance_name](OutputA,InputB, ControlC) ；第一个端口为输出，第二个端口是输入，第三个端口是控制输入端。如果nmos和rnmos开关的控制输入为0，pmos和rpmos开关的控制为1，那么开关关闭，即输出为z；如果控制是1，输入数据传输至输出；如图5-5所示。与nmos和pmos相比，rnmos和rpmos在输入引线和输出引线之间存在高阻抗(电阻)。因此当数据从输入传输至输出时，对于rpmos和rmos，

存在数据信号强度衰减。

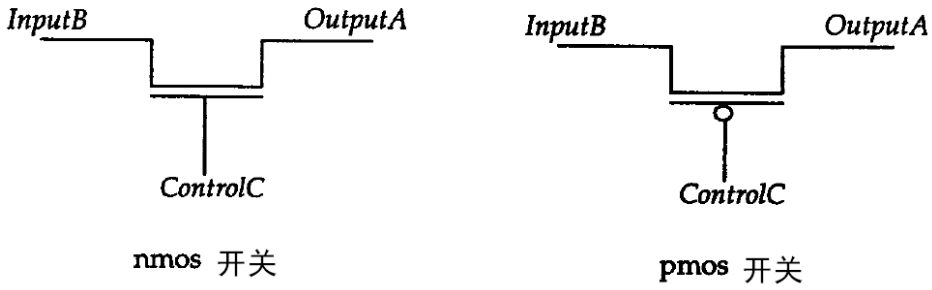


图5-5 nmos和pmos开关

例如：

```
Pmos P1 (BigBus, SmallBus, GateControl ) ;
rnmos RN1 (ControlBit, ReadyBit, Hold ) ;
```

第一个实例为一个实例名为P1 的pmos开关。开关的输入为SmallBus，输出为BigBus，控制信号为GateControl。这些开关的真值表如下所示。表中的某些项是可选项。例如，1/z表明，根据输入和控制信号的强度，输出既可以为1，也可以为z。

pmos rpmos		控制			
		0	1	x	z
数据	0	0	z	0/z	0/z
	1	1	z	1/z	1/z
	x	x	z	x	x
	z	z	z	z	z

nmos rnmos		控制			
		0	1	x	z
数据	0	z	0	0/z	0/z
	1	z	1	1/z	1/z
	x	z	x	x	x
	z	z	z	z	z

cmos(mos求补)和rcmos(cmos的高阻态版本)开关有一个数据输出，一个数据输入和两个控制输入。这两个开关实例语句的语法形式如下：

```
(r)cmos [instance_name](OutputA, InputB, NControl, PControl);
```

第一个端口为输出端口，第二个端口为输入端口，第三个端口为n通道控制输入，第四个端口为是P通道控制输入。cmos(rcmos)开关行为与带有公共输入、输出的pmos(rpmos)和nmos(rnmos)开关组合十分相似。参见图5-6。

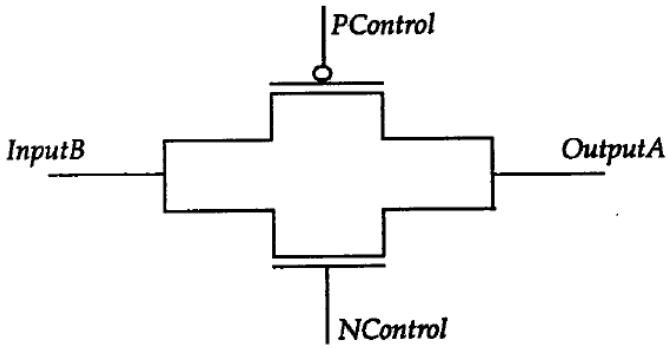


图5-6 (r)cmos开关

5.7 双向开关

双向开关有：tran rtran tranif0 rtranif0 tranif1 rtranif1这些开关是双向的，即数据可以双向流动，并且当数据在开关中传播时没有延时。后4个开关能够通过设置合适的控制信号来关闭。tran和rtran开关不能被关闭。tran或rtran(tran的高阻态版本)开关实例语句的语法如下：(r)tran[instance_name](SignalA, SignalB);端口表只有两个端口，并且无条件地双向流动，即从SignalA向SignalB，反之亦然。其它双向开关的实例语句的语法如下：gate_type[instance_name] (SignalA, SignalB, ControlC) ;
前两个端口是双向端口，即数据从SignalA流向SignalB，反之亦然。第三个端口是控制信号。如果对tranif0和tranif0，ControlC是1；对tranif1和rtranif1，ControlC是0；那么禁止双向数据流动。对于rtran、rtranif0和rtranif1，当信号通过开关传输时，信号强度减弱。

5.8 门时延

可以使用门时延定义门从任何输入到其输出的信号传输时延。门时延可以在门自身实例语句中定义。带有时延定义的门实例语句的语法如下：

gate_type[delay][instance_name](terminal_list) ;

时延规定了门时延，即从门的任意输入到输出的传输时延。当没有强调门时延时，缺省的时延值为0。门时延由三类时延值组成：

1) 上升时延
2) 下降时延
3) 截止时延

门时延定义可以包含0个、1个、2个或3个时延值。下表为不同个数时延值说明条件下，各种具体的时延取值情形。

	无时延	1个时延(d)	2个时延(d1, d2)	3个时延 (dA, dB, dC)
上升	0	d	d1	dA
下降	0	d	d2	dB
to_x	0	d	min① (d1, d2)	min (dA, dB, dC)
截止	0	d	min (d1, d2)	dC

① m i n 是m i n i m u m 的缩写词。

注意转换到x的时延(to_x)不但被显式地定义，还可以通过其它定义的值决定。下面是一些具体实例。注意Verilog HDL模型中的所有时延都以单位时间表示。单位时间与实际时间的关联可以通过`timescale编译器指令实现。在下面的实例中,notN1(Qbar, Q);因为没有定义时延，门时延为0。下面的门实例中，nand #6 (Out, In1, In2) ;所有时延均为6，即上升时延和下降时延都是6。因为输出决不会是高阻态，截止时延不适用于与非门。转换到x的时延也是6。and #(3,5) (Out, In1, In2, In3);在这个实例中，上升时延被定义为3，下降时延为5，转换到x的时延是3和5中间的最小值，即3。在下面的实例中，

```
notif1 #(2,8,6) ( Dout, Din1, Din2);
```

上升时延为2，下降时延为8，截止时延为6，转换到x的时延是2、8和6中的最小值，即2。对多输入门（例如与门和非门）和多输出门(缓冲门和非门)总共只能够定义2个时延(因为输出决不会是z)。三态门共有3个时延，并且上拉、下拉电阻实例门不能有任何时延。min:typ:max时延形式门延迟也可采用min:typ:max形式定义。形式如下：minimum: typical: maximum最小值、典型值和最大值必须是常数表达式。下面是在实例中使用这种形式的实例。nand #(2:3:4, 5:6:7) (Pout, Pin1, Pin2);选择使用哪种时延通常作为模拟运行中的一个选项。例如，如果执行最大时延模拟，与非门单元使用上升时延4和下降时延7。程序块也能够定义门时延。

5.9 实例数组

当需要重复性的实例时，在实例描述语句中能够有选择地定义范围说明(范围说明也能够在模块实例语句中使用)。这种情况的门描述语句的语法如下：

```
gate_type
[delay]instance_name[leftbound:rightbound](list_of_terminal_names);
```

leftbound和rightbound值是任意的两个常量表达式。左界不必大于右界，并且左、右界两者都不必限定为0。示例如下。

```
wire [3:0] Out, InA, InB;
. . .
Nand Gang [3:0] (Out, InA, InB) ;带有范围说明的实例语句与下述语句等价：
nand
Gang3 (Out[3],InA[3],InB[3]),
```

```
Gang2 (Out[2],InA[2],InB[2]),
Gang1 (Out[1],InA[1],InB[1]),
Gang0 (Out[0],InA[0],InB[0]);
```

注意定义实例数组时，实例名称是不可选的。

5.10 隐式线网

如果在Verilog HDL模型中一个线网没有被特别说明，那么它被缺省声明为1位线网。但是`default_nettype编译指令能够用于取代缺省线网类型。编译指令格式如下：

```
`default_nettype net_type
```

例如：

```
`default_nettype wand
```

根据此编译指令，所有后续未说明的线网都是wand类型。`default_nettype编译指令在模块定义外出现，并且在下一个相同编译指令或`resetall编译指令出现前一直有效。

5.11 简单示例

下面是图5-7中4-1多路选择电路的门级描述。注意因为实例名是可选的(除用于实例数组情况外)，在门实例语句中没有指定实例名。

```
module MUX4x1 (Z , D0 , D1 , D2 , D3 , S0 , S1) ;
output Z;
input D0 , D1 , D2 , D3 , S0 , S1;
and (T0 , D0 , S0bar , S1bar) ,
(T1 , D1 , S0bar , S1) ,
(T2 , D2 , S0 , S1bar),
(T3 , D3 , S0 , S1),
not (S0bar , S0),
(S1bar , S1);
Or (Z , T0 , T1 , T2 , T3 );
endmodule
```

如果或门实例由下列的实例代替呢?

```
Or Z (Z , T0 , T1 , T2 , T3); //非法的Verilog HDL表达式。
```

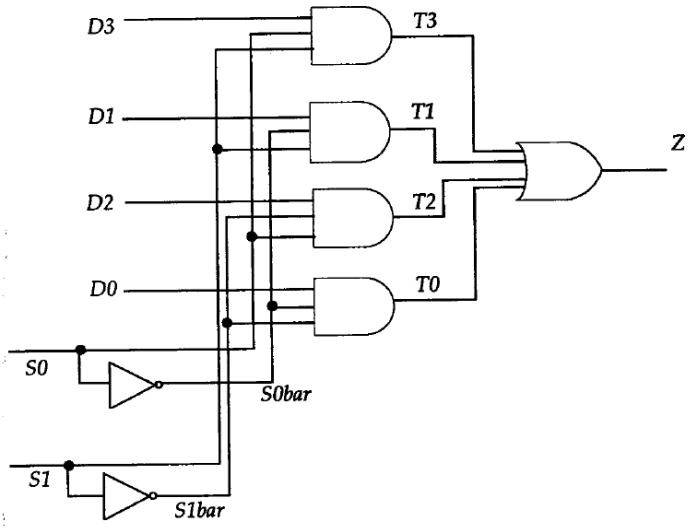


图5-7 4-1多路选择电路

注意实例名还是Z，并且连接到实例输出的线网也是Z。这种情况在Verilog HDL中是不允许的。在同一模块中，实例名不能与线网名相同。

5.12 2-4 解码器举例

图5-8中显示的2-4解码器电路的门级描述如下：

```
module DEC2×4 (A , B , Enable , Z);
input A , B , Enable;
output [0:3] Z;
wire Abar, Bbar;
not # ( 1 , 2 )
V0 (Abar , A) ,
V1 (Bbar, B);
nand # (4,3)
N0 (Z[3], Enable, A,B) ,
N1 (Z[0], Enable, Abar,Bbar) ,
N2 (Z[1], Enable, Abar,B) ,
N3 (Z[2], Enable, A,Bbar) ,
endmodule
```

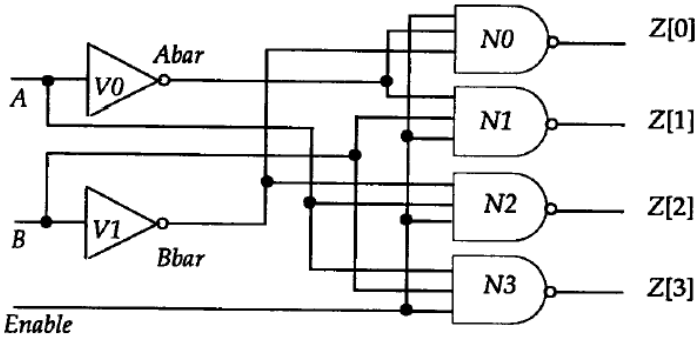


图5-8 2-4解码器电路

5.13 主从触发器举例

图5-9所示的主从D触发器的门级描述如下：

```
module MSDFF (D , C , Q , Q b a r) ;
input D , C ;
output Q , Qbar;
not
NT1 (Not D , D) ,
NT2 (Not C , C) ,
NT3 (Not Y , Y) ;
nand
ND1 (D1 , D , C) ,
ND2 (D2 , C , NotD) ,
ND3 (Y , D1 , Ybar) ,
ND4 (Ybar , Y , D2) ,
ND5 (Y1 , Y , NotC) ,
ND6 (Y2 , NotY , NotC) ,
ND7 (Q , Qbar , Y1) ,
ND8 (Qbar , Y2 , Q) ;
endmodule
```

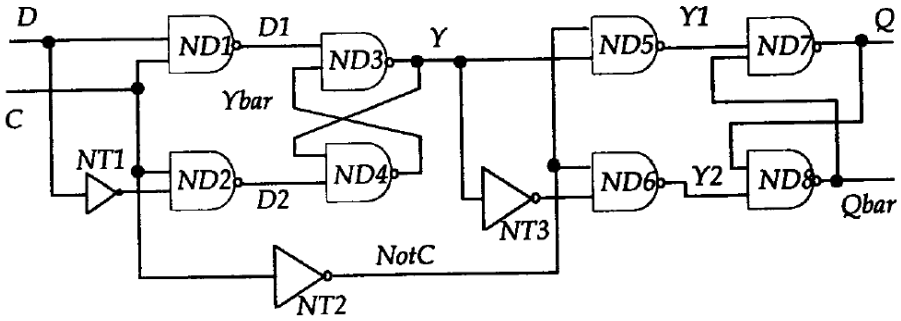


图5-9 主从触发器

5.14 奇偶电路

图5-10所示的9位奇偶发生器门级模型描述如下：

```
module Parity_9_Bit (D, Even,Odd) ;
input [0:8] D ;
output Even, Odd;
xor # ( 5 , 4 )
XE0 (E0 , D[0] ,D[1]) ,
XE1 (E1 , D[2] ,D[3]) ,
XE2 (E2 , D[4] ,D[5]) ,
XE3 (E3 , D[6] ,D[7]) ,
XF0 (F0 , E0 , E1) ,
XF1 (F1 , E2 , E3) ,
XH0 (H0 , F0 , F1) ,
XEVEN (Even, D[8], H0) ;
not #2
XODD (Odd, Even) ;
endmodule
```

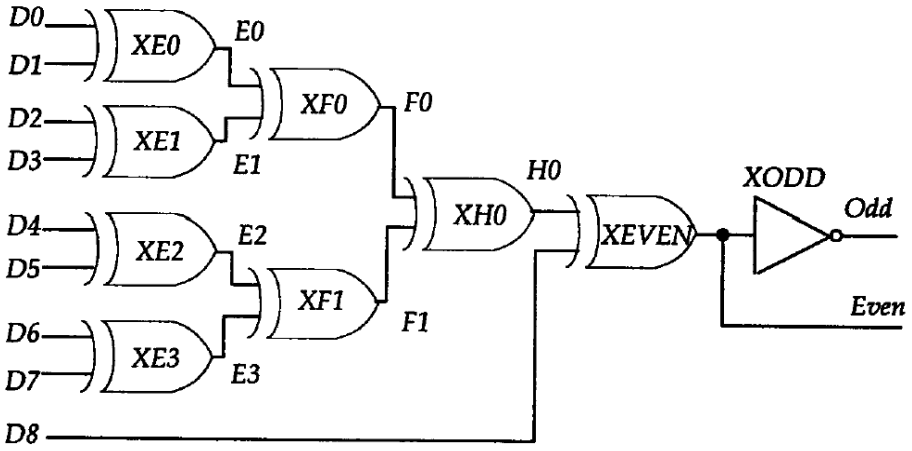


图5-10 奇偶发生器

习题

1. 用基本门描述图5-11显示的电路模型。编写一个测试验证程序用于测试电路的输出。使用所有可能的输入值对电路进行测试。

2. 使用基本门描述如图5-12所示的优先编码器电路模型。当所有输入为0时，输出Valid为0，否则输出为1。并且为验证优先编码器的模型行为编写测试验证程序。

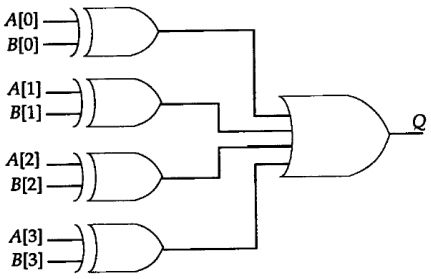


图5 - 11 A不等于B的逻辑

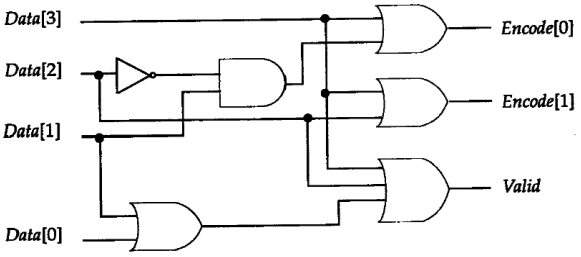


图5-12 优先编码器

第 6 章　用户定义的原语

在前一章中，我们介绍了Verilog HDL提供的内置基本门。本章讲述Verilog HDL指定用户定义原语UDP的能力。UDP的实例语句与基本门的实例语句完全相同，即UDP实例语句的语法与基本门的实例语句语法一致。

6.1 UDP 的定义

使用具有如下语法的UDP说明定义UDP。

```
primitive UDP_name (OutputName, List_of_inputs )
Output_declaration
List_of_input_declarations
[Reg_declaration]
[Initial_statement]
table
List_of_tabel_entries
endtable
endprimitive
```

UDP的定义不依赖于模块定义，因此出现在模块定义以外。也可以在单独的文本文件中定义UDP。UDP只能有一个输出和一个或多个输入。第一个端口必须是输出端口。此外，输出可以取值0、1或x(不允许取z值)。输入中出现值z以x处理。UDP的行为以表的形式描述。在UDP中可以描述下面两类行为：

1) 组合电路

2) 时序电路(边沿触发和电平触发)

6.2 组合电路 UDP

在组合电路UDP中，表规定了不同的输入组合和相对应的输出值。没有指定的任意组合输出为x。下面以2-1多路选择器为例加以说明。

```
primitive MUX2x1 (Z, Hab, Bay, Sel) ;
output Z;
input Hab,Bay, Sel;
table
```

```
// Hab Bay Sel : Z 注：本行仅作为注释。
0 ? 1 : 0 ;
1 ? 1 : 1 ;
? 0 0 : 0 ;
? 1 0 : 1 ;
0 0 x : 0 ;
1 1 x : 1 ;
endtable
endprimitive
```

字符?代表不必关心相应变量的具体值，即它可以是0、1或x。输入端口的次序必须与表中各项的次序匹配，即表中的第一列对应于原语端口队列的第一个输入(例子中为Hab)，第二列是Bay，第三列是Sel。在多路选择器的表中没有输入组合0 1x项(还有其它一些项)；在这种情况下，输出的缺省值为x(对其它未定义的项也是如此)。

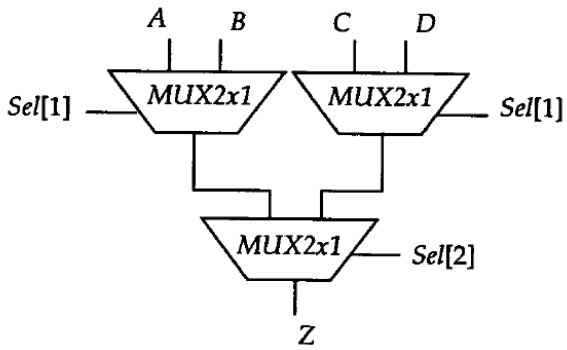


图6-1 使用UDP构造的4-1多路选择器

图6-1为使用2-1多路选择器原语组成的4-1多路选择器的示例。

```
module MUX4x1 (Z, A, B, C, D, Sel ) ;
input A, B, C, D;
input [2:1] Sel;
output Z;
parameter tRISE = 2, tFALL = 3;
MUX2x1 #(tRISE, tFALL)
(TL, A, B, Sel[1]) ,
(TP, C, D, Sel[1]) ,
(Z, TL, TP, Sel[2]) ;
endmodule
```

如上例所示，在UDP实例中，总共可以指定2个时延，这是由于UDP的输出可以取值0、1或x(无截止时延)。

6.3 时序电路 UDP

在时序电路UDP中，使用1位寄存器描述内部状态。该寄存器的值是时序电路UDP的输出值。共有两种不同类型的时序电路UDP：一种模拟电平触发行为；另一种模拟边沿触发行为。时序电路UDP使用寄存器当前值和输入值决定寄存器的下一状态(和后继的输出)。

1、初始化状态寄存器

时序电路UDP的状态初始化可以使用带有一条过程赋值语句的初始化语句实现。形式如下：initial reg_name = 0,1,or x;

初始化语句在UDP定义中出现。

2、电平触发的时序电路UDP

下面是D锁存器建模的电平触发的时序电路UDP示例。只要时钟为低电平0，数据就从输入传递到输出；否则输出值被锁存。

```
primitive Latch (Q, Clk, D) ;
output Q;
reg Q;
input Clk, D;
table
// Clk D Q(State) Q( n e x t )
0 1 : ? : 1 ;
0 0 : ? : 0 ;
1 ? : ? : - ;
endtable
endprimitive
```

“-”字符表示值“无变化”。注意UDP的状态存储在寄存器D中。

3、边沿触发的时序电路UDP

下例用边沿触发时序电路U D P为D边沿触发触发器建模。初始化语句用于初始化触发器的状态。

```
Primitive D_Edge_FF (Q, Clk, Data) ;
output Q ;
reg Q ;
input Data, Clk;
initial Q = 0;
table
// Clk Data Q (State) Q( n e x t )
(01) 0 : ? : 0 ;
(01) 1 : ? : 1 ;
(0x) 1 : 1 : 1 ;
```

```
(0x) 0 : 0 : 0 ;
// 忽略时钟负边沿：
(?0) ? : ? : - ;
// 忽略在稳定时钟上的数据变化:
? (??): ? : - ;
endtable
endprimitive
```

表项(01)表示从0转换到1，表项(0 x)表示从0转换到x，表项(? 0)表示从任意值(0 , 1或x)转换到0，表项(? ?)表示任意转换。对任意未定义的转换，输出缺省为x。

假定D_Edge_FF为UDP定义，它现在就能够象基本门一样在模块中使用，如下面的4位寄存器所示。

```
module Reg4 (Clk, Din, Dout) ;
input Clk ;
input [0:3] Din;
output [0:3] Dout;
D_Edge_FF
DLAB0 (Dout[0],Clk, Din[0]),
DLAB1 (Dout[1],Clk, Din[1]),
DLAB2 (Dout[2],Clk, Din[2]),
DLAB3 (Dout[3],Clk, Din[3]),
endmodule
```

4、边沿触发和电平触发的混合行为

在同一个表中能够混合电平触发和边沿触发项。在这种情况下，边沿变化在电平触发之前处理，即电平触发项覆盖边沿触发项。下例是带异步清空的D触发器的UDP描述。

```
 D_Async_FF (Q, Clk, Clr, Data) ;
output Q;
reg Q;
input Clr, Data, Clk;
table
// Clk Clr Data Q (State) Q( n e x t )
(01) 0 0 : ? : 0 ;
(01) 0 1 : ? : 1 ;
(0x) 0 1 : 1 : 1 ;
(0x) 0 0 : 0 : 0 ;
// 忽略时钟负边沿：
(?0) 0 ? : ? : - ;
(??) 1 ? : ? : 0 ;
?  1 ? : ? : 0;
endtable
```

```
endprimitive
```

6.4 另一实例

下面是3位表决电路的U D P描述。如果输入向量中存在2个或更多的1，则输出为1。

```
primitive Majority3(Z, A, B, C) ;
input A, B, C;
output Z
table
//A B C : Z
0 0 ? : 0 ;
0 ? 0 : 0 ;
? 0 0 : 0 ;
1 1 ? : 1 ;
1 ? 1 : 1 ;
? 1 1 : 1 ;
endtable
endprimitive
```

6.5 表项汇总

出于完整性考虑，下表列出了所有能够用于U D P原语中表项的可能值。

符　号	意　义	符　号	意　义
0	逻辑0	(AB)	由A变到B
1	逻辑1	*	与(??)相同
x	未知的值	r	上跳变沿，与(01)相同
?	0、1或**x**中的任一个	f	下跳变沿，与(10)相同
b	0或1中任选一个	p	(01)、(0**x**)和(**x**1)的任一种
－	输出保持	n	(10)、(1**x**)和(**x**0)的任一种

习题

1. 组合电路UDP与时序电路UDP如何区别?
2. UDP可有一个或多个输出，是否正确?
3. 初始语句可用于初始化组合电路UDP吗?
4. 为图5-12中显示的优先编码器电路编写UDP描述。使用测试激励验证描述的模型。
5. 为T触发器编写UDP描述。在T触发器中，如果数据输入为0，则输出不变化。如果数据输入是1，那么输出在每个时钟沿翻转。假定触发时钟沿是时钟下跳沿，使用测试激励验

证所描述的模型。

6. 以UDP方式为上跳边沿触发的J K触发器建模。如果J和K两个输入均为0，则输出不变。如果J为0，K为1，则输出为0。如果J是1，K是0，则输出是1。如果J和K都是1，则输出翻转。使用测试激励验证描述的模型。