迈航科技

单片机那些事儿

初级篇——单片机概述

残弈悟恩

2014

官方淘宝店铺：HTTP://SHOP109195762.TAOBAO.COM

郑重声明

本资料以残弈悟恩开发的 MGMC-V1.0 实验板为硬件平台，以残弈悟恩编写的《深入浅出玩转 51 单片机》为辅助教程，以残弈悟恩录制的《31 天环游单片机》为基础视频。

本资料以个人学习和工作的经验为素材，以单片机初学者、单片机项目开发者为对象。教大家如何走进单片机的世界，如何开发工程项目。限于时间和水平关系，资料中难免有过失之处，望各位高手批评指教，多多拍砖，拍累了，你们休息，我继续上路。

版本：20140401（V1.0）　　制造者：残弈悟恩

让爱充满大地——花 1 秒时间，拯救 1 个人，传递 1 份爱

声明：只是残弈悟恩爱心的喷发，我得不到一分钱，各位不要多想，谢谢！

你知道吗？在非洲北边的某个地区，每一秒都有许许多多的人正在挨饿，每一天至少有一位儿童死于营养不足。你的一次点击就能让某位穷人得到 1.1 杯食物。当然你可以不相信有这样的链接或者是骗点击什么的。事实上，网站确实是帮穷人得 1.1 杯食物的，只要你点进去单机一下中间的黄色按钮，就会出来一系列介绍各种商品的网页（绝对免费的并且不会下载任何软件，也不会有电脑病毒），同时也会有人因为您的一次点击而得到 1.1 杯食物，食物是由商家提拱的，但爱心却是您献出的。如果你觉得残弈悟恩在忽悠大家，你不妨可以在网上查一下是真与假。

看到这本资料的朋友多数都是电子爱好者、单片机初学者，或者干电子这一行的，管你穷学生还是穷工人，只要能上网，只要愿花一秒种就可以了。人生在世，有两件事不能等：一、孝顺；二、行善。无论你是 LED 小灯、普通灯泡也好，还是荧光灯也吧，最重要就是要懂得用自身的光去照耀别人，光的强度并不重要。

点击链接：

http://www.thehungersite.com/clickToGive/home.faces?siteId=1&link=ctg_ths_home_from_ths_thankyou_sitenav

第三章　单片机概述

无论怎么样讲述单片机，首先必须得从应用讲起，否则有人又会用轻蔑的口吻为我：饿了，单片机能充饥吗？我会热情的回答一句：这个要看对象，选对的人，做对的事。

3.1 高谈阔论单片机的应用

若直接端上来一盘单片机的菜肴，大家或许会觉得有点陌生。那就先来说说大家熟悉的电脑。相信看到此资料的每位都用过电脑吧，这不是纸质的书籍，要看到，肯定得用电脑，没有质疑。就如下面美丽的一幅图画（如图 3-1 所示），必经电脑合成处理，否则美女怎么能坐到电脑上看书呢？

图 3-1 用电脑 PS 的美图

我们在学电脑时，并不明确学完有何用途，但当学完了之后，才发现用电脑可以看电影、玩游戏、上网、聊天、编程、PS 图片等。这时你用电脑做什么，你并不明确，以后按工作所需，你可能需要用电脑画图、编程、做文字处理等。毫无疑问，电脑只是你工作、学习的一种工具罢了。

说完了电脑，在说单片机。单片机渗透在我们生活的各个领域，几乎很难找到哪个领域没有单片机的踪迹。导弹的导航装置，飞机上各种仪表的控制，工业自动化过程的实时控制和数据处理，广泛使用的各种智能 IC 卡，民用豪华轿车的安全保障系统，录像机、摄像机，全自动洗衣机的控制，以及程控玩具、电子宠物等，这些都离不开单片机。更不用说自动控制领域的机器人、智能仪表、医疗器械以及各种智能机械了。

单片机的应用之广泛，无法一一列举，这里大致列举几种，以便说明。

1. 智能仪器

单片机具有体积小、功耗低、控制功能强、扩展灵活、微型化和使用方便等优点，广泛应用于仪器仪表中，结合不同类型的传感器，可实现诸如电压、电流、功率、频率、湿度、温度、流量、速度、厚度、角度、长度、硬度、压力等物理量的测量。采用单片机控制使得仪器仪表数字化、智能化、微型化，且功能比起采用电子或数字电路更加强大。例如图 3-2 所示的数字电压表。

图 3-2 数字电压表示意图

2. 工业控制

单片机可以构成形式多样的控制系统、数据采集系统、通信系统、信号检测系统、无线感知系统、测控系统、机器人等应用控制系统。例如及当今非常流行的物联网系统，如图 3-3 所示。

图 3-3 家用物联网系统

3. 家用电器

家用电器广泛采用了单片机控制，从电饭煲、洗衣机、电冰箱、空调机、彩电，其他音响视频器材、电子秤、家庭影院（如图 3-4）。

图 3-4 家庭影院示意图

4. 网络和通信

现代的单片机普遍具备通信接口，可以很方便地与计算机进行数据通信，为在计算机网络和通信设备间的应用提供了极好的物质条件，通信设备基本上都实现了单片机智能控制，从手机、电话机、小型程控交换机、楼宇自动通信呼叫系统、列车无线通信，再到日常工作中随处可见的移动电话、 无线电对讲机（如图 3-5 所示）。

图 3-5 无线对讲机

5. 设备领域

图 3-6 呼吸机实物图

单片机在医用设备中的用途亦相当广泛，例如医用呼吸机（如图 3-6），各种分析仪，

监护仪，超声诊断设备及病床呼叫系统等。

此外，单片机在工商、金融、科研、教育、电力、通信、物流和国防航空航天等领域都有着十分广泛的用途。

看着上面一张张图画，虽美，但对于在校的穷学生来说，是不是觉得很飘渺呢？对于我们来说，不必考虑能否做出上面的一个个产品，只需把它当成一个工具罢了，再无需加太多的形容词。我们现在要做的就是，选择、坚持、学习它，至于以后的路，很长、变化很大，且行且继续追求好了。

3.2 解刨单片机

三步不出车,满盘皆是输。三页再不说单片机长啥样，或者是否有头、有尾、有内脏，那就是臭资料了。其实单片机即不是高富帅，也不是白富美，别激动。

3.2.1 单片机的外型图

一、单片机的封装

说道单片机的外型图，必需提到一个词——封装。封装概念不大，但是类型繁多，这里推荐大家去看看我的新浪博客，地址如下：

http://blog.sina.com.cn/s/blog_7b665eab01011bgr.html

其实现在主要在 EDNChina 和 ChinaAET 上写博客，新浪上没写过，这里不是为骗点击量，没意思，也需要。若果看完了上面的博文，那么接下来的这些就好理解了。单片机的封装形式种类繁多，这里只贴 PDIP40（如图 3-7 所示）、LQFP44（如图 3-8 所示）和 PLCC44（如图 3-9 所示）三种的样式，别的大家自行查阅。

图 3-7 PDIP40 的封装示意图

图 3-8 LQFP44 的封装示意图　　3-9 PLCC44 的封装示意图

接下来我们先不说每个引脚的含义，都后面我们连同最小系统的搭建一起讲述。有外形可以看出，无非都是一些黑不拉几的东西，在加无数条铁腿，那内部是什么东西呢，又是怎么工作的呢？那就带着这些问题，我们来看看它的五脏吧。

二、单片机的引脚功能图

这里我们再来说说各个引脚的定义，其中 LQFP44 和 PLCC44 的大家自行查阅官方数据手册，这里以 DIP40 的为例，其中管脚分布图如图 3-10 所示。

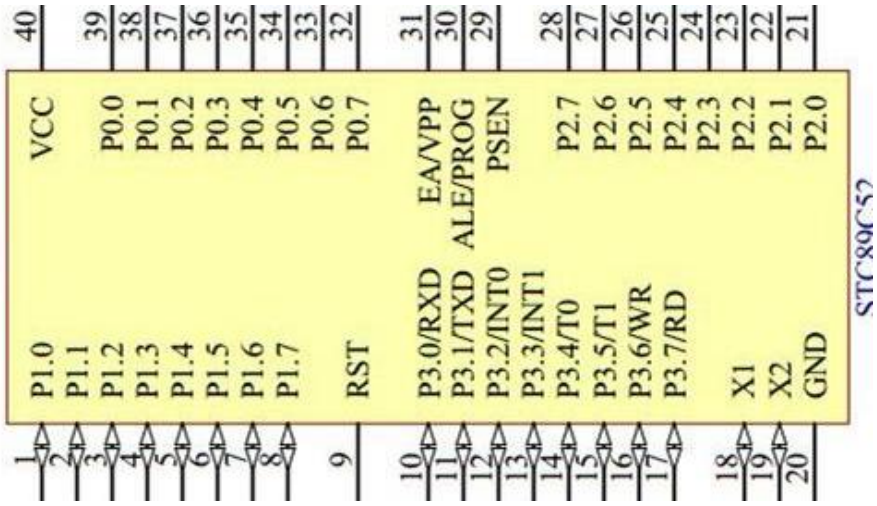

图 3-10 STC89C52 单片机（PDIP40）的引脚分布图

便于讲述和大家理解，这里将着 40 个管脚分为三类，分别为：

- 电源、时钟、复位管脚，如：VCC、GND、X1、X2、RST，必须掌握。
- 程序存储器的扩展管脚，如：EA/VPP、ALE/PROG、PSEN，**了解即可**。
- I/O 口引脚，如 P0、P1、P2、P3，必须掌握。

各个引脚的具体含义，我们在稍后讲述。

3.2.2 单片机的内脏

既然认识了单一的外表，是不是应该看看它的华丽内脏呢？那就先来张图，以便读者琢磨，内部结构如图 3-11 所示。曾记否，残弈悟恩刚开始学单片机时，买了一张“天价”（老板要了我 8 元，现在想起来都心疼）的 A3 纸，将其内部结构（我当初画的视乎笔这复杂多了）画到了纸上，之后折起来装在身上，有空拿出来看看，感觉还不错，最后直接贴在了宿舍墙上，让大家看，才是真的看。

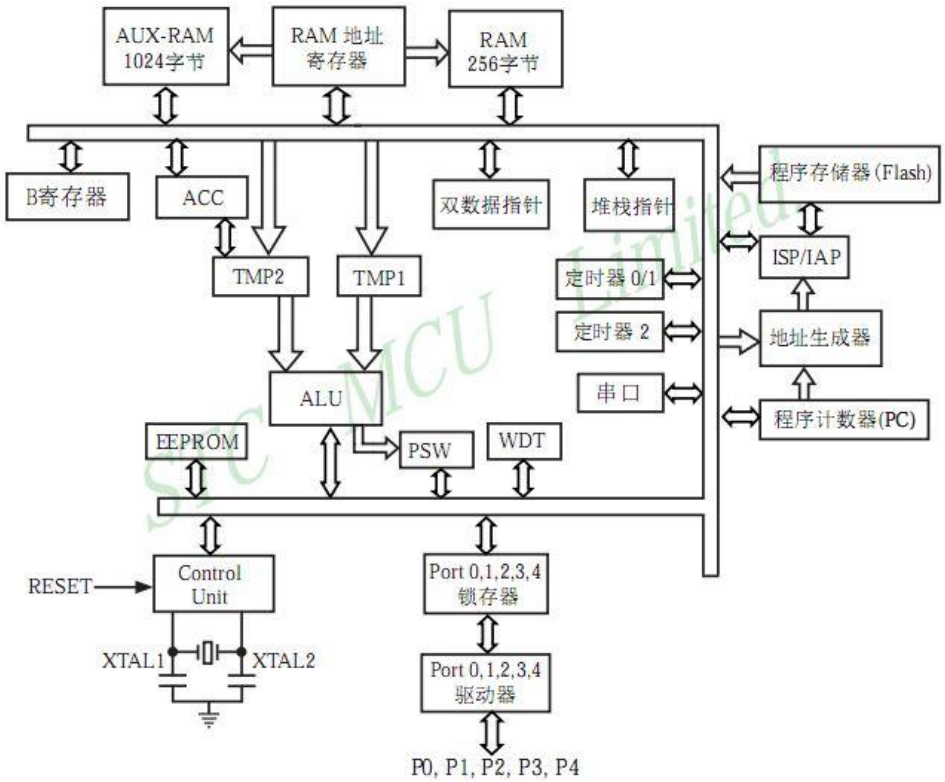

图 3-11 单片机内部结构图

由图可知，单片机主要包括以下 8 大部分：

（1）中央处理单元—CPU（8 位）。用于数据处理，位操作（置位、位清零）等。

（2）只读存储器—ROM（8K）。用于永久性存储应用程序。

（3）随机存取存储器—RAM（256B）。存储程序运行中的数据和变量。

（4）并行输入/输出口—I/O 口（32 线）。用作系统总线、扩展外存、外围设备的控制。

（5）串行输入/输出口—UART（2 线）。用于串口通信和一些串行芯片的扩展。

（6）定时/计时器—T/C（8、13、16 位自增量可编程）。它与 CPU 之间各自独立工作，当它记满时，CPU 会做出相应的动作（中断、置位）。

（7）时钟和复位电路。分为内部振荡器和外接振荡器。

（8）中断系统。5、6 个中断源，2 个优先级，可编程进行控制。

（5）、（6）、（8）我们以后慢慢讲述，这里先说说（1）～（4）、（7）五个部分吧。

一、单片机的大脑—CPU

单片机的 CPU 是完整的 8 位微计算机。这个 8 位微计算机包含 CPU、位寄存器、I/O 口和指令集。其中 CPU 内部包含：运算器、控制器、存储器。

1. 运算器

运算器包括：

（1）算术逻辑运算单元—ALU，主要功能有算术运算、逻辑运算。

（2）累加器—A，相当于数据加工厂。

（3）位处理器，主要进行位运算。

（4）BCD 码修正电路，主要运行十进制数的运算和处理。

（5）程序状态字寄存器—PSW，各个位的定义如表 3-1 所示。

表 3-1 程序状态寄存器

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name	CY	AC	F0	RS1	RS0	OV	F1	P
Address	D7H	D6H	D5H	D4H	D3H	D2H	D1H	D0H

该寄存器属于特殊功能寄存器，地址为：0xD0，可以位寻址。各个位的具体含义这里不赘，读者可以参考 STC 官方数据手册的第 45 页。

2. 控制器

单片机的指挥部件，主要任务是识别指令，控制各功能部件，保证各部分有序工作。主要包括指令寄存器、指令译码器、程序计数器、程序地址寄存器、条件转移逻辑电路和时序控制逻辑电路。

这些知识点读者需要结合汇编语言和微机原理来理解，这里就不详细介绍了，向大家推荐一本资料，名称为《精通 MCS-51 单片机　绝世秘籍》，讲述得还不错。

3. 存储器

52 单片机的存储器采用了哈佛结构。有一根地址和数据总线，程序存储器空间和数据存储器空间采用独立编址，拥有各自的寻址方式和寻址空间。

MCS-52 单片机在物理结构上有四个存储空间，分别为：

（1）片内程序存储器（ROM）　　（2）片外程序存储器（ROM）

（3）片内数据存储器（RAM）　　（2）片外数据存储器（RAM）

在逻辑上，即从用户的角度上，8051 单片机有三个存储空间：

1）片内外统一编址的 64K 的程序存储器地址空间（MOVC）。

2）256B 的片内数据存储器的地址空间（MOV）。

3）以及 64K 片外数据存储器的地址空间（MOVX）。

在访问三个不同的逻辑空间时，应采用不同形式的指令（具体读者可以查阅相关书籍），以产生不同的存储器空间的选通信号，单片机存储器的空间结构如图 3-12 所示。

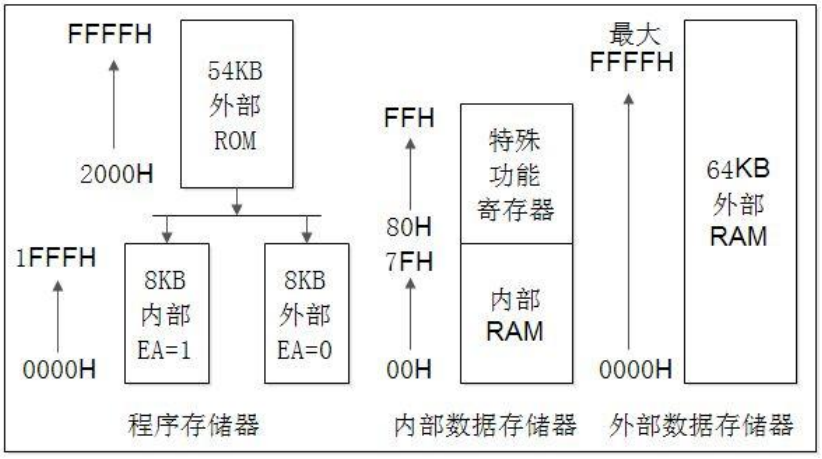

图 3-12 52 单片机内部存储结构示意图

（1）程序内存 ROM

该内存的寻址范围为：0000H~FFFFH，容量就为 64K（2^{16}）。主要作用是存放程序及程序运行时所需的常数。

EA = 1 时，寻址内部 ROM；EA = 0 时，寻址外部 ROM。EA 是单片机的 31 管脚，由此可

见，若该管脚接高电平，上电复位后单片机从内部程序存储器开始读取并执行程序；若接低电平，上电复位之后单片机从外部程序存储器开始读取并执行程序。对于初学者，暂时不需要外扩程序存储器，因此该管脚（31 脚）接高电平就是了。

特别提醒，这里有七个具有特殊含义的单元，分别为：

1）0000H——系统复位，PC 指向此处；

2）~（7）都为中断向量的入口地址，中断章节在续，这里不赘。

顺便说说 29（PSEN）、30（ALE/PROG#）管脚。

PSEN（29）—程序存储器允许输出控制端。在读外部程序存储器时 PSEN 低电平有效，以实现外部程序存储器单元的读操作，由于现在我们使用的单片机内部已经有足够大的 ROM，所以几乎没人再去外扩 ROM 了，因此这个引脚大家只需了解即可。

ALE/PROG（30 脚）—在单片机扩展外部 RAM 时，ALE 用于控制把 P0 口的输出低 8 位地址锁存器起来，以实现地位地址和数据的隔离。当 ALE 为高电平是，允许地址锁存信号，当访问外部存储器时，ALE 信号负跳变讲 P0 口上低 8 位地址信号送入锁存器；当 ALE 是低电平时，P0 口上的内容和锁存器输出一直。关于锁存器和该端口的应用我们在后面再做详解，这里不赘。

（2）内部数据存储器 RAM

数据存储器也称为随机存取数据存储器。数据存储器分为内部数据存储器和外部数据存储器。52 单片机内部有 256 个字节的存储空间(不同型号有别)，片外最多可扩展 64K 的 RAM，片内 RAM 用“MOV”指令访问，片外 RAM 用“MOVX”指令访问，都是用来存放执行的中间结果和过程数据的。该存储器在物理上和逻辑上都分为两个地址空间，即：

（1）数据存储器空间（低 128 字节）。寻址范围为：00H~7FH。

这部分寄存器主要包含：工作寄存器区、位寻址区、数据缓冲区和堆栈数据区三个部分。这里简述一下堆栈，别的读者可以自行查阅相关资料。

堆栈都是一种数据项按序排列的数据结构，只能在一端（称为栈顶）对数据项进行插入和删除。不同的是，堆的顺序随意；栈的顺序是后进先出。堆栈的主要作用有：保护断点、现场保护、临时暂存数据。

（2）特殊功能寄存器空间（高 128 字节），寻址范围为：80H~FFH。

单片机是通过特殊功能寄存器（SFR）对各种功能部件进行集中控制的，这里就不一一列举了，我们在后续学习中慢慢掌握就是。

由此可见，这两个空间是相连的，单从用户的角度而言，低 128 字节才是真正的数据存储器，单片机运行程序时才能自由的读写数据。高 128 字节的空间已经被厂商定制，用户只需按官方出示的数据手册操作就是了。

（3）外部数据存储器 RAM 的扩展

前面说过，52 单片机内部有 256 字节的数据存储器（真正可用的才 128 字节），这些存储器通常被用作工作寄存器、堆栈、零时变量的存储等，一般情况够用了，但是如果系统要存储大量数据，例如跑 uCOS 操作系统时，那么片内的数据存储器就不够用了，需要进行扩展；再如后面我们要讲述的 LD3320 芯片，就可以将其看做是一个外扩存储器，这样可以大大的简化程序的编写。

单片机中常用的数据存储器是静态 RAM 存储器（SRAM），这里以 6264 为例，来讲述外部 RAM 的扩展。Intel 6264 是 8K×8 的 SRAM，单一的+5V 电源，所有的输入端和输出端都与 TTL 电平兼容。它的电路原理图逻辑符合如图 3-13 所示，其中，CE1 为片选信号 1（低电平有效），CE2 为片选信号 2（高电平有效），OE 为输出允许信号，WE 为写信号，A0~A12 为 13

根地址线，D0~D7 为 8 位数据线。

图 3-13 6264RAM 的引脚图

（1）Intel 6264 的操作方式

Intel 6264 的操作方式由 OE、WE、CE1、CE2 共同作用决定。

①写入：当 WE 和 CE1 为低电平，且 OE 和 CE2 为高电平时，数据输入缓冲器打开，数据由数据线 D7～D0 写入被选中的存储单元。

②读出：当 OE 和 CE1 为低电平，且 WE 和 CE2 为高电平时，数据输出缓冲器选通，被选中单元的数据送到数据线 D7～D0 上。

③保持：当 CE1 为高电平，CE2 为任意时，芯片未被选中，处于保持状态，数据线呈现高阻状态。

（2）Intel 6264 的扩展方法

外部数据存储器的扩展，说白了就是三种总线（数据总线、地址总线、控制总线）的连接方式，其中 A0~A12 为地址总线，D0~D7 为数据总线，ALE、WR、RD 为控制总线，因此该存储器与单片机的连接可用 3-14 的示意图表示。

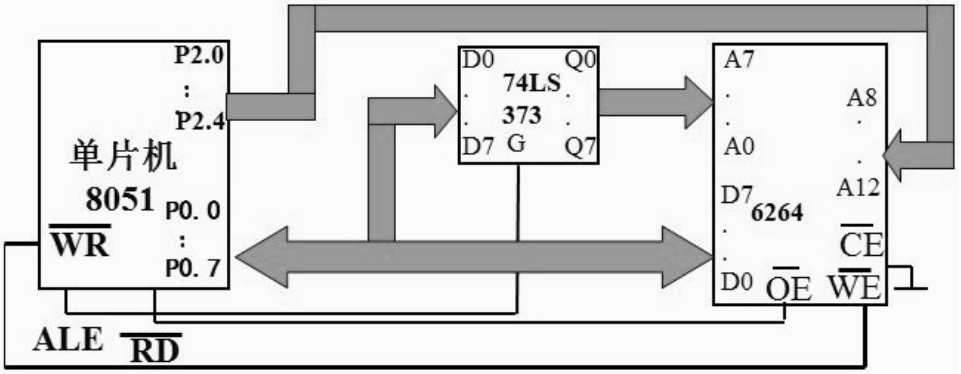

图 3-14 6264 与单片机的连接示意图

这里并没有画出真正原理图，取而代之的是用了示意图，这样做的好处是便于读者理解，具体电路的连接和数据的读写会在以后章节讲述。

二、单片机的 I/O 口

残弈悟恩在编写的《深入浅出玩转 51 单片机》中说过，玩单片机，主要就是玩 32 个 I/O 口（型号不同，可能 I/O 数量不同），**让其在合适的时间出现合适的高低电平**。无论单片机对外界进行何种控制，或接受外部的何种输入，都是通过 I/O 口进行的，因为这些 I/O 口就是单片机与外界沟通的桥梁。51 单片机总共有四个 8 位输入输出端口，每个端口都有锁存器、输出驱动器和输入缓冲器。4 个 I/O 端口都能做输入输出口用，其中 P0 和 P2 还可用于对外部存储器的访问。

51 系列单片机有 4 个 I/O 端口，每个端口有 8 位，合计 32 个引脚。每个端口都包括一个锁存器(即专用寄存器 P0～P3)、一个输出驱动器和输入缓冲器。通常把 4 个端口笼统的表示为 P0～P3。这 32 个引脚分别对应单片机的 32~37（P0）、1~8（P1）、21~28（P2）、10~17（P3）。

备注：其中 P0 口为双向口，P1~P3 为准双向口。

在无片外扩展存储器的系统中，这 4 个端口的每一位都可以作为通用 I/O 口使用。在具有片外扩展存储器的系统中，P2 口作为高 8 位地址线，P0 口分时作为低 8 位地址线和双向数据总线。

51 单片机 4 个 I/O 端口线路设计的非常巧妙，学习 I/O 端口逻辑电路，不但有利于正确合理地使用端口，而且会给设计单片机外围逻辑电路有所启发。下面分别介绍一下输入/输出端口的结构。

1. P0 口的内部结构

一、P0 口的内部结构

图 3-15 为 P0 口的某位 P0.n(n=0~7)的结构图，它由一个输出锁存器、两个三态输入缓冲器和输出驱动电路及控制电路组成。从图中可以看出，P0 口既可以作为 I/O 用，也可以作为地址/数据线用。

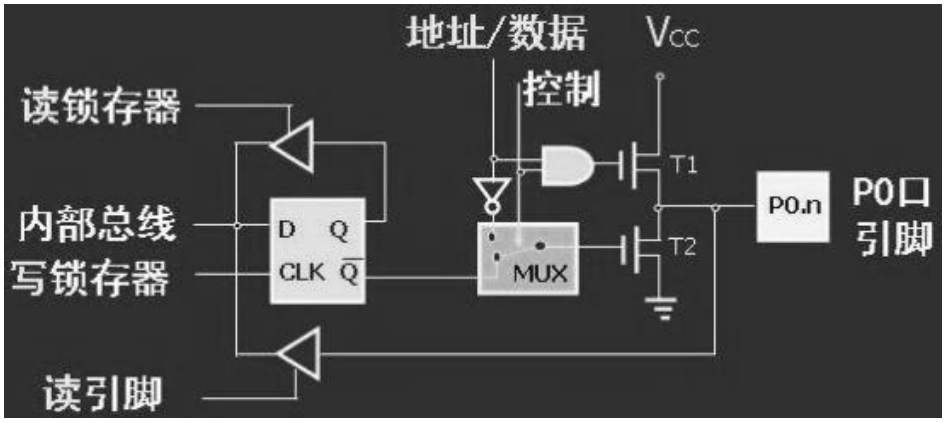

图 3-15 P0 口的内部结构图

二、P0 口作为普通 I/O 口

1.输出时，CPU 发出控制电平“0”封锁“与”门，则输出的上拉场效应管 T1（N 沟道的）截止，同时使多路开关 MUX 把锁存器与输出驱动场效应管 T2 栅极接通。故内部总线与 P0 口同通。由于输出驱动级是漏极开路电路的，若驱动 NMOS 或其它拉电流负载时，需要外接上拉电阻，若不接就只能输出低电平，而输不出高电平。P0 的输出级可驱动 8 个 LSTTL 负载。

2.输入时，分为读引脚和读锁存器。

（1）读引脚。图 3-15 下面的一个缓冲器用于读端口引脚数据。当执行一条由端口输入的指令时，读脉冲把该三态缓冲器打开，这样端口引脚上的数据经过缓冲器读入到内部总线。

（2）读锁存器。图 3-15 上面的一个缓冲器用于读端口锁存器数据。

读锁存器的原因：如果此时该端口的负载恰是一个晶体管（NPN 型）基极，且原端口输出值为 1，那么导通了的 PN 结会把端口引脚高电平拉低。若此时直接读端口引脚的信号，将会把原端口输出的高电平误读为低电平。现采用读输出锁存器代替读引脚，这样就等价于通过上面的三态缓冲器读锁存器 Q 端的信号，过程如图 3-15 所示。因而通过读输出锁存器可避免上述可能发生的错误。

这里说明两点，望读者注意。

A）P0 口必须接上拉电阻，原因上面已经说这了。

B）在读信号数据之前，先要向相应的锁存器做写 1 操作。至于这个原因各位高手众说纷纭，有些从工艺角度解释，有些从是否对寄存器写过“0”的角度解释，有些从内部结构解释，鉴于各种情况，笔者的建议是在读操作之前，最好写上一句：P0=0xFF，反正又累不死人，^_^。

三、P0 作为地址/数据总线

在系统扩展时，P0 端口作为地址/数据总线使用时，分为：

（1）P0 引脚输出地址/数据信息：CPU 发出控制电平“1”，打开“与”门，又使多路开关 MUX 把 CPU 的地址/数据总线与 T2 栅极反相接通，输出地址或数据。由图 11-14 可以看出，上下两个 FET 处于反相，构成了推拉式的输出电路，其带负载能力大大增强。

P0 作为地址/数据总线----真正的双向口

（2）P0 引脚输出地址/输入数据：输入信号是从引脚通过输入缓冲器进入内部总线。此时，CPU 自动使 MUX 向下，并向 P0 口写“1”，“读引脚”控制信号有效，下面的缓冲器打开，外部数据读入内部总线。

2. P2 的内部结构

一、P2 口的内部结构

图 3-16 为 P2 口某一位的内部结构示意图。其构成与 P0 稍有差异，望读者注意。

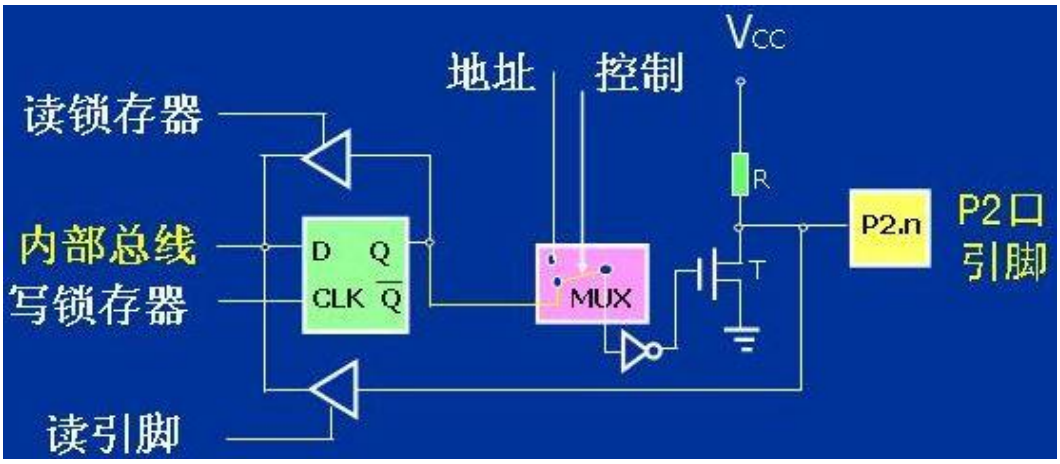

图 3-16 P2 口内部结构示意图

二、P2 口作为普通 I/O 口：CPU 发出控制电平“0”，使多路开关 MUX 倒向锁存器输出 Q 端，构成一个准双向口。其功能与 P1 相同。

三、P2 口作为地址总线：在系统扩展片外存储器时，CPU 发出控制电平“1”，使多路开关 MUX 倒向内部地址线，此时 P2 输出高 8 位地址。

3. P1 口、P3 口的内部结构

一、P1 口的内部结构

它由一个输出锁存器、两个三态输入缓冲器和输出驱动电路组成，标准的准双向 I/O 口。其内部结构如图 3-17 所示。

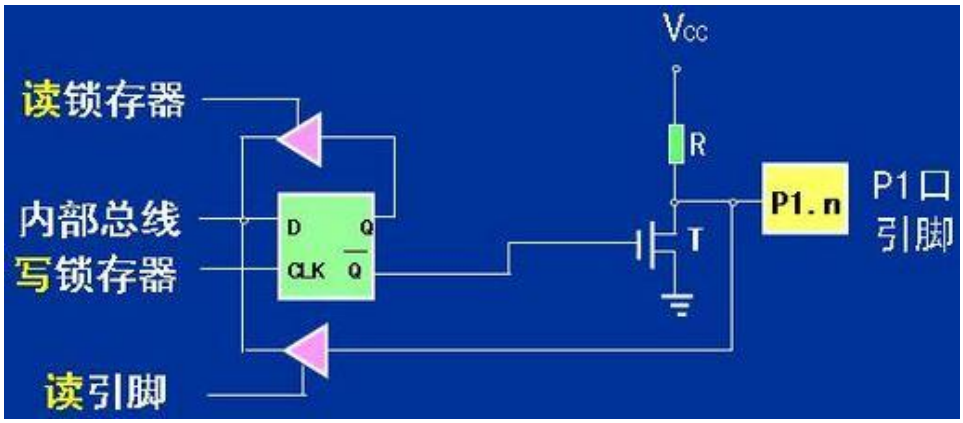

图 3-17 P1 口内部结构图

二、P3 口的内部结构

（1）作为通用 I/O 口与 P1 口类似，标准的准双向 I/O 口(W=1)。内部结构如图 3-18 所示。

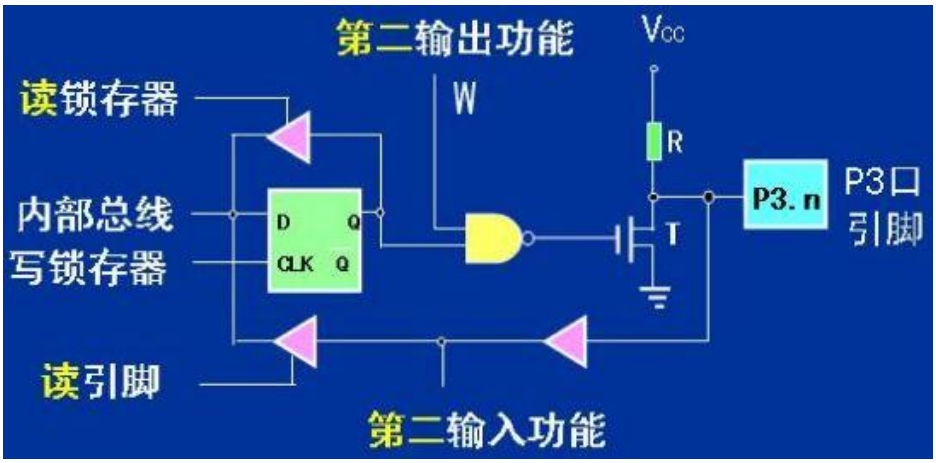

图 3-18 P3 口内部结构图

（2）P3 第二功能(Q=1)。此时引脚部分输入(Q=1、W=1），部分输出(Q=1、W 输出)。

P3 第二功能各引脚功能定义：

P3.0（RXD）：串行口输入

P3.1（TXD）：串行口输出

P3.2（INT0）：外部中断 0 输入（下降沿中断或低电平中断）

P3.3（INT1）：外部中断 1 输入（下降沿中断或低电平中断）

P3.4（T0）：定时器/计数器 0 外部输入

P3.5（T1）：定时器/计数器 1 外部输入

P3.6（WR）：外部写控制

P3.7（RD）：外部读控制

综上所述：当 P0 作为 I/O 口使用时，特别是作为输出时，输出端属于开漏电路，必须外接上拉电阻才会有高电平输出；如果作为输入，必须先向相应的锁存器写“1”，才不会影响输入电平。当 CPU 内部控制信号为“1”时，P0 口作为地址/数据总线使用，这时，P0 口

就无法再作为 I/O 口使用了。

P1、P2 和 P3 口为准双向口，在内部差别不大，但使用功能有所不同。

P1 口是用户专用 8 位准双向 I/O 口，具有通用输入/输出功能，每一位都能独立地设定为输入或输出。当有输出方式变为输入方式时，该位的锁存器必须写入“1”，然后才能进入输入操作。

P2 口是 8 位准双向 I/O 口。外接 I/O 设备时，可作为扩展系统的地址总线，输出高 8 位地址，与 P0 口一起组成 16 位地址总线。对于 8031 而言，P2 口一般只作为地址总线使用，而不作为 I/O 线直接与外部设备相连，关于这些知识点，笔记 20 的单片机体系结构中会做详细介绍。

P3 口也是 8 位准双向 I/O 口，只是该端口除了 I/O 口应有的功能以外，还具有第二种特殊功能，例如串口通信、外部中断、计数器输入、外部存储器控制端子。

这样 52 的 32 个 I/O 口就讲解完了，其实这些 I/O 口是相当简单，因为它们都是双向的，做输入、输出都一样。随着读者深入的学习的，以后要是涉及到像 STM32 这样的 ARM 核的处理器时，端口才说复杂，因为其端口是可以配置的，什么意思，就是可以通过设置其内部的寄存器将 I/O 口配置成 8 种模式，例如模拟输入、悬空输入、上拉输入、下拉输入、推挽输出、开漏输出、推挽复用输出、开漏复用输出。是不是很强大，那读者就赶紧玩好单片机之后去膜拜、膜拜它的强大吧。

三、复位和晶振电路

这里我们在说说，单片机中最简单，却又很重要的两个电路，那就是复位和晶振电路。

1. 复位电路

什么是复位电路，用于复位的电路就是复位电路，那什么是复位，所谓复位就是利用它把单片机当前的运行状态恢复到起始状态。就像计算器的清零按钮作用一样，当读者进行完了一个题目的计算后肯定是要清零的是吧。还有常说的一句话：重新来过中的重新也可理解为是复位，复位电路如图 3-19 所示。

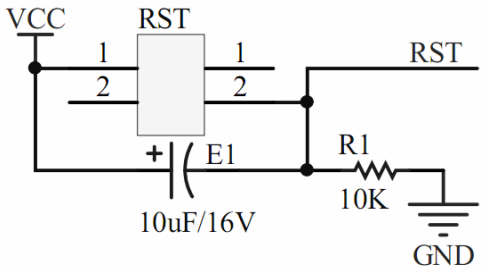

图 3-19 复位电路

复位电路可分为：手动复位和自动复位。MGMC-V1.0 实验板上采用的是手动复位，当然该电路也包含了上电自动复位（直接把 RST 开关咔嚓掉）。

STC89C52 单片机是高电平复位，所以这里设计成了高电平复位电路，具体复位过程是，上电的一瞬间，电容 E1 正端为高电平（5V），负端为低电平（0V），这样就会对电容充电，从而在 R1 上有电流流过，这样，在 RST 端就会有高电平出现，之后随着电容的充电，电流一直下降，当电容充满电后，电流变为 0，所以 RST 就一直保持低电平，单片机开始工作，复位时间大概是 250MS（残弈悟恩用示波器在电容为 10uF 时测得的数据），具体时间读者可以计算一下。以后在调试程序或者程序跑飞时，就可以按一下复位按钮，PC 初始化为 0000H，

使单片机从 0000H 单元开始执行程序。除 PC 之外，复位操作还会影响到别的寄存器，例如 P0~P3（寄存器的名称）的复位值都为：FFH，SP 的为：07H，剩余的寄存器全为：00H。

补充：其中 RST 接单片机的复位端子（9 管脚）。

2. 晶振电路

晶振的作用是为单片机提供时钟，若没有晶振，单片机的程序就会乱跑，也有可能直接不跑。举个例子，若学生和老师没有一个统一的时钟，老师上课的时间是学生下课的时间，学生上课是老师休息的时间，那这课还能上吗？再者为何全中国要统一用北京时间，肯定不是上海人用上海时间，兰州人用兰州时间，这样岂不也乱乎也，时钟电路如图 3-20 所示。

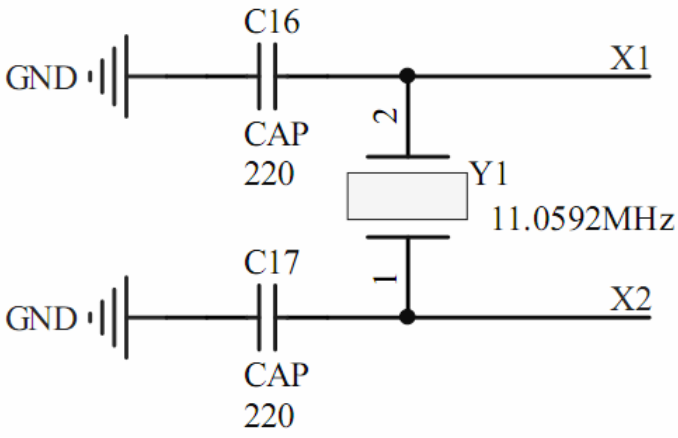

图 3-20 晶振电路

接着说说那两个电容（C16、C17），电容在该电路中叫负载电容（或者起振电容），晶振上电启动后会振荡产生脉冲波形，相当于提供给单片机大动脉，但往往伴随着有谐波参杂在主波形中，影响单片机的工作稳定性，所以加了电容将这些谐波虑掉，虑到哪儿，当然是地，因而两电容都接地了，这样就可以起到一个并联谐振的作用，从而使它的脉冲更平稳与协调。

再说三个名词：**时钟周期、机器周期、指令周期**。

（1）时钟周期。单片机的基本时间单位，也即外接晶振的振荡周期。MGMC-V1.0 实验板使用的是 11.0592Mhz（为何用 11.0592MHz，笔记 10 会给读者一个满意的答案）的晶振，那么时钟周期就为：1/11.0592us。

（2）机器周期。CPU 完成一个基本操作所需要的时间。STC89C52 是 12T 的单片机，什么意思，意思是 12 个时钟周期为一个机器周期。也即机器周期=时钟周期/12=晶振周期/12=12/11.592MHz（歉意：《深入浅出玩转 51 单片机》中有错）。其实现在 STC 公司生产了好多 1T 的单片机，例如 STC12C5A60S2，这样机器周期 = 晶振周期。当初残弈悟恩上课时，有同学说那我直接给接个 1GHz 的晶振，单片机不就也上 G 呢，这可万万不行，为何？读者好好阅读一下官方的数据手册就知道了，再者也没有这么大的晶振，除非读者自己造一个，^_^。

指令周期主要用在汇编中，所以这里不赘。

补充：X1、X2 接单片机的晶振管脚，也即 18、19 脚。

3.2.3 再废话几句存储器

我自认为，单片机的存储器有些抽象，大家说呢？既然 ROM 是只读存储器，那难道“谁”都写不进数据吗？

存储器一直贯穿于嵌入式的开发中，单片机开发中也不例外。读者都知道，要让单片机

按人为要求有序运行，必须首先要编写程序，同时，程序运行中，又和数据在打交道。这样，既要存放程序，又要存放数据，因此单片机中也引入了程序存储器和数据存储器。既然这样，我们再接着废话几句。

一、ROM（只读存储器）

ROM 是 Read-Only Memory 的英文简写。ROM 所存数据，一般是装入整机前事先写好的，整机工作过程中只能读出，而不像随机存储器那样能快速地、方便地加以改写。ROM 所存数据稳定，断电后所存数据也不会改变。其结构较简单，读出较方便，因而常用于存储各种固定程序和数据。

ROM 的分类比较多，例如有 PROM、EPROM、OTPROM、EEPROM（这个在笔记 11 将会做详细的介绍）、Flash ROM 等，其中 Flash 又分 NOR Flash、NAND Flash。

二、RAM（随机存储器）

RAM 是 random access memory 的英文缩写。存储单元的内容可按需求随意取出或存入，且存取的速度与存储单元的位置无关。这种存储器在断电时将丢失其存储内容，故主要用于存储短时间使用的程序。

按照存储信息的不同，随机存储器又分为静态随机存储器（Static RAM，SRAM)和动态随机存储器（Dynamic RAM,DRAM)。

以上这些内容读者不需记忆，以后碰见了知道有这么个回事就 OK 了。或者以后学 ARM、FPGA 等时要详细的学习 NOR Flash、NAND Flash、SRAM、SDRAM 等。

好了，熟悉了 ROM 和 RAM 的定义之后，再回过头去看看单片机（STC89C52）中的 ROM 和 RAM，是否还记得笔者在实例 8 之后简单说了一下，若不记得了也没事，接下来还有更详细的说明了。

三、单片机的 ROM（FLASH）和 RAM（SRAM）

（1）单片机的 ROM（Flash)。单片机 Flash 主要用作程序存储器，就是代替上面提到的 ROM，最大的优点是降低了芯片的成本并且可以电擦写。目前市场上单片机的 Flash 寿命相差比较大，有的长达 40 年之久，擦写次数从 1000~10 万不等。

在单片机中用来存储程序数据及常量数据或变量数据，凡是 c 文件以及 h 文件中所有代码、全局变量、局部变量、‘const’限定符定义的常量数据、startup.asm 文件中的代码（类似 ARM 中的 bootloader 或者 X86 中的 BIOS，一些低端的单片机是没有这个的）通通都存储在 ROM 中。

（2）单片机的 RAM（SRAM)。SRAM 是数据存储器，跟计算机里面的内存差不多，主要是用来存放程序运行中的过程数据，掉电后数据会丢失。所以程序在上电时需要初始化（也即复位），否则上电后的数据是一个随机数，可能导致程序崩溃。

用来存储程序中用到的变量。凡是整个程序中，所用到的需要被改写的量，都存储在 RAM 中，“被改变的量”包括全局变量、局部变量、堆栈段。

打开 STC 官方单片机（STC89C52）的数据手册第 2 页可知，该单片机的 Flash 为：8K，SRAM 为：512 字节。由此可见该型号单片机的数据存储器（SRAM）不是很大，因而有了在编

写程序时，在某些数组前一般加“code”关键词（前面实例中运用很多），就是将这些数组数据存储在程序存储器（Flash）中，以便腾出空间来运行程序。接下来详细了解一下这些内部的渠渠道道。

程序经过编译、汇编、链接后，生成 hex 文件。用专用的烧录软件，通过烧录器（其实 STC 公司的单片机也可以不用烧录器，用串口就可以了）将 hex 文件烧录到 ROM 中（究竟是怎样将 hex 文件传输到 MCU 内部的 ROM 中的呢？），因此，这个时候的 ROM 中，包含所有的程序内容：无论是一行一行的程序代码，函数中用到的局部变量，头文件中所声明的全局变量，‘const’声明的只读常量，都被生成了二进制数据，包含在 hex 文件中，全部烧录到了 ROM 里面，此时的 ROM，包含了程序的所有信息，正是由于这些信息，“指导”了 CPU 的所有动作。

可能有人会有疑问，既然所有的数据在 ROM 中，那 RAM 中的数据从哪里来？什么时候 CPU 将数据加载到 RAM 中？会不会是在烧录的时候，已经将需要放在 RAM 中数据烧录到了 RAM 中？

要回答这个问题，首先必须明确一条：ROM 是只读存储器，CPU 只能从里面读数据，不能往里面写数据，掉电后数据依然保存在存储器中；RAM 是随机存储器，CPU 既可以从里面读出数据，又可以往里面写入数据，掉电后数据不保存，这是条永恒的真理，读者们要刻在骨头里面，记在心里面。

清楚了上面的问题，那么就很容易想到，RAM 中的数据不是在烧录的时候写入的，因为烧录完毕后，拔掉电源，当再给 MCU 上电后，CPU 也能正常执行动作，RAM 中照样有数据，这就说明：RAM 中的数据不是在烧录的时候写入，同时也说明，在 CPU 运行时，RAM 中已经写入了数据。关键就在这里：这个数据不是人为写入的，那肯定是 CPU 写入的，既然是 CPU 写入的，那又是什么时候写入的呢？娓娓道来，不用着急。

上回说到，ROM 中包含所有的程序内容，在 MCU 上电时，CPU 开始从第 1 行代码处执行指令。这里所做的工作是为整个程序的顺利运行做好准备，或者说是对 RAM 的初始化（注：ROM 是只读不写的），工作任务主要有下面几项。

1）为全局变量分配地址空间——如果全局变量已赋初值，则将初始值从 ROM 中拷贝到 RAM 中，如果没有赋初值，则这个全局变量所对应的地址下的初值为 0 或者是不确定的。当然，如果已经指定了变量的地址空间，则直接定位到对应的地址就行，那么这里分配地址及定位地址的任务由“链接器”完成。

2）设置堆栈（堆、栈又是何方妖孽，稍等再来解释）段的长度及地址——用 C 语言开发的单片机程序里面，普遍都没有涉及到堆栈段长度的设置，但这不意味着不用设置。堆栈段主要是用来在中断处理时起“保存现场”及“现场还原”的作用，其重要性不言而喻。而这么重要的内容，也包含在了编译器预设的内容里面，确实省事，可并不一定省心。平时怎么就没发现呢？奇哉怪也。

3）分配数据段（data），常量段（const），代码段（code）的起始地址。代码段与常量段的地址可以不管，它们都是固定在 ROM 里面的，无论它们怎么排列，都不会对程序产生影响。但是数据段的地址就必须得关心。数据段的数据是要从 ROM 拷贝到 RAM 中去的，而在 RAM 中，既有数据段（data），也有堆栈段（stack），还有通用的工作寄存器组。通常，工作寄存器组的地址是固定的，这就要求在决定地址数据段时，不能使数据段覆盖所有的工作寄存器组的地址。必须引起特别关注。

这里所说的“第一行代码处”，并不一定是读者自己写的程序代码，绝大部分都是编译器代劳的，或者是编译器自带的 demo 程序文件。因为，读者自己写的程序（C 语言程序）里面，并不包含这些内容。高级一点的单片机，这些内容，都是在 startup 的文件里面。仔细阅读，有益无害。

通常的做法是：普通的 flashMCU（单片机）是在上电时或复位时，PC 指针里面存放的是“0000”，表示 CPU 从 ROM 的 0000 地址开始执行指令，在该地址处放一条跳转指令，使程序跳转到_main 函数中，然后根据不同的指令，一条一条的执行，当中断发生时（中断数量也很有限，2~5 个中断），按照系统分配的中断向量表地址，在中断向量里面，放置一条跳转到中断服务程序的指令，如此如此，整个程序就跑起来了。决定 CPU 这样做，是这种 ROM 结构所造成的。

其实，这里面，C 语言编译器作了很多的工作，只是，读者们不知道而已。如果读者们仔细阅读编译器自带的 help 文件就会发现很多，也是了解编译器最好的途径。

是不是看着这冗长、无味的文字，又想放弃呢？冲动的想法可以有，决定不能下。

一边是悬崖峭壁，一边是万丈深渊。就在这样的求学路上，他们一走就是 10 多年。每当天下雨，家长们需要拿着铁锹护送孩子们上学。出身决定了他们的命运，没有选择只有继续。我们也是一样，只有继续才是最好的选择。

大山深处的希望路：

http://v.youku.com/v_show/id_XMzIxMzEyNDQw.html?from=y1.2-1-92.3.3-2.1-1-1-2

谁又知道，每天用生命护送孩子几十里路的张克荣老师，一个月才领着 200 元的工资了？他为了在浑水中救孩子，得了重感冒，但无论怎样，他还是没有放弃，我们有何理由放弃呢？

3.3 电源

要让单片机工作起来，电源肯定是少不了。就如，人要工作，必须要吃饱、睡好，否则肯定是没精力工作。

3.3.1 单片机的电源

对于单片机来说，型号不同，可能工作电压不同。例如我们所用的 STC89C52 单片机工作电压为 5V，其电压输入正、负端子分别为：VCC（40 管脚）和 GND（20 管脚），恰好，电脑的 USB 输出端子为 5V，我这样，我们就可以直接用 USB 来供电了，其供电原理图如图 3-21 所示，这样，我们再用一根 USB 线连接单片机与电脑就是了。

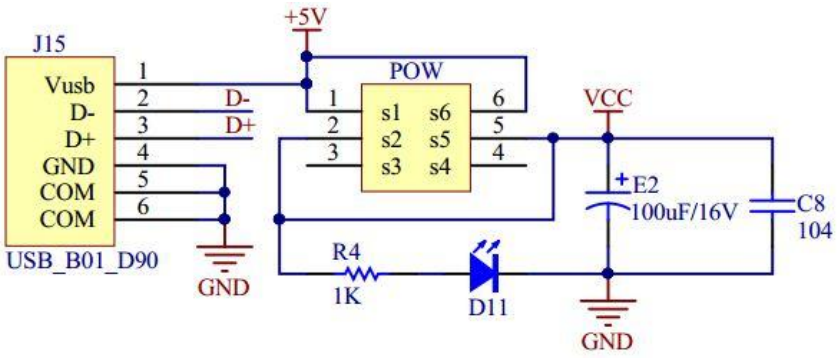

图 3-21 USB 供电原理图

那要是以后我们用额定电源为 3.3V 的单片机怎么办了，5V 的电源肯定是不行，或者以后我们 DIY 小车，一般小车电机工作的电压比 5V 大，这时又得怎么办了，这就需要我们对电压进行转换，将输入的电压转换成我们想要的电压，此时我们就得考虑以下的电路设计了。

3.3.2 LDO 与 DC-DC 电路

无论是平时的学习中，还是以后的工作中，读者都会接触 LDO 和 DC-DC 这一类的电源产品，但作为玩单片机的初学者来说，可能对这些东西了解的不够深刻，在这里残弈悟恩就给大家介绍一下 DC-DC 和 LDO，希望对读者有所帮助，别走进公司了，对着一个 BOOST 或 BUCK 电路还说什么 LDO，那样连外行听了都会笑掉大牙，^o _o^。

一、LDO 花落知多少？

LDO 是 low dropout voltage regulator 的缩写，就是低压差线性稳压器。

低压降（LDO）线性稳压器的成本低，噪音低，静态电流小，这些是它的突出优点。它需要的外接元件也很少，通常只需要一两个旁路电容。这里举个例子，5V→3.3V 的电路如图 3-22 所示。

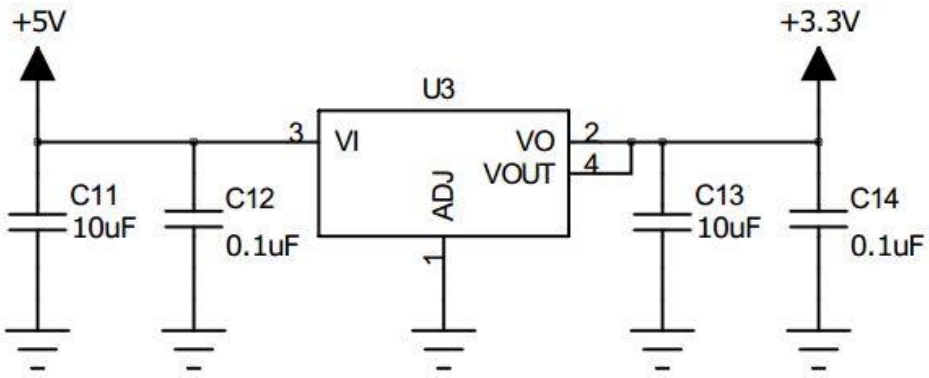

图 3-22 5V→3.3V 电路原理图

该电路相当简单，就是将 5V 的电压转换成 3.3V，转换所用器件为 AMS1117-3.3，前后所用的四个电容都为滤波电容，大家注意，这里 10uF 和 0.1uF 的放置是有顺序的，为何？残弈悟恩先不说了，嘿嘿。

二、DC-DC 又知多少呢？

DC-DC 的意思是直流变(到)直流(不同直流电源值的转换)，只要符合这个定义都可以叫 DC-DC 转换器，包括 LDO。但是一般的说法是把直流变(到)直流由开关方式实现的器件叫 DC-DC。

DC-DC 的现实过程，其实内部是先把 DC 直流电源转变为交流电源 AC。通常是一种自激震荡电路，所以外面需要电感等分立元件。然后在输出端再通过积分滤波，又回到 DC 电源。由于产生了 AC 电源，所以可以很轻松的进行升压和降压。两次转换，必然会产生损耗，这就是大家都在努力研究的如何提高 DC-DC 效率的问题。

这里就不讲述了，给大家布置两个作业。

（1）将 5V 的电压转换为 27V（参考：用 Boost 电路）。

（2）将 5V 的电压转换为-27V（参考：电压器）。

具体过程、原理图后续讲述，敬请期待。

三、DC-DC 和 LDO 的选择依据

如果输入电压和输出电压很接近，最好是选用 LDO 稳压器，可达到很高的效率。所以，在把锂离子电池电压转换为 3V 输出电压的应用中大多选用 LDO 稳压器。即使电池的能量最后有百分之十是没有使用的，但 LDO 稳压器仍然能够保证电池的工作时间较长，同时噪音较低。

如果输入电压和输出电压不是很接近，就要考虑用开关型的 DC-DC 了。从上面的原理可以知道，LDO 的输入电流基本上是等于输出电流的，如果压降太大，耗在 LDO 上的能量就会太大，因而效率就不高。

DC-DC 转换器包括升压、降压、升/降压和反相等电路。DC-DC 转换器的优点是效率高、可以输出大电流、静态电流小。随着集成度的提高，许多新型 DC-DC 转换器仅需要几只外接电感器和滤波电容器。但是，这类电源控制器的输出脉动和开关噪音较大、成本相对较高。

总的来说，升压是一定要选 DC-DC 的；降压，是选择 DC-DC 还是 LDO，要在成本、效率、噪声和性能上先做比较，后选择。

四、DC-DC 和 LDO 应用对比

首先从效率上说，DC-DC 的效率普遍要远高于 LDO，这是其工作原理决定的。

其次，DC-DC 有 Boost、Buck、Boost/Buck、(有人把 Charge Pump 也归为此类)。而 LDO 只有降压型。

再次，也是很重要的一点，DC-DC 因为其开关频率的原因导致其电源噪声很大，远比 LDO 大的多，大家可以关注 PSRR 这个参数。所以当考虑到比较敏感的模拟电路时候，有可能就要牺牲效率为保证电源的纯净而选择 LDO。

还有，通常 LDO 所需要的外围器件简单，占面积小，而 DC-DC 一般都会用到电感、二极管、大电容、有的还会要 MOSFET、特别是 Boost 电路，需要考虑电感的最大工作电流，二极管的反向恢复时间，大电容的 ESR 等，所以从外围器件的选择来说比 LDO 复杂，而且占面积也相应的会大很多。

3.4 单片机最小系统

学习单片机，最重要的一环就是动手实践。学习过程中，自己若有一块单片机最小系统板，那非常有利于动手实践。万丈高楼平地起，有了这块小板，你即将有了建设高楼的基石。

3.4.1 什么是单片机最小系统

残弈悟恩眼中的单片机最小系统与别人的多，望读者熟知。单片机最小系统主要由电源，复位、振荡电路以及程序下载电路（别人将下载电路没算，那怎么下载程序呢？奇哉怪也！）。具体各个部分是干嘛的，若读者知道，恭喜你；若不知道，你还努力。好吧，笔者还是以图说话，最小系统原理图如图 3-23 所示。

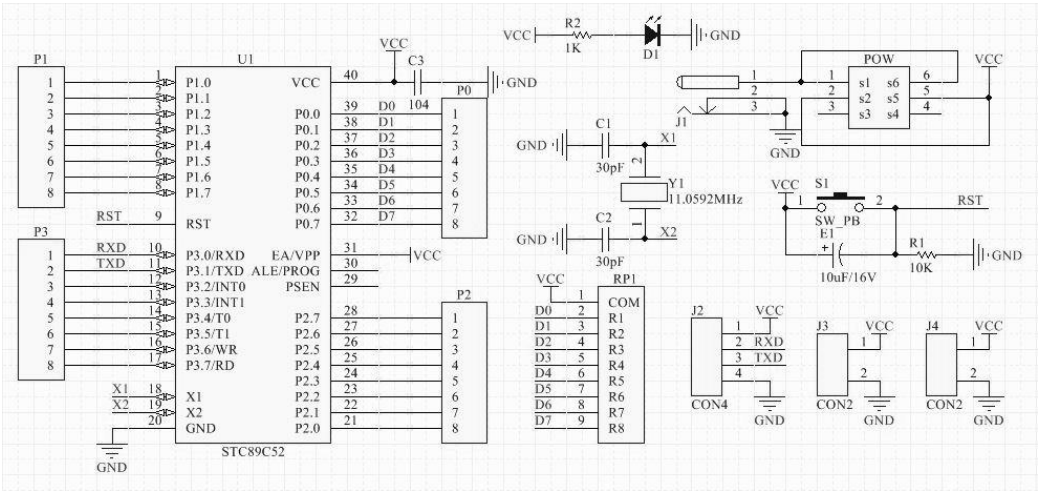

图 3-23 最小系统原理图

3.4.2 搭建最小系统的各种需求

一、工具

工欲善其事，必先利其器。做一个东西，工具是必需。工具没有什么定论，这里简述几种，如烙铁、锡丝、镊子、斜口钳、吸焊笔、万用表，这些是必须要具备的，剩下的像剥线钳、螺丝刀等，用到了再买也来的急。最后像示波器、信号发生器都比较昂贵，读者可以根据自己的经济实力，自行斟酌，残弈悟恩就不废话了，常用工具如图 20-6 所示。

图 3-24 焊接和测试所需的基本工具

二、元器件

接着再看原理图中所需要元件实物，因为没有实物，光原理图是不会工作的，同样，只会原理，没有实际做过东西的人，并不被公司所看好。这些东西需要读者“省吃俭用”，完了去电子市场或网上购买的，最好两者都好好参与一下。在电子市场，读者即可以体验到电子产品和电子元器件的复杂性，又可以认识无知的自己。书中只有示意图，没有实物（除非读者是刘谦将图中的示意图变成实物）。元器件名称比较多，这里不赘，对于读者来说，掌握这些元器件的特性是很有必要的。就像笔者当初工作的部门老大所说的：若一个厨师连辣椒是辣的，食盐是咸的都不知道，那还怎么做饭。因此读者很有必要抽时间去补补电子元器件的特性、功能、封装等知识点，别大学毕业了，都不知道电解电容有正负之分，甚至有人会问电阻有没有正负等这样的笑话。搭建最小系统所需元件如图 3-25 所示。

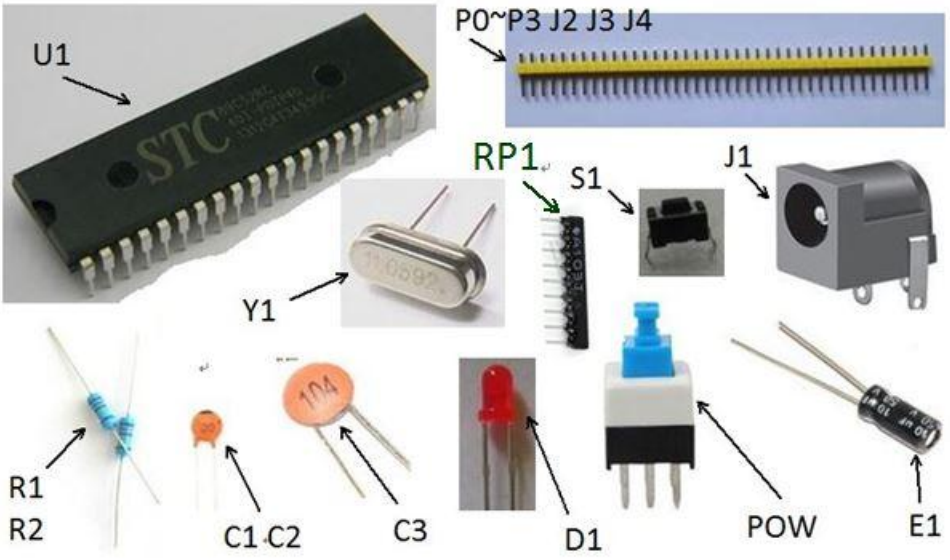

图 3-25 制作最小系统需要准备的元件

3.5 四种最小系统的搭建方法

3.5.1 万用板（洞洞板）搭建版

如果读者想和残弈悟恩一样，用万用板（洞洞板）来搭建一个最小系统，那就还需要认识一下万用板，掌握焊接方法以及焊接的技巧。

一、万用板（洞洞板）

读者一看到“万用”二字，非常激动，以为这东西既能帮你写作业，又能帮你洗衣服，还能帮你买早餐。也许还会有读者一看到“洞洞”二字，认为不是“蜂窝”就是“莲花洞”，要知答案，请看下图，万用板如图 3-26 所示。

这时，读者或许又得跑一趟电子市场，再买几块万用板回来。若读者很懒，那就多买几块，留着以后 DIY 用。

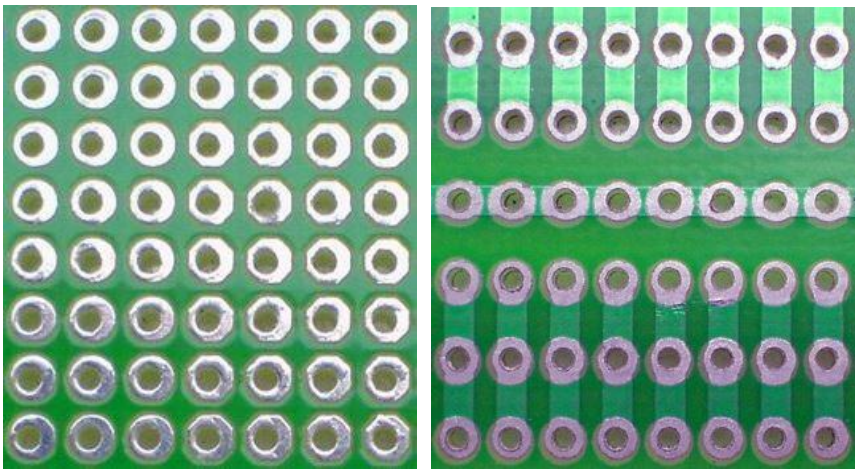

图 3-26 万用板示意图（左：单孔板、右：连孔板）

二、导线

要焊接板子，还有一样少不了，那就是导线，若懒得买，剪一段网线也是可以的，只要你不嫌导线硬（硬了不好焊接）。一般导线分两种：多股和单股。笔者建议买彩色的单股导线，单股的比较好焊接，彩色的焊完电路板看上去比较“漂亮”，导线如图 3-27 所示。

图 3-27 导线示意图

若读者想法比较多，技术也好，喜欢用焊锡直接当导线，那肯定也是可以的，不妨看看残弈悟恩曾经用导线和锡丝焊接的板子（当然两种可以并存），板子如图 3-28 所示。

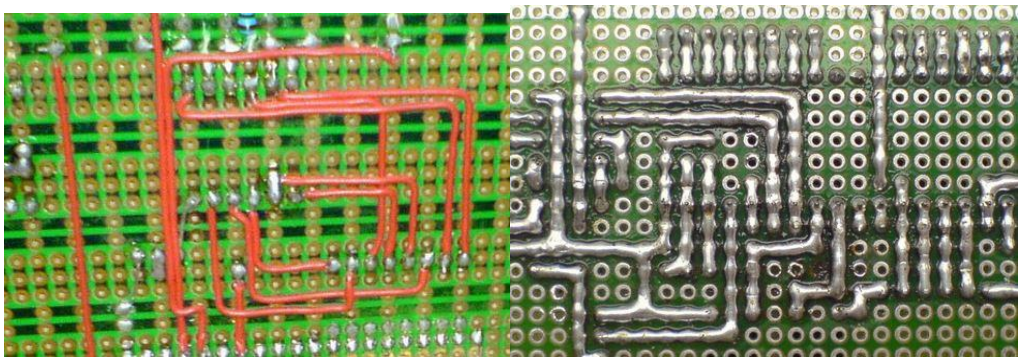

图 3-28 用万用板焊接的电路板（左：导线法、右：锡丝法）

三、焊接技巧的介绍

焊接过程，肯定是先上烙铁，对电路板进行预热，再上焊锡，等焊点饱满、圆滑之后，再撤焊锡，最后再撤烙铁。关于焊接，多练才是硬道理，技巧什么的都是花架子。这里总结几点关于万用板的焊接方法，这些方法是笔者焊接过程中总结的，仅供读者参考，最主要的还是要读者自己行动，光说不练肯定是不行的，死记硬背也是永远掌握不了焊接技巧的。现对焊接技巧总结如下：

1、初步确定电源、地线的布局。电源贯穿电路的始终，合理的电源布局能对简化电路起到关键的作用。某些洞洞板布置有贯穿整块板子的铜箔，应将其用作电源线和地线。如果无此类铜箔，读者也需要对电源线、地线的布局有个初步的规划。

2、善于利用元器件的引脚。洞洞板的焊接需要大量的跨接、跳线等，不要急于剪断元器件多余的引脚，有时候直接跨接到周围待连接的元器件引脚上会事半功倍。另外，本着构建节约型社会的目的，可以把剪断的元器件引脚收集起来作为跳线。

3、善于设置跳线。特别要强调这一点，多设置跳线不仅可以简化连线，而且要美观得多，如图 3-31。

4、善于利用元器件自身的结构。图 3-29 这是一个利用了元器件自身结构的典型例子。图 20-30 中的轻触式按键有 4 只脚，其中两两相通，我们便可以利用这一特点来简化连线，电气相通的两只脚充当了跳线，读者可以对照图 3-30 好好体会一下。

5、善于利用排针。笔者特别喜欢使用排针，因为排针有许多灵活的用法。比如两块板子相连，就可以用排针和排座，排针既起到了两块板子间的机械连接作用又起到电气连接的作用。这点残弈悟恩借鉴了电脑的板卡连接法，处处留心皆学问啊。

6、在需要的时候隔断铜箔。在使用连孔板的时候，为了充分利用空间，必要时可用小刀割断某处铜箔，这样就可以在有限的空间放置更多的元器件。

7、充分利用双面板。双面板比较昂贵，既然选择它就应该充分利用它。双面板的每一个焊盘都可以当作过孔，灵活实现正反面电气连接。

8、充分利用板上的空间。芯片座里面隐藏元件，既美观又能保护元件，如图 3-32 所示。

图 3-29 矩阵键盘正面图

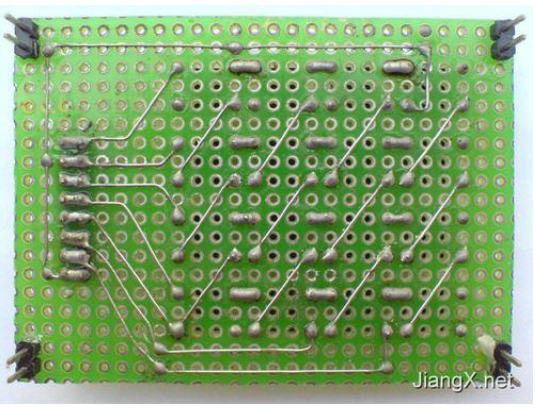

图 3-30 矩阵键盘反面图

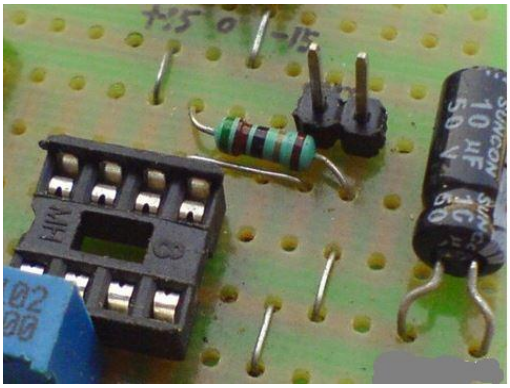

图 3-31 善于设置跳线的电路板

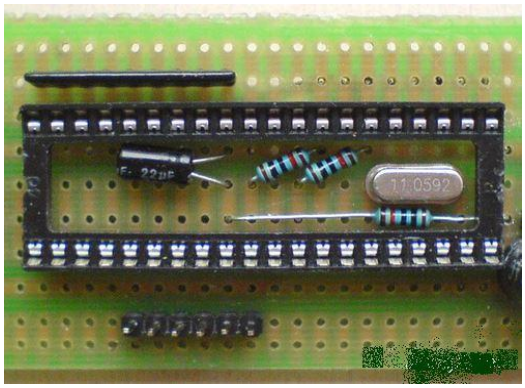

图 3-32 芯片底座下隐藏元件图

四、用万用板焊接的单片机最小系统

此时的读者应该不是一个简单的新手了，可以说是行走在通向“高手”行列的大道上。读者即有了最小系统的理论知识（原理图），又有了搭建最小系统所需的基本工具和电子元器件，同时掌握了焊接技巧，那还不赶紧行动，等什么，再等黄花菜都凉了。

最后欣赏一下残弈悟恩焊接的最小系统，如图 3-33 所示，或许读者刚开始焊接不是太好，但这没事，每个人做事都是由不会到会，有简单到复杂，由不好到好的嘛。但有一点，只要读者多焊接、多总结，肯定能让万用板成为读者电子设计的一个有力助手。

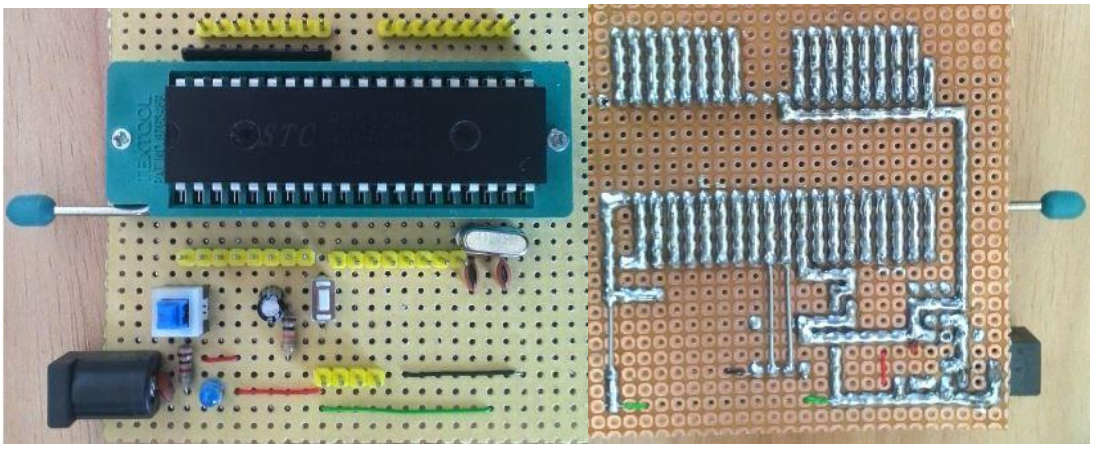

图 3-33 万用板焊接的最小系统图（左：正面图、右：反面图）

最后补充一点，可能为了简化电路，这里没加下载电路，或者只引出四个端子。若下载端口是用排针引出来的，那就还需借助一个 USB 转 TTL 电平模块（串口转 TTL 电平模块也可

以）来给最小系统下载程序，模块如图 3-34 所示。那如果手上已经有了 MGMC-V1.0 实验板，读者就不需要购买 USB 转 TTL 模块了，直接焊接一个 IC 测试座（如图 3-35 所示），这样直接取下单片机，插到 MGMC-V1.0 实验板上，下载程序，完了再插过去，不过这样在调试阶段比较麻烦。当然还可以直接将最小系统的下载电路与 MGMC-V1.0 实验板相连，也能下载程序，说白了，只要读者懂了原理，完成某件事的方法是多种多样的。

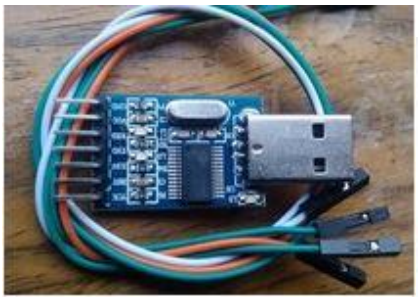

图 3-34 USB 转 TTL 模块

图 3-35 IC 测试座（40P）

3.5.2 面包板搭建版

无论在学校，还是在公司，还有一种搭建测试环境的方法，那就是用面包板，这下读者一看，激动坏了，搞设计这么饿，终于搞出吃的来了，要是残弈悟恩再讲讲“牛奶板”，这样面包、牛奶都有了，生活肯定是乐滋滋啊。残弈悟恩告诉读者，想法很美好，可事实很残酷啊。因为这东西不能吃，那是什么东西呢？稍等片刻，广告之后精彩不断，^_^。

这里先介绍一下用面包板来搭建一个什么系统，是大到“神州飞船”，还是小到单片机控制一个 LED 小灯。飞船太复杂，那就搭建一个单片机史上最简单的系统，用单片机去控制一个 LED 小灯。

电路包括：单片机+晶振电路+复位电路+电源，至于原理，相信读者看着实物图再结合前面的最小系统原理图，脚指头都能想到吧，但一定不要错，否则你会失败的很惨。

广告之后，电视剧 Go On。继续解释面包板，它不是吃的，而是专为电子电路的无焊接试验设计制造的一种板子。由于各种电子元器件可根据需要随意插入或拔出，免去了焊接，节省了电路的组装时间，而且元件可以重复使用，所以非常适合电子电路的组装、调试和训练。其实物图如图 3-36 所示。有了面板板，还需杜邦线，否则电路如何连接起来，杜邦线实物图见 3-37。

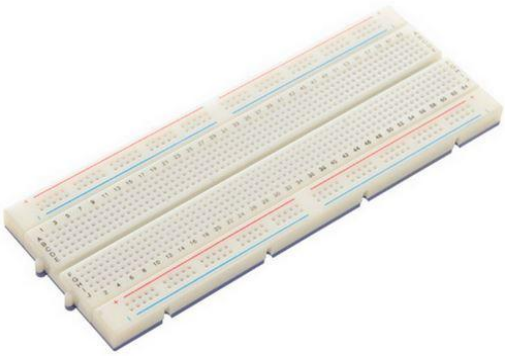

图 3-36 面包板实物图

图 3-37 杜邦线实物图

有了这些实物，接下来就是将其按原理图插接元件（单片机等下载好程序之后再插）、连线杜邦线，之后编写程序，将程序下载到单片机中，再插入到面包板上，这里采用电池供电，整个系统搭建完毕之后如图 3-38 所示。

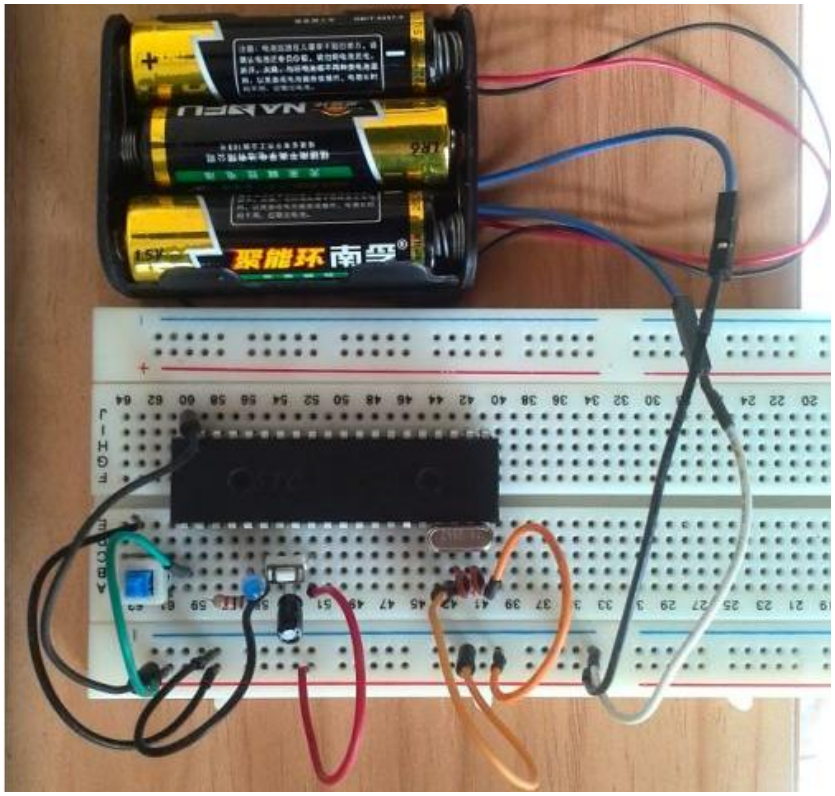

图 3-38 用面包搭建的点灯系统

3.5.3 DIY PCB 板搭建版

所谓的 DIY PCB，意思是用覆铜板来自己腐蚀印刷线路板，怎么腐蚀，常用的方法有两种，分别为：热转印法和感光法。这两种方法和下面要讲的工厂 PCB 法都需要画 PCB，关于 PCB 的绘制方法笔记 19 也做了大量讲解，这里不赘。

笔者将热转印法和感光法统称之为 DIY PCB 法，将工厂制作 PCB 法称之为工厂法。或许读者要问，这两种（DIY 法和工厂法）方法哪个更好呢？残弈悟恩先不急着回答读者的问题，先说这两种方法的适合人群，看完大伙就明白了。

（1）DIY PCB 法适合“我在读书、我要做实验板、我没钱”的人。

（2）工厂 PCB 法适合“我有钱做 PCB、我用的是老板的钱、我家开制版厂”的人。

读者认为那种好呢？

接着主题，DIY 法分两种，这两种又各自有优缺点，选择了 DIY 法的人再来一次选择吧，选择依据请看下面的优缺点。

（1）热转印制板法：

优点：快速、方便、安全、直观、成功率高；

缺点：需要价格昂贵的激光打印机

（2）感光制板法：

优点：无需价格昂贵的激光打印机

缺点：速度慢、有点麻烦、不安全、不直观、成功率低。

如果读者和残弈悟恩一样，穷到家了，那肯定是选择了 DIY 法中的感光制板法，这里笔者应该一步一步的讲解其过程，可是书本有限，网络无限，读者百度一下“感光 PCB”，下面肯定是一大堆的制作过程介绍，因此残弈悟恩也就不浪费大家的 money 了，过程自己查阅。笔者说明两点，望读者多多注意。第一点，曝光的时间很重要，需要多试几次，再来定论；第二点，显影液的浓度和温度很重要，同样需要多试，没有定论。最后看张笔者用感光

法制作的 PCB 实物图，如图 3-39 所示。

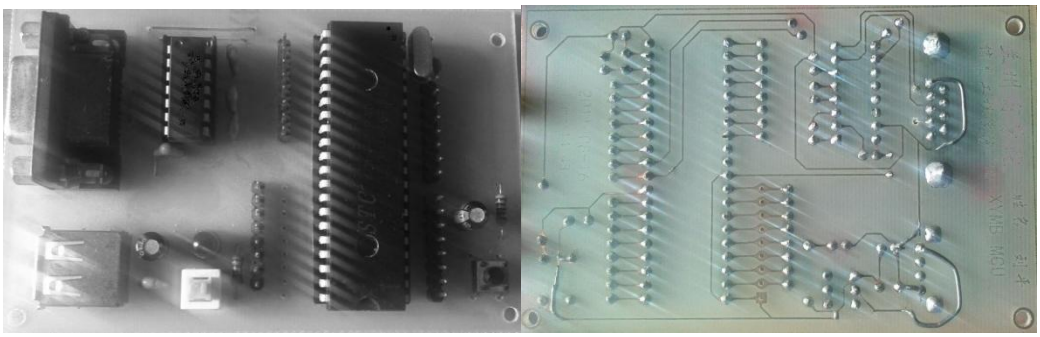

图 3-39 感光板制作的 PCB 实物图

3.5.4 工厂 PCB 板搭建版

这是有钱人玩的“游戏”。此种方法需要大家会绘制 PCB，这个知识点，残弈悟恩会在后面做详解，这里先不赘。对于初学者来说很少用此方法来制板，若真的想打一块板子，体验一下工厂板的 PCB，读者画好之后一定要仔细检查，多检查几遍无妨，别等板子打样回来一调试，发现就是因为自己小小的疏忽而功亏一篑，多可惜啊！

总结以上四种版本，笔者强烈建议读者掌握前两种，后两种就仁者见仁，智者见智了。