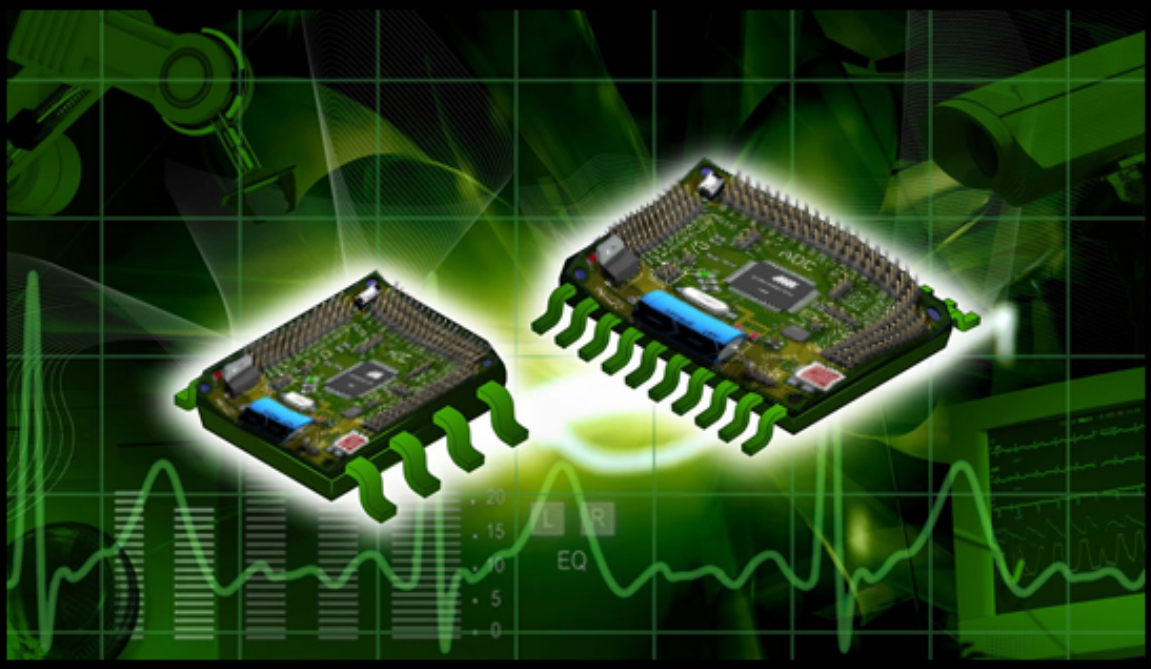
单片机应用系统开发典型实例系列

单片机开发实例大全

——OFweek电子工程网编辑团队出品



OFweek电子工程网 创新设计系列电子书

——单片机 C51 编程规范

1 单片机 C51 编程规范— 前言

为了提高源程序的质量和可维护性，从而最终提高软件产品生产力，特编写此规范。

2 单片机 C51 编程规范—范围

本标准规定了程序设计人员进行程序设计时必须遵循的规范。本规范主要针对 C51 编程语言和 keil 编译器而言，包括排版、注释、命名、变量使用、代码可测性、程序效率、质量保证等内容。

3 单片机 C51 编程规范—总则

格式清晰

注释简明扼要

命名规范易懂

函数模块化

程序易读易维护

功能准确实现

代码空间效率和时间效率高

适度的可扩展性

4 单片机 C51 编程规范—数据类型定义

编程时统一采用下述新类型名的方式定义数据类型。

建立一个 datatype.h 文件，在该文件中进行如下定义：

```
typedef bit BOOL; // 位变量 //

typedef unsigned char INT8U; // 无符号 8 位整型变量 //

typedef signed char INT8S; // 有符号 8 位整型变量 //

typedef unsigned int INT16U; // 无符号 16 位整型变量 //

typedef signed int INT16S; // 有符号 16 位整型变量 //

typedef unsigned long INT32U; // 无符号 32 位整型变量 //

typedef signed long INT32S; // 有符号 32 位整型变量 //

typedef float FP32; // 单精度浮点数（32 位长度） //

typedef double FP64; // 双精度浮点数（64 位长度） //
```

5 单片机 C51 编程规范—标识符命名

5.1 命名基本原则

命名要清晰明了，有明确含义，使用完整单词或约定俗成的缩写。通常，较短的单词可通过去掉元音字母形成缩写；较长的单词可取单词的头几个字母形成缩写。即“见名知意”。

命名风格要自始至终保持一致。

命名中若使用特殊约定或缩写，要有注释说明。

除了编译开关/头文件等特殊应用，应避免使用以下划线开始和/或结尾的定义。

同一软件产品内模块之间接口部分的标识符名称之前加上模块标识。

5.2 宏和常量命名

宏和常量用全部大写字母来命名，词与词之间用下划线分隔。对程序中用到的数字均应用有意义的枚举或宏来代替。

5.3 变量命名

变量名用小写字母命名，每个词的第一个字母大写。类型前缀（u8s8 etc.）全局变量另加前缀 g_。

局部变量应简明扼要。局部循环体控制变量优先使用 i、j、k 等；局部长度变量优先使用 len、num 等；临时中间变量优先使用 temp、tmp 等。

5.4 函数命名

函数名用小写字母命名，每个词的第一个字母大写，并将模块标识加在最前面。

5.5 文件命名

一个文件包含一类功能或一个模块的所有函数，文件名称应清楚表明其功能或性质。

每个.c 文件应该有一个同名的.h 文件作为头文件。

6 单片机 C51 编程规范－注释

6.1 注释基本原则

有助于对程序的阅读理解，说明程序在“做什么”，解释代码的目的、功能和采用的方法。

一般情况源程序有效注释量在 30%左右。

注释语言必须准确、易懂、简洁。

边写代码边注释，修改代码同时修改相应的注释，不再有用的注释要删除。

6.2 文件注释

文件注释必须说明文件名、函数功能、创建人、创建日期、版本信息等相关信息。

修改文件代码时，应在文件注释中记录修改日期、修改人员，并简要说明此次修改的目的。所有修改记录必须保持完整。

文件注释放在文件顶端，用“/*……*/”格式包含。

注释文本每行缩进 4 个空格；每个注释文本分项名称应对齐。

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

    文件名称：

    作 者：

    版 本：

    说 明：

    修改记录：

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

6.3 函数注释

6.3.1 函数头部注释

函数头部注释应包括函数名称、函数功能、入口参数、出口参数等内容。如有必要还可增加作者、创建日期、修改记录（备注）等相关项目。

函数头部注释放在每个函数的顶端，用“/*……*/”的格式包含。其中函数名称应简写为 FunctionName（），不加入、出口参数等信息。

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

    函数名称：
```

函数功能：

入口参数：

出口参数：

备 注：

****

*/

6.3.2 代码注释

代码注释应与被注释的代码紧邻，放在其上方或右方，不可放在下面。如放于上方则需与其上面的代码用空行隔开。一般少量注释应该添加在被注释语句的行尾，一个函数内的多个注释左对齐；较多注释则应加在上方且注释行与被注释的语句左对齐。

函数代码注释用“//…//”的格式。

通常，分支语句（条件分支、循环语句等）必须编写注释。其程序块结束行“}”的右方应加表明该程序块结束的标记“end of ……”， 尤其在多重嵌套时。

6.4 变量、常量、宏的注释

同一类型的标识符应集中定义，并在定义之前一行对其共性加以统一注释。对单个标识符的注释加在定义语句的行尾。

全局变量一定要有详细的注释，包括其功能、取值范围、哪些函数或过程存取它以及存取时的注意事项等。

注释用“//…//”的格式。

7 单片机 C51 编程规范－函数

7.1 设计原则

函数的基本要求：

*正确性：程序要实现设计要求的功能。

*稳定性和安全性：程序运行稳定、可靠、安全。

*可测试性：程序便于测试和评价。

*规范／可读性：程序书写风格、命名规则等符合规范。

*扩展性：代码为下一次升级扩展留有空间和接口。

*全局效率：软件系统的整体效率高。

*局部效率：某个模块／子模块/函数的本身效率高。

编制函数的基本原则：

*单个函数的规模尽量限制在 200 行以内（不包括注释和空行）。一个函数只完成一个功能。

*函数局部变量的数目一般不超过 5～10 个。

*函数内部局部变量定义区和功能实现区（包含变量初始化）之间空一行。

*函数名应准确描述函数的功能。通常使用动宾词组为执行某操作的函数命名。

*函数的返回值要清楚明了，尤其是出错返回值的意义要准确无误。

*不要把与函数返回值类型不同的变量，以编译系统默认的转换方式或强制的转换方式作为返回值返回。

*减少函数本身或函数间的递归调用。

*尽量不要将函数的参数作为工作变量。

7.2 函数定义

*函数若没有入口参数或者出口参数，应用 void 明确申明。

*函数名称与出口参数类型定义间应该空一格且只空一格。

*函数名称与括号（）之间无空格。

*函数形参必须给出明确的类型定义。

*多个形参的函数，后一个形参与前一个形参的逗号分割符之间添加一个空格…

——uCOS-II 在 51 单片机上的移植

引言：随着各种应用电子系统的复杂化和系统实时性需求的提高，并伴随应用软件朝着系统化方向发展的加速，在 16 位/32 位单片机中广泛使用了嵌入式实时操作系统。然而实际使用中却存在着大量 8 位单片机，从经济性考虑，对某些应用场合，在 8 位 MCU 上使用操作系统是可行的。从学习操作系统角度，uC/OS- II for 51 即简单又全面，学习成本低廉，值得推广。

结语：μC/OS-II 具有免费、简单、可靠性高、实时性好等优点，但也有缺乏便利开发环境等缺点，尤其不像商用嵌入式系统那样得到广泛使用和持续的研究更新。但开放性又使得开发人员可以自行裁减和添加所需的功能，在许多应用领域发挥着独特的作用。当然，是否在单片机系统中嵌入 μC/OS-II 应视所开发的项目而定，对于一些简单的、低成本的项目来说，就没必要使用嵌入式操作系统了。

uC/OS-II 原理：

uCOSII 包括任务调度、时间管理、内存管理、资源管理（信号量、邮箱、消息队列）四大部分，没有文件系统、网络接口、输入输出界面。它的移植只与 4 个文件相关：汇编文件（OS_CPU_A.ASM）、处理器相关 C 文件（OS_CPU.H、OS_CPU_C.C）和配置文件（OS_CFG.H）。有 64 个优先级，系统占用 8 个，用户可创建 56 个任务，不支持时间片轮转。它的基本思路就是 “近似地每时每刻总是让优先级最高的就绪任务处于运行状态” 。为了保证这一点，它在调用系统 API 函数、中断结束、定时中断结束时总是执行调度算法。原作者通过事先计算好数据，简化了运算量，通过精心设计就绪表结构，使得延时可预知。任务的切换是通过模拟一次中断实现的。

uCOSII 工作核心原理是：近似地让最高优先级的就绪任务处于运行状态。

操作系统将在下面情况中进行任务调度：调用 API 函数（用户主动调用），中断（系统占用的时间片中断 OsTimeTick（），用户使用的中断）。

调度算法书上讲得很清楚，我主要讲一下整体思路。

（1） 在调用 API 函数时，有可能引起阻塞，如果系统 API 函数察觉到运行条件不满足，需要切换就调用 OSSched（）调度函数，这个过程是系统自动完成的，用户没有参与。OSSched（）判断是否切换，如果需要切换，则此函数调用 OS_TASK_SW（）。这个函数模拟一次中断（在 51 里没有软中断，我用子程序调用模拟，效果相同），好象程序被中断打断了，其实是 OS 故意制造的假象，目的是为了任务切换。既然是中断，那么返回地址（即紧邻 OS_TASK_SW（）的下一条汇编指令的 PC 地址）就被自动压入堆栈，接着在中断程序里保存 CPU 寄存器（PUSHALL）……。堆栈结构不是任意的，而是严格按照 uCOSII 规范处理。OS 每次切换都会保存和恢复全部现场信息（POPALL），然后用 RETI 回到任务断点继续执行。这个断点就是 OSSched（）函数里的紧邻 OS_TASK_SW（）的下一条汇编指令的 PC 地址。切换的整个过程就是，用户任务程序调用系统 API 函数，API 调用 OSSched（），OSSched（）调用软中断 OS_TASK_SW（）即 OSCtxSw，返回地址（PC 值）压栈，进入 OSCtxSw 中断处理子程序内部。反之，切换程序调用 RETI 返回紧邻 OS_TASK_SW（）的下一条汇编指令的 PC 地址，进而返回 OSSched（）下一句，再返回 API 下一句，即用户程序断点。因此，如果任务从运行到就绪再到运行，它是从调度前的断点处运行。

（2）中断会引发条件变化，在退出前必须进行任务调度。 uCOSII 要求中断的堆栈结构符合规范，以便正确协调中断退出和任务切换。前面已经说到任务切换实际是模拟一次中断事件，而在真正的中断里省去了模拟 （本身就是中断嘛）。只要规定中断堆栈结构和 uCOSII 模拟的堆栈结构一样，就能保证在中断里进行正确的切换。任务切换发生在中断退出前，此时还没有返回中断断点。仔细观察中断程序和切换程序最后两句，它们是一模一样的，POPALL+RETI。即要么直接从中断程序退出，返回断点；要么先保存现场到 TCB，

等到恢复现场时再从切换函数返回原来的中断断点（由于中断和切换函数遵循共同的堆栈结构，所以退出操作相同，效果也相同）。用户编写的中断子程序必须按照 uCOSII 规范书写。任务调度发生在中断退出前，是非常及时的，不会等到下一时间片才处理。OSIntCtxSw（）函数对堆栈指针做了简单调整，以保证所有挂起任务的栈结构看起来是一样的。

（3）在 uCOSII 里，任务必须写成两种形式之一（《uCOSII 中文版》p99 页）。在有些 RTOS 开发环境里没有要求显式调用 OSTaskDel（），这是因为开发环境自动做了处理，实际原理都是一样的。uCOSII 的开发依赖于编译器，目前没有专用开发环境，所以出现这些不便之处是可以理解的。

移植过程：

（1）拷贝书后附赠光盘 sourcecode 目录下的内容到 C：\YY 下，删除不必要的文件和 EX1L.C，只剩下 p187（《uCOSII》）上列出的文件。

（2）改写最简单的 OS_CPU.H

数据类型的设定见 C51.PDF 第 176 页。注意 BOOLEAN 要定义成 unsigned char 类型，因为 bit 类型为 C51 特有，不能用在结构体里。

EA=0 关中断；EA=1 开中断。这样定义即减少了程序行数，又避免了退出临界区后关中断造成的死机。

MCS-51 堆栈从下往上增长（1=向下，0=向上），OS_STK_GROWTH 定义为 0

#define OS_TASK_SW（） OSCtxSw（） 因为 MCS-51 没有软中断指令，所以用程序调用代替。两者的堆栈格式相同，RETI 指令复位中断系统，RET 则没有。实践表明，对于 MCS-51，用子程序调用入栈，用中断返回指令 RETI

出栈是没有问题的，反之中断入栈 RET 出栈则不行。总之，对于入栈，子程序调用与中断调用效果是一样的，可以混用。在没有中断发生的情况下复位中断系统也不会影响系统正常运行。详见《uC/OS-II》第八章 193 页第 12 行

（3）改写 OS_CPU_C.C

我设计的堆栈结构如下图所示：

TCB 结构体中 OSTCBStkPtr 总是指向用户堆栈最低地址，该地址空间内存放用户堆栈长度，其上空间存放系统堆栈映像，即：用户堆栈空间大小=系统堆栈空间大小+1。

SP 总是先加 1 再存数据，因此，SP 初始时指向系统堆栈起始地址（OSStack）减 1 处（OSStkStart）。很明显系统堆栈存储空间大小=SP-OSStkStart。

任务切换时，先保存当前任务堆栈内容。方法是：用 SP-OSStkStart 得出保存字节数，将其写入用户堆栈最低地址内，以用户堆栈最低地址为起址，以 OSStkStart 为系统堆栈起址，由系统栈向用户栈拷贝数据，循环 SP-OSStkStart 次，每次拷贝前先将各自栈指针增 1。

其次，恢复最高优先级任务系统堆栈。方法是：获得最高优先级任务用户堆栈最低地址，从中取出“长度”，以最高优先级任务用户堆栈最低地址为起址，以 OSStkStart 为系统堆栈起址，由用户栈向系统栈拷贝数据，循环“长度”数值指示的次数，每次拷贝前先将各自栈指针增 1。

用户堆栈初始化时从下向上依次保存：用户堆栈长度（15），PCL，PCH，PSW，ACC，B，DPL，DPH，R0，R1，R2，R3，R4，R5，R6，R7。不保存 SP，任务切换时根据用户堆栈长度计算得出。

OSTaskStkInit 函数总是返回用户栈最低地址。

操作系统 tick 时钟我使用了 51 单片机的 T0 定时器，它的初始化代码用 C 写在了本文件中。

最后还有几点必须注意的事项。本来原则上我们不用修改与处理器无关的代码，但是由于 KEIL 编译器的特殊性，这些代码仍要多处改动。因为 KEIL

缺省情况下编译的代码不可重入，而多任务系统要求并发操作导致重入，所以要在每个 C 函数及其声明后标注 reentrant 关键字。另外，“pdata”、“data”在 uCOS 中用做一些函数的形参，但它同时又是 KEIL 的关键字，会导致编译错误，我通过把“pdata”改成“ppdata”， “data”改成“ddata”解决了此问题。OSTCBCur、OSTCBHighRdy、OSRunning、OSPrioCur、OSPrioHighRdy 这几个变量在汇编程序中用到了，为了使用 Ri 访问而不用 DPTR，应该用 KEIL 扩展关键字 IDATA 将它们定义在内部 RAM 中。

（4）重写 OS_CPU_A.ASM

A51 宏汇编的大致结构如下：

NAME 模块名 ;与文件名无关

;定义重定位段 必须按照 C51 格式定义，汇编遵守 C51 规范。段名格式为：？PR？函数名？模块名

;声明引用全局变量和外部子程序 注意关键字为“EXTRN”没有‘E’

全局变量名直接引用

无参数/无寄存器参数函数 FUNC

带寄存器参数函数 _FUNC

重入函数 _? FUNC

分配堆栈空间

只关心大小，堆栈起点由 keil 决定，通过标号可以获得 keil 分配的 SP 起点。切莫自己分配堆栈起点，只要用 DS 通知 KEIL 预留堆栈空间即可。

STACK 段名与 STARTUP.A51 中的段名相同，这意味着 KEIL 在 LINK 时将把两个同名段拼在一起，我预留了 40H 个字节，STARTUP.A51 预留了 1 个字节,LINK 完成后堆栈段总长为 41H。查看 yy.m51 知 KEIL 将堆栈起点定在 21H，长度 41H，处于内部 RAM 中…

——基于单片机的大棚温湿度控制系统设计

0 引言

植物的生长都是在一定的环境中进行的，在生长过程中受到环境中各种因素的影响，其中影响最大的是温度和湿度。若昼夜的温度和湿度变化很大，其对植物生长极为不利。因此必须对温度和湿度进行监测和控制，使其适合植物的生长，以提高其产量和质量。

本系统就是针对大棚内温度、湿度，研究单片机控制的温室大棚自动控制，综合考虑系统的精度、效率以及经济性要求多方面因素之后，设计一种基于计算机自动控制的大棚温湿度控制系统。

本系统实现的蔬菜大棚温湿度控制系统的目标功能如下：

（1）系统能对大棚环境温湿度进行采集和显示（现场观温、湿度，软件记录）。

（2）能通过上位机端远程设定蔬菜的生长期适宜温湿度。由主控机统一设置系统时间和温度湿度修正值。

（3）当大棚的环境温湿度参数超过设定的上下限值时控制相应的系统启动。

（4）可实时显示当前温度、时间、报警阈值等信息，并可查询各时间段的温湿度情况，并加以控制。

1 系统各组成模块

本系统通过温度传感器 DS18B20 采集温度，HM1500LF 采集湿度，经过含有单片机的检测系统的进一步分析处理，通过通信线路将信息上行到 PC 机，在 PC 机上可对温湿度信号进行任何分析、处理。用户可以通过下位机中的键盘输入温湿度的上下限值和预置值，也可以通过上位机进行输入，从而实现上位机对大棚内作物生长的远程控制。如果环境的实时参数超越上下限值，系统自动启动执行机构调节大棚内温度和湿度状态，直到温湿度状态处于上下限值内为止。如果有预置初值，且与当前状态不相等时，系统也会启动执行机构实时动态调节温湿度状态，直到所处的平衡状态与预置值相等为止。

上位机即 PC 机使用 DELPHI 软件编写的一个数据库管理系统，可直接设置温度的上下限值和读取下位机的数据，并对下位机内的控制设备进行操作，调节大棚内温湿度状态。形成作物生长的走势图，从而通过生长走势图得出适合各种作物生长的最佳环境参数条件，为今后的温室种植提供参考。

上下位机之间通过符合串行总线 RS 232 标准的通信通道以事先约定的协议进行通信。系统原理图如图 1 所示。

2 总体电路及工作过程说明

使用智能温度传感器 DS18B20 进行组网来测量各个采集点的温度，HM1500LF 来采集湿度，单片机 AT89S52 作为该系统的处理核心，单片机根据温湿度传感器检测到的数据，把各个测量点的温湿度存储并显示在 LCD 液晶显示器上，同时显示在 PC 机上。

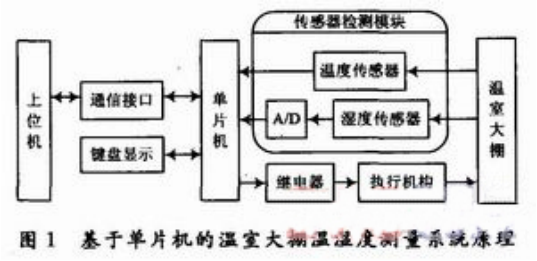


图 1 基于单片机的温室大棚温湿度测量系统原理

3 数据采集模块

本模块主要采用 DS18B20 采集温度，HM1500LF 采集湿度，由单片机 AT89S52 作总的控制并显示与传输。具体原理图如图 2 所示。

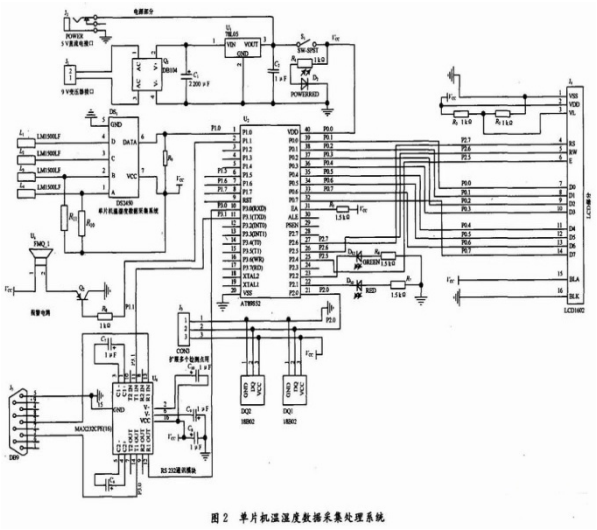


图 2 单片机温湿度数据采集处理系统

3．1 温度传感器 DS18B20

DS18B20 是 Dallas 公司生产的一线式数字温度传感器，采用的是单总线数据传输方式，数据的输入、输出都通过同一条线，因此对时序有很高的要求，为了保证时序，需要做精确的延时，较短的延时可以通过用_nop_（）来实现，根据 DS18B20 的读写时序，用到的延时有 15μs，45μs，90μs，270μs，540μs 等，因这些延时为 15μs 的整数倍，因此可编写一个 Delay15（n）函数，用该函数进行大约 15μs×n 的延时，非常方便。程序如下：

```
void delay15(unsigned char n)
{ do{
    _nop_();
    ;
    _nop_();//共 13 个
_nop_();//
    n--;
    }while(n! =0);
}
```

有了比较精确的延时保证，就可以对 DS18B20 进行初始化、数据写、数据读。根据时序图，不难写出相应的函数。

3．2 湿度传感器 HM1500LF

湿度传感器 HM1500LF 是法国 Humirel 公司生产的一种低价位的线性电压输出湿度传感器，HM1500LF 的测湿元件选用湿敏电容，利用电容量与相对湿度的函数关系即可测量湿度。DS2450 是美国 Dallas 公司最新推出的一种符合单总线协议的可组网集成 A／D 芯片，四个湿度传感器分别接到一片 DS2450 的四个模拟电压输入通道 A，B，C，D 上，电路采用+5 V 电源供电，必须在上电完毕后向地址 1CH 写入 40H，使模拟电路永久地保持在工作状态。利用该电路湿度检测信号在测量现场就被直接转换为数字信号，因此 HM1500LF 和 DS2450 组合在一起，就构成一个单总线数字湿度传感器模块。

4 单片机软件

整个系统的功能是由硬件电路配合软件来实现的，当硬件基本定型后，软件的相应子程序模块就大体定下来了。单片机程序组成如图 3 所示。

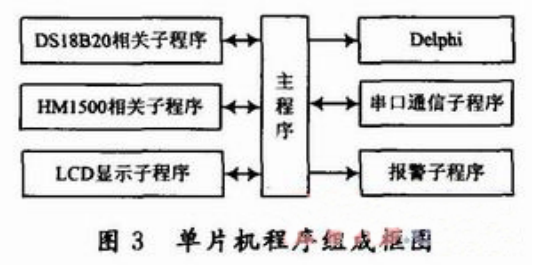


图 3 单片机程序组成框图

4．1 系统的主程序设计

主程序是系统的监控程序，在程序运行的过程中必须先经过初始化，流程图如图 4 所示。系统在初始化完成后就进入温度测量程序，实时地测量当前的温度并通过显示电路在 LCD 上显示。程序中以中断的方式来重新设定温度的上下限。根据硬件设计完成对温度的控制。系统软件设计的总体流程图如图 4 所示…

——基于 L297/L298 芯片步进电机的单片机控制

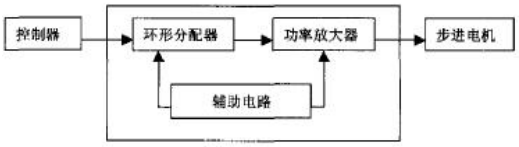


图 1 典型步进电机控制框图

1 引言

步进电动机是一种将电脉冲信号转换成角位移或线位移的精密执行元件，由于步进电机具有控制方便、体积小等特点，所以在数控系统、自动生产线、自动化仪表、绘图机和计算机外围设备中得到广泛应用。微电子学的迅速发展和微型计算机的普及与应用，为步进电动机的应用开辟了广阔前景，使得以往用硬件电路构成的庞大复杂的控制器得以用软件实现，既降低了硬件成本又提高了控制的灵活性，可靠性及多功能性。市场上有很多现成的步进电机控制机构，但价格都偏高。应用 SGS 公司推出的 L297 和 L298 两芯片可方便的组成步进电机驱动器，并结合 AT89C52 单片机进行控制，即可以实现用相对便宜的价格组成一个性能不错的步进电机驱动电路。

2 工作原理

由于步进电机是一种将电脉冲信号转换成直线或角位移的执行元件，它不能直接接到交直流电源上，而必须使用专用设备-步进电机控制驱动器 典型步进电机控制系统如图 1 所示：控制器可以发出脉冲频率从几赫兹到几十千赫兹可以连续变化的脉冲信号，它为环形分配器提供脉冲序列。环形分配器的主要功能是把来自控制环节的脉冲序列按一定的规律分配后，经过功率放大器的放大加到步进电机驱动电源的各项输入端，以驱动步进电机的转动。环形分配器主要有两大类：一类是用计算机软件设计的方法实现环分器要求的功能，通常称软环形分配器。另一类是用硬件构成的环形分配器，通常称

为硬环形分配器。功率放大器主要对环形分配器的较小输出信号进行放大。以达到驱动步进电机目的。

3 硬件组成

文中所控制的步进电机是四相单极式 35BY48HJ120 减速步进电动机。本文所设计的步进电机控制驱动器的框图如图 2 所示。它由 AT89C52 单片机、光电耦和器、集成芯片 L297 和 L298 组成。AT89C52 是美国 ATMEL 的低电压、高性能 8 位 CMOS 单片机。片内置 8K 字节可重复擦写的

Flash 闪速存储器。256 字节 RAM。3 个 16 位定时器。可编程串行 UART 通道。对完成步进电机的简单控制已足以胜任。

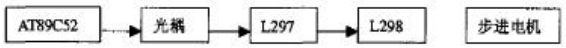
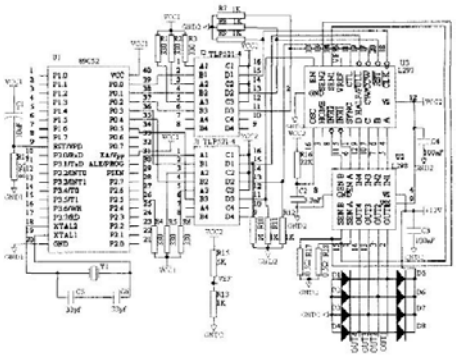


图 2 本文提出的步进电机控制驱动器框图

L297 是步进电动机控制器(包括环形分配器)。L298 是双 H 桥式驱动器。它们所组成的微处理器至双桥式步进电动机的接口如图 3 所示。这种方式结合的优点是，需要的元件很少。从而使得装配成本低，可靠性高和占空间少。

并且通过软件开发。可以简化和减轻微型计算机的负担。另外，L297 和 L298 都是独立的芯片。所以应用是十分灵活的。



L297 芯片是一种硬件环分集成芯片。它可产生四相驱动信号，用于计算机控制的两相双极或四相单极步进电机 它的心脏部分是一组译码器它能产生各种所需的相序。这一部分是由两种输入模式控制，方向控制（CW/CCW） 和 HALF/FULL 以及步进式时钟 CLOCK.它将译码器从一阶梯推进至另一阶梯。译码器有四个输出点连接到输出逻辑部分，提供抑制和斩波功能所需的相序。因此 L297 能产生三种相序信号，对应于三种不同的工作方式：即半步方式（HALF STEP）；基本步距（FULL STEP，整步）一相激励方式；基本步距两相激励方式。脉冲分配器内部是一个 3bit 可逆计数器，加上一些组合逻辑。产生每周期 8 步格雷码时序信号，这也就是半步工作方式的时序信号。此时 HALF/FULL 信号为高电。若 HALF/FULL 取低电平，得到基本步距工作方式。即双四拍全阶梯工作方式。

L297 另一个重要组成是由两个 PWM 斩波器来控制相绕组电流，实现恒流斩波控制以获得良好的矩频特性。每个斩波器由一个比较器、一个 RS 触发器和外接采样电阻组成，并设有一个公用振荡器，向两个斩波器提供触发脉冲信号。图 3 中，频率 f 是由外接 16 脚的 RC 网络决定的， 当 R》10kΩ 时，f=1/0.69RC。当时钟振荡器脉冲使触发器置 1，电机绕组相电流上升，采样电阻的 R 上电压上升到基准电压 Uref 时，比

较器翻转，使触发器复位，功率晶体管关断，电流下降，等待下一个振荡脉冲的到来。这样，触发器输出的是恒频 PWM 信号，调制 L297 的输出信号，绕组相电流峰值由 Uref 确定。L297 的 CONTROL 端的输入决定斩波器对相位线 A、B、C、D 或抑制线 INH1 和 INH2 起作用。CONTROL 为高电平时，对 A、B、C、D 有控制作用；而为低电平时，则对 INH1 和 INH2 起控制作用，从而可对电动机转向和转矩进行控制。

L298 芯片是一种高压、大电流双全桥式驱动器，其设计是为接受标准 TTL 逻辑电平信号和驱动电感负载的，例如继电器、圆筒形线圈、直流电动机和步进电动机 具有两抑制输入来使器件不受输入信号影响。每桥的三级管的射极是连接在一起的，相应外接线端可用来连接外设传感电阻。可安置另一输入电源，使逻辑能在低电压下工作。L298 芯片是具有 15 个引出脚的多瓦数直插式封装的集成芯片。

图 3 中.AT89C52 通过串口经 MAX232 电平转换之后与微机相连。接受上位机指令。向 L297 发出时钟信号、正反转信号、复位信号及使能控制等信号。电路中，电阻 R13，R15 用来调节斩波器电路的参考电压，该电压将与通过管脚 13，14 所反馈的电位的大小比较，来确定是否进行斩波控制，以达到控制电机绕组电流峰值、保护步进电机的目的

由于 L297 内部带有斩波恒流电路，绕组相电流峰值由 Uref 确定。当采用两片 L297 通过 L298 分别驱动步进电机的两绕组，且通过两个 D/A 转换器改变每相绕组的 Uref 时，即组成了步进电机细分驱动电路。另外，为了有效地抑制电磁干扰，提高系统的可靠性，在单片机与步进电动机驱动回路中利用两个 16 引脚光电耦合器件 TLP521-4 组成如图 3 所示的隔离电路。其作用是切断了单片机与步进电动机驱动回路之间电的直接联系，实现了单片机与驱动回路系统地线的分别联接。防止处于大电流感性负载下工作的驱动电路产生的干扰信号以及电网负载突变产生的干扰信号通过线路串入单片机，影响单片机的正常工作…

继续阅读文章

——基于单片机的智能太阳能路灯控制系统设计方案

摘要：随着世界能源危机日益严重，利用太阳能成为解决能源问题的一大途径，在此背景下开发智能太阳能路灯意义重大。本文介绍了智能太阳能路灯系统的组成及工作原理，采用 LPC935 单片机作为主控制器，结合密封铅酸蓄电池充电专用芯片 UC3906，实现了对密封铅酸蓄电池最佳充电所需的全部控制和检测功能，延长了系统的使用寿命。通过热释电红外、微波双鉴传感器技术及以无线通讯技术，实现了红外微波探测、相邻路灯间的无线通讯以及主副灯的智能化切换，达到了节能减排的效果。

随着科学技术的迅速发展，世界能源危机日益严重，利用常规能源已不能适应世界经济快速增长的需要，开发和利用新能源越来越引起各国的重视。太阳能源本身的安全可靠、无噪声、无污染和可再生性的特点，加之现今光伏技术的逐渐成熟，利用光伏发电成为解决能源问题的一大途经。

智能太阳能路灯是利用太阳能组件的光生伏特效应，将光能转换为电能，并储存在蓄电池中供负载使用，它是集太阳能光伏技术、蓄电池技术、照明光源技术于一体的新兴技术。太阳能路灯控制器是应用于太阳能光伏系统中，协调太阳能电池板、蓄电池、负载的工作，使整个太阳能光伏系统高效，安全的运作。

1 智能太阳能路灯系统总体方案

智能太阳能路灯系统的由太阳能电池板、蓄电池、LED 灯（ 主灯、副灯）和控制器组成（ 如图 1 所示） 。白天太阳能电池板接受太阳辐射能并转化为电能输出，经过充电控制电路储存在蓄电池中； 晚间当光线照度降低时，控制器使副灯点亮，进行指示性照明。当控制器监测到有人经过时控制器同时点亮主灯和副灯，同时和相邻前后的灯通讯，控制邻灯主灯和副灯同时点亮，保证行人在该路段的照明。控制器检测到蓄电池充电或放电超出一定范围时，控制器切断充放电回路，保证电池不被损坏。遇到连续阴雨天季节可切换成市电照明，避免蓄电池长期亏电。

2 控制系统硬件电路图设计

系统硬件是基于 P89PLC935 单片机作为主控制器的基础，设计出符合功能要求的各个子模块，原理见图 2。

（ 1） 控制器

控制器选用 P89LPC935 单片机，它是一款单片封装的微控制器，适合于本系统要求的高集成度、底成本的场合，可以满足多方面的性能要求，LPC935 采用了高性能的处理器结构，指令执行时间只需 2 － 4 个时钟周期，6 倍于标准 80C51，同时，LPC935 集成了许多系统级的功能，这样可大大减少元件的数目，它的 8KBROM 能满足本系统程序存储器的要求，不需用扩展 EPROM。

该单片机内置的 2 个 4 路输入的 8 位 A/D 转换器，不需再单独选用 A/D 转换器，简化了外围硬件电路，P89LPC935 内部的看门狗电路及低电压掉电

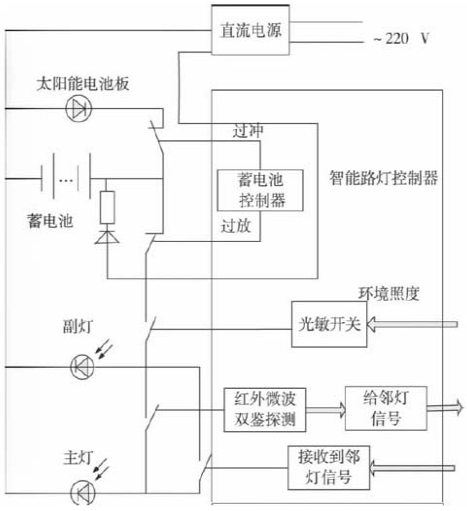


图 1 智能太阳能路灯系统总体方案

检测可在电源故障和受到强电磁干扰时使系统可靠复位，提高了系统的安全可靠性。

（ 2） 环境照度的检测

本系统采用光敏开关检测环境照度。环境照度检测是整个路灯的总开关，只有在夜晚，环境照度较低的情况下，主副灯、人体感应单元及相应的控制电路开始工作，白天均不工作。白天时光敏电阻阻值小，比较器 LM358 负端电压高于正端电压，比较器输出低电平，单片机接收到低电平，屏蔽各种通讯和感应信号，夜晚光敏电阻阻值大，比较器负端电压小于正端，输出高电平，单片机控制接收感应信号和通讯信号。

（ 3） 人体感应单元

本系统采用被动式热释电红外、微波双鉴传感器作为人体感应单元。由于人体都有恒定的体温，一般在 36. 5℃，所以会发出特定波长，一般是 10μm 左右的红外线。人体发射的 10um 左右的红外线通过菲涅尔滤光片增强后聚集到热释电元件上，热释电元件接受到人体红外辐射温度发生变化时失去电荷平衡，向外释放电荷，经后续电路检测处理并产生报警信号［2］，但是，热气流，暖风也会造成被动式热释电红外探头发出错误信号，造成和相邻灯之间的误通讯。为了避免误通讯…

——基于 MCS-51 单片机的智能机器人迷宫车设计

机器人应当具有几个特征：移动功能，执行功能、感觉和智能。目前全世界各国举办的涉及硬件，软件仿真的机器人大赛不下 20 余类。各种各样的机器人比赛都有一个共同的宗旨：培养科学创新精神，激发思维的想象力，鼓励理论与实践的结合。不仅如此，现在已经有越来越多的自动控制产品已经介入生产，在农业、工业上都有广泛的应用。新的工作方式将大大的缩短了人工作业的时间，并且减轻了人的体力劳动的支出。走迷宫的微型机器鼠主要是基于自动引导小车（Auto Guided Vehicle，AGV）的原理，实现机器鼠识别路线，判断并自动躲避障碍，选择正确的行进路线走出迷宫。在此选择制作一个简易的行进装置，使其能顺利的走避障或是迷宫。为了实现小车识别路线，判断并自动躲避障碍，选择正确的行进路线，障碍判断采用单光束反射取样红外传感器，驱动电机采用直流电机，控制核心采用 MCS-51 单片机。控制上采用分时复用技术，仅用一块单片机实现了信号采集、线路判断、电机控制等功能。迷宫由 16×16 个区组成。起点设在拐角处，终点设在中央，占 4 个区。每个区为 180 mm×180 mm 大小，间壁高为 50 mm，厚度为 12 mm，侧面涂白色，底面涂黑色，如图 1 所示。

1 迷宫车控制系统的总体设计方案

迷宫车由墙壁传感器、单片机控制板、动力及转向系组成的，控制框图如图 2 所示。

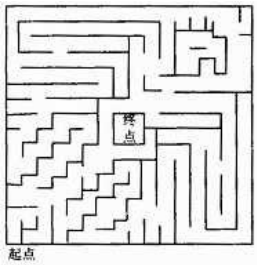


图 1 机器鼠比赛的迷宫

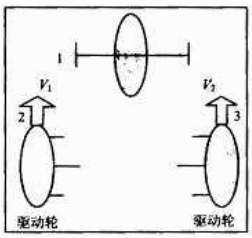


图 2 迷宫车的控制框图

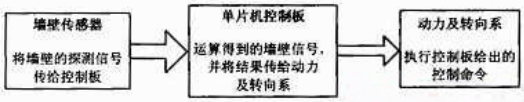


图 3 车轮移动方式图

迷宫车采用轮式移动方式。优点是：结构和控制简单而且技术成熟。从选定电动机转速和轮胎直径，可以简单地计算出小车的速度。但是，有关路面的阻力或上坡的驱动转矩等成为重要的因素。考虑这点，在轮胎上使用无线遥控车的塑胶轮胎。如图 3 所示，前轮 1 为万向脚轮或球形轮，后轮 2 和后轮 3 为独立驱动轮，利用它们的转速差实现转向。这种组合的特点是机枢组合容易，而且当 2 个驱动轮以相同速度、相反方向转动时车体能绕 2 个驱动轮连线的中点自转，值得注意的是自转中心与车体中心不一致。

迷宫车车身材料的选择。迷宫车使用的材料大部分用于结构，一般应采用金属材料。迷宫车承载和运动不应产生严重的变形和断裂，从力学角度讲即具有足够的强度。迷宫车负载小，自重轻，对寿命的要求不高。因此，选用铁皮。

1．1 迷宫车控制电路的设计

控制电路主要由电机驱动电路，单片机接口电路，电源电路和传感器电路组成。控制框图如图 4 所示。

图 4 迷宫车控制框图

（1）红外线光感电路传感器通过发光二极管发出红外线，若有障碍物在前方，红外线会被反射回来，被感光三极管接收，单片机程序对信号进行比较处理，按设定的动作要求向后轮的两个电机发出控制命令，控制小车行进。

（2）电机驱动电路采用 89S51 单片机，通过 L293D 芯片来控制两个驱动电机动作。89S51 根据红外传感器对外界进行探测后反馈回来的信号，依据迷宫车探路算法，判定迷宫车行进方向，分别向左右两个驱动电机发出控制指令，该信号经 L293D 芯片驱动后，直接控制相应电机动作，使迷宫车按既定动作进行前进、后退、转向。

1．2 迷宫车控制程序设计

控制算法：

迷宫车一般有四种控制算法：

（1）靠左算法

①默认靠左走法。即一直沿着左墙壁走，左边有墙时一直沿着左边墙壁前进，当左边没有墙时左转，然后继续靠左边墙壁运行。该算法使用于最简单的迷宫走法。如图 5 所示，其中虚线表示小车前进的路线。

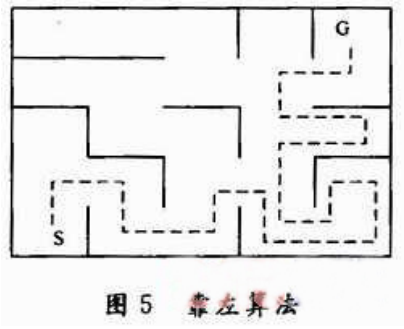


图 5 靠左算法

②算法流程图如图 6 所示。

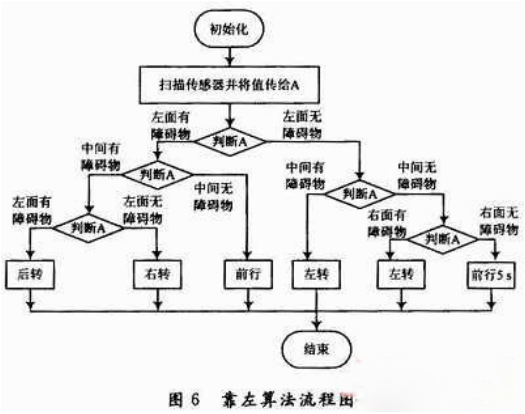


图 6 靠左算法流程图

③靠左算法的弊端。按照该算法，小车在走第二遍迷宫的时候，可以一次性的走出迷宫，但是这条道路不一定是最短的道路；如果迷宫本身存在“孤岛”，那么小车很有可能走不出迷宫。如图 7 所示。图 7 中两个圆点分别表示入口和出口，小车从入口进入迷宫，靠左前进则会导致小车一直按照虚线所描绘出的路线一直在迷宫里循环，终走不出迷宫。

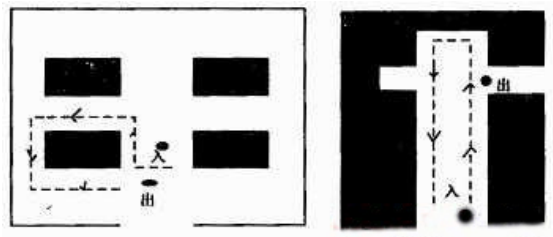


图 7 迷宫图

（2）靠前算法

①靠前算法即一直沿着前方道路前行（前方没有任何障碍时一直前进），当前边没有墙时判断左边，左边没墙左转，左边有墙则判断右边。然后重复该循环。该算法使用于最简单的迷宫走法。

②算法流程图如图 8 所示。

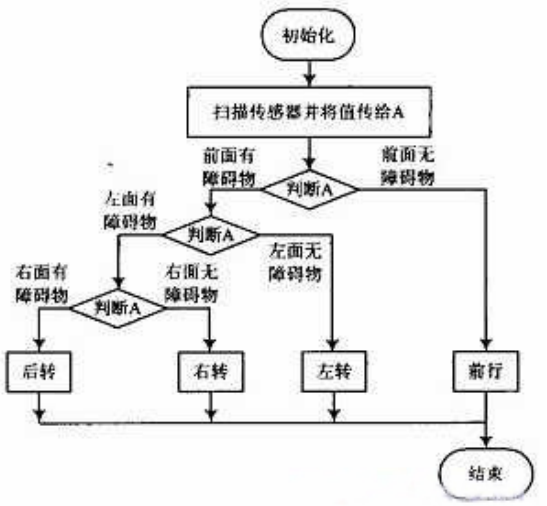


图 8 靠前算法流程图

③程序思路：前边没墙靠前走，前面有墙判断左边，左边没墙靠左转，左边前边都有墙再判断右边，右边没有墙靠右转，三面都墙直接后退转 180°，继续靠前走。0 表示有墙，1 表示没墙，p3．0 表示左传感器；p3．1 表示前传感器；p3．2 表示右传感器。

转角控制思路：计算轮子的转速，测出小车转 90°时每个轮子所行的路程，用路程来除速度，求出一个时间作为小车转弯时的延迟时间，再通过调试进一步精确转角…

——基于单片机的智能排插的设计方案

- **摘要：**顺应节能减排、环保的时代潮流，在不更换旧家用电器的基础上，设计一款基于 51 单片机控制的智能排插，赋予传统家电以遥控、定时开关、无待机功耗等功能，从而实现旧家电的节能减排和智能化目的。

1 引言

随着节能减排、绿色环保意识的不断增强，人们对家用电器的节能减排提出了更高的要求。但并不是每家每户都有资金去更换使用多年、现在仍可以使用的家用电器。为此，本文设计研发了一款基于 51 单片机控制的节能环保的智能排插，旧一代的家电通过使用这个排插可以蜕变为智能化家电，具备遥控、定时开关、无待机功耗等功能，节约了家电更新换代的成本。该排插现场运行效果良好，符合一般用户需求，具有一定的推广应用价值。

2 智能排插基本工作原理

为方便用户使用，本设计将系统分成两部分：排插控制系统和排插遥控器系统（如图 1、图 2 所示），它们分别由一片 AT89S52 单片机控制。在排插控制系统中，单片机通过无线接收模块接收遥控器的遥控命令、解码并进行相关操作；单片机通过控制与排插插孔相连的继电器来控制插孔的通断电。

当然，插孔的通断电也可以通过手动开关按钮来操控。LED 指示灯用于指示排插当前的工作状况。在排插遥控系统中，无线发射模块发送遥控指令，对排插进行定时开关或立即开关等远程操作。本系统采用成都市飞宇达实业有限公司生产的液晶显示模块 FYD12864-0402B，内置 ST7920 液晶控制器。该 LCD 用来显示遥控器人机交互界面。

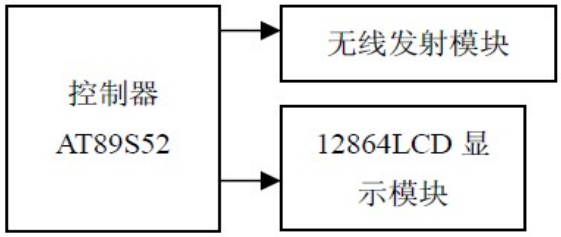


图 1 排插控制系统

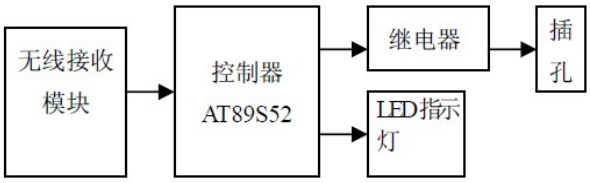


图 2 排插遥控系统

3 智能排插硬件设计

3.1 无线通讯模块与 51 单片机接口及通讯技术

3.1.1 无线收发模块与 51 单片机硬件接口

在排插系统中，为了降低生产成本，我们选用 Atmel 公司的单片机 AT89S52 为控制单元，采用 PT2272 与 PT2262 配对编、解码芯片构成无线通讯模块。无线发射模块和无线接收模块电路图如图 3、图 4 所示。

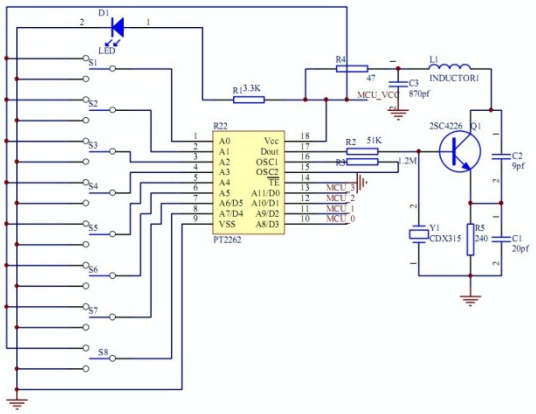


图 3 无线发射模块电路图（点击图片放大）

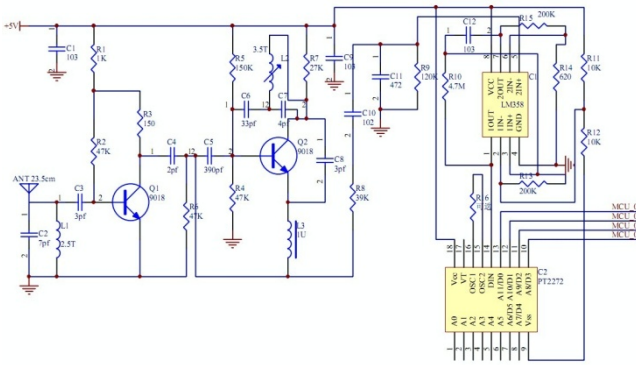


图 4 无线接收模块电路图（点击图片放大）

图 4 中，解码芯片 PT2272 的 17 脚（VT）为接收有效状态指示，它与排插控制系统上的单片机 I/O 口 P3.2（INT0）相连；10~13 脚（D0～D3）为解码后的四个数据脚，它们分别与单片机 I/O 口 P2.0～P2.3 相连；1~8 脚为三态（VSS、VDD、空）编码脚；LM358 的 1 脚（RXD）输出的是解调后的方波信号。

从逻辑分析仪采样结果分析得知，当 PT2272 接收到有效编码数据时，VT 端会输出高电平。利用此特性，可以将单片机外部中断触发方式设置为下降沿触发…

继续阅读文章

——单片机自编程及 Bootloader 设计

- Bootloader 是在单片机上电启动时执行的一小段程序。也称作固件，通过这段程序，可以初始化硬件设备、建立内存空间的映射图，从而将系统的软硬件环境带到一个合适的状态，以便为最终调用应用程序准备好正确的环境。

Boot 代码由 MCU 启动时执行的指令组成。这里的 loader 指向 MCU 的 Flash 中写入新的应用程序。因此，Bootloader 是依赖于特定的硬件而实现的，因此，在众多嵌入式产品中目前还不可能实现通用 Bootloader。

Bootloader 的最大优点是：在不需要外部编程器的情况下，对嵌入式产品的应用代码进行更新升级。它使得通过局域网或者 Intemet 远程更新程序成为可能。例如，如果有 5 000 个基于 MCU 的电能表应用程序需要更新，电能表制造商的技术人员就可以避免从事对每一个电能表重新编程的巨大工作量，通过使用 Bootloader 的功能，由控制中心通过电能表抄表系统网络，远程对 5 000 个电表重新编程。可见，Bootloader 功能对于嵌入式系统的广泛应用具有十分重要的意义。

1 78K0/Fx2 系列单片机简介

78K0/Fx2 系列是带 CAN 控制器的 8 位单片机，该系列单片机广泛应用于汽车电子，智能仪表等领域。其内置 POC（可编程上电清零电路）/LVI（可编程低电压指示器），单电压自编程闪存，引导交换功能（闪存安全保护），具有低功耗、宽电压范围、超高抗干扰等性能。

78K0 系列单片机支持自编程（Self-programming）。所谓自编程，是指用 Flash 存储器中的驻留的软件或程序对 Flash 存储器进行擦除/编程的方法。通过单片机的自编程功能，可以设计 Bootloader 程序，通过串口等通信接口实现对产品重新编程、在线升级的功能。

以 μPD78F0881 为例。μPD78F0881 为 78KO/Fx2 系列中的一款 44 管脚单片机，内置 32 KB Flash ROM，2 KB RAM，

自带 2 个串行通信接口。其内部 Flash 结构如图 1 所示。为了方便实现擦除和编程，人为地将整个 Flash 分成若干个 block，每

个 block 大小为 1 KB。block 为自编程库函数中空白检测、擦除、校验的最小单位。blockO 从地址 0000H 开始，程序都从 0000H 开始执行。block0～block3 共 4 KB 存储空间为 Bootloader 程序存储区域。block4～block31 为应用程序存储区域。

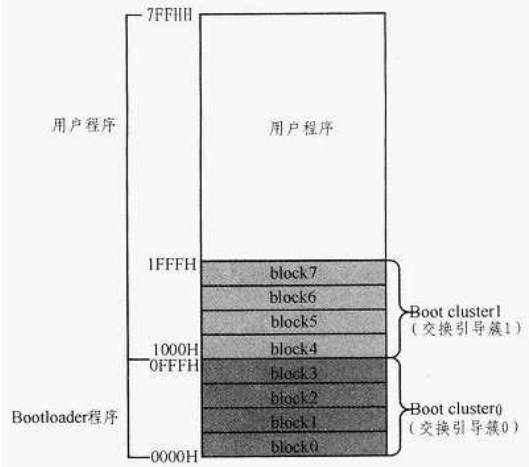


图 1 内部 Flash 结构图

为了防止 Bootloader 自身的升级失败，设计了引导交换功能。该功能定义 2 个簇，即 Boot cluster0 和 Boot cluster1。Boot clustee0 为 block0～block3 的 4 KB 存储空间，Boot cluster1 为 block4～block7 的 4 KB 存储空间。因此，实际运用过程中，一般把应用程序的开始定义在 2000H，也就是从 block8 开始。

Flash 地址为 0000H～FFFFH。7FFFFH～FFFFH 存储空间为保留区域以及特殊功能寄存器区域等，用户无法对其进行编程。

2 自编程

2.1 自编程环境

2.1.1 硬件环境

FLMDO 引脚是 78KO/Fx2 系列单片机为 Flash 编程模式设置的，用于控制 MCU 进入编程模式。在通常操作情况下，FLMDO 引脚下拉到地。要进入自编程模式，必须使 FLMDO 引脚置成高电平。因此，通过一个普通 I/O 接口控制 FLMD0 引脚的电平。如图 2 所示。

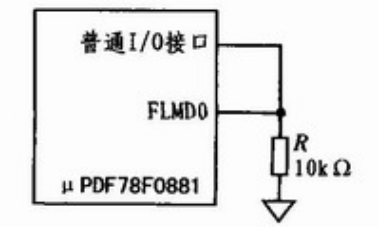


图 2 FLMD0 引脚连接示意图

2.1.2 软件环境

1）使用通用寄存器 bank3，自编程库函数，需要调用通用寄存器 bank3。因此，在自编程时，不能对通用寄存器 bank3 操作。

2）使用 100 B RAM（入口 RAM）作为隐藏 ROM 中函数的工作区，入口 RAM，是 Flash 存储器自编程样例库所使用的 RAM 区域。用户程序需要保留着块区域，当调用库时，需要指定这片区域的起始地址。入口 RAM 地址可以指定在 FB00h～FE20h 之间。

3)4～256 B RAM 作为数据缓冲区，必须是 FE20H～FE83H 以外的内部高速 RAM 区域。

4）最大 39 B RAM 作为隐藏 ROM 函数的堆栈。

5）隐藏 ROM 中的函数被 0000H～7FFFH 中的应用程序调用。

2.2 自编程流程

自编程功能利用自编程软件库完成用户程序对 Flash 内容的重新编程。如果在自编程的过程中有中断发生，那么自编程将暂停来响应中断。中断结束，自编程模式恢复后，自编程过程将继续进行。采用汇编语言编写 78K0/Fx2 自编程软件库，如表 1 所示。

表 1 78K0/Fx2 自编程库
Tab. 1 78K0/Fx2 self-programming library

库函数名称	调用函数	说明
自编程启动函数	CALL ! _FlashStart	声明自编程的开始
初始化函数	CALL ! _FlashEnv	入口 RAM 的初始化，FLMD0 引脚置高电平
模式检测函数	CALL ! _CheckFLMD	确认 FLMD0 引脚电平
Block 空白检测函数	CALL ! _FlashBlockCheck	确认制定 Block 是否为空白
Block 擦除函数	CALL ! _FlashBlockErase	擦除指定 Block
字写入函数	CALL ! _FlashWordWrite	向指定地址写入 1-64 个字数据
Block 验证函数	CALL ! _FlashBlockVerify	-
自编程结束函数	CALL ! _FlashEnd	声明自编程结束
读取信息函数	CALL ! _FlashGetInfo	读取 Flash 信息
改变信息设置函数	CALL ! _FlashSetInfo	改变 Flash 信息设置

自编程操作流程如图 3 所示，当单片机收到自编程执行信号时，开始进入自编程模式。将 FLMDO 引脚设置成高电平，初始化入口 RAM，为自编程库函数开辟空间。当确认 FLMD0 为自编程状态时，开始检查需要编程区域是否为空白区域。当被编程区域不是空白区域时，先将其擦除，然后在此区域进行编程。编程结束后进行校验。若校验无误，则将 FLMDO 引脚设置成低电平，退出自编程模式。

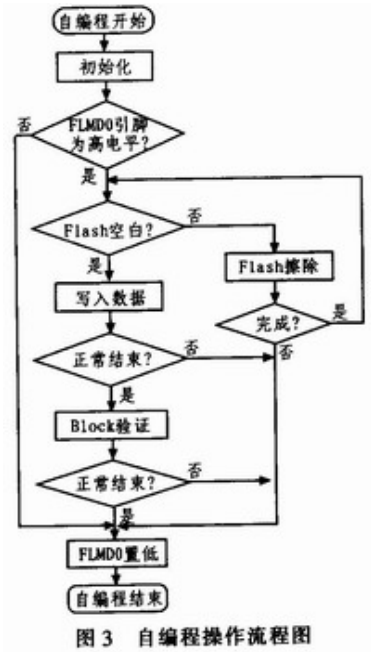


图 3 自编程操作流程图

3 引导交换（boot swap）

产品程序的升级包括应用程序的升级和引导程序（Bootloader 自身）的升级。为了防止引导程序在升级的过程中发生错误，从而导致 MCU 无法启动，设计了引导交换功能。以图 4 说明引导交换的实现过程。

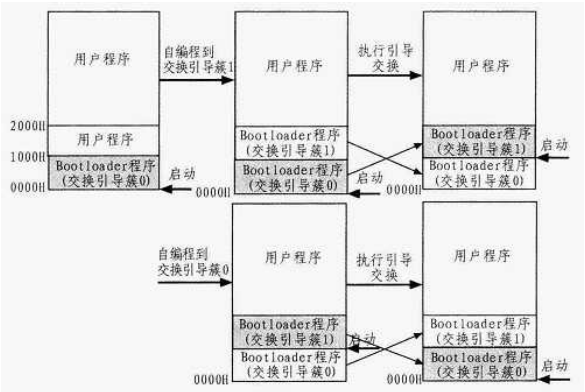


图 4 引导交换功能描述图

1）旧的 Boot 程序首先将新的 Boot 程序编程到交换引导簇 1（Boot cluster 1），然后设置启动交换标志位，并强迫看门狗复位。

2）复位启动后，MCU 看到交换标志位，便从交换引导簇 1 处开始启动。交换引导簇 1 处的新 Boot 程序将检查交换标志位。如果交换标志位被置 1，则新的 Boot 程序将擦除交换引导簇 0（Boot cluster 0）区域，并将自身复制到交换引导簇 O，然后将交换标志位清零，强迫看门狗复位。

——基于单片机和 USB 接口的数据采集系统设计

- 在工业生产和科学技术研究中，常利用 PC 或工控机对各种数据进行采集，以获得所需要的控制信息和实验数据。传统的数据采集系统多以 ISA，EISA 或 PCI 插卡的形式完成数据传输，这种方式存在安装麻烦，受计算机插槽数量、地址、中断资源限制，可扩展性差等缺点。由于通用串行总线(Universal Serial Bus．USB)具有自动被系统识别．自动安装驱动程序、自行进行系统配置，以及支持不同速率的同步和异步传输方式，支持热插拔和即插即用(Plug and Play，PNP)等优点，已逐渐成为现代数据传输的发展趋势。目前实现 USB 数据传送多采用专用的 USB 接口芯片，文献采用的 PDIUSBDI2 可支持 USBl．1 协议，文献

E37 采用的接口芯片为 USBl00 也仅支持 USBl．1 协议，文献采用 CP2102 符合 USB2．0 协议，其通用的驱动程序可将设备作为虚拟的 COM 端口设备进行操作，文献采用 Philips 公司 ISPl581 芯片作为 USB2．0 的接 VI 芯片'。这里采用 Cypress 公司的 CY7C68013 作为 USB 接口芯片，设计实现了基于单片机和 USB2．0 的数据采集系统。该系统可实现单通道模拟信号的采集，主机应用程序负责启动和停止采样，采样间隔时间由主机应用程序设置调整，采样数据传给主机应用程序显示并保存。

1 系统硬件设计

1．1 系统硬件组成

整个系统的硬件结构如图 l 所示。AT89C52 为主控单片机，负责控制 A／D 转换、上传采集数据、接收并执行主机的命令。CY7C68013 为 USB 接口芯片。A／D 转换芯片采用 TI 公司生产的 TLC549，AD780 是一款高精度参考电压芯片，可为 TLC549 提供 2．5 V 或 者 3．0 V 的参考电压。系统+5 V 电源由主机的 USB 接口提供，CY7C68013 所需的电源为+3．3 V，由+5 v 电源接稳压芯片 APlll7 提供，图中没有画出。

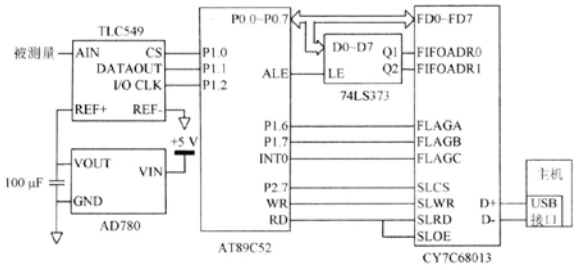


图 l 系统的硬件连接图

1．2 TLC549

TLC549 是以 8 位开关电容逐次逼近 A／D 转换器为基础而构造的 CMOS A／D 转换器，将其设计成能通过三态输出与微处理器或外围设备串行接 VI。TLC549 用输入／输出时钟(I／O CLOCK)和芯片选择(CS)输入作数据控制，转换结果由 DATAOUT 引脚输出。I／o CLOCK 端的最高频率可达 1．1 MHz。TLC 一 549 片内系统时钟工作在 4 MHz(不需要外部时钟)。片内系统时钟使内部器件的操作独立于串行输入／输出时序并允许 TLC549 像许多软件和硬件所要求的那样工作。I／O CLOCK 和内部系统时钟可以实现高速数据传送，使得 TLc549 可实现 40 kHz 的采样频率。TLC549 具有通用控制逻辑及自动工作或在微

处理器控制下工作的片内采样／保持电路，差分高阻抗基准电压输入端，易于实现比例转换的高速转换器，定标及隔离电路。整个开关电容逐次逼近转换器电路的设计允许在小于 17μs 的时间内，以最大误差±0．5 为最低有效位的精度实现转换。

1．3 CY7(368013 及其固件程序

EZ—USB FX2 系列芯片 CY7C68013 是业界第一个支持 USB2．0，同时向下兼容 USBl．1 规范的单片机，为描述方便以下简称该芯片为 FX2。FX2 支持全速传输(12 Mb／s)和高速传输(480 Mb／s)，该芯片将 USB2．O 收发器、串行接口引擎 SIE、增强的 8051 内核、GPIF 等集成于一体。FX2 内含 4 KB 的端点缓冲区 F1FO，可以被配置为具有不同大小缓冲区的 IN 或 OUT 端点(EP2，EP4，EP6，EP8)，具有 USB 协议所规定的 4 种传输方式，即控制方式、中断方式、批量传输、和同步传输方式。Cypress 公司为 FX2 提供了完善的软件开发工具包，降低了开难度，加快了开发进度。

FX2 可以工作在 3 种不同的模式下完成 USB 数据的传输，即：Ports 模式、GPIF 模式和 Slave FIFO 模式。Ports 模式下其 uSB 数据的传输主要在 FX2 的 8051 内核参与下完成，数据传输通过执行指令实现，因此数据的传输率比较低，对大批量数据传输一般采用后两种方式。GPIF 方式，称为通用可编程接口方式，在此模式下，FX2 的 FIFO 是由内部的 GPIF 控制的，FX2 利用由软件编程输出读写控制波形读取 FIFO 标志，控制 FIFO 的选通，并且对外部设备提供了用户专用接口，可以对许多通用总线接口进行访问，如 ASIC，DSP 和存储器等。文献利用 FX2 的 GPIF 方式构建了 LISB 数据传输通道。Slave FIFO 方式是将 FX2 的 FIFO 作为外部控制器(如 FPGA 或单片机)的从属 FIFO，外部控制器可像普通 FIFO 操作一样对 FX2 的 FIFO 进行读写，而不考虑该包的大小，传输速率可明显提高，文中 FX2 在 Slave FIFO 模式下工作。FX2 有 3 种封装形式：128 引脚、100 引脚和 56 引脚，这里选用 FX2 的 56 引脚的封装形式。

FX2 芯片在使用时必须先下载固件程序，固件程序主要负责完成芯片初始化，对芯片进行必要的配置、处理设备请求、进行数据传输等相应工作。用户通过编写适当的固件程序完成对 FX2

的设置。Cypress 公司提供了一个固件程序开发框架可以大大简化 FX2 芯片固件程序的开发难度。通过编写用户初始化函数 TD_Init()，用户可以规定各种端点资源的使用以及配置外围接口的输入 / 输出等。其主要配置语句如下：

```
IFCONFIG = 0xCB;
    //FX2 配置为异步 Slave FIFO 方式,内部时钟 48 MHz
EP2CFG = 0xA0;          //EP2,4×512 B,BULK,OUT
EP2FIFOCFG = 0x10;          //EP2 自动 OUT,8 位
EP6CFG = 0xE0;          //EP6,4×512 B,BULK,IN
EP6FIFOCFG = 0x08;          //EP6 自动 IN,8 位
EP4CFG = 0x20;                  //EP4 无效
EP8CFG = 0x60;                  //EP8 无效
PINFLAGSAB = 0x8E;
        //FLAGA 固定为 EP6FF,FLAGB 固定为 EP2EF
PINFLAGSCD = 0x04;          //FLAGC 固定为 EP2PF
PORTACFG |= 0x40;           //PA7 引脚配置为 SLCS
FIFOPINPOLAR = 0x00;        //所有引脚低电平有效
EP2FIFOPFH = 0x00;
            //FP2 端点的整个 FIFO 大于等于 1 时,
                            FP2 的可编程标志激活
EP2FIFOPFL = 0x01;
```

固件程序将 FX2 配置为异步 Slave FIFl0 模式，总线宽度 8 位，在 4 个端点中，EP4 和：EP8 未被使用，EP2 和 EP6 的配置如表 1 所示。由于采用自动输入 / 输出模式，主机和单片机通过旁路 FX2 的 CPU 直接连接，所有数据被直接通过 FIFO 管道提交，不需固件程序干预。在 FX2 的 slave FIF0 模式下，FIFOADR[1：0]引脚作为地址线选择某个端点，SLCS 相当于片选信号，SLwR(写)与单片机的 wR 引脚相连，SLRD(读)和 SLOE(输出使能)与单片机的 RD 引脚相连。单片机通过访问地址为 0x00 的外部存储器的方式就可以实现对 EP2 的访问，同理可访问 EP6 端点。 继续阅读文章

——嵌入式 TCP/IP 协议单片机技术在网络通信中的应用

- 在因特网上，TCP/IP 协议每时每刻保证了数据的准确传输。在数据采集领域，如何利用 TCP/IP 协议在网络中进行数据传输成为一个炙手可热的话题。在本系统中，笔者利用 TCP/IP 协议中的 UDP（用户数据报协议）、IP（网络报文协议）、ARP（地址解析协议）及简单的应用层协议成功地实现了单片机的网络互连，既提高了数据传输的速度，又保证了数据传输的正确性，同时也扩展了数据传输的有效半径。

1 TCP/IP 协议简介

TCP/IP 协议是一套把因特网上的各种系统互连起来的协议组，保证因特网上数据的准确快速传输。参考开放系统互连（OSI）模型，TCP/IP 通常采用一种简化的四层模型，分别为：应用层、传输层、网络层、链路层。

（1）应用层

网络应用层要有一个定义清晰的会话过程，如通常所说的 Http、Ftp、Telnet 等。在本系统中，单片机系统传递来自 Ethernet 和数据终端的数据，应用层只对大的数据报作打包拆报处理。

（2）传输层

传输层让网络程序通过明确定义的通道及某些特性获取数据，如定义网络连接的端口号等，实现该层协议的传输控制协议 TCP 和用户数据协议 UDP。在本系统中使用 UDP 数据报协议。

（3）网络层

网络层让信息可以发送到相邻的 TCP/IP 网络上的任一主机上，IP 协议就是该层中传送数据的机制。同时建立网络间的互连，应提供 ARP 地址解析协议，实现从 IP 地址到数据链路物理地址的映像。

（4）链路层

由控制同一物理网络上的不同机器间数据传送的底层协议组成，实现这一层协议的协议并属于 TCP/IP 协议组。在本系统中这部分功能由单片机控制网卡芯片 CS8900 实现。

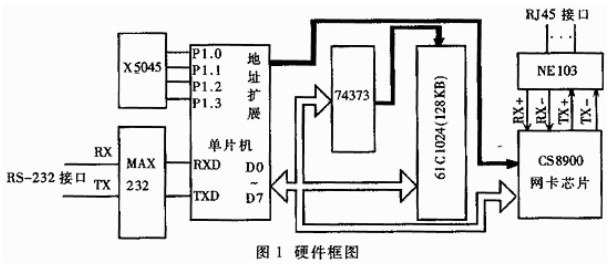


图 1 硬件框图

2 硬件框图

如图 1 所示，系统提供 RJ45 接口连接 Ethernet 网络，并且提供一个串口给用户使用。系统板可以将从 Ethernet 上过来的 IP 数据报解包后送给串口，也可将从串口过来的数据封装为 IP 包送到局域网中。外部 RAM 使用 61C1024（128KB），从而为数据处理提供了很大的缓存;使用 E2PROM——X25045，既可以

作为看门狗使用，也可以将 IP 地址、网卡物理地址和其他参数保存在里面。

CS8900 芯片是 Cirrus Logic 公司生产的一种局域网处理芯片，它的封装是 100-pin TQFP，内部集成了在片 RAM、10BASE-T 收发滤波器，并且提供 8 位和 16 位两种接口，本文只介绍它的 8 位模式。

NE103 是一种脉冲变压器，在 CS8900 的前端对网络信号进行脉冲波形变换。

3 工作原理

3.1 CS8900 的工作原理

CS8900 与单片机按照 8 位方式连接，网卡芯片复位后默认工作方式为 I/O 连接，基址是 300H，下面对它的几个主要工作寄存器进行介绍（寄存器后括号内的数字为寄存器地址相对基址 300H 的偏移量）。

·LINECTL（0112H）

LINECTL 决定 CS8900 的基本配置和物理接口。在本系统中，设置初始值为 00d3H，选择物理接口为 10BASE-T，并使能设备的发送和接收控制位。

·RXCTL（0104H）

RXCTL 控制 CS8900 接收特定数据报。设置 RXTCL 的初始值为 0d05H，接收网络上的广播或者目标地址同本地物理地址相同的正确数据报。

·RXCFG（0102H）

RXCFG 控制 CS8900 接收到特定数据报后会引发接收中断。RXCFG 可设置为 0103H，这样当收到一个正确的数据报后，CS8900 会产生一个接收中断。

·BUSCT（0116H）

BUSCT 可控制芯片的 I/O 接口的一些操作。设置初始值为 8017H，打开 CS8900 的中断总控制位。

·ISQ（0120H）

ISQ 是网卡芯片的中断状态寄存器，内部映射接收中断状态寄存器和发送中断状态寄存器的内容。

·PORT0（0000H）

发送和接收数据时，CPU 通过 PORT0 传递数据。

·TXCMD（0004H）

发送控制寄存器，如果写入数据 00C0H，那么网卡芯片在全部数据写入后开始发送数据。

·TXLENG（0006H）

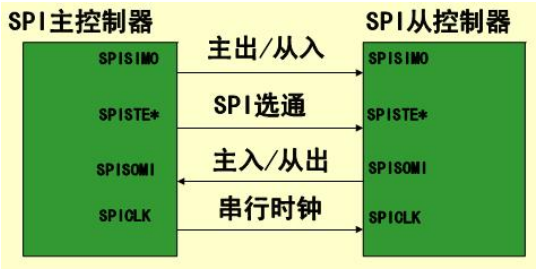
发送数据长度寄存器，发送数据时，首先写入发送数据长度，然后将数据通过 PORT0 写入芯片。

以上为几个最主要的工作寄存器（为 16 位），CS8900 支持 8 位模式，当读或写 16 位数据时，低位字节对应偶地址，高位字节对应奇地址。例如，向 TXCMD 中写入 00C0H，则可将 00h 写入 305H，将 C0H 写入 304H。

系统工作时，应首先对网卡芯片进行初始化，即写寄存器 LINECTL、RXCTL、RCCFG、BUSCT。发数据时，写控制寄存器 TXCMD，并将发送数据长度写入 TXLENG，然后将数据依次写入 PORT0 口，如将第一个字节写入 300H，第二个字节写入 301H，第三个字节写入 300H，依此类推。网卡芯片将数据组织为链路层类型并添加填充位和 CRC 校验送到网络同样，单片机查询 ISO 的数据，当有数据来到后，读取接收到的数据帧。读数据时，单片机依次读地址...

继续阅读文章

——51 单片机模拟 SPI 总线的方法



- **1 引言**

SPI（Serial Peripheral Interface--串行外设接口）总线系统是一种同步串行外设接口，它可以使 MCU 与各种外围设备以串行方式进行通信以交换信息。外围设置 FLASHRAM、网络控制器、LCD 显示驱动器、A/D 转换器和 MCU 等。SPI 总线系统可直接与各个厂家生产的多种标准外围器件直接接口，该接口一般使用 4 条线：串行时钟线（SCK）、主机输入/从机输出数据线 MISO、主机输出/从机输入数据线 MOSI 和低电平有效的从机选择线 SS（有的 SPI 接口芯片带有中断信号线 INT 或 INT、有的 SPI 接口芯片没有主机输出/从机输入数据线 MOSI）。由于 SPI 系统总线一共只需 3～4 位数据线和控制即可实现与具有 SPI 总线接口功能的各种 I/O 器件进行接口，而扩展并行总线则需要 8 根数据线、8～16 位地址线、2～3 位控制线，因此，采用 SPI 总线接口可以简化电路设计，节省很多常规电路中的接口器件和 I/O 口线，提高设计的可靠性。由此可见，在 MCS51 系列等不具有 SPI 接口的单片机组成的智能仪器和工业测控系统中，当传输速度要求不是太高时，使用 SPI 总线可以增加应用系统接口器件的种类，提高应用系统的性能。

2 SPI 总线的组成

利用 SPI 总线可在软件的控制下构成各种系统。如 1 个主 MCU 和几个从 MCU、几个从 MCU 相互连接构成多主机系统（分布式系统）、1 个主 MCU 和 1 个或几个从 I/O 设备所构成的各种系统等。在大多数应用场合，可使用 1 个 MCU 作为控机来控制数据，并向 1 个或几个从外围器件传送该数据。从器件只有在主机发命令时才能接收或发送数据。其数据的传输格式是高位（MSB）在前，低位（LSB）在后。SPI 总线接口系统的典型结构如图 1 所示。

当一个主控机通过 SPI 与几种不同的串行 I/O 芯片相连时，必须使用每片的允许控制端，这可通过 MCU 的 I/O 端口输出线来实现。但应特别注意这些串行 I/O 芯片的输入输出特性：首先是输入芯片的串行数据输出是否有三态控制端。平时未选中芯片时，输出端应处于高阻态。若没有三态控制端，则应外加三态门。否则 MCU 的 MISO 端只能连接 1 个输入芯片。其次是输出芯片的串行数据输入是否有允许控制端。因此只有在此芯片允许时，SCK 脉冲才把串行数据移入该芯片；在禁止时，SCK 对芯片无影响。若没有允许控制端，则应在外围用门电路对 SCK 进行控制，然后再加到芯片的时钟输入端；当然，也可以只在 SPI 总线上连接 1 个芯片，而不再连接其它输入或输出芯片。

3 在 MCS-51 系列单片机中的实现方法

对于不带 SPI 串行总线接口的 MCS-51 系列单片机来说，可以使用软件来模拟 SPI 的操作，包括串行时钟、数据输入和数据输出。对于不同的串行接口外围芯片，它们的时钟时序是不同的。对于在 SCK 的上升沿输入（接收）数据和在下降沿输出（发送）数据的器件，一般应将其串行时钟输出口 P1.1 的初始状态设置为 1，而在允许接收后再置 P1.1 为 0。这样，MCU 在输出 1 位 SCK 时钟的同时，将使接口芯片串行左移，从而输出 1 位数据至 MCS-51 单片机的 P1.3 口（模拟 MCU 的 MISO 线），此后再置 P1.1 为 1，使 MCS-51 系列单片机从 P1.0（模拟 MCU 的 MOSI 线）输出 1 位数据（先为高位）至串行接口芯片。至此，模拟 1 位数据输入输出便宣告完成。此后再置 P1.1 为 0，模拟下 1 位数据的输入输出……，依此循环 8 次，即可完成 1 次通过 SPI 总线传输 8 位数据的操作。对于在 SCK 的下降沿输入数据和上升沿输出数据的器件，则应取串行时钟输出的初始状态为 0，即在接口芯片允许时，先置 P1.1 为 1，以便外围接口芯片输出 1 位数据（MCU 接收 1 位数据），之后再置时钟为 0，使外围接口芯片接收 1 位数据（MCU 发送 1 位数据），从而完成 1 位数据的

传送。

图 2 所示为 MCS-51 系列单片机与存储器 X25F008（E2PROM）的硬件连接图，图 2 中，P1.0 模拟 MCU 的数据输出端（MOSI），P1.1 模拟 SPI 的 SCK 输出端，P1.2 模拟 SPI 的从机选择端，P1.3 模拟 SPI 的数据输入端（MISO）。下面介绍用 MCS-51 单片机的汇编语言模拟 SPI 串行输入、串行输出和串行输入/输出的 3 个子程序，实际上，这些子程序也适用于在串行时钟的上升沿输入和下降沿输出的其它各种串行外围接口芯片（如 A/D 转换芯片、网络控制器芯片、LED 显示驱动芯片等）。对于下降沿输入、上升沿输出的各种串行外围接口芯片，只要改变 P1.1 的输出电平顺序，即先置 P1.1 为低电平，之后再次置 P1.1 为高电平，再置 P1.1 为低电平……，则这些子程序也同样用。

3.1 MCU 串行输入子程序 SPIIN

从 X25F008 的 SPISO 线上接收 8 位数据并放入寄存器 R0 中的应用子程序如下：

SPIIN：SETB P1.1 ； 使 P1.1（时钟）输出为 1

CLR P1.2 ； 选择从机

MOV R1，#08H ； 置循环次数

SPIIN1： CLR P1.1 ； 使 P1.1（时钟）输出为 0

NOP ； 延时

NOP

MOV C，P1.3 ； 从机输出 SPISO 送进位 C

RLC A ； 左移至累加器 ACC

SETB P1.1 ； 使 P1.0（时钟）输出为 1

DJNZ R1，SPIIN1 ； 判断是否循环 8 次（8 位数据）

MOV R0，A ； 8 位数据送 R0

RET

3.2 MCU 串行输出子程序 SPIOUT

将 MCS-51 单片机中 R0 寄存器的内容传送到 X25F008 的 SPISI 线上的程序如下：

SPIOUT：SETB P1.1 ； 使 P1.1（时钟）输出为 1

CLR P1.2 ； 选择从机

MOV R1，#08H ； 置循环次数

MOV A，R0 ； 8 位数据送累加器 ACC

SPIOUT1：CLR P1.1 ； 使 P1.1（时钟）输出为 0

NOP ； 延时

NOP

RLC A ； 左移至累加器 ACC 最高位至 C

MOV P1.0，C ； 进位 C 送从机输入 SPISI 线上

SETB P1.1 ； 使 P1.1（时钟）输出为 1

DJNZ R1，SPIOUT1 ； 判是否循环 8 次（8 位数据）

RET

——AVR 单片机在 LED 遥控照明中的应用

- **引言**

LED 照明已经进入了家庭用户，与传统的照明设备（如白炽灯、荧光灯）相比，具有光源单色纯度高、色彩多样、效率高、光强度可调等优点。针对传统照明亮度不易调节、开关位置固定的问题，本文基于 AVR 单片机设计了一种 LED 遥控照明系统，提出了 LED 照明灯的驱动与亮度调节的方法。

1 LED 照明灯控制系统原理

系统原理图如图 1 所示。当红外接收器接收到红外遥控信号

时，通过外部中断将 AVR 单片机从休眠模式中唤醒；AVR 单片机开始解析红外信号，如果与系统地址匹配，则将根据解析到的命令改变 LED 恒流源驱动的输入，从而改变 LED 灯的状态。

图 1　LED 照明控制系统原理图

2 系统硬件设计

2．1 控制器

控制器采用 AVR 单片机 ATmega8。ATmega8 是 Atmel 公司在 2002 年推出的一款 AVR 单片机，采用小引脚封装。ATmega8 内部集成 8 KB 的可编程 F1ash、512 字节 EEPROM 和 1KB 内部 SRAM；3 个 PWM 通道，可实现任意小于 16 位、相位和频率可调的 PWM 脉宽调制输出；1 个可编程的串行 USART 接口，支持同步、异步以及多机通信自动地址识别；5 种省电模式。本系统中，控制器 ATmega8 的主要作用为：解析红外信号，对 LED 驱动器进行控制。

2．2 红外接收模块

红外接收模块主要器件采用 IRM-2368V，常用于家庭 DVD、电视、空调等家电的遥控中。IRM-2368V 具有以下特点：工作电压为 2．4～6 V；灵敏度高，抗干扰能力强；能直接将遥控信号从载波中提取出来，输出匹配 TTL、CMOS 电平，可与单片机直接接口；遥控距离可达 12m。图 2 为红外接收模块原理图。其中 PD2 复用为 ATmega8 的外部中断 INTO，电源部分使用系统的 5 V 供电。

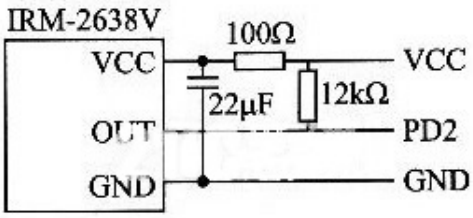


图 2　红外接收模块原理图

2．3 LED 驱动模块

LED 驱动模块采用 HV9910 集成芯片。它具有如下特点：高能效超过 90%；8～450 V 的宽电压输入；输出电流从几 mA 到 1A 可调；能驱动多达百个 LED 灯；PWM 调节电流。图 3 是 LED 恒流源驱动原理图，该驱动电路为典型 buck-boost 转化器设计。驱动器中输入电源电压 Vin=12V，驱动 3～6 个 3 50mA 高亮度 LED 灯。

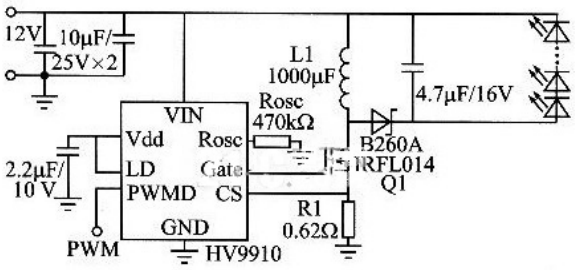


图 3　LED 驱动原理图

HV9910 工作时，内部振荡频率 fosc 由引脚 Rosc 上的电阻决定。本设计中 Rosc 取 470 kΩ，将 MOSFET 管 Q1 的 gate 端开关频率设定为 50kHz。R osc 与 fosc 满足以下关系式（Rosc 的单位是 kΩ）：

$$f_{osc}=25\,000/(R_{osc}+22)\ \text{kHz}$$

每个 LED 灯工作时压降约为 3 V，当有 3 只 LED 灯串联在输出端时，驱动器输出电压 Vled=91 V。可得 LED 满电流工作时 Q1 管的控制信号占空比 D 为：

$$D=V_{led}/(V_{leds}+V_{in})=0.43$$

Q1 的导通时间 Ton=D / fosc=8．6μs，输出电流 Iled=350 mA，谐波电流抑制在 30%以内，则可由下式得出电感 L1 的值：

$$L_1=V_{in}\cdot T_{on}/(0.3I_{led})=0.98\ \text{mH}$$

本方案中 L1 实际使用 1 mH。

R1 上的反馈电压与 HV9910 内部比较电压 250 mV 相比较，若反馈电压大于 250 mV，则关断 Q1。由谐波电流关系式可求出 R1：

$$R_1 = 250\ \text{mA}/(I_{led} + 0.5 \times 0.3 \times I_{led}) = 0.62\ \Omega$$

——基于 AVR 单片机 Mega16 的电子时钟设计

- **1 引言**

数字钟能长期、连续、可靠、稳定地工作；同时还具有体积小，功耗低等特点，便于携带，使用方便。数字钟是采用数字电路实现对“时、分、秒”数字显示的计时装置，广泛应用于个人家庭、车站、码头、办公室等公共场所，已成为人们日常生活中不可缺少的必需品。由于数字集成电路的发展和石英晶体振荡器的广泛应用，使得数字钟的精度远远超过老式钟表，钟表的数字化给人们生产生活带来了极大的方便，而且大大地扩展了钟表原先的报时功能。传统 MCS51 系列单片机的所有数据处理都基于一个累加器，因此累加器与程序存储器、数据存储器之间的数据转换就成了单片机的瓶颈；在 AVR 单片机中，寄存器由 32 个通用工作寄存器组成，并且任何一个寄存器都能充当累加器，从而有效避免累加器的瓶颈效应，提高系统性能。

AVR 系列的单片机不仅具有良好的集成性能，而且都具备在线编程接口，其中的 Mega 系列还具备 JTAG 仿真和下载功能；含有片内看门狗电路、片内程序 Flash、同步串行接口 SPI；多数 AVR 单片机还内嵌了 A / D 转换器、EEPROM、模拟比较器、PWM 定时计数器等多种功能；AVR 单片机的 I / O 接口具有很强的驱动能力，灌人电流可直接驱动继电器、LED 等元件，从而省去驱动电路，节约系统成本。

2 整体设计思路

利用 Mega16 单片机内部时钟作为时间基准，通过软件编程控制可编程器件 Mega16，实现秒、分、时、日、月、年的控制，最终通过 LCD 液晶显示屏显示结果。此外还可以实现时间调整、定时等多种实用功能。整个设计分硬件和软件两大部分。硬件部分采用 Mega16 单片机作为可编程芯片，1602 字符液晶作为信号显示；软件部分利用 C 语言作为设计语言，对 Mega16 进行编程实现各种功能。

3 硬件设计

硬件设计电路分解为 Mega16 单片机、晶体振荡器和 802 / 1602 字符液晶显示 3 个部分，其结构简单，经济实惠。Mega16 单片机内部晶体振荡器的外接电路。由两个 15 pF 的电容 C7 和 C8、晶体振荡器 Y2（f=7.328 MHz）构成，其电路如图 1 所示。图中 X1 和 X2 分别接 Mega16 的 12 和 13 两个脉冲控制端，使得 Mega16 的内部脉冲电路为电子时钟和整个系统时钟提供脉冲。

图 1 所示给出了采用 Mega16 单片机外加电源及晶体振荡器构成最小单片机系统。配合单片机开发的设计、调试和下载，最终将时钟信息从 PB0～PB7 端口输出到字符液晶显示。

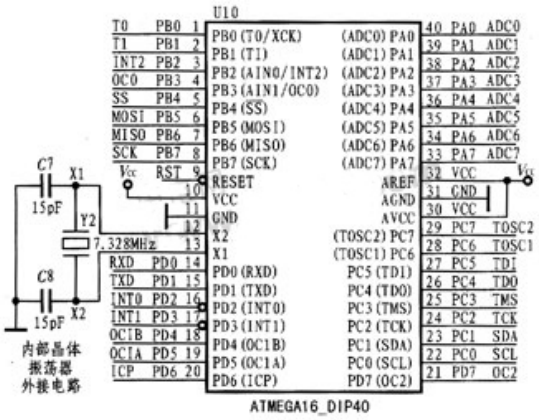


图 1 内部晶体振荡器外接电路

图 2 给出 1602 字符液晶作为信号显示部分。字符液晶采用 4 位模式与单片机的 PB 端口相连。 继续阅读文章

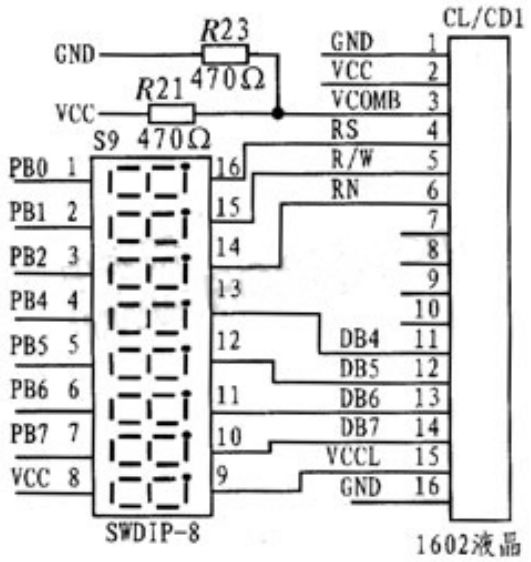


图 2 1602 字符液晶电路连接

电子工程网

简介

电子工程：

功率设计

可编程逻辑

IC 设计

MCU/控制技术

缓冲/存储技术

放大/调整/转换

封装/测试

嵌入式设计

数字信号处理

传感技术

RF/无线

光电/显示

网络/协议

EMC/EMI/ESD 设计

推荐专题

精华电子书推荐


电子工程师创新设计宝典
电源技术创新专辑
OFweek电子工程网编辑团队出品
创新节能设计
产品新知
反激电源
LED驱动电源
开关电源
便携设备电源
逆变电源
市场趋势
OFweek电子工程网
创新设计系列电子书

ADI推出业界首款双端口低功耗H类VDSL线路驱动器

Analog Devices, Inc.（ADI），全球领先的高性能信号处理解决方案供应商及信号调节技术领先者。

ADI 可变增益放大器具有最高RF与IF频率性能与集成度

日前，德州仪器（TI）宣布推出一款在日前可用 6 Gbps 转接驱动器 / 均衡器中具有最低工作功耗。

TI推出最低功耗6 Gbps双通道单信道SATA转接驱动器

日前，德州仪器（TI）宣布推出一款在日前可用 6 Gbps 转接驱动器 / 均衡器中具有最低工作功耗。

瑞萨推出V850ES/Jx3 少管脚系列32位低功耗微控制器

日前，瑞萨电子开始在中国推广78款超低功耗内置闪存32位微控制器，并于即日起开始提供样品。

OFweek EE.ofweek.com

电子工程网

中国领先的电子工程专业媒体

2010年11月

绿色节能设计特刊

- 设计方案
  太阳能大功率LED路灯系统设计
- 技术热点
  新型可调光LED灯具应用设计关键
- 热点推荐
  现代无线充电技术大揭秘
- 工程案例
  全面提升电源能效新技术工程范例
- 技术精粹
  非接触超低功耗红外接近感应技术
- 热点推荐
  汽车磷酸铁锂动力电池监测管理系统BMS
- 设计实现
  电池供电装置的超低功耗设计
- 节能设计
  新型逆变器优化光伏系统设计

Http://ee.ofweek.com


OFweek Ee.ofweek.com
电子工程网
中国领先的电子工程专业媒体
工程师创新设计灵感的源泉