【小蜜蜂笔记】系列 www.xmf393.com

51 单片机原理与应用开发学习笔记与题库

小蜜蜂老师 欧浩源

佛山市图志科技有限公司 广东职业技术学院

2021 年 10 月 19 日

目　录

【小蜜蜂老师简介】：欧浩源，专注于嵌入式开发与物联网应用的基础教育学。
中国计量大学，机械设计制造及其自动化（光机电一体化），本科。
中国计量大学，计算机应用技术（嵌入式应用），研究生。
广东职业技术学院，从事物联网技术应用专业的教学与科研工作。
【电子邮箱】：ohy3686@qq.com
【资源网站】：www.xmf393.com
【淘宝小店】：xmfkj.taobao.com
【小蜜蜂笔记】公众号：xmf393
【B 站 和 CSND 博客】：小蜜蜂老师的干货铺

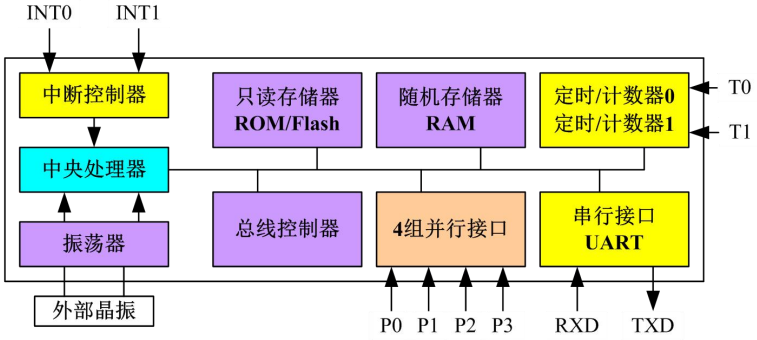
第 1 章 经典 51 内核资源总览图

你不要当 51 单片机是科技产品，请把它看作一款高级玩具。你不是在进行学习开发，你是在玩喜欢的游戏。学习会让懒人痛苦，而玩耍会给懒人快乐，没有愉悦的心情，还谈什么学好学精呢？

正所谓：知之者，不如好知者；好知者，不如乐知者。

经典的 51 单片机里面都有些什么东西呢？我用一张图把核心的资源浓缩起来，展示给你...**麻雀虽小，五脏俱全**，是不是没有想象中那么复杂呀！

经典51内核资源分布浓缩图



中断向量列表

中断号	中断向量	中断源
----	0x0000	系统复位
0	0x0003	外部中断INT0
1	0x000B	定时/计数TF0
2	0x0013	外部中断INT1
3	0x001B	定时/计数TF1
4	0x0023	串口中断RI/TI

P3端口复用功能

P3.0	RXD
P3.1	TXD
P3.2	INT0
P3.3	INT1
P3.4	T0
P3.5	T1
P3.6	/WR
P3.7	/RD

外设相关寄存器定义

中断系统	IE寄存器	EA	----	----	ES	ET1	EX1	ET0	EX0
	IP寄存器	----	----	PT2	PS	PT1	PX1	PT0	PX0
	TCON寄存器	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
定时/计数	THx寄存器	计数初值高8位							
	TLx寄存器	计数初值低8位							
	TMOD寄存器	GATE	C/T	M1	M0	GATE	C/T	M1	M0
串口通信	SCON寄存器	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
	PCON寄存器	SMOD	----	----	----	----	----	----	----
	SBUF寄存器	数据发送/数据接收缓冲器							

1.1 四组 8 位并行 I/O 端口

P0 端口：PC 门，集电极开路输出，必须接上拉电阻才能输出高电平。

P1 端口：无特殊之处。

P2 端口：访问外部存储器时，作高 8 位地址用。

P3 端口：功能复用端口，详情看上图。

1.2 三大外设

外部中断、定时/计数、串行通信

1.3 五个中断源

2 个外部中断、2 个定时/计数中断、1 个串行通信中断

注意：要记住中断号与中断源的对应关系...编程时需要用到！

1.4 三大特殊功能寄存器

TCON 寄存器、SCON 寄存器、TMOD 寄存器

第 2 章 重要外设特殊功能寄存器概览

要用好任何一款单片机，首先要掌握该单片机的**功能**与**特性**，然后就要去要学会怎么样**使用这些功能**和**控制这些特性**。不管是复杂的处理器还是简单的单片机，其功能大多通过外设来体现，而**外设**与**内核**的信息交互则主要通过**特殊功能寄存器**和**中断系统**来实现。因此，嵌入式设计师在做底层的程序开发时，实际上大部分的工作都在**操作特殊功能寄存器**和**处理响应各种中断请求**。

对于刚刚入门的新手，要想快速掌握成单片机的应用开发，是绕不开对单片机特殊功能寄存器了解和编程的。然而，比较幸运的是，在 51 单片机的应用程序设计中，**常用外设特殊功能寄存器真的非常少，其内容也不是太复杂。**

2.1 经典型 51 单片机的重要外设

对于绝大多数的单片机和微处理器，其基本架构都是“**内核+外设**”，而控制外设的重要途径就是**特殊功能寄存器**。作为一个嵌入式设计师，如果做底层的程序开发，很多时候都需要跟各种特殊功能寄存器打交道。51 单片机的外设不多，可以分成三大块：**外部中断**、**定时/计数器**和**串行接口**。

要想把这些外设应用好，首先得把控制这些外设的特殊功能寄存器弄明白，其实也不多，主要的也就下面几个：

IE **寄存器**：中断控制寄存器。

IP **寄存器**：中断优先级寄存器。

TCON **寄存器**：中断状态标志寄存器。

TMOD **寄存器**：定时/技术模式控制寄存器。

THx 和 TLx **寄存器**：定时/计数器的计数初值寄存器。

SCON **寄存器**：串口控制寄存器。

SBUF **寄存器**：串行接口收据发送和接收缓冲器。

2.2 重要的特殊功能寄存器

【1】 IE 寄存器

EA	--	--	ES	ET1	EX1	ET0	EX0
总开关			串口	定时器 1	外部中断 1	定时器 0	外部中断 0
0：禁止中断			1：使能中断				

【2】IP 寄存器(设置中断优先级，用的不多)

--	--	PT2	PS	PT1	PX1	PT0	PX0
		定时器 2	串口	定时器 1	外部中断 1	定时器 0	外部中断 0
0：低优先级			1：高优先级				

【3】 TCON 寄存器

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1：定时/计数器 T1 溢出中断请求标志位（0：无中断请求； 1：有中断请求）。
TR1：定时/计数器 T1 启动位（0：停止定时/计数器 1； 1：启动定时/计数器 1）。
TF0：定时/计数器 T0 溢出中断请求标志位（0：无中断请求； 1：有中断请求）。
TR0：定时/计数器 T0 启动位（0：停止定时/计数器 1； 1：启动定时/计数器 1）。
IE1：外部中断 INT1 中断请求标志位（0：无中断请求； 1：有中断请求）。
IT1：外部中断 INT1 触发方式控制位（0：低电平触发； 1：下降沿触发）。
IE0：外部中断 INT0 中断请求标志位（0：无中断请求； 1：有中断请求）。
IT0：外部中断 INT0 触发方式控制位（0：低电平触发； 1：下降沿触发）。

【4】 TMOD 寄存器（只能字节寻址）

GATE	C/T	M1	M0	GATE	C/T	M1	M0

寄存器可以分成 2 个部分：
高 4 位：控制定时/计数器 T1。
低 4 位：控制定时/计数器 T0。
GATE 设置为 0：由 TR0 和 TR1 来启动定时/计数器。
GATE 设置为 1：由外部中断引脚 INT0 和 INT1 来启动定时/计数器。
C/T 设置为 0：定时功能。
C/T 设置为 1：计数功能。
M1、M0：设置定时/计数器的工作方式。
00：13 位定时/计数器，最大计数为 8192。
01：16 位定时/计数器，最大计数为 65536。
10：8 位自动重装模式，最大计数为 256。
11：T0 分为 2 个独立的 8 位定时/计数器，T1 此方式停止计数。

【4】 SCON 寄存器

SM0	SM1	SM2	REN	TB8	RB8	RI	TI

TI：发送完成标志。
　　当 SUBF 中的数据发送完成后由硬件置 1，需要手动软件清 0。
RI：接收完成标志。
　　当 SUBF 接收到一个字节数据后由硬件置 1，需要手动软件清 0。
RB8：模式 2 或模式 3 中，接收到的第 8 位数据（由第 0 位开始计算）。
TB8：模式 2 或模式 3 中，等待发送的第 8 位数据（由第 0 位开始计算）。
REN 设置为 0：禁止接收。
REN 设置为 1：允许接收。
SM2：多机通信控制位，仅在模式 2 和模式 3 中才有效。
SM1、SM0：设置串口通信的工作模式。
00：模式 0，本质上是一个同步移位寄存器（波特率为：fosc/12）。
01：模式 1，8 位的 UART 模式（波特率：可变）。
10：模式 2，9 位的 UART 模式（波特率：fosc/32 或 fosc/64），用于多机通信。
11：模式 3，9 位的 UART 模式（波特率：可变），用于多机通信。

第 3 章 程序开发流程与设计要点

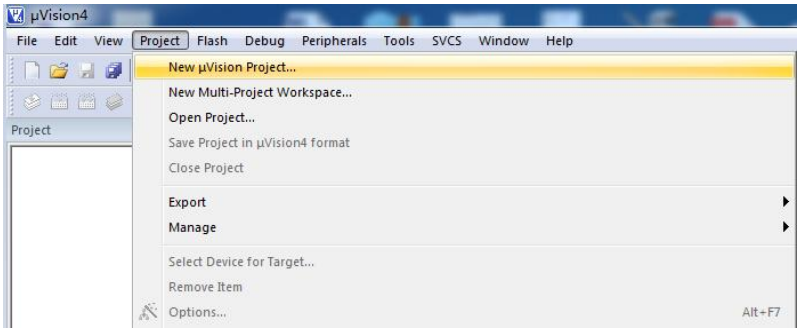
单片机的程序开发在电脑上完成，而程序的运行则在芯片中进行。程序的开发和运行分别是在不同的平台上进行的，所以，程序开发之后需要下载到芯片之中。常规的嵌入式或者其他单片机的开发都需要有调试过程，在仿真器的支持下能够进行在线调试跟踪。然而，51 单片机的开发大多数情况下都是 ISP 下载，也就是说你只能下载程序，运行观察结果，不能够进行仿真调试。虽然这个对于开发来说不太方便，也不够规范的但是 51 单片机的资源不多，外设也复杂，其应用场景亦相对简单，即使不能仿真调试，也是可以轻松胜任的。

3.1 单片机的集成开发环境

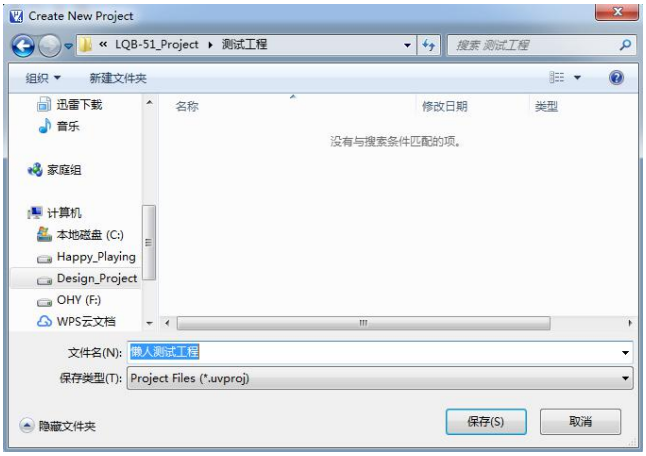
目前 51 单片机基本上都是用 Keil 集成开发环境，注意：你要安装的是 C51。

3.2 项目开发流程

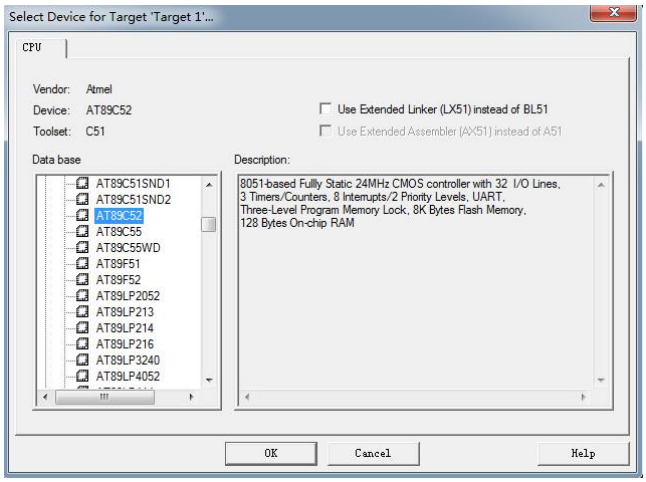
【1】建立新工程或者打开已有工程。



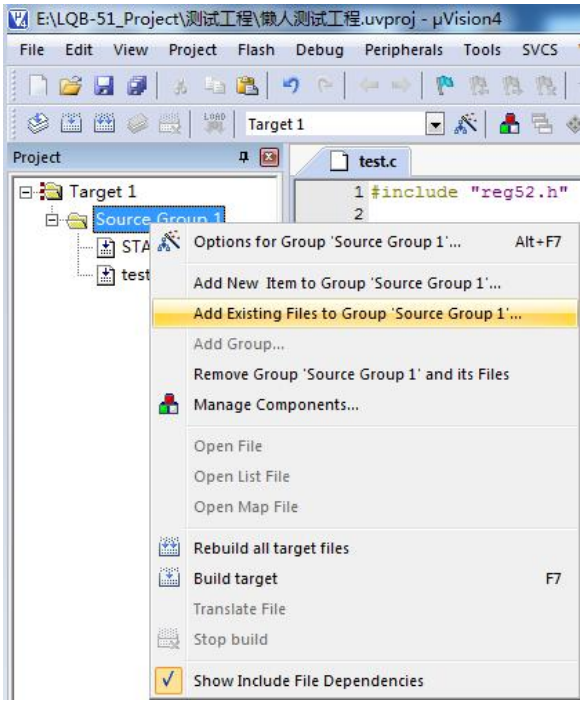
【2】如果是新建工程，单击“New uVision Project”菜单项，输入工程名字。



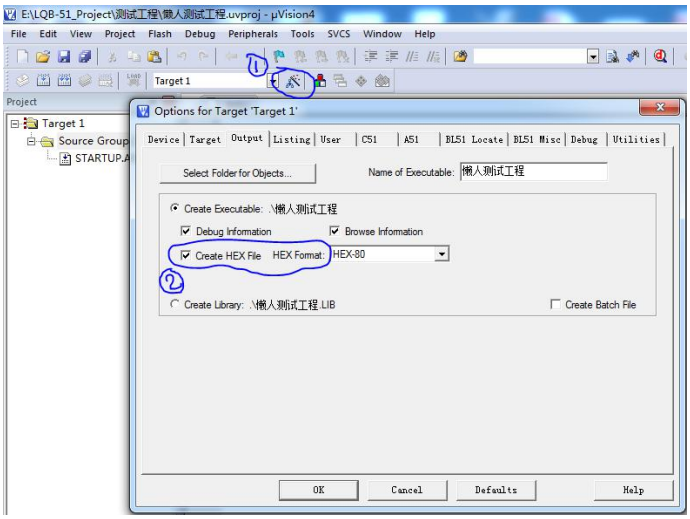
【3】为新工程选择芯片类型，我们选择 Atmel 公司的 89C52。



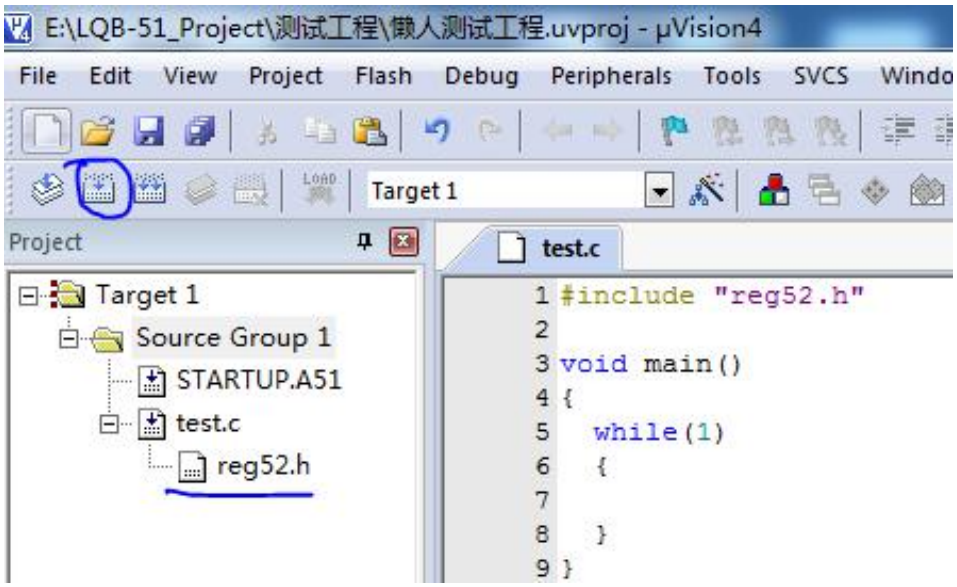
【4】给新建的工程中添加代码文件，如果没有代码文件则需要先新建一个。



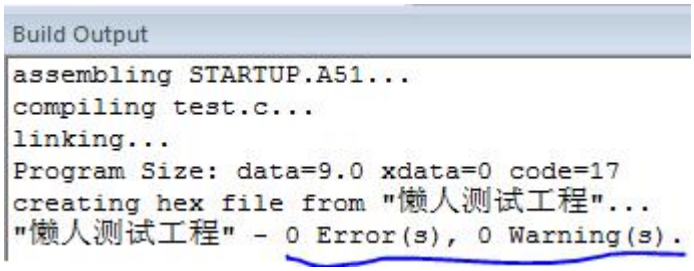
【5】配置工程的选项参数，主要的设置是输出 HEX 文件。



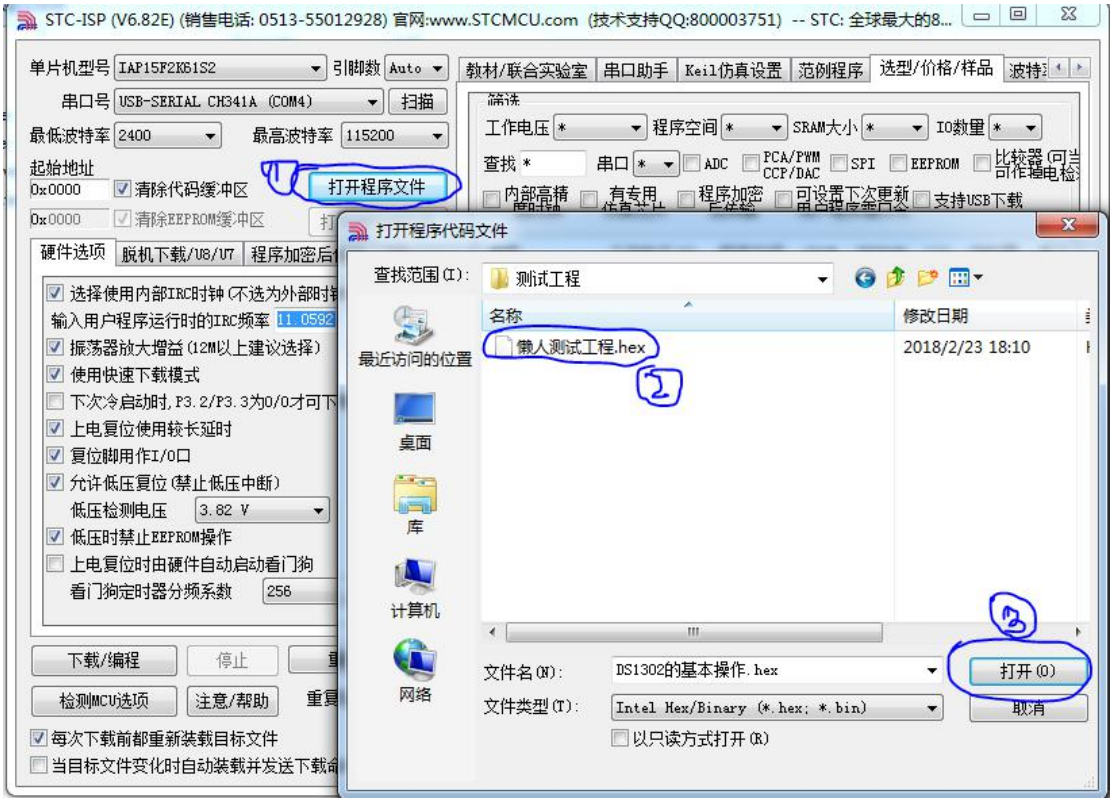
【6】编写代码，根据不同的芯片引入相应的头文件。



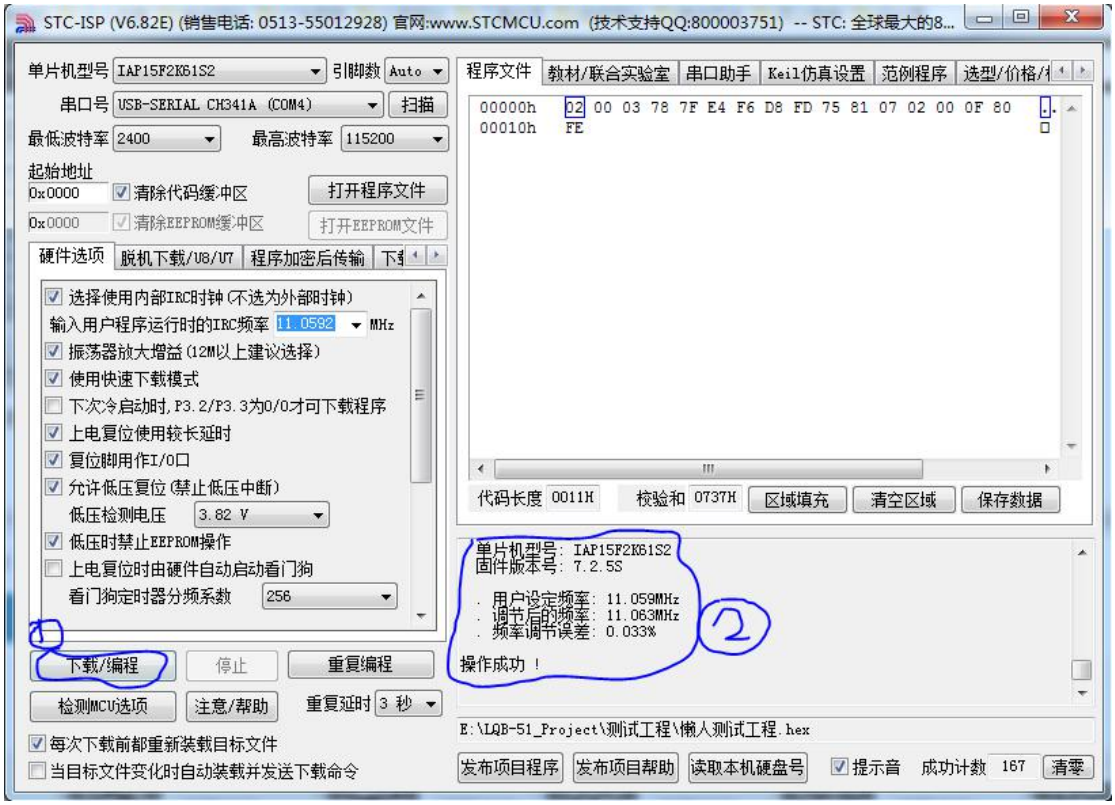
【7】编译代码，有时候允许有警告存在，但必须检查该警告是否可以忽略。



【8】打开 ISP 下载软件，导入 HEX 文件，进行代码下载。



【9】将程序代码烧写到芯片中，在断电状态下单击“下载”按钮，然后上电，程序就自动下载到芯片了。



3.3 程序设计要点

【1】引入芯片的头文件

寄存器实际上就是芯片中的一个**内存单元**。在单片机的程序设计中，操作**特殊功能寄存器**的时候，需要知道这个**内存单元的地址**。我们要记住寄存器的地址很难，而且代码编写也比较复杂。为了方便记忆和使用，每一个特殊功能寄存器都有一个**名字**。在程序设计中你想直接使用这些寄存器名字，你就需要引入对应当芯片头文件，例如 89C52 单片机，你需要引入“reg52.h”文件。

test.c | reg52.h

```
10  #define __REG52_H__
11
12  /*  BYTE Registers  */
13  sfr P0    = 0x80;
14  sfr P1    = 0x90;
15  sfr P2    = 0xA0;
16  sfr P3    = 0xB0;
17  sfr PSW   = 0xD0;
18  sfr ACC   = 0xE0;
19  sfr B     = 0xF0;
20  sfr SP    = 0x81;
21  sfr DPL   = 0x82;
22  sfr DPH   = 0x83;
23  sfr PCON  = 0x87;
24  sfr TCON  = 0x88;
25  sfr TMOD  = 0x89;
```

【2】中断服务函数的格式

经典的 51 单片机中有 5 个中断源。每个中断源都有固定的入口地址（也就是中断向量）和中断号。我们在设计 51 单片机中断相关的应用程序时，不需要记住中断向量地址，但需要知道每个中断源所对应的中断号。

一般情况下，跟中断处理相关的函数有两个。其一为中断初始化函数，其二为中断服务函数。初始化函数就是一个普通的函数，而中断服务函数则有特殊的编程格式要求：

A. 中断函数没有返回值，也不能带参数。

B. 函数名后面要跟一个关键字 interrupt，说明这是一个中断服务函数。

C. 在关键字 interrupt 后面要跟上中断号，说明这个中断服务函数是为那个中断源服务的。

中断服务函数的格式为：

```
void 函数名() interrupt  中断号
{
    ----函数体----
}
```

例如，定时器 0 的中断服务函数，我们 C 语言可以这样写：

```
/*================初始化定时器0==================*/
void Init_Timer0()
{

}

/*============定时器0中断服务函数===============*/
void Interrupt() interrupt 1
{

}

/*==================主函数=======================*/
main()
{
  Init_Timer0();
  while(1);
}
```

3.4 小结

在单片机或嵌入式的程序设计中，有很多代码是可以重用的，甚至可以直接复制粘贴。如果你能够在平时的学习和开发中不断归纳积累，整理出一个属于自己的应用程序库，那么，对于您日后的项目开发会事半功倍，如虎添翼。

第 4 章 三大外设开发与可重用代码

在传统 51 内核的单片机中，主要有外部中断、定时/计数和串行通信三大外设。在其他的嵌入式芯片中，这三个外设也是必不可少的。三大外设的程序设计实际上就成了 51 单片机最基础核心，**掌握好这三个部分的设计思路和具体编码是学习单片机开发的必经之路，没有捷径可达。**

编写三大外设的程序实际上有很多类似的地方。它们都需要进行初始的寄存器配置，也就是初始化，它们都和中断有关，都需要编写中断服务函数。因此，对于每一个外设，我们至少都需要编写一个初始化函数和中断服务函数。这些函数在不同的项目应用中设计思路差不多，甚至一样，代码的重用度也很高。

4.1 外部中断模块

在 51 单片机中有 INT0 和 INT1 两个外部中断输入，外部的触发信号可以是低电平触发，也可以是下降沿触发，由寄存器 TCON 的 IT0 和 IT1 位来决定。

```
/*================外部中断0初始化函数===============*/
void Init_Int0()
{
  IT0 = 1;              //0--低电平触发；1--下降沿触发
  EX0 = 1;              //使能外部中断0
  EA = 1;               //使能总中断
}

/*===============外部中断0中断服务函数==============*/
void SeviceInt0() interrupt 0
{
  /*----编写中断服务函数逻辑代码----*/
}
```

4.2 定时/计数模块

在 51 单片机中有 T0 和 T1 两个定时器，对内部信号可以定时，对外部信号可以计数。在初始化的时候，需要确定该模块是定时功能还是计数功能，工作模式是什么，计数初值是多少。

```
/*================定时器0初始化函数================*/
void Init_Timer0()
{
  TMOD = 0x01;          //定时器0定时功能，16位模式
  TH0 = (65536 - 50000) / 256;  //最大计数值的高8位
  TL0 = (65536 - 50000) % 256;  //最大计数值的低8位
  ET0 = 1;              //使能定时器0中断
  EA = 1;               //使能总中断
  TR0 = 1;              //启动定时器0
}

/*================定时器0中断服务函数===============*/
void SeviceTimer0() interrupt 1
{
  TH0 = (65536 - 50000) / 256;    //重装最大计数值
  TL0 = (65536 - 50000) % 256;
  /*----编写中断服务函数逻辑代码----*/
}
```

4.3 串行通信模块

在 51 单片机的串行接口中，**产生波特率需要占用定时器 T1**，发送数据时，将内容放到 SBUF 中，数据发送完毕，TI 会自动置 1。当完整接收到一个数据后，RI 会自动置 1，这时从 SBUF 中将内容读出即可。不管是 TI 还是 RI 标志位，都需要手动清 0。

一般情况下，发送数据采用查询方式，接收数据采用中断方式。

```
/*==================串行接口初始化函数================*/
void Init_Uart()
{
  TMOD=0x20;          //定时器1工作模式为自动重装
  TH1=0xfd;           //设置波特率为9600
  TL1=0xfd;           //11.0592M或12M的12分频
  TR1=1;              //启动定时器1
  SCON = 0x50;        //串口参数为模式1和允许接收
  ES=1;               //使能串口中断
  EA=1;               //使能总中断
}
/*================发送单个字节函数====================*/
void SendByte(unsigned char dat)
{
  SBUF = dat;       //将数据放进SBUF缓冲器
  while(TI == 0); //等待发送数据完成
  TI = 0;           //清除发送完成标志
}
/*================串行接口中断服务函数==============*/
void SeviceUart() interrupt 4
{
  if(RI == 1)       //接收到一个完整的字节
  {
    RI = 0;         //清除接收完成标志
    /*----编写中断服务函数逻辑代码----*/
  }
}
```

4.4 小结

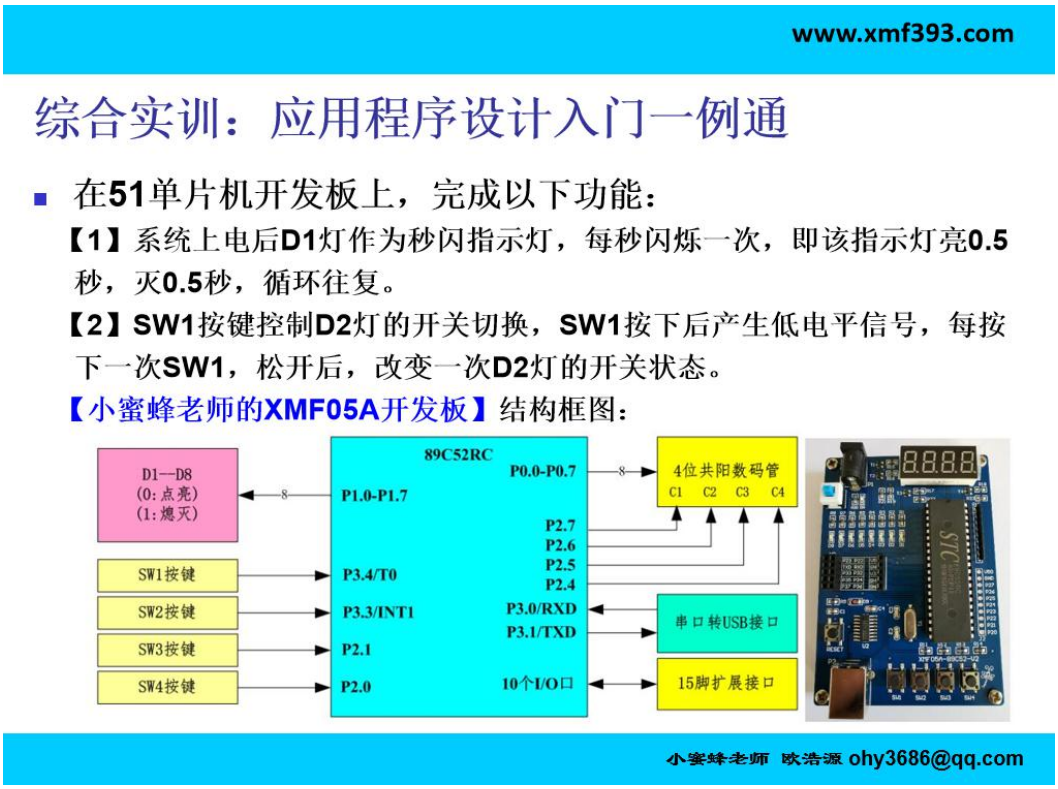
掌握了 51 单片机的三大外设，就等于掌握了 51 单片机的核心资源，也是进行拓展开发的一个重要编程基础。在日后的应用系统中，可能会扩展很多其他功能的外设，但在单片机核心控制中，实际上也是通过这些基础外设实现的。例如要扩展 WIFI 接口进行物联网应用，实际上就是通过串口和蓝牙模块进行数据通信，本质上是在操作一个串口。

第 5 章 应用程序设计入门一例通

再高级的微控制器也是解决问题的一个工具。如何灵活运用已掌握的外设模块来满足功能需求，以解决实际问题，是我们学习嵌入式和单片机开发的根本目标。你的程序写得很快，算法设计很妙，每个外设也用得很溜，这并不能代表你解决问题的综合能力就很好。这需要通过实际开发的磨练与经验的积累。

实际上，每个工程师在不断的开发过程中，多多少少都会形成自己的一套思维模式和开发习惯。因此，在这一节我所阐述的也只是本人的思维模式与开发习惯，借此抛砖引玉，可以作为参考。

5.1 题目功能需求



5.2 基本设计思路

我的设计模式是自上而下，从整体到局部。在编写代码之前，我的脑子里面基本上对应用程序已经有了比较完整的轮廓，包括应该定义那些变量，应该安排那些函数，程序框架大致如何搭建，技术难点大概出现在什么地方等等。

从本题的需求分析中，我们可以获得以下的信息：

【1】秒闪功能，需要使用定时器 0，得安排一个初始化函数 Init_Timer0()和一个中断服务函数 Sevice_Timer0()。

【2】 定时器 0 使用 16 位模式，在 12MHz 晶振情况下，最大的定时值也只有 65.535ms，因此要定时 500ms 不能够直接实现，那么就需要多次定时进行累加。把定时器 0 的间隔定时确定为 50ms 比较方便，累计 10 次就是 500ms 了，那么就需要一个定时累计变量 count_t。

【3】 按键功能，首先要定义一个按键扫描函数 Scan_Keys()，在扫描过程中需要做去抖动处理，那么就要安排一个简易的延时函数 Delay()。

【4】再检查思考一下，还有什么吗？没有了，那就开始编写代码吧...

5.3 程序框架搭建

严格来说，在开始编写代码之前，应该将程序的流程图和数据定义表做出来。不过，对于比较简单的功能应用，可以通过编写代码搭建整体框架的方式来体现。这就跟建设框架结构的房子一样，先把毛坯房建好，再去逐个单元进行装修。这样对应用程序有一个相对全面的掌握，每一个代码的编写都有全局的考虑，每一个函数都可以前后呼应，对程序的整体运行流向了然于胸，在最后的运行调试，自然可以一气呵成，就算稍有错漏，也可以精确定位，快速解决。

本题在基本思路分析指导下，可以搭建出程序框架结构。

```
#include "reg52.h"

sbit D1 = P1^0;
sbit D2 = P1^1;
sbit SW1 = P3^4;

/*=========================简单的延时函数=======================*/
void Delay(unsigned char t)
{
  while(t--);
}

/*=====================定时器0初始化函数=====================*/
void Init_Timer0()
{  }

/*=====================定时器0中断服务函数====================*/
unsigned char count_t = 0;          //定时器0中断累计变量
void Service_Timer0() interrupt 1
{  }

/*=======================按键扫描处理函数=====================*/
void Scan_Keys()
{  }

/*===========================主函数=========================*/
void main()
{
  Init_Timer0();                    //初始化定时器0，并启动
  while(1)
  {
    Scan_Keys();                    //在死循环中，反复扫描按键
  }
}
```

5.4 逐个函数实现

【1】 定时器初始化函数：

```
/*======================定时器0初始化函数======================*/
void Init_Timer0()
{
  TMOD = 0x01;              //定时器0，定时功能，16位模式
  //在12MHz晶振的51单片机系统中，定时器计数脉冲周期为：1us。
  TH0 = (65536 - 50000) / 256;     //定时50ms的计数初值
  TL0 = (65536 - 50000) % 256;
  ET0 = 1;                         //使能定时器0
  EA = 1;                          //使能总中断
  TR0 = 1;                         //启动定时器0
}
```

【2】 定时器中断服务函数。

```
/*======================定时器0中断服务函数======================*/
unsigned char count_t = 0;         //定时器0中断累计变量
void Service_Timer0() interrupt 1
{
  count_t++;
  if(count_t == 10)                //定时间隔0.5秒到
  {
    D1 = ~D1;                      //实现D1灯的秒闪功能
    count_t = 0;                   //定时中断累计变量清0
  }
}
```

【3】 按键扫描函数。

```
/*======================按键扫描处理函数======================*/
void Scan_Keys()
{
  if(SW1 == 0)                     //扫描按键输入信号
  {
    Delay(100);                    //第1次发生低电平，延时片刻
    if(SW1 == 0)                   //再次读取按键输入信号
    {
      while(SW1 == 0);             //等待按键松开
      D2 = ~D2;                    //切换D2灯的开关状态
    }
  }
}
```

5.5 小结

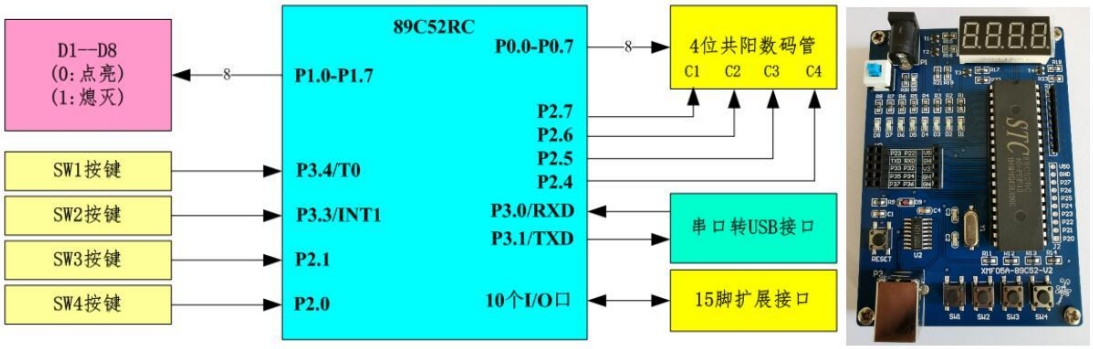
作为 51 单片机应用开发的入门基础学习，本学习笔记到这里就结束了。实际上 51 单片机的常用核心内容也差不多也是这些。至于一些讲述 51 单片机开发好几百页的书，里面很多内容都是 51 单片机外部的各种功能外设，与 51 内核资源无太大关系，只是利用 51 单片机操作那些功能外设而已。对于这部分的内容，你学习的不是 51 单片机了，而是那些功能外设。然而，在实际的应用中，更多的是要应用各种功能外设来解决各种问题。所以，一个单片机的应用高手并不是仅仅把 51 单片机本身学好就够了，而是在他心中掌握了多少功能模块的应用，设计了多少应用的代码，积累了多少项目的经验。

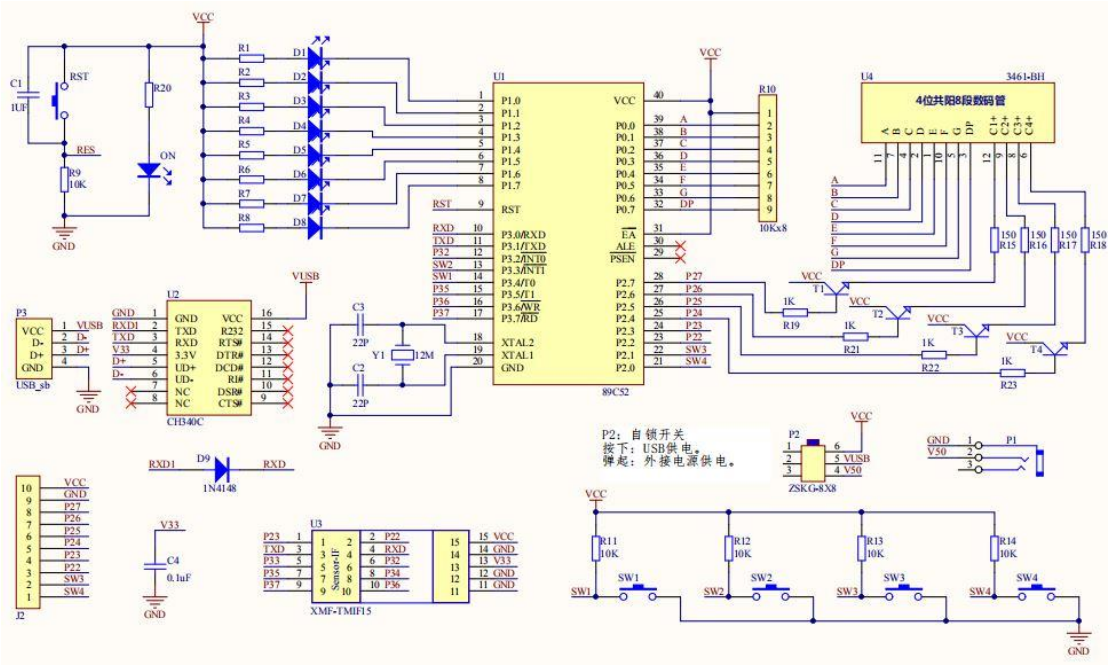
学习笔记结束了，而你们的征途才刚刚开始。在接下来的 2 章中，给你们设计两组题库，一个是入门基础案例，一个是进阶拓展案例。学习 51 单片机不能停留在纸面上。多动手，多实践才是有效途径，加油!!!

第 6 章　51 单片机入门基础案例项目

以小蜜蜂老师研制的 XMF05A 开发板为平台，针对 51 单片机的基础核心要点，设计了 15 个入门基础案例，配套案例项目的实现录制了《51 单片机原理与应用开发》视频教程 22 集（见附录 1），可在线观看和网盘下载。

6.1　开发板 XMF05A 功能定义与原理图





6.2 入门基础案例项目 15 个

【01】LED 灯闪烁

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、在 keil C51 开发环境中新建工程。

2、编写代码，在主函数中实现 D1 灯的循环闪烁。

3、利用 ISP 软件，将生成的 HEX 文件烧写到开发板的单片机中。

4、运行程序，观察结果，总结 51 单片机项目开发的基本流程。

【02】LED 跑马灯

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、8 个 LED 灯同时闪烁 3 次。

2、从 D1 灯到 D8 灯依次点亮，然后，从 D1 灯到 D8 灯依次熄灭。

3、从 D1 灯到 D8 灯依次点亮，然后，从 D8 灯到 D1 灯依次熄灭。

4、循环实现以上 3 个功能。

【03】单个数码管显示数字

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、逐段点亮最右边一位共阳数码管。

2、最右边一位共阳数码管依次显示数字 0 到 9。

3、循环实现以上 2 个功能。

【04】多个数码管静态显示

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、最右边一位共阳数码管依次显示数字 0 到 9。

2、最右边两位共阳数码管依次显示数字 0 到 9。

3、全部四位共阳数码管依次显示数字 0 到 9。

4、循环实现以上 3 个功能。

【05】多个数码管动态显示

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、系统上电后，全部四位共阳数码管依次显示数字 0 到 9。

2、数码管由 0 开始，每隔一小段时间进行加 1 累计，并在数码管上显示。

3、当累计数值达到 10000 时，复位，又从 0 开始进行加 1 显示。

【06】指示灯与数码管的综合应用

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、系统上电后，全部 LED 灯熄灭，四位数码管显示“0-00”。

2、D1 灯循环闪烁，每闪烁一次，数码管最右边的两位数字加 1 累计，当累计值达到 100 时，复位，由从 0 开始累计。

3、D1 灯每闪烁 10 次，D2 灯的状态翻转 1 次；D2 灯每翻转 1 次，数码管最左边的一位数字加 1 累计，当累计值到 10 时，复位，由从 0 开始累计。

4、循序实现以上两个灯光的闪烁和累计数据显示。

【07】 单按键控制灯光开关

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、上电后，D1 灯闪烁 3 次，模拟灯光检测。

2、循环扫描 SW1 按键的状态。当检测到 SW1 按键按下时，等待其松开后，切换 D1 灯的开关状态。

【08】多按键联合控制灯光开关

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、D1 灯为开关指示灯，D6 和 D8 为照明灯，SW1 按键为总开关，SW2 按键为照明控制开关。系统上电后，总开关关闭。

2、SW1 按键第 1 次按下松开后，总开关打开，D1 灯点亮，此时 SW2 按键有效，可以控制 D6 和 D8 灯；在总开关打开的状态下，SW1 按键按下松开后，总开关关闭，D1 灯熄灭，D6 和 D8 灯也熄灭，SW2 按键无效，按下不起作用。

3、在总开关打开的情况下，SW2 按键有效。SW2 按键第 1 次按下松开后，D6 灯点亮，SW2 按键第 2 次按下松开后，D8 灯点亮,SW2 按键第 3 次按下松开后，D6 和 D8 灯全部熄灭，如此循环往复。

【09】数码管显示按键触发次数

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、系统上电后，全部 LED 灯熄灭，四位数码管显示“0--0”。最左边的数码管显示 SW1 按键按下的次数，最右边的数码管显示 SW2 按键按下的次数。

2、两个按键每次按下松开后，对应数码管上的数字加 1 累计，当累计值达到 10 时，复位，又从 0 开始累计。

3、在按下任何一个按键的时候，必修保持数码管的正常动态显示，不能出现显示中断等异常情况。

【10】外部中断控制灯光开关

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、初始化 P3.3（SW2 按键）为外部中断，下降沿触发，使能相关中断。

2、在主函数中，控制 D8 灯循环闪烁。

3、在外部中断服务函数中，切换 D1 灯的开关状态。

【11】外部信号计数的基本应用

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、初始化 T0（P3.4）为计数功能，8 位重装模式，计数初值为 253，使能相关中断，启动定时/计数器 T0。

2、当 SW2 按键（P3.4）按下 3 次，产生 3 个计数脉冲后，计数器溢出中断。在中断服务函数中，切换 D1 灯的开关状态。

【12】内部信号定时实现秒闪灯

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、初始化 T0 为定时功能，16 位模式，定时时长为 50ms，使能相关中断，启动定时/计数器 T0。

2、在中断服务函数中，每隔 1 秒钟翻转 1 次 D1 灯的状态，实现秒闪灯。

【13】基于定时器的秒表实现

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、系统上电后，四位数码管显示“0.00.0”，即“分.秒.零点一秒”。SW1 按键为秒表启动，SW2 为秒表暂停，SW3 按键为秒表清 0。

2、初始化 T0 为定时功能，16 位模式，定时时长为 50ms，使能相关中断，启动定时/计数器 T0。

3、在定时中断服务函数中，实现 0.1 秒定时，完成秒表的计数功能，分钟的最大计数为 9，即当计数值达到 10 分钟后，复位，从 0 开始计时。

【14】串口数据收发基础

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、初始化串口为 8 位 UART 模式，允许接收，波特率 2400，使能相关中断。

2、成功接收到上位机的一个字节数据后，加 1 后，发送回上位机。

【15】串口远程控制灯光

新建工程、编写程序，在 51 单片机开发板 XMF05A 上，实现以下功能：

1、初始化串口为 8 位 UART 模式，允许接收，波特率 2400，使能相关中断。

2、成功接收到上位机的一个字节数据后，解析并执行相应操作：

0xA1：点亮 D1 灯，返回信息“LED 灯 D1 点亮了!”。

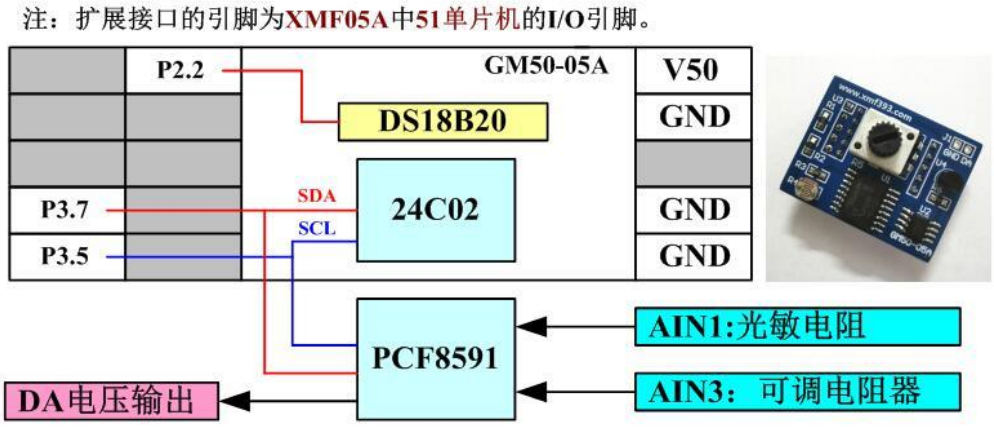
0xA1：点亮 D1 灯，返回信息“LED 灯 D1 熄灭了!”。

其他数据：不操作任何灯光，返回信息“这是一个错误的命令!”。

第 7 章　51 单片机进阶拓展案例项目

51 单片机开发板 XMF05A 搭载多功能综合扩展模块 GM50，在掌握入门案例项目的基础上进行拓展训练。就 DS18B20 温度采样、24C02 数据读写、PCF8591 电压采集、DA 电压输出几个典型应用设计了 9 个进阶拓展案例。

7.1 多功能拓展模块 GM50 的功能结构



7.2 进阶拓展案例项目 9 个

【01】24C02 单字节保存开机次数

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、系统上电后，从 24C02 存储器的 0x00 地址单元读出一个字节数据，进行加 1 后，判断是否大于 99，如果大于 99 则进行清 0 操作。然后将该数据作为系统开机的次数，重新写入 24C02 存储器的 0x00 地址单元。

3、在数码管靠右显示系统开机次数，没有使用的数码管关闭。

【02】24C02 多字节保存按键按下次数

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、四个数码管从左至右，分别显示 SW1 按键、SW2 按键、SW3 按键和 SW4 按键的按下次数，四个按键的按下次数同时依次保存在 24C02 存储器的 0x02、0x03、0x07 和 0x08 四个地址单元中。

3、系统上电后，从 24C02 存储器的四个特定地址单元中将历史按键按下次数读出，并显示到数码管上。

4、循环扫描四个按键状态。当有按键按下时，在该按键的历史按下次数基础上进行加 1 累计，当累计值达到 10 时，复位，又从 0 开始累计。将该数值写入到 24C02 的正确地址单元，并在数码管上显示，然后等到按键松开。在等待按键松开过程中，不能影响数码管正常的动态显示。

【03】PCF8591 单通道电压采样与显示

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、循环采样可调电阻的输出电压，即 PCF8591 通道 3 的 A/D 转换结果。采样结果读出后，进行电压值换算，保留 2 位小数，靠右显示到数码管上。

【04】可调电压控制灯光变化

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、循环采样可调电阻的输出电压，即 PCF8591 通道 3 的 A/D 转换结果。采样结果读出后，进行电压值换算，保留 2 位小数，靠右显示到数码管上。

3、根据可调电阻的输出电压，控制 8 个 LED 灯亮灭：

4.5V < 当前电压： D8 D7 D6 D5 D4 D3 D2 D1，点亮。

4.0V < 当前电压 <= 4.5V： D8 D7 D6 D5 D4 D3 D2，点亮。

3.5V < 当前电压 <= 4.0V： D8 D7 D6 D5 D4 D3，点亮。

3.0V < 当前电压 <= 3.5V： D8 D7 D6 D5 D4，点亮。

2.5V < 当前电压 <= 3.0V： D8 D7 D6 D5，点亮。

2.0V < 当前电压 <= 2.5V： D8 D7 D6，点亮。

1.5V < 当前电压 <= 2.0V： D8 D7，点亮。

1.0V < 当前电压 <= 1.5V： D8，点亮。

当前电压 <= 1.0V： 全部 LED 灯熄灭。

【05】PCF8591 双通道电压采样与显示

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、D8 为光敏电阻电压采样指示灯，D7 为可调电阻电压采样指示灯。

3、循环采样 PCF8591 指定通道的 A/D 转换结果，采样结果读出后，进行电压值换算，保留 2 位小数，靠右显示到数码管上。

4、系统上电后，默认采样光敏电阻电压，并进行显示。

5、SW1 按键按下后，选择 PCF8591 通道 1，D8 灯点亮，进行光敏电阻电压采样；SW2 按键按下后，选择 PCF8591 通道 1，D7 灯点亮，进行可调电阻电压采样；在按键按下时，不能影响电压的正常采样和数码管的正常显示。

【06】PCF8591 的 DAC 输出

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、四个按键分别控制 PCF8591 的 DAC 输出不同电压，D8、D7、D6 和 D5 分别作为不同输出电压的指示灯。

3、系统上电后，PCF8591 的 D/A 通道输出 1V 电压，D8 灯点亮，其余熄灭。

4、循环扫描四个按键的状态，不同按键按下选择不同电压输出：

按下 SW1 按键：输出 1.0V 电压，D8 灯点亮，其余熄灭。

按下 SW2 按键：输出 2.0V 电压，D7 灯点亮，其余熄灭。

按下 SW3 按键：输出 3.5V 电压，D6 灯点亮，其余熄灭。

按下 SW4 按键：输出 4.3V 电压，D5 灯点亮，其余熄灭。

5、用万用表的电压档测量 GM50 模块的 J1 接口，观测输出电压实际数值。

【07】可调电阻控制 DAC 输出

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能扩展模块 GM50 的组合套件上，实现以下功能：

1、将 IIC 的底层驱动代码文件正确移植到工程中。

2、循环采样可调电位器的输出电压，即 PCF8591 通道 3 的 A/D 转换结果。采样结果读出后，进行电压值换算，保留 2 位小数，靠右显示到数码管上。

3、用 PCF8591 通道 3 的采样结果作为 ADC 的参数，控制输出模拟电压。

4、用万用表的电压档测量 GM50 模块的 J1 接口，观察比较电压表测量的电压值与数码管显示的电压值。

【08】DS18B20 温度采样与显示

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能综合扩展模块 GM50 的组合套件上，实现以下功能：

1、将 DS18B20 的底层驱动代码文件正确移植到工程中。

2、循环采集 DS18B20 的温度数据，进行合理的换算后，结果保留 1 位小数，在数码管中靠右进行显示，没有用到的数码管关闭。

3、D1 灯作为采集指示灯，启动温度转换时点亮，结果换算完成熄灭。

【09】温度与电压数据采样上报

新建工程、编写程序，在 51 单片机开发板 XMF05A 与多功能扩展模块 GM50 的组合套件上，采样温度与电压，上报到“嵌入式与物联网串口数据监测助手”上位机软件中进行显示：

1、将 DS18B20 和 IIC 的底层驱动代码文件正确移植到工程中。

2、初始化串口为 8 位 UART 模式，允许接收，波特率 2400，使能相关中断。

3、循环采样 DS18B20 的温度数据和 PCF8591 通道 3 可调电阻的输出电压，经过合理的换算后，温度保留 1 位小数，电压保留 2 位小数，在数码管上显示，显示的内容由 SW1 按键控制选择切换。

4、系统上电后，默认显示温度数据，按下 SW1 按键后，切换至电压显示，再次按下 SW1 按键，又切换到温度显示，如此循环，并累计按键按下的次数。

5、当串口成功接收到一个字节数据后，进行解析，如果为上位机抄收数据帧的命令字“0xA3”，即根据“嵌入式与物联网串口数据监测助手”的通信规约，构建上报数据帧，并通过串口上传到上位机进行数据处理与显示。

【注】：点击“嵌入式与物联网串口数据监测助手”界面下方“通信规约定义”按钮，可查看通信规约的明细定义。

附录 1：欧浩源讲《51 单片机原理与应用开发》视频 22 集目录

【51 单片机入门 01】51 单片机应用开发快速入门
【51 单片机入门 02】LED 跑马灯的实现
【数码管显示 01】数码管的基本工作原理
【数码管显示 02】单个数码管显示数字
【数码管显示 03】多位数码管的静态显示
【数码管显示 04】数码管动态显示基本原理
【数码管显示 05】4 位共阳数码管的动态显示
【数码管显示 06】指示灯与数码管的综合应用
【按键处理 01】按键的工作原理与设计基本思路
【按键处理 02】单按键控制灯光开关
【按键处理 03】多按键联合控制灯光开关
【按键处理 04】数码管显示按键触发次数
【外部中断 01】中断的基本概念与执行过程
【外部中断 02】51 单片机的中断系统
【外部中断 03】51 单片机的外部中断原理与应用
【定时计数 01】51 单片机定时计数器的基本原理
【定时计数 02】外部信号计数的基本应用
【定时计数 03】内部信号定时的基本应用
【定时计数 04】基于 51 单片机定时计数器的秒表
【串行接口 01】51 单片机串行接口基本原理
【串行接口 02】51 单片机串口数据收发基础
【串行接口 03】51 单片机串口控制灯光开关

附录 2：《51 单片机原理与应用开发》教学资源目录汇总专题

资源目录汇总 【XMF05A】51单片机原理与应用开发-教学资源目录汇总（暨小蜜蜂老师的蓝桥杯单片机资料汇总） 置顶

2019-09-10 小蜜蜂 阅读(9032) 赞(46)

广东职业技术学院，欧浩源，以51单片机开发套件XMF05A+GM50为核心，配套教学视频、项目案例、源码分析、课件题库等全方位教学资源，面向课堂教学，服务拓展创新，对接蓝桥杯单片机大赛，形成一个完整的岗课赛证创生态体系。

这是小蜜蜂老师原创和整理的 51 单片机教学资源的链接汇总目录专题，包含：视频教程在线观看、网盘下载、51 开发套件资料、教学精华笔记、案例项目源码、课件题库整理、小蜜蜂老师的蓝桥杯单片机资源链接汇总入口...形式多样，内容丰富，持续更新，欢迎观看！

专题网址：https://www.xmf393.com/2019/09/10/51mcu/