|  |  |  |  |
| --- | --- | --- | --- |
| **学生学号** | 0122015710114 | **实验课成绩** |  |

**学 生 实 验 报 告 书**

|  |  |
| --- | --- |
| **实验课程名称** | 单片机及嵌入式系统原理 |
| **开 课 学 院** | 信息工程学院 |
| **指导教师姓名** | 周伟 |
| **学 生 姓 名** | 胡姗 |
| **学生专业班级** | 信息2001 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2022 | -- | 2023 | 学年 | 第 | 一 | 学期 |

**实验教学管理基本规范**

实验是培养学生动手能力、分析解决问题能力的重要环节；实验报告是反映实验教学水平与质量的重要依据。为加强实验过程管理，改革实验成绩考核方法，改善实验教学效果，提高学生质量，特制定实验教学管理基本规范。

1. 本规范适用于理工科类专业实验课程，文、经、管、计算机类实验课程可根据具体情况参照执行或暂不执行。
2. 每门实验课程一般会包括许多实验项目，除非常简单的验证演示性实验项目可以不写实验报告外，其他实验项目均应按本格式完成实验报告。
3. 实验报告应由实验预习、实验过程、结果分析三大部分组成。每部分均在实验成绩中占一定比例。各部分成绩的观测点、考核目标、所占比例可参考附表执行。各专业也可以根据具体情况，调整考核内容和评分标准。
4. 学生必须在完成实验预习内容的前提下进行实验。教师要在实验过程中抽查学生预习情况，在学生离开实验室前，检查学生实验操作和记录情况，并在实验报告第二部分教师签字栏签名，以确保实验记录的真实性。
5. 教师应及时评阅学生的实验报告并给出各实验项目成绩，完整保存实验报告。在完成所有实验项目后，教师应按学生姓名将批改好的各实验项目实验报告装订成册，构成该实验课程总报告，按班级交课程承担单位（实验中心或实验室）保管存档。
6. 实验课程成绩按其类型采取百分制或优、良、中、及格和不及格五级评定。

**附表：实验考核参考内容及标准**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 观测点 | 考核目标 | 成绩组成 |
| 实验预习 | 1. 预习报告 2. 提问 3. 对于设计型实验，着重考查设计方案的科学性、可行性和创新性 | 对实验目的和基本原理的认识程度，对实验方案的设计能力 | 20% |
| 实验过程 | 1. 是否按时参加实验 2. 对实验过程的熟悉程度 3. 对基本操作的规范程度 4. 对突发事件的应急处理能力 5. 实验原始记录的完整程度 6. 同学之间的团结协作精神 | 着重考查学生的实验态度、基本操作技能；严谨的治学态度、团结协作精神 | 30% |
| 结果分析 | 1. 所分析结果是否用原始记录数据 2. 计算结果是否正确 3. 实验结果分析是否合理 4. 对于综合实验，各项内容之间是否有分析、比较与判断等 | 考查学生对实验数据处理和现象分析的能力；对专业知识的综合应用能力；事实求实的精神 | 50% |

实验课程名称： 单片机及嵌入式系统原理

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验项目名称** | **数据采集显示和变送** | | | **实验成绩** |  |
| **实 验 者** | **胡姗** | **专业班级** | **信息2001** | **组 别** |  |
| **同 组 者** | **无** | | | **实验日期** | **2022年12月10日** |
| 第一部分：实验预习报告（包括实验目的、意义，实验基本原理与方法，主要仪器设备及耗材，实验方案与技术路线等）  **一、实验目的**  1、能够灵活使用1602液晶显示任意字符串。  2、掌握A/D的基本概念和性能指标。  3、掌握PCF8591的I2C软件编写。  **二、实验基本原理**  PCF8591是一个单电源低功耗8位CMOS数据采集器，有4路模拟输入，1路模拟输出，一个串行I2C总线接口与单片机通信。 PCF8591的ADC是逐次逼近型的，转换速率算是中速，但是它的速度瓶颈在 I2C 通信上。由于 I2C 通信速度较慢，所以最终的 PCF8591 的转换速度，直接取决于 I2C 的通信速率。由于 I2C 速度的限制，所以 PCF8591 得算是个低速的 AD 和 DA 的集成，通过双排插针，AIN0测的是电位器分压值，AIN1和AIN2测的是GND的值，AIN3测的是+5V的值。但是当输入信号超过基准电压时，AD得到的始终是最大值255，即AD无法测量超过基准电压的信号。同时输入电压也不能超过VCC(+5V)，否则会损坏ADC芯片。  **三、实验内容**  通过PCF8591的I2C通信接口取得ADC芯片AIN0的测量值，将测量值转换成电压值，与基准电压2.5V进行比较，得到AIN0测量值占基准电压的百分比，然后将AIN0测量值和百分比输出到LCD1602上显示。 | | | | | |
| 第二部分：实验过程记录（可加页）（包括实验原始数据记录，实验现象记录，实验过程发现的问题等）  在main函数中首先使能总中断，T0定时10ms，初始化LCD1602，每300ms取得一次AIN0的测量值及其占基准电压的百分比，刷新LCD1602的显示。  void main()  {  unsigned char val;  unsigned char str[10];  unsigned char por[6];  EA = 1;  ConfigTimer0(10);  InitLcd1602();  while (1)  {  if (flag300ms)  {  flag300ms = 0;  LcdShowStr(0, 0, "AIN0:");  val = GetADCValue(0);  ValueToString(str, val);  LcdShowStr(6, 0, str);  val = GetADCValue(0);  ValueToPor(por,val);  LcdShowStr(0,1,por);  }  }  }  取得AIN0的测量值函数如下，首先产生总线起始信号，然后寻址PCF8591，如果无应答，产生总线停止信号，返回0，如果有应答，写入控制字节，选择通道，然后切换成读操作，读取该通道的测量值，最后产生总线停止信号，返回读取的测量值。  unsigned char GetADCValue(unsigned char chn)  {  unsigned char val;  I2CStart();  if (!I2CWrite(0x90))  {  I2CStop();  return 0;  }  I2CWrite(0x40|chn);  I2CStart();  I2CWrite(0x91);  I2CReadACK();  val = I2CReadNAK();  I2CStop();  return val;  }  将测量值转换成电压值函数如下，因基准电压2.5V的测量结果为255，所以val = (val\*25) / 255得到的结果是电压值乘以十。然后通过val/10和val%10得到电压值的个位和小数位。  void ValueToString(unsigned char \*str, unsigned char val)  {  val = (val\*25) / 255;  str[0] = (val/10) + '0';  str[1] = '.';  str[2] = (val%10) + '0';  str[3] = 'V';  str[4] = '\0';  }  将测量值转换成占基准电压百分比的函数如下，首先得到电压值的十倍，然后通过num=(val\*100)/25得到占基准电压的百分比。  void ValueToPor(unsigned char \*str,unsigned char val)  {  unsigned char num;  val = (val\*25) / 255;  num=(val\*100)/25;  str[2]=(num%10)+'0';  num=num/10;  str[1]=(num%10)+'0';  num=num/10;  str[0]=(num%10)+'0';  str[3]='%';  str[4]='\0';  }  教师签字\_\_\_\_\_\_\_\_\_\_ | | | | | |
| 第三部分 结果与讨论（可加页）  一、实验结果分析（包括数据处理、实验现象分析、影响因素讨论、综合分析和结论等）  LCD1602第一行显示AIN0测量得到的电压值，第二行显示AIN0测量的电压值占基准电压的百分比。  IMG_20221217_112350_edit_10221320085939  通过调节R62，AIN0的采样值发生改变，同时所占百分比的显示也发生改变。  IMG_20221217_112913_edit_10310749805717  二、思考题  （1）请简要说明 AD 分辨率的计算方法；  答：n位ADC的分辨率为满刻度量程与的比值，即。  （2）请总结 AIN0 采样值随 R62 调节的变化规律。  答：随着R62调节，AIN0采样值从0.0V 到2.5V以0.1V均匀变化。  **附录：**  **Lcd1602.c文件：**  #include <reg52.h>  #define LCD1602\_DB P0  sbit LCD1602\_RS = P1^0;  sbit LCD1602\_RW = P1^1;  sbit LCD1602\_E = P1^5;  void LcdWaitReady()  {  unsigned char sta;  LCD1602\_DB = 0xFF;  LCD1602\_RS = 0;  LCD1602\_RW = 1;  do {  LCD1602\_E = 1;  sta = LCD1602\_DB;  LCD1602\_E = 0;  } while (sta & 0x80);  }  void LcdWriteCmd(unsigned char cmd)  {  LcdWaitReady();  LCD1602\_RS = 0;  LCD1602\_RW = 0;  LCD1602\_DB = cmd;  LCD1602\_E = 1;  LCD1602\_E = 0;  }  void LcdWriteDat(unsigned char dat)  {  LcdWaitReady();  LCD1602\_RS = 1;  LCD1602\_RW = 0;  LCD1602\_DB = dat;  LCD1602\_E = 1;  LCD1602\_E = 0;  }  void LcdSetCursor(unsigned char x, unsigned char y)  {  unsigned char addr;  if (y == 0)  addr = 0x00 + x;  else  addr = 0x40 + x;  LcdWriteCmd(addr | 0x80);  }  void LcdShowStr(unsigned char x, unsigned char y, unsigned char \*str)  {  LcdSetCursor(x, y);  while (\*str != '\0')  {  LcdWriteDat(\*str++);  }  }  void InitLcd1602()  {  LcdWriteCmd(0x38);  LcdWriteCmd(0x0C);  LcdWriteCmd(0x06);  LcdWriteCmd(0x01);  }  **I2C.c文件：**  #include <reg52.h>  #include <intrins.h>  #define I2CDelay() {\_nop\_();\_nop\_();\_nop\_();\_nop\_();}  sbit I2C\_SCL = P3^7;  sbit I2C\_SDA = P3^6;  void I2CStart()  {  I2C\_SDA = 1;  I2C\_SCL = 1;  I2CDelay();  I2C\_SDA = 0;  I2CDelay();  I2C\_SCL = 0;  }  void I2CStop()  {  I2C\_SCL = 0;  I2C\_SDA = 0;  I2CDelay();  I2C\_SCL = 1;  I2CDelay();  I2C\_SDA = 1;  I2CDelay();  }  bit I2CWrite(unsigned char dat)  {  bit ack;  unsigned char mask;  for (mask=0x80; mask!=0; mask>>=1)  {  if ((mask&dat) == 0)  I2C\_SDA = 0;  else  I2C\_SDA = 1;  I2CDelay();  I2C\_SCL = 1;  I2CDelay();  I2C\_SCL = 0;  }  I2C\_SDA = 1;  I2CDelay();  I2C\_SCL = 1;  ack = I2C\_SDA;  I2CDelay();  I2C\_SCL = 0;  return (~ack);  }  unsigned char I2CReadNAK()  {  unsigned char mask;  unsigned char dat;  I2C\_SDA = 1;  for (mask=0x80; mask!=0; mask>>=1)  {  I2CDelay();  I2C\_SCL = 1;  if(I2C\_SDA == 0)  dat &= ~mask;  else  dat |= mask;  I2CDelay();  I2C\_SCL = 0;  }  I2C\_SDA = 1;  I2CDelay();  I2C\_SCL = 1;  I2CDelay();  I2C\_SCL = 0;  return dat;  }  unsigned char I2CReadACK()  {  unsigned char mask;  unsigned char dat;  I2C\_SDA = 1;  for (mask=0x80; mask!=0; mask>>=1)  {  I2CDelay();  I2C\_SCL = 1;  if(I2C\_SDA == 0)  dat &= ~mask;  else  dat |= mask;  I2CDelay();  I2C\_SCL = 0;  }  I2C\_SDA = 0;  I2CDelay();  I2C\_SCL = 1;  I2CDelay();  I2C\_SCL = 0;  return dat;  }  **main.c**  #include <reg52.h>  bit flag300ms = 1;  unsigned char T0RH = 0;  unsigned char T0RL = 0;  void ConfigTimer0(unsigned int ms);  unsigned char GetADCValue(unsigned char chn);  void ValueToString(unsigned char \*str, unsigned char val);  void ValueToPor(unsigned char \*str, unsigned char val);  extern void I2CStart();  extern void I2CStop();  extern unsigned char I2CReadACK();  extern unsigned char I2CReadNAK();  extern bit I2CWrite(unsigned char dat);  extern void InitLcd1602();  extern void LcdShowStr(unsigned char x, unsigned char y, unsigned char \*str);  void main()  {  unsigned char val;  unsigned char str[10];  unsigned char por[6];  EA = 1;  ConfigTimer0(10);  InitLcd1602();  while (1)  {  if (flag300ms)  {  flag300ms = 0;  LcdShowStr(0, 0, "AIN0:");  val = GetADCValue(0);  ValueToString(str, val);  LcdShowStr(6, 0, str);  val = GetADCValue(0);  ValueToPor(por,val);  LcdShowStr(0,1,por);  }  }  }  unsigned char GetADCValue(unsigned char chn)  {  unsigned char val;  I2CStart();  if (!I2CWrite(0x90))  {  I2CStop();  return 0;  }  I2CWrite(0x40|chn);  I2CStart();  I2CWrite(0x91);  I2CReadACK();  val = I2CReadNAK();  I2CStop();  return val;  }  void ValueToString(unsigned char \*str, unsigned char val)  {  val = (val\*25) / 255;  str[0] = (val/10) + '0';  str[1] = '.';  str[2] = (val%10) + '0';  str[3] = 'V';  str[4] = '\0';  }  void ValueToPor(unsigned char \*str,unsigned char val)  {  unsigned char num;  val = (val\*25) / 255;  num=(val\*100)/25;  str[2]=(num%10)+'0';  num=num/10;  str[1]=(num%10)+'0';  num=num/10;  str[0]=(num%10)+'0';  str[3]='%';  str[4]='\0';  }  void ConfigTimer0(unsigned int ms)  {  unsigned long tmp;  tmp = 11059200 / 12;  tmp = (tmp \* ms) / 1000;  tmp = 65536 - tmp;  tmp = tmp + 12;  T0RH = (unsigned char)(tmp>>8);  T0RL = (unsigned char)tmp;  TMOD &= 0xF0;  TMOD |= 0x01;  TH0 = T0RH;  TL0 = T0RL;  ET0 = 1;  TR0 = 1;  }  void InterruptTimer0() interrupt 1  {  static unsigned char tmr300ms = 0;    TH0 = T0RH;  TL0 = T0RL;  tmr300ms++;  if (tmr300ms >= 30)  {  tmr300ms = 0;  flag300ms = 1;  }  } | | | | | |