**实训日志**

|  |  |  |
| --- | --- | --- |
| 学校：宝鸡文理学院 | 专业：电子科学与技术 | 学号：201691074036 |
| 日期：2018/9/9 | 星期：星期三 | 姓名：刘超飞 |
| 内容：（课程主要内容，代码/作业，收获，感悟，建议等）   1. 课程主要内容   串口：  UART：  串行通信：    串口通信里的几个参数：  波特率：  数据位：  停止位：  奇偶校验位：  流控制位：  串口开发流程：  串口初始化：ser\_init()  >配置数据位、停止位  >配置波特率  >串口一系列使能  读写：  读数据：char ser\_getc()  写数据：ser\_putc(char c);  串口控制led：  uartandled.rar  串口控制蜂鸣器：  uartandbzz.rar  rfid开发流程：  1.定义读写buf[16]  unsigned char rbuf[16];  unsigned char wbuf[16];  2.配置IO口时钟  LPC\_SYSCON->SYSAHBCLKCTRL |= 1 << 16;  3.串口初始化  ser\_init();  4.rfid初始化  rfid\_init();  5.while()：循环处理读写业务  {  1.定义一个字符用来存储串口接到的数据  int c;  2.从串口读取字符  c = ser\_getc();  3.判断读取到的字符是r还是w  如果是r：  读取rfid的卡片信息  如果是w：  读取剩余串口信息(16个字节)  将获取到的串口信息(想要设置的卡片信息)写入rfid卡中  switch(c)  {  case 'r':  rfid\_read(rbuf);//用来读取rfid卡片的数据  //将读到的数据，即rbuf中的数据通过串口发送给串口助手  for(i = 0; i < 16; i++)  {  ser\_putc(rbuf[i]);  }  break;  case 'w':  //读取剩余16字节的数据，将其存入wbuf中，便于写入rfid卡  for(i = 0; i < 16; i++)  {  wbuf[i] = ser\_getc();  }  //往rfid卡中写入数据  rfid\_write(wbuf);  break;  }  }  ser\_init();  /\*需要添加的头文件\*/  #include "serial.h"  /\*函数声明\*/  void ser\_init(void)；  ser\_getc();  /\*需要添加的头文件\*/  #include "serial.h"  /\*  \*函数名：ser\_getc  \*函数功能：从串口中读取一个字节  \*函数参数：void  \*函数返回值：int：返回串口中读到的数据  \*/  int ser\_getc(void)  ser\_putc();  /\*需要添加的头文件\*/  #include "serial.h"  /\*  \*函数名：ser\_putc  \*函数功能：往串口中写入一个字节  \*函数参数：char c:想要写入的字符  \*函数返回值：void  \*/  void ser\_putc(char c)；  rfid\_init();  /\*头文件\*/  #include "rfid.h"  void rfid\_init(void);  rfid\_read();  /\*头文件\*/  #include "rfid.h"  /\*函数名：rfid\_read  \*函数功能：是将rfid卡中的数据读取出来存入rbuf中  \*函数参数：  \* unsigned char \*rbuf：存储rfid卡信息的内存的首地址  \*函数返回值：成功返回0，失败返回-1  \*/  int rfid\_read(unsigned char \*rbuf);  rfid\_write();  /\*头文件\*/  #include "rfid.h"  /\*函数名：rfid\_write  \*函数功能：是将wbuf中的数据写入rfid卡中  \*函数参数：  \* unsigned char \*wbuf：存储想要写入rfid卡中的信息的首地址  \*函数返回值：成功返回0，失败返回-1  \*/  int rfid\_write(unsigned char \*wbuf)；  Linux操作系统下的串口编程：  <1>初始化串口  /\*需要添加的头文件\*/  #include "serial.h"  /\*  \*函数名：init\_uart()  \*函数功能：串口初始化  \*函数参数：void  \*函数返回值：int：返回串口的文件描述符  \*/  int init\_uart();  <2>往串口中写入数据  /\*需要添加的头文件\*/  #include "serial.h"  /\*  \*函数名：writecmd()  \*函数功能：发送一个命令给m0  \*函数参数：int fd: 初始化串口返回的文件描述符  \* const char \*str：想要发送的指令  \*函数返回值：int：成功返回0  \*/  int writecmd(int fd, const char \*str);  <3>从串口中读取数据  /\*需要添加的头文件\*/  #include "serial.h"  /\*  \*函数名：myrecvmsg()  \*函数功能：接收m0传过来的数据  \*函数参数：int fd: 初始化串口返回的文件描述符  \* const char \*buf：接收m0数据的内存空间首地址  \* int data\_len:想要接收的字符个数  \*函数返回值：int：成功返回0，失败返回-1  \*/  int myrecvmsg(int fd, unsigned char \*buf, int data\_len);  1》初始化串口：  int fd = -1；  fd = init\_uart();  char buf[20] = {0};  2.给m0发送指令  writecmd(fd, "read\_rfid");  3.从串口中读取m0发送过来的数据  myrecvmsg(fd, buf, 17);   1. 作业   .h 文件  #ifndef \_\_MYSTRING\_H\_\_  #define \_\_MYSTRING\_H\_\_  char \*mystrcpy(char \*dest, char \*src);  int mystrlen(const char \*src);  char \*mystrtok(char \*src, char c);  char \*mystrcat(char \*s1, const char \*s2);  int myatoi(char \*nptr);  #endif  .c 文件  #include <stdio.h>  #include <string.h>  char \*mystrcpy(char \*dest, char \*src)  {  if(NULL == dest || NULL == src)  {  puts("NULL pointer error.");  return NULL;  }  char \*p = dest;  while(\*src)  {  \*p = \*src;  src++;  p++;  }  \*p = 0;  return dest;  }  int mystrlen(const char \*src)  {  if(NULL == src)  {  puts("src is null.");  return -1;  }    int len = 0;  while(\*src)  {  len++;  src++;  }  return len;  }  char \*mystrtok(char \*src, char c)  {  if(NULL == src)  {  return NULL;  }  char \*p = src;  while(\*p)  {  if(\*p == c)  {  \*p = 0;  break;  }  p++;  }  return src;  }  char \*mystrcat(char \*s1, const char \*s2)  {  if(NULL == s1 && NULL != s2)  {  return s2;  }  if(NULL != s1 && NULL == s2)  {  return s1;  }  if(NULL == s1 && NULL == s2)  {  return NULL;  }  char \*p = s1 + strlen(s1);  while(\*s2)  {  \*p = \*s2;  s2++;  p++;  }  \*p = 0;  return s1;  }  int myatoi(char \*nptr)  {  if(NULL == nptr)  {  puts("nptr is null.");  return -1;  }  int sum = 0;  while(\*nptr)  {  sum = sum \* 10 + (\*nptr - '0');  nptr++;  }  return sum;  }  #include <stdio.h>  #include "mystring.h"  #include <string.h>  int main()  {  char s1[20] = {0};  char s2[] = "farsight";  char \*p = mystrcpy(s1, s2);  puts(p);  int len = mystrlen(s2);  printf("%d\n",len);  //p = mystrtok(s2, 'i');  //puts(p);  memset(s1, 0, 20);  mystrcpy(s1, "hello ");  mystrcat(s1, s2);  puts(s1);  char s[] = "521";  int ret = myatoi(s);  printf("ret:%d\n",ret);  return 0;  }   1. 收获与建议   今天主要学习了裸机开发中的串口调试部分。串口的初始化，串口的写入与输出，有两种编写方法：keil5 环境，linux 环境。  从编写难度方面来看，linux 环境写的编写更简单，它只需要写出头文件，函数声明，函数调用这些就可以实现串口的调试功能；在keil5 环境下编写则需要查看芯片手册，查看串口的应用规则，然后才能编写，全英文的芯片手册对我这种英语大白来说简直可以说是天书。但是芯片手册中的内容又是串口调试的基础，如果不懂得串口运行的原理，纵使在linux 环境下也程序也是无从编起的。  总体感觉对于c语言学习来说，理论基础的重要性是不言而喻的，前期的积累才能出现后期的勃发。 | | |