Linux 下 ARM 和单片机的串口通信设计

摘要：介绍 Linux 环境下串口通信的设计方法和步骤，并介绍了 ARM9 微处理器 s3c2440 在 Linux 下和 C8051Fxxx 系列单片机进行串行通信的设计方法，给出了硬件连接和通信程序流程图。该方法可靠、实用，适用于大多数 Linux ARM 和单片机串口通信的场合。

关键词： Linux; ARM; 单片机； 串口通信

0 引言

数据采集系统中由于单片机侧重于控制，数据处理能力较弱，对采集的数据进行运算处理比较繁琐，如果通过串口与上位机通信，利用上位机强大的数据处理能力和友好的控制界面对数据进行处理和显示则可以提高设计效率。串口通信以其简单的硬件连接，成熟的通信协议，成为上下位机之间通信的首选。移植了 Linux 操作系统的 s3c2440 可以在 Linux 环境下操作串口，降低了串口操作的难度，可以使开发者集中精力开发大规模的应用程序，而不必在操作底层设计上耗费时间。

1 硬件连接

s3c2440 是三星公司生产的基于 ARM9 核的处理器，采用 3.3 V 电压供电；C8051Fxxx 系列单片机是美国 CYGNAL 公司推出的与 8051 兼容的高性能高速单片机，采用 3.3 V 电压供电。两者供电电压相同，所以进行串行口通信时不需要进行电平转换。硬件连接采用最常用的 TXD,RXD,GND 三线连接方式。注意采用交叉连接方式，即 TXD RXD,RXD TXD.

2 Linux 下串口通信

2.1 Linux 下串口设备描述

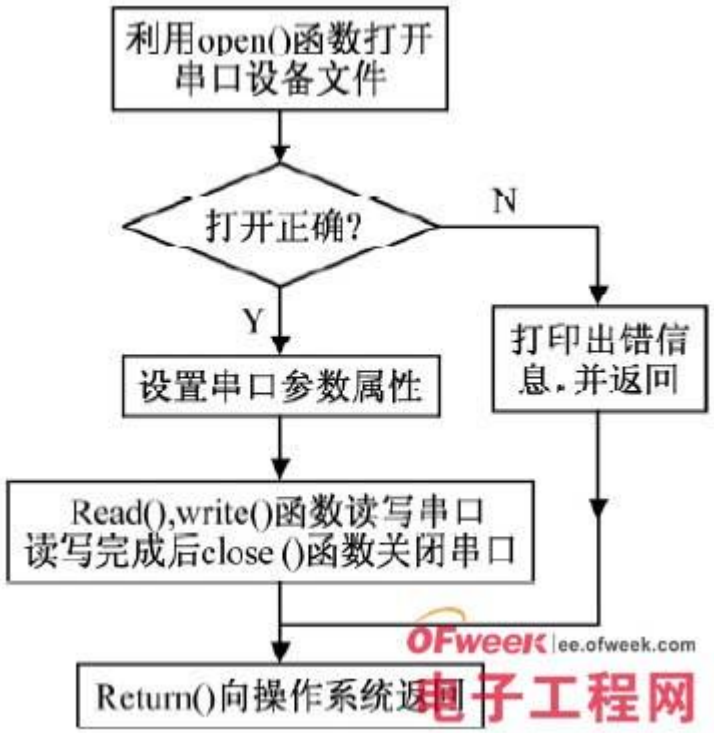
s3c2440 上移植了 Linux 2.6.32 操作系统，加载了 s3c2440 的串口驱动程序，通过 Linux 提供的串口操作函数和文件操作函数把对串口的操作等同于文件操作，降低了串口的操作难度，提高了效率。在程序中设备和文件都是通过文件描述符来操作的，文件描述符在 Linux 内核中是一个非负整数。Linux 设备文件都存放在“/dev”目录下，串口也不例外，在/dev 中可以找到串口对应的设备文件，本文对应的串口 1 的设备文件路径是“/dev /ttySAC1”.

2.2 Linux 下串口通信程序设计

串口通信需要设置一些参数，如波特率、数据位、停止位，输入输出方式等。这些参数都存在于 Linux 提供的 termios 结构中，该结构是 Linux 系统用于查

询和操作各个终端的一个标准接口，定义在头文件< ter-mios. h > 中，如下所示：

STruct termios{tcflag_t c_iflag; /* 输入标志* /tcflag_t c_oflag; /* 输出标志* /tcflag_t c_cflag /* 控制标志* /tcflag_t c_lflag /* 本地标志* /cc_t c_cc[NCCS]; /* 控制特性* /} ;Linux 串口通信步骤可分为以下三步，操作流程如图 1 所示。



▲图 1

第一步： 打开串口调用 open（ ） 函数打开串口设备文件，若出错则返回 - 1,成功则返回文件句柄。

#define UART1 /dev /ttySAC1int fd;fd = open（ “UART1”,O_RDWR） /* 以可读可写方式打开串口设备* /

第二步： 设置串口属性函数 tcsetattr （ ） 可以设置串口的结构属性，tcgetatt（ ） 可以得到串口的结构属性。在 termios 结构中，？ &=s ?最重要的是 c_cflag,用户通过对其进行赋值可以实现串口波特率、数据位、停止位、奇偶校验位等参数的设置。c_cc 数组中的两个变量 VMIN 和 VTIME 判断是否返回输入，c _cc[VTIME]设定字节输入时间计时器，c _cc[VMIN]设定满足读取功能的最低接收字节数。这两个变量的值要设定合理,才能保证串口的通信成功率。

int set_attr （ int fd）{struct termios newtio,oldtio;tcgetattr （ fd,&oldtio） ;cfsetispeed（ &newtio,B9600） ; /* 设置读波特率为 9600*

/cfsetospeed（ &newtio,B9600） ; /* 设置写波特率为 9600* /memset （ &newtio,0, sizeof （ newtio） ）; newtio. c_cflag = CS8 | CREAD; /* 设置数据位为 8 位并且使能接收* /newtio. c_cflag & = ~ PARENB; /* 不进行奇偶校验* /newtio. c_cflag & = ~ CSTOPB; /* 1 位停止位* /newtio. c_cc[VMIN]= 1; /* 当接收到一个字节数据就读取* /newtio. c_cc[VTIME]= 0; /* 不使用计时器* /tcflush（ fd,TCIOFLUSH） ; /* 刷清输入输出缓冲区* /tcsetattr（ fd,TCSANOW,&newtio） /* 使设置的终端属性立即生效* /}

第三步： 串口读写，串口关闭设置完通信参数后，就可以用标准的文件读写命令 read（ ） 和 write（ ） 操作串口了。最后在退出之前，用 close（ ） 函数关闭串口。

void rd_wr（ ）{write（ fd,wbuf,10） ;usleep（ 500000） ; /* 延时 50 ms 等待下位机发送数据* /read（ fd, rbuf,10） ;printf（ “read string is %s \n”, rbuf） ;}

3 通信程序设计

ARM 与单片机的串口通信程序包括两方面： 一方面是作为上位机的 ARM 的串口通信程序，另一方面是作为下位机的单片机的串口通信程序。在通信之前必须制定合理的通信协议以保证通信的可靠性和成功率。现约定双方通信协议如下：

（ 1） 波特率为 9600 bit /s,帧格式为 1 - 8 - N - 1（ 1 位起始位，8 位数据位，无奇偶校验，1 位停止位） ;（ 2） 由于上位机 ARM 的速度远远高于下位机单片机的速度，所以采用上位机主动联络，下位机等待的方式。在数据传送前 ARM 先发送联络信号/0xaa,单片机收到后回答一个/0xbb,表示可以发送，否则继续联络；（ 3） 单片机端可以有中断和查询方式收发串口数据。本文采用中断方式；（ 4） ARM 处理器 s3c2440 采用 UART1 和单片机通信，UART0 则作为 s3c2440 终端控制台。

3.1 上位机 ARM 的通信程序设计

由于 s3c2440 移植了定制和裁剪后的 Linux2.6.32 内核的操作系统，对串口的操作采用上述的 Linux 下串口操作方法，程序流程图如图 2 所示。

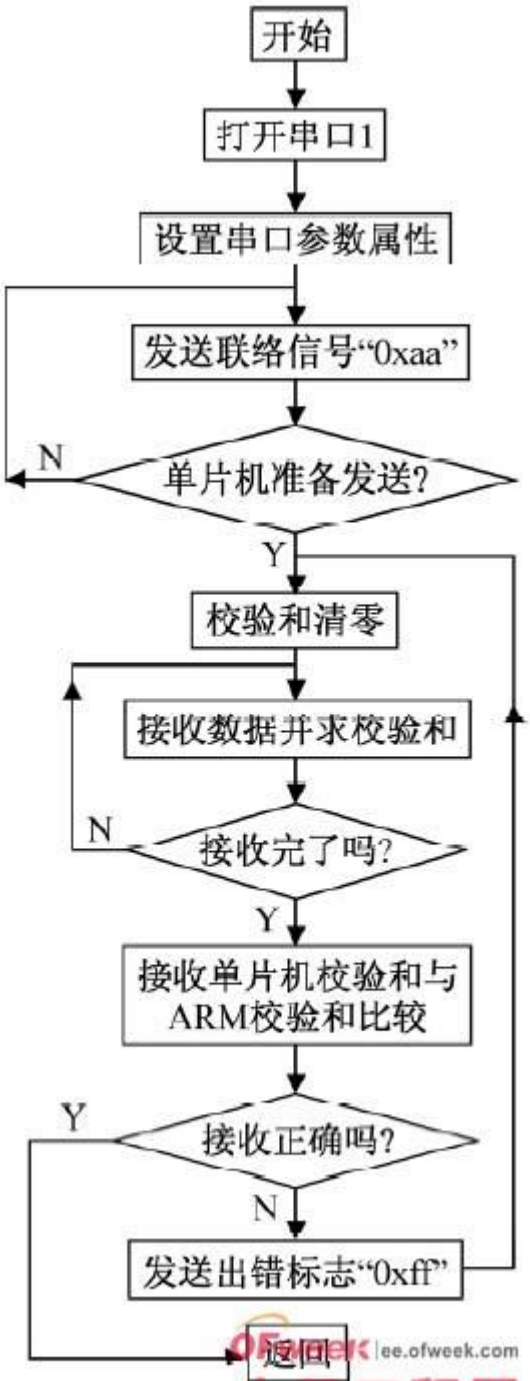
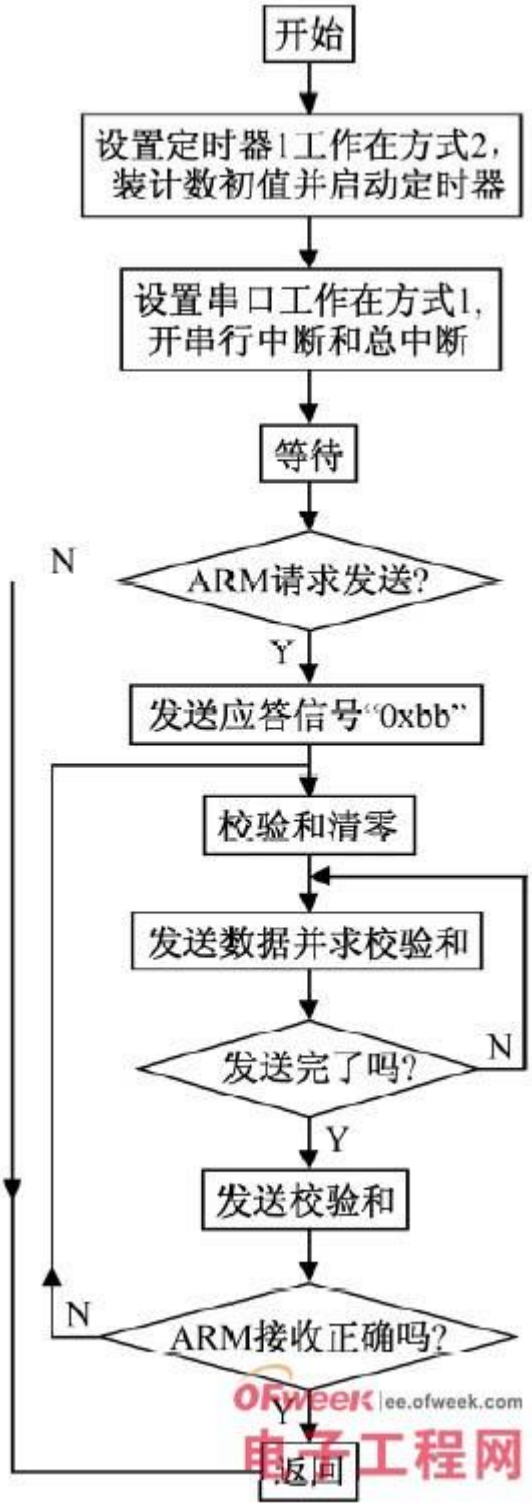


图 2



3.2 下位机单片机的通信程序设计

选用 C8051F021 的定时器 T1 作为波特率发生器，晶振采用 11.0592 MHz，定时器工作在方式 2，计数初值为 0xfd，串口工作在串行方式 1（ 1－8－N－1），采用中断方式收发数据。

4 结束语

随着近年来嵌入式 Linux 在国内的应用范围日益壮大，基于 ARM 平台的嵌入式 Linux 设备也将会越来越多地用在数据采集中作为上位机对数据进行处理、显示、存储、发送。本文所介绍方案适用于大多数场合 Linux 下 ARM 和单片机的串口通信设计，设计人员只需根据自己的实际需要修改或重新制定通信协议即可。另外需要注意的是由于上位机 ARM 的速度比单片机快很多，所以一次不能发送过多的数据，否则极有可能使发送缓冲区溢出而出现数据丢失的现象，开发人员要根据通信双方设备的状况选择合适的帧长度，以达到最佳的传输状态。