ARM-Linux 平台下 GPS 信号的采集与处理研究

陈未峰,李 兵

(西华大学 数学与计算机学院,四川 成都 610039)

摘 要: GPS 信息的采集是导航定位系统的重要组成部分。分析了 GPS 普遍采用的 NMEA-0183 通信协议,然后介绍了目标平台及交叉编译环境的建立。在此基础上实现了嵌入式 ARM-Linux 平台下 GPS 的数据采集与处理,为导航定位系统或者 GIS 系统的应用奠定了基础。

关键词: NMEA-0183 协议;GPS;ARM-Linux;交叉编译

中图分类号: TP316.81 **文献标识码**: A

Study on GPS signal acquisition and processing based on ARM-Linux platform

CHEN Wei-Feng, LI Bing

(School of Mathematics and Computer Engineering, XIHUA University, Chengdu 610039, China)

Abstract: The GPS data acquisition is the important element of navigation orientation system. Firstly, analyzing NMEA-0183 communication protocol used generally by GPS. Secondly, introducing goal platform and building cross compiling environment. Then the GPS data acquisition and processing system based on embedded ARM-Linux platform has been realized, which plays an important role in navigation orientation system or GIS application.

Key words: NMEA-0183 protocol; GPS; ARM-Linux; cross compile

定位和导航是很多便携移动设备以及汽车电子设备的重要功能之一,所以 GPS 在上述设备中得到了广泛的应用。基于 ARM-Linux 的嵌入式平台以其开放性、安全性、健壮性和稳定性越来越成为各种便携设备和车载导航设备的主要开发平台。如何实现 GPS 模块和嵌入式 ARM-Linux 平台之间的通信成了实现系统定位导航的基础。

本文主要研究 GPS 模块与 ARM-Linux 平台之间采用异步串行传送方式进行数据传送的问题,利用多线程编程技术实现 GPS 信号采集与处理,并介绍了一种 WGS 坐标向地方坐标的转换方法。与 GPS 通信可选的协议有很多种,目前普遍采用的是 NMEA-0183 通信协议。

1 NMEA-0183 通信协议

NMEA-0183 协议[1]是为了在不同的 GPS 导航设备中建立统一的海事无线电技术委员会(BTCM)标准,由美国国家海洋电子协会 NMEA (National Marine Electronics Association) 制定的通信协议,其中规定了海用和陆用 GPS 接收设备输出的定位位置数据、时间、卫星状态、接收机状态等信息。除 NMEA-0183 协议之外,还有差分用的 RTCMSC-104 格式,各个厂商互不兼容的二进制格式等,但以 NMEA-0183 使用最广泛。为实现 ARM-LINUX 平台与 GPS 之间的通信,应清楚协议规定的 GPS 输出的数据格式和报文。NMEA-0183 规定的格式如下:

波特率:4 800 b/s

数据位:8 bit

奇偶校验:无

开始位:1 bit

停止位:1 bit

报文格式:报文的语句串(十进制 ASCII 码)格式全部信息如图 1。

图 1 中具体内容:$为串头,表示串开始;GP 为交谈

$GPXXX	,ddd	…	,ddd	*hh	<CR><LF>

图 1 NMEA-0183 报文格式

识别符。XXX 为语句名，NMEA 规定的常用语句有以下 6 种：GGA，卫星定位信息；GLL，地理位置-经度和纬度；GSA，GNSS DOP 偏差信息，说明卫星定位的信号的优劣情况；GSV，GNSS 天空范围内的卫星；RMC，最基本的 GNSS 信息，指能够达到定位目的的基本信息等语句。ddd 为数据字段，字母或数字，“，”为域分隔符；* 表示串尾；hh 表示$与 * 之间所有字符代码的校验和；<CR>为回车控制符；<LF>为换行控制符。

在实际的 GPS 应用中，并不会用到 NMEA 的全部信息，而是根据具体的需要，从中选取有用的信息，忽略其余的信息内容。下面以 GPRMC 语句为例来介绍。该语句包含时间、日期、方位、速度和磁偏角等信息，基本上可以满足一般的导航需求。GPRMC 语句的结构为：$GPRMC，<1>，<2>，<3>，<4>，<5>，<6>，<7>，<8>，<9>，<10>，<11>，*hh<CR><LF>。

数据区说明如下：

(1)UTC 时间，hhmmss.sss(时分秒.毫秒)格式；

(2)定位状态，A=有效定位，V=无效定位；

(3)纬度 ddmm.mmmm (度分)格式(前面的 0 也将被传输)；

(4)纬度半球 N(北半球)或 S(南半球)；

(5)经度 dddmm.mmmm(度分)格式(前面的 0 也将被传输)；

(6)经度半球 E(东经)或 W(西经)；

(7)地面速率(000.0~999.9 节，前面的 0 也将被传输)；

(8)地面航向(000.0~359.9 度，以真北为参考基准，前面的 0 也将被传输)；

(9)UTC 日期，ddmmyy(日月年)格式；

(10)磁偏角(000.0~180.0 度，前面的 0 也将被传输)；

(11)磁偏角方向，E(东)或 W(西)。

2 目标平台

本文中使用的是以 SAMSUNG 公司的 ARM9 系列中的 16/32 位 RISC 处理器 S3C2410A 芯片为核心的目标平台。S3C2410A 包含一个 16/32 位的 RISC(ARM920T)CPU 内核、独立的 16 KB 的指令和 16 KB 数据缓存(cache)、用于虚拟内存管理的 MMU 单元、LCD 控制器(STN&TFT)、非线性(NAND)Flash、系统管理器(包括片选逻辑控制和 SDRAM 控制器)及 3 个通道的异步串口(UART)。目标板资源包括 S3C2410 的微处理器，主频 200 MHz；16 MB 的 Flash；64 MB 的 SDRAM；RS-232C UART 接口；LCD 液晶显示屏。

在目标板上选配的 GPS 模块是 HIMARK 公司的 GPS 模块，此模块是符合民用标准的 GPS 接收器，信号接收性能好，功耗较小，整体工作比较稳定。整体硬件设计框图如图 2 所示。

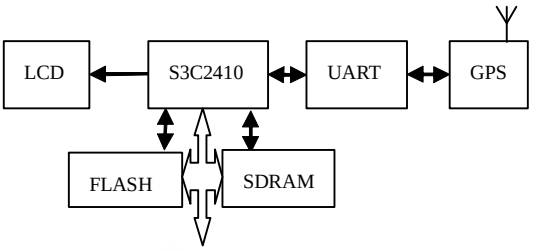

图 2 目标平台

3 交叉编译环境的建立

基于嵌入式 Linux 操作系统的应用开发模式通常都是宿主机+目标机[2]。目标机用于运行操作系统和系统应用软件，而目标板所用到的操作系统的内核编译、应用程序的开发和调试则需要通过宿主机(资源丰富处理能力强的 PC)来完成，称之为交叉编译。双方之间一般通过串口、并口或以太网接口建立连接关系。

(1)配置 minicom：在宿主机 Redhat Linux 9.0 的 X windows 界面下新建终端，在终端命令提示符后输入 minicom-s，回车，然后按照提示设置波特率 115200，8 位数据，1 位停止位，无流控，保存退出。

(2)TFTP 服务的配置：在终端中运行 setup->system service->ftp 增加 TFTP 服务，并去掉 ipchains 和 iptables 两项，然后在 Firewall configuration，选中 no firewall，保存退出，执行 service xinetd restart 启动 TFTP 服务。

(3)NFS 服务器的配置：在终端中运行 setup->system service->NFS，增加 NFS 服务，然后编辑文件 exports，添加与目标机共享的目录，并设置目标机对目录的访问权限，重新启动 NFS 服务。

(4)Linux 内核移植：通过并口，宿主机向目标开发板的 Flash 烧写引导程序 ppcboot，烧写完毕后通过 TFTP 服务把经过裁剪的 Linux 内核镜像文件以及根文件系统下载到目标板的 RAM 中，然后由 ppcboot 完成内核及根文件系统从内存到 Flash 的烧写。最后需要在宿主机安装主编译器 Arm-linux-gcc，用来交叉编译应用程序。

4 GPS 信号的采集和处理

为实现 ARM-Linux 平台下 GPS 信号的采集与处理，涉及到 Linux 下串口编程技术，首先给出 Linux 串口通信的原理，然后利用多线程编程技术来完成 GPS 数据采集与 NMEA 数据格式的解析，因解析后得到的 GPS 定位坐标属于 WGS84 坐标，需转换到相应的 54、80 坐标或地方坐标供用户标图定位所用，因此介绍了一种坐标转换方法。

4.1 GPS 数据采集与处理

大多数 GPS 接收机与各种处理器平台进行数据交换时，都采用异步串行传送方式，提供一个符合 RS-232C 电气标准的数据接口。

在 Linux 操作系统中，所有设备以设备文件的形式存储在目录/dev/下，串口设备文件为/dev/ttyS*，在 Linux

中，若要设置串口的参数，如改变串口的波特率、字符大小等，可通过 POSIX 标准终端接口[3]，该接口被称为 termios，在系统头文件<termios.h>中定义。它包括一个数据结构和一系列操纵这些数据结构的函数组成。有关串口的所有参数配置都保存在接口 termios 的结构 struct termios 中，该结构定义如下：

```
struct  termios
{
tcflag_t  c_iflag;  /* 输入模式标志 */
tcflag_t  c_oflag;  /* 输出模式标志 */
tcflag_t  c_cflag;  /* 控制模式标志 */
tcflag_t  c_lflag;  /* 本地模式标志 */
cc_t      c_cc[NCCS];/* 特殊控制字符 */
}
```

其中的 c_iflag 成员是用来控制输入处理选项的，它影响到终端驱动程序将输入发送给程序前是否对其进行处理，及怎样对其进行处理。c_oflag 成员是用来控制输出数据的处理，并决定在发送输出数据到显示屏和其他输出设备之前，终端驱动程序是否以及如何来处理它们。c_cflag 用于存放各种决定终端设备硬件特性的控制标志，如串口的波特率、奇偶校验、停止位、数据位等。存放在 c_lflag 中的本地模式标志用来操纵串口如何处理输入字符，比如是否将输入字符显示到显示屏上，一般可通过此成员来设置串口为正规模式或是非正规模式。c_cc 数组成员用来定义支持的特殊控制字符以及一些 timeout 参数。

除了上面的这个包含串口参数配置的数据结构之外，termios 中还包含许多控制串口特性的函数。其中重要的几个函数如：tegetattr( )、tesetattr( )、cfsetispeed( )、cfsetospeed( )、tcflush( )。tegetattr( )用来初始化一个 termios 数据结构，之后可使用其他的函数来操纵由 tegetattr ( )返回的数据结构。完成这些操作后，使用 tesetattr( )来更新串口的设置。cfsetispeed ( ) 用来设置串口的输入速度。cfsetospeed( )用来设置串口的输出速度。tcflush( )用来清除所有队列在串口的输入与输出。

在 Linux 下采用多线程编程技术可大大节省系统的开销，方便各线程之间通信，提高应用程序的响应，改善程序结构，从而可以提高嵌入式系统的性能。本文就是利用 Linux 下多线程编程来实现 GPS 数据的采集和处理，在 GPS 模块的初始化 GPS_Initial 函数中创建接收线程 GPS_Thread_Port_Svc，在接收线程中调用 GPS 信息语句的解析函数 GPS_Parse_Data(buf_GPS，&gps_data)，进一步调用语句字符串解析函数 GPS_Parse_Data_Line (char*str_gprs_data_line，GPS_DATA_TYPE*GPS_DATA)。在 GPS 语句的处理过程中，需对所读取的语句进行鉴别区分，只选取其中要用的信息进行处理而忽略其余的信息，这就要根据 NMEA-0183 协议中规定的语句格式来进行解析。图 3 给出了 GPS 数据处理流程。

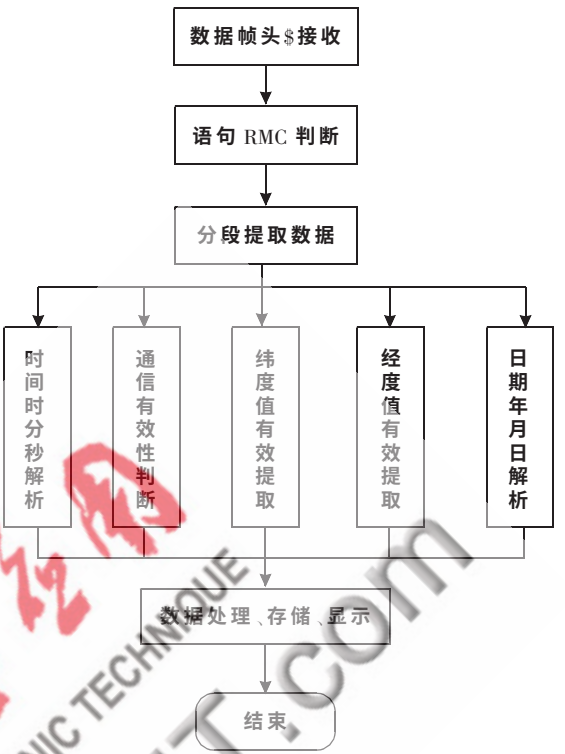

图 3 数据处理流程

下面是程序实现的关键函数部分代码。

```
/* 包含必要的头文件 */
#include <termios.h>
#include <stdio.h>
#include <fcntl.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
int  GPS_Initial(int  n_tty_no)
{
int  ret_tty=-1;
int  ret_thread=-1;
ret_tty=OpenComPort(n_tty_no,9600,8,"1",'N');/*
打开串口，并设置通信属性，如波特率，数据位，有无奇偶校验，停止位等 */
ret_thread=pthread_create(&pthr_id,NULL,GPS_
Thread_Port_Svc,NULL);/* 创建接收线程 */
…
}
void*GPS_Thread_Port_Svc(void*pData)
                                    {/* 接收线程函数 */
unsigned  char  buf_GPS[256];
int  ret_rd_com;
while(1)  {
```

```
    bzero(buf_GPS, sizeof(buf_GPS));
    ret_rd_com=ReadComPort((void*)buf_GPS,
    sizeof(buf_GPS));              /* 读串口,接收数据 */
    GPS_Parse_Data(buf_GPS,&gps_data);
                                   /* 调用解析语句函数,*/
    …}
```

在 GPS_Parse_Data (buf_GPS,&gps_data) 函数中每接收一个 GPS 语句,调用一次字符串解析函数。

```
GPS_Parse_Data_Line (char*str_gprs_data_line, GPS_DATA_TYPE*GPS_DATA){
    char*temptr;
    temptr=str_gprs_data_line;
            /* 字符串赋给临时指针,然后对其解析 */
    if (strncmp (temptr, RMC_DATA_L, strlen (RMC_DATA_L))==0)
                                   {/* 解析 RMC 语句串 */
        startchar(temptr, ',');
    temptr=temptr+strlen(temptr)+12;
                                   /*$GPRMC, HHMMSS.SSS*/
    GPS_DATA->Time.Flag=temptr[0];
    temptr=temptr+27;
        /*A, DDMM.MMMM, N, DDDMM.MMMM, E*/
    …}
```

以上只是 GPS 信息处理的部分代码。经过交叉编译调试下载至目标平台上,运行后可得到本地地理位置信息。实验所得数据为:时间 10:28:35;纬度:北纬 30°46';经度:东经 103°57'。用户也可以根据需要选择提取 GPS 的其他语句,只需编写解析相应语句字符串的代码即可。

4.2 GPS 坐标的转换

上述所得的结果属于 WGS84 坐标,而在工程上实用的大多是国家坐标系,因此 GPS 数据采集结果的使用就存在与国家坐标系间的坐标转换问题。一般要通过两步转换:首先将上述实测经纬度坐标即 WGS84 大地坐标 (L,B) 转换为对应于 WGS84 椭球的高斯平面坐标 (X_{84},Y_{84}),然后再经过平面坐标转换,将高斯平面坐标 (X_{84},Y_{84}) 强制附合到本地高斯平面坐标系统[4]。

(1)高斯投影换算

将 GPS 采集所得出的大地坐标 (L,B) 转换为高斯平面坐标 (X_{84},Y_{84})。有关的推导过程较复杂,本文只给出结果:

$$X_{84}=X+\frac{l^2}{2}N\sin B\cos B+\frac{l^2}{24}N\sin B\cos^5 B(5-t^2+9\eta^2+4\eta^4)+\frac{l^6}{720}N\sin B\cos^5 B(61-58t^2+t^4)$$

$$Y_{84}=lN\cos B+\frac{l^3}{6}N\cos B(1-t^2+\eta^2)+\frac{l^5}{120}N\cos^5 B(5-18t^2+t^4+14\eta^2-58\eta^2t^2)+500\,000$$

式中,$l=(L-l_0)/\rho$,l_0 为投影带中央子午线经度,$\rho=206265$ s/rad;卯酉圈曲率半径 $N=\alpha/\sqrt{1-e^2\sin^2 B}$;椭球第一偏心率 $e=2\alpha-\alpha^2$;辅助变量 $t=\mathrm{tg}B$;辅助变量 $\eta=e'\cos B$;椭球第二偏心率 $e'=\sqrt{a^2/b^2-1}$ (a,b 分别为参考椭球的长短半径);扁率 $\alpha=(a-b)/a$;X 为赤道至纬度为 B 的平行圈的子午线弧长,其计算公式为

$$X=c\int_0^B[1+(e')^2\cos^2 B]^{-\frac{3}{2}}\mathrm{d}B$$

由上述原理利用 EXCEL 就可以算出对应的高斯平面坐标。

(2)平面坐标转换

平面坐标转换的目的就是将高斯投影换算得出平面坐标 (X_{84},Y_{84}) 转换为当地国家坐标系的平面坐标。下面介绍一种平均转轴相似转换法,以转换为北京 54 坐标系下的平面坐标 (X_{54},Y_{54}) 为例,说明该方法实现过程。

首先,根据公共点分别在 WGS84 和北京 54 系中的高斯平面坐标,求出该点在两个坐标系中同一边的方位角之差 $\Delta\alpha$ 和长度比例系数 κ,然后按下式计算任一点在北京 54 系中的平面坐标。

$$X_{54}=x_0+\kappa\cos\Delta\alpha(x_{84}-x_0')-\kappa\sin\Delta\alpha(y_{84}-y_0')$$

$$Y_{54}=y_0+\kappa\cos\Delta\alpha(x_{84}-y_0')-\kappa\sin\Delta\alpha(x_{84}-x_0')$$

式中,x_0,y_0 为公共点在北京 54 中的重心坐标;x_0',y_0' 为公共点在 WGS84 中的重心坐标;κ 为同一边在北京 54 和 WGS84 中的边长之比,当有两条以上公共边时,分别求出,取平均值;$\Delta\alpha$ 为同一边在北京 54 与 WGS84 中的方位角之差,$\Delta\alpha=\alpha_{54}-\alpha_{84}$,当有两个以上公共点时,分别求出,取平均值。

将得到的 X_{54},Y_{54} 坐标可应用于 GIS 系统标图,实现导航定位。

GPS 信息的获取是设计导航定位系统的首要环节,本文有针对性的研究了 ARM-Linux 平台与 GPS 接收模块之间的串行通信。基于 NMEA-0183 通信协议,在 Linux 下通过多线程编程实现了 GPS 基本定位信息的采集与处理,所得数据满足精度要求,为导航定位系统或 GIS 系统提供了数据基础。

参考文献

[1] 陈石磊.FPGA 与 GPS-OEM 板的串行通讯系统设计[J].电子元器件应用,2008,10(5):12-13.

[2] 蓝炳伟,王涛.构建嵌入式 Linux 应用开发环境[J].网络安全技术与应用,2008(2):57-58.

[3] 马文辉,李兰友.Linux 环境下的串口通信[J].仪器仪表用户,2005(12):32-34.

[4] 杨贵军,武文波.PDA 环境下 GPS 信号的接收和处理方法[J].单片机与嵌入式系统应用,2005(9):25-26.

(收稿日期:2009-03-12)