Comp 3980 Computer Systems Technology - October 2013

Data Communications and Internetworking Option

Assignment #3

Works in groups of two, chosen from within your own teams.

Objective:

To gain experience with principles of GPS operation, and the design and implementation of GPS applications on Embedded platforms such as Raspberry Pi.

Background:

Location finding applications are beginning to proliferate into every aspect of mobile computing, business, social media, and all manner of handheld devices including smart phones.

An added component in this project will be the implementation of a GPS location finding application on a small system such as the Raspberry Pi platform. As such this will be a Linux-based implementation using the Open Source GPSD utility.

This project will be expanded next term to include the sending of GPS data to remote web server with a mapping application.

Your Mission:

The theoretical principles of RF and GPS have been covered in class. In this project you will design and develop a minimal location finding application using C or C++. The ***gpsd*** daemon and its associated API library will be used implement your design. A GPS dongle will be provided to each team, together with a Raspberry Pi unit.

The first step will be to get the Pi configured and operational. You will be required to purchase a Class 10 SD card and install a distribution of your choice on it. A WiFi adapter will also be provided to you.

Test your complete Pi and GPS setup with an application such as ***cgps*** or ***xgps***. Once the testbed has been verified as functional you will begin your development work on it. Note that the coding part can be done on any Linux machine; you can transfer your code to the Pi later and compile and test it there.

Constraints

- This is to be a Linux-based implementation using C or C++.
- The software implementation must follow the following **minimum** architectural model:

  dcgps.c

  - This file will contain the main function only. You will malloc the gps_data_t structure in this function, open the stream to gpsd, and start the stream.
  - In addition this function will also perform any cleanup or free operations when the application terminates.

  gps-utils.c

  - This file will contain at least two functions. The largest function will be the main GPS loop which will continuously read sensor data and send it to another function (described below) to be printed out.
  - Another function that could go in here is a function that performs error checking and reports abnormal conditions such as the sensor returning errors or termination of the stream.

  gpsprint.c

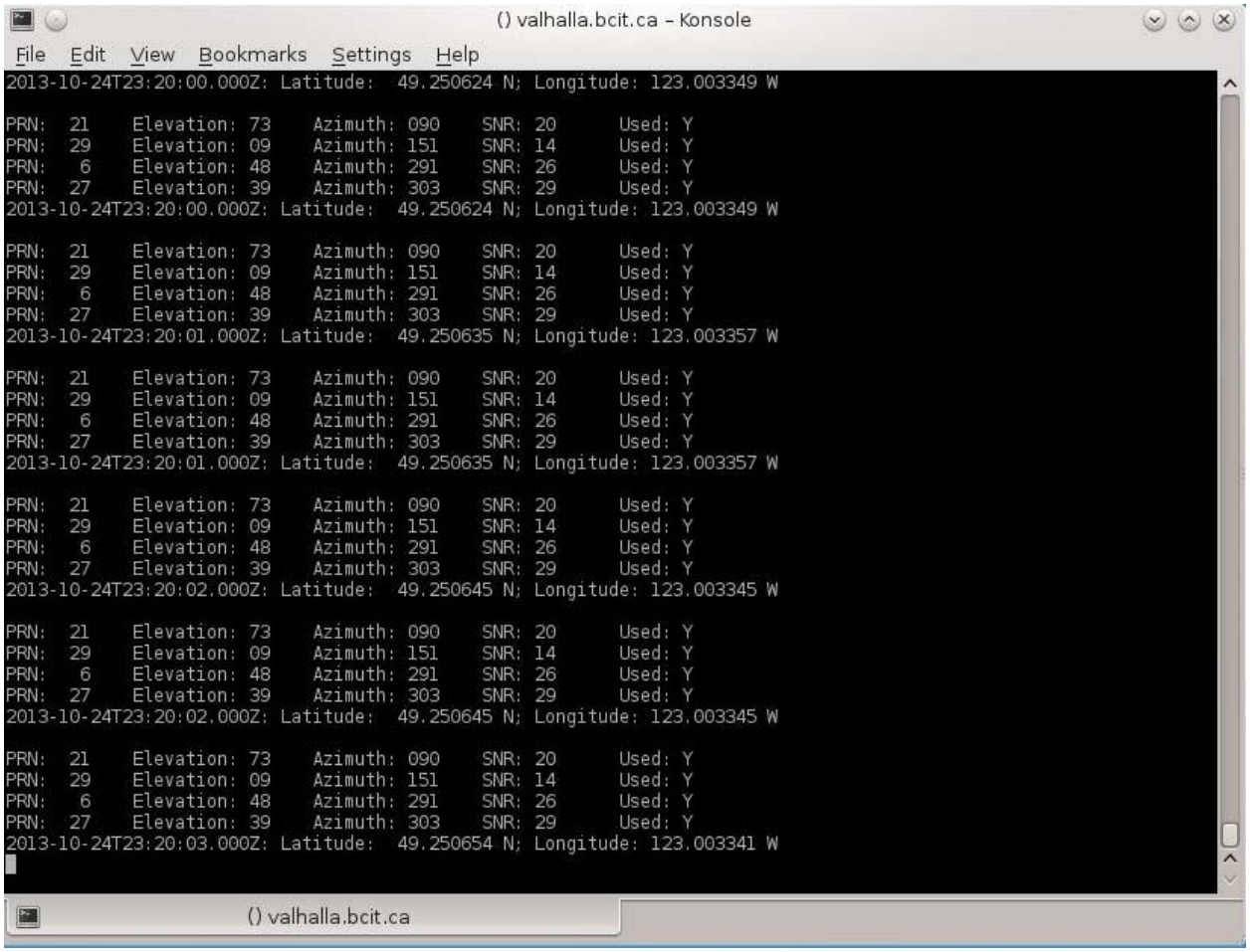
  - This file will primarily consist of one complete function that will handle all of the printing out of GPS report data. One of its arguments will be a pointer to a **gps_data_t** structure, which will be passed to it from the read loop described above in **gps-utils.c**.

- The following is a minimum list of report data parameters that your application will print out to the screen:

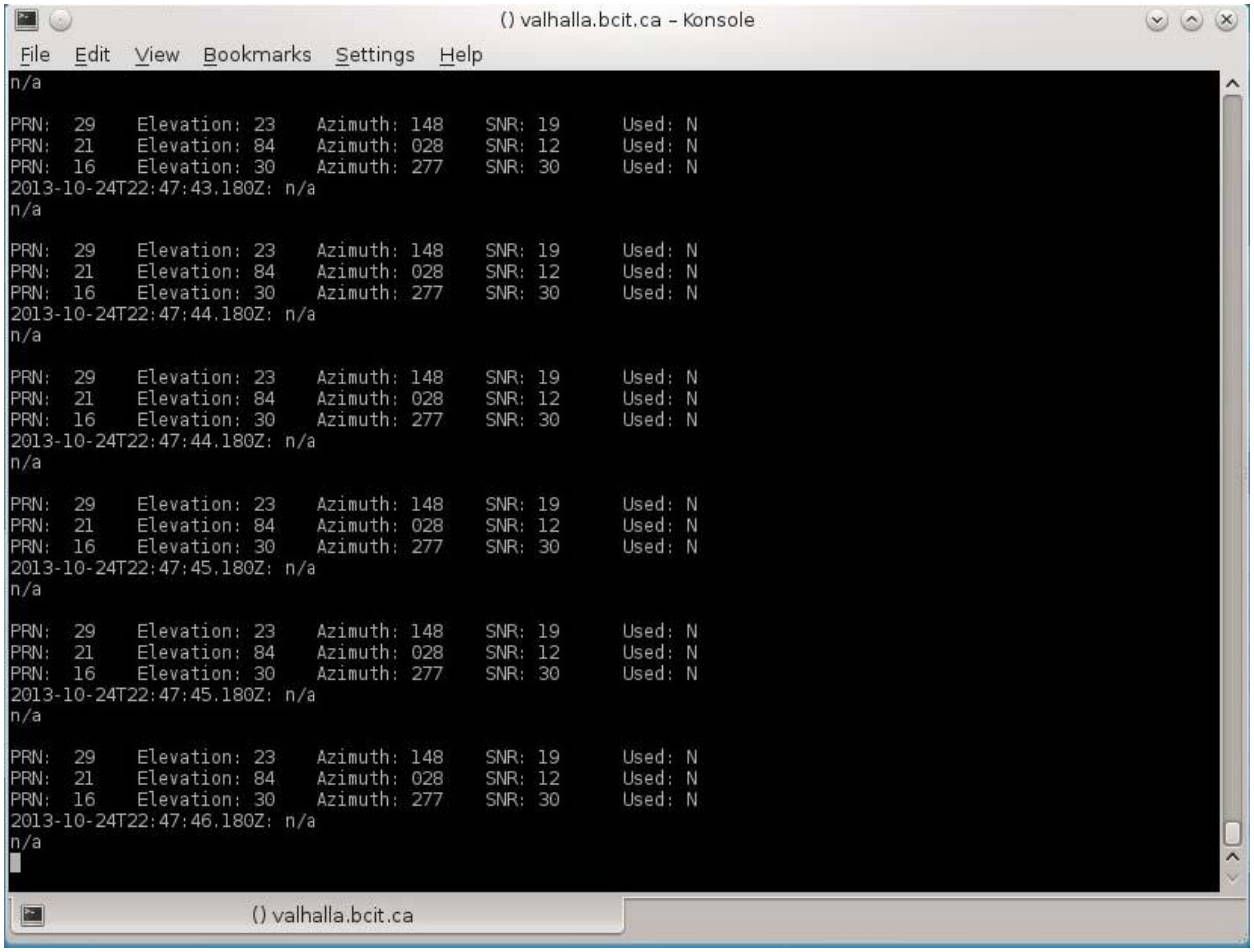
  - Time stamp (UTC)
  - Latitude/Longitude
  - PRN
  - Elevation
  - Azimuth
  - SNR
  - Used flag (Y or N)

- Note that you are not required to provide a graphical display of satellites and parameters. This is to be a basic console application. Although, as usual, bonus marks will be awarded for extra work including user-level control functions, graphical display, etc.

- The following is an screenshot of an application that meets the minimum requirements (it shows a fix obtained using 4 satellites) in a basic console:



- The next screenshot shows an example of three visible satellites but no fix due to poor signal conditions:



- You are required to provide detailed and clear headers in your code. The way this project is laid out it can be divided up into three main sections. Aside from the main file, which is minimal, the other two files can be assigned to individual team members.
- Not to put too fine a point on this, I will be looking for individual effort on this project.
- You are required to submit a detailed test document showing all of the tests you performed to validate and test the functionality of your application. This will include screenshots and explanations.
- In addition you will also provide a small user guide.
- You are required to demonstrate the functionality of your application on the due date.

Marking Guide

Design Work :	15
Code Quality:	10
Functionality:	60
Testing & Documentation:	15
Total:	100

To Be Submitted (on disk): (Due Date: November 13, 2013 - 0930 hrs.)

1. State transition diagrams and pseudocode (in PDF format) for the application.

2. All of the C/C++ source code, Makefiles, executable, and any other material that will be required to generate an executable version of your program.

3. You are required to demonstrate your application in the lab on the day the assignment is due.