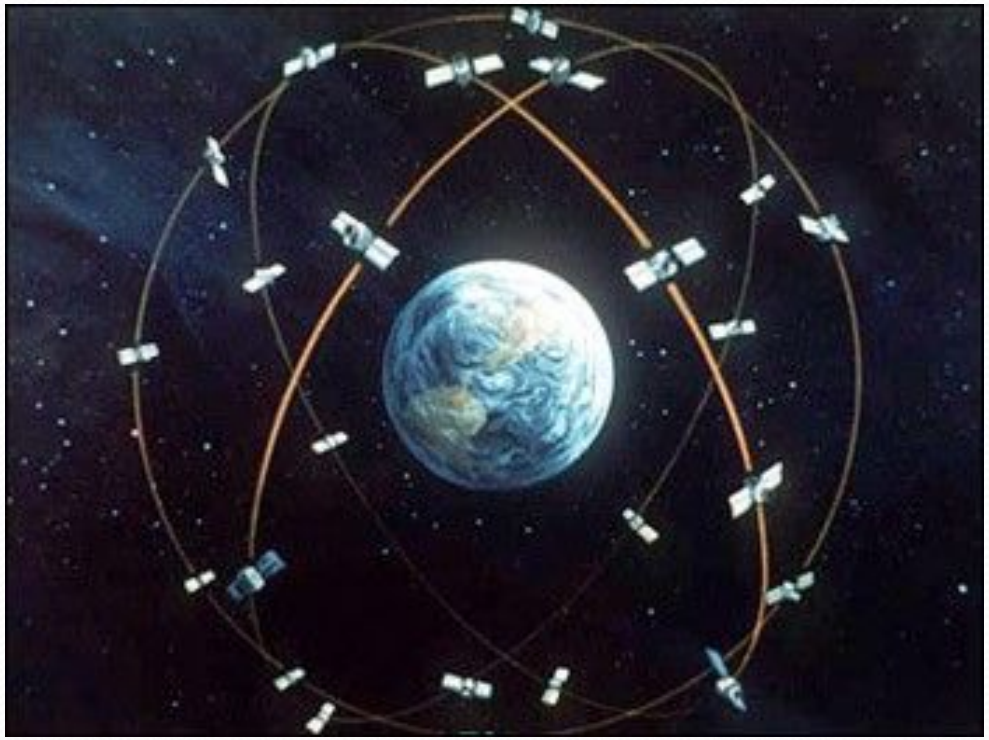
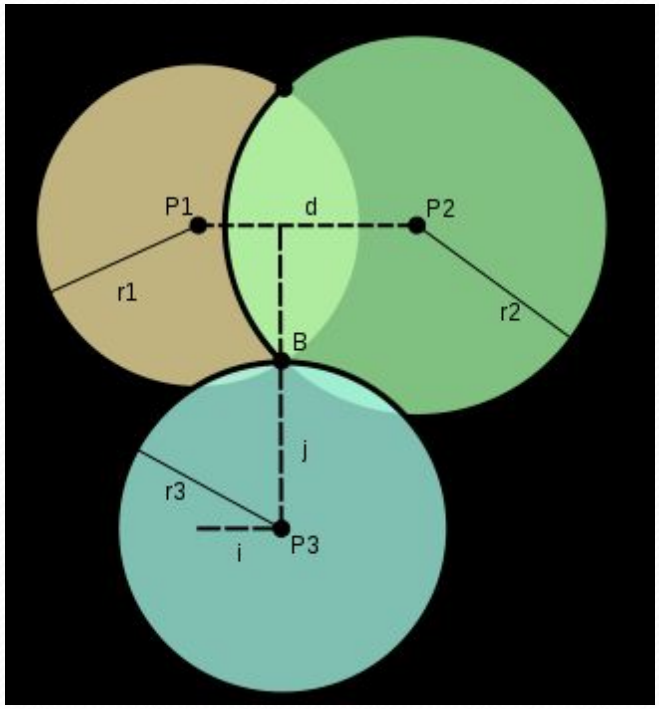
Embedded System
&
Embedded Linux Development
Part 4

Index of today's topic

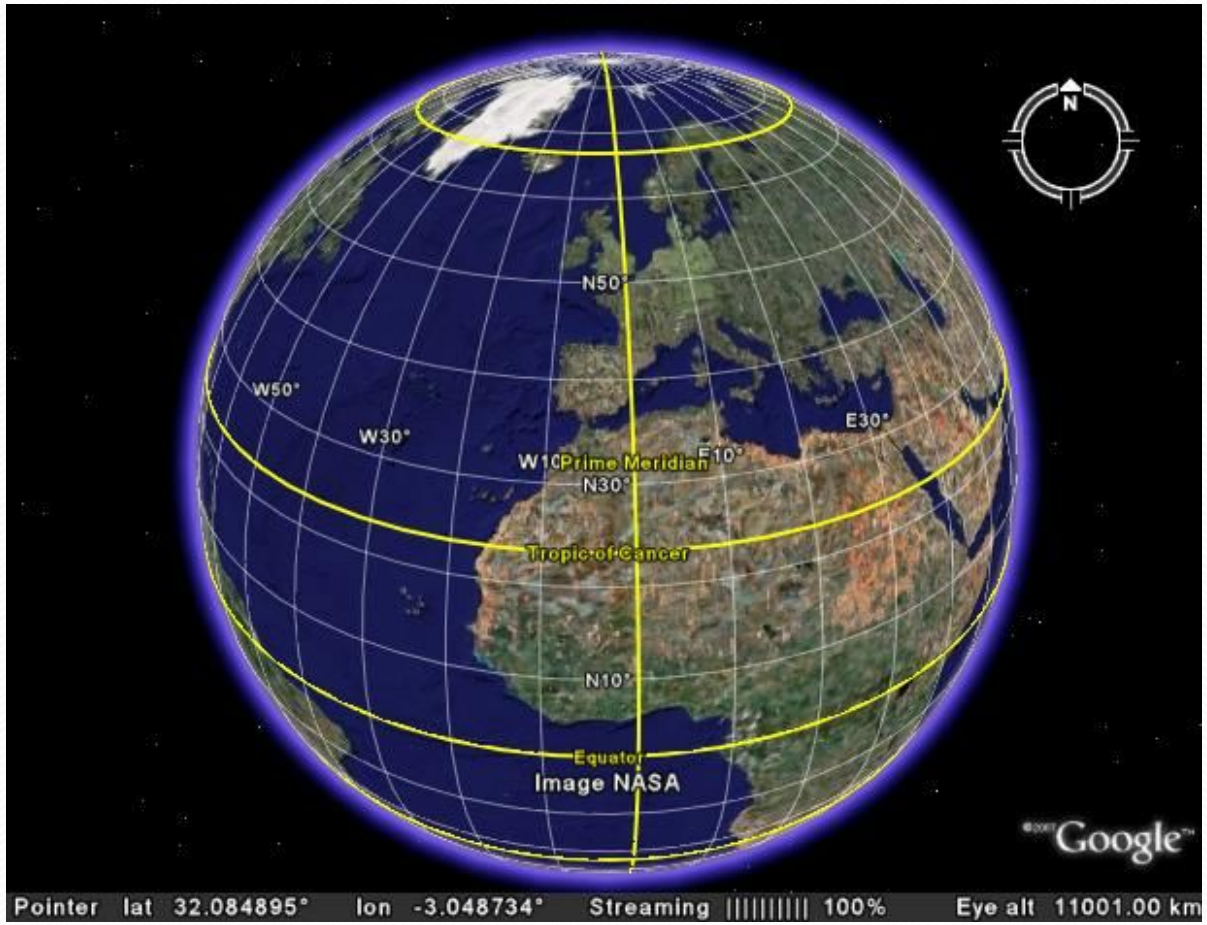
- GPS hardware connection
- GPS data format
- Get current location from GPS
- Save it to KML & export to Google Earth

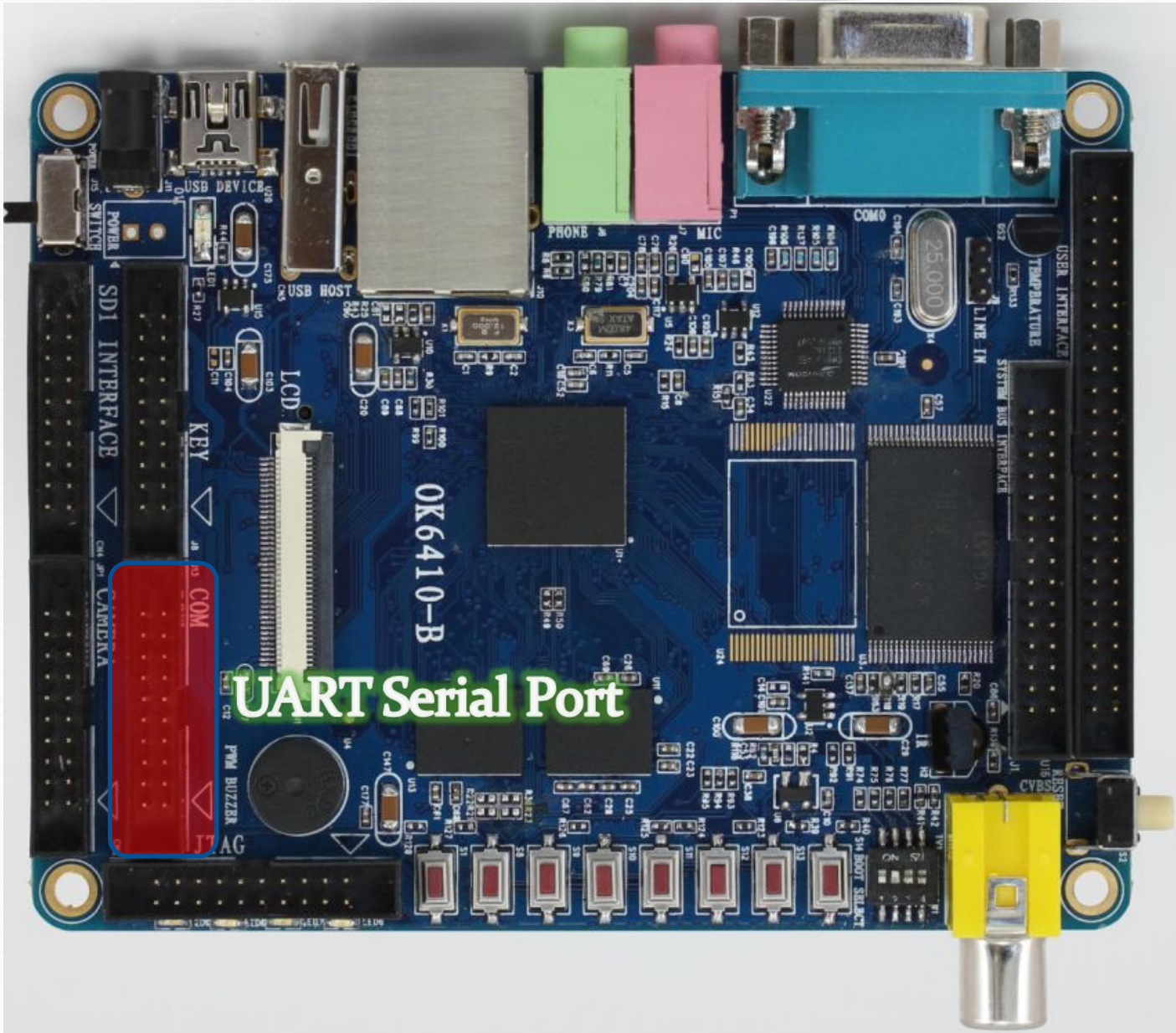
What's GPS?



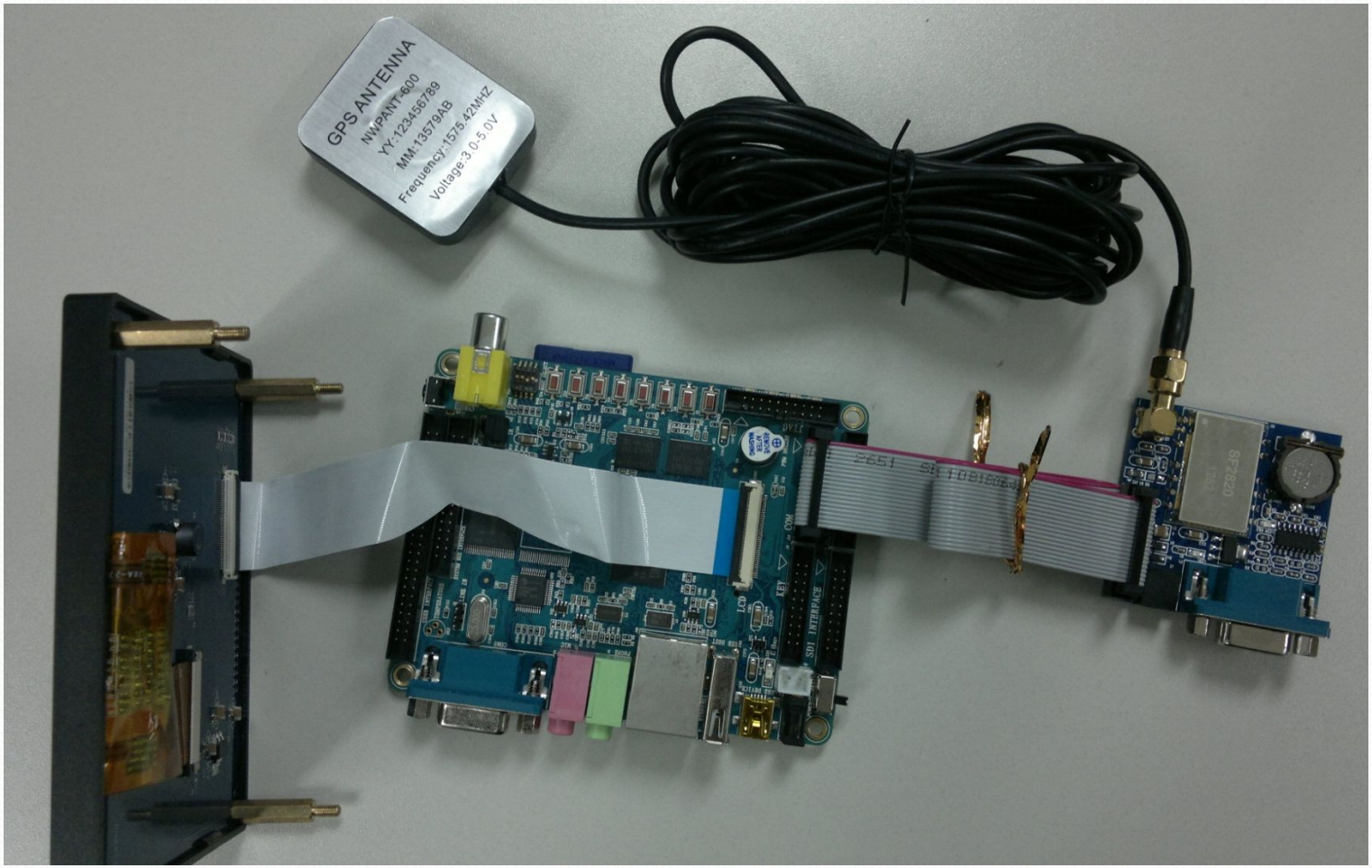


Latitude, Longitude, Altitude



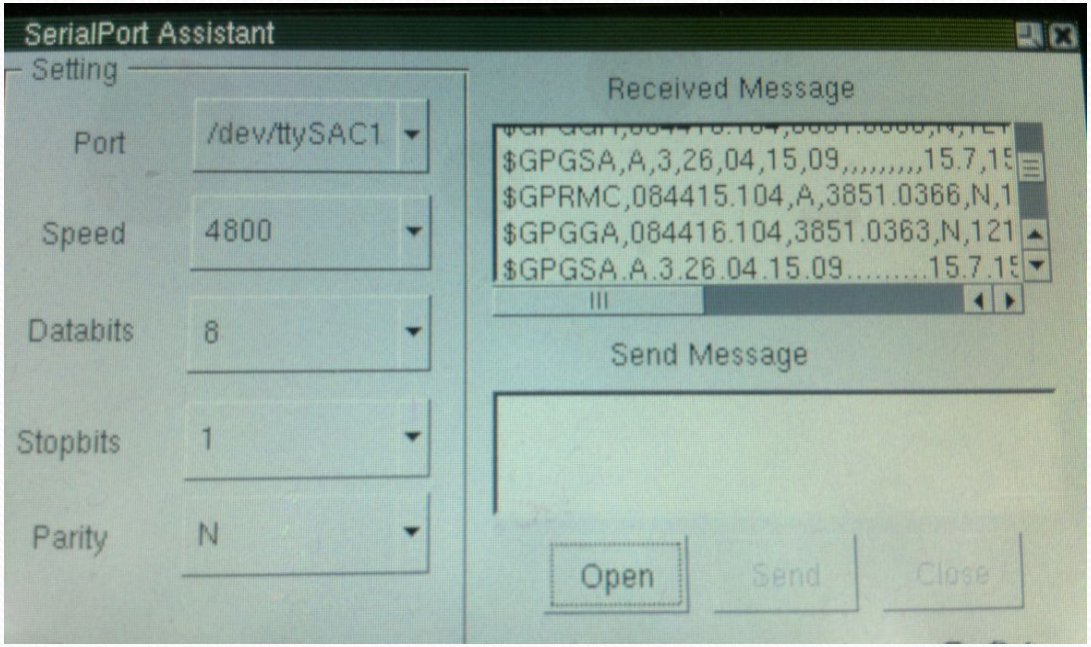

USB DEVICE
USB HOST
PHONE
MIC
COM0
25.000
USER INTERFACE
TEMPERATURE
LINE IN
SYSTEM BUS INTERFACE
SD1 INTERFACE
CAMERA
KEY
LCD
OK6410-B
COM
PWM BUZZER
JTAG
IR
CVBS
UART Serial Port

Connect GPS module to FL6410



Test GPS

- The bit of serial data of GPS module is 4800 bps



- A group of GPS data will be printed like this:
  - **$GPGGA**,082006.000,3852.9276,N,11527.4283,E,1,08,1.0,20.6,M,,,,0000*35
  - **$GPRMC**,082006.000,A,3852.9276,N,11527.4283,E,0.00,0.0,26109,,*38
  - **$GPVTG** ,0.0,T,,M,0.00,N,0.0,K*50

GPS data format

- GPS module output followings NMEA format
  - http://www.gpsinformation.org/dale/nmea.htm
- The output of GPS is always a line of data start with "$GP???"
- You can read data line by line

RMC (Recommended Minimum)

- **$GPRMC,123519,A,3851.0127,N,12129.8392,E,022.4,084.4,221013,003.1,W,*6A**
  - **123519 format: hhmmss.ssss UTC**
  - A Status A=active or V=Void.
  - **3851.0127,N format: ddmm.mmmm N(latitude North/South)**
  - **12129.8392,E format: ddmm.mmmm E(longitude East/West)**
  - 022.4 Speed over the ground in knots
  - 084.4 Track angle in degrees True
  - **221013 Date - 22rd of October 2013**
  - 003.1,W Magnetic Variation
  - *6A The checksum data, always begins with *

Read data from GPS

- Open Serial Port
  - use **open()** function
- Read data to a buffer
  - use **read()** function
- Parse the GPS data
  - find the "**$GPRMC**" datda
    - use **strstr()** to find a string in a string
  - Parse a complex string to several variables
    - use **sscanf()** to do the parsing

Exercise 1

Get the latitude, longitude, ground speed. Print them out like this:

latitude=**North 38.850211'**

longitude=**East 121.497320'**

ground speed=**41.4 kph(Kilometers Per Hour)**

Hint: 1 knot = 1.85200 kph

Exercise 1

```
$GPGSV,3,2,12,28,31,182,39,02,19,264,17,23,19,101,45,09,13,132,40*79
$GPGSV,3,3,12,12,13,322,,01,10,067,21,32,01,036,,24,00,291,*73
$GPRMC,012434.000,A,3851.0242,N,12129.8749,E,0.14,327.99,271114,,,A*6D
$GPGGA,012435.000,3851.0242,N,12129.8748,E,1,06,2.0,133.2,M,4.4,M,,0000*6B
$GPGSA,A,3,23,09,28,10,17,06,,,,,,,3.9,2.0,3.3*39
$GPRMC,012435.000,A,3851.0242,N,12129.8748,E,0.13,323.40,271114,,,A*6A
$GPGGA,012436.000,3851.0242,N,12129.8748,E,1,06,2.0,133.1,M,4.4,M,,0000*6B
$GPGSA,A,3,23,09,28,10,17,06,,,,,,,3.9,2.0,3.3*39
$GPRMC,012436.000,A,3851.0242,N,12129.8748,E,0.14,331.27,271114,,,A*6C
$GPGGA,012437.000,3851.0242,N,12129.8747,E,1,06,2.0,133.0,M,4.4,M,,0000*64
$GPGSA,A,3,23,09,28,10,17,06,,,,,,,3.9,2.0,3.3*39
$GPRMC,012437.000,A,3851.0242,N,12129.8747,E,0.10,327.37,271114,,,A*60
$GPGGA,012438.000,3851.0242,N,12129.8747,E,1,06,2.0,132.9,M,4.4,M,,0000*63
```

Exercise 1

```
latitude=North 38.51.015800'
longitude=West 121.29.887699'
ground speed = 0.314840 kph
latitude=South 0.0.000000'
longitude=West 0.0.000000'
ground speed = 0.000000 kph
latitude=South 0.0.000000'
longitude=West 0.0.000000'
ground speed = 0.000000 kph
latitude=North 38.51.015701'
longitude=West 121.29.887800'
ground speed = 0.277800 kph
```

Exercise 2

Time synchronizing

1. Get the GPS time from GPS input data
2. Set GPS time to Linux system.

- tips: **mktime(), settimeofday(),localtime()**.
- **mktime()**:converts a broken-down time structure, expressed as local time, to calendar time representation.
- **settimeofday()**:can set the time as well as a timezone.
- **localtime()**:take an argument of data type time_t which represents calendar time.When interpreted as an absolute time value, it represents the number of seconds elapsed since the Epoch, 1970-01-01 00:00:00 +0000 (UTC).

Exercise 2

```
[root@FORLINX6410]# ./gpsDemo3
Current DateTime : Thu Nov 27 01:27:07 2014

[root@FORLINX6410]#
```

Exercise 3

- Read data from GPS, if current position is available, save it to a KML file. put the current time in the description.
  - Hint: use fopen(), fwrite() to write a file in filesystem.
- Upload this file to PC and open it with Google map or Google Earth. See what you get.

Keyhole Markup Language (KML)

- KML is an XML schema for expressing geographic annotation and visualization within Internet-based, two-dimensional maps and three-dimensional Earth browsers. KML was developed for use with Google Earth, which was originally named Keyhole Earth Viewer.

KML Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
  <name>Place Name</name>
  <description>Place Description</description>
  <Point>

  <coordinates>121.497320,38.850211,0</coordinat
  es>
  </Point>
</Placemark>
</Document>
</kml>
```

Usage

- KML file can be opened by Google Earth or Google Map

- Embedded System can record its location or track into a KML file and transfer it to PC

KML Sample

```
[root@FORLINX6410]# ./gpsDemo4
[root@FORLINX6410]# ls
gpsDemo1    gpsDemo2    gpsDemo3    gpsDemo4    mydata.kml
[root@FORLINX6410]#
```

KML Sample