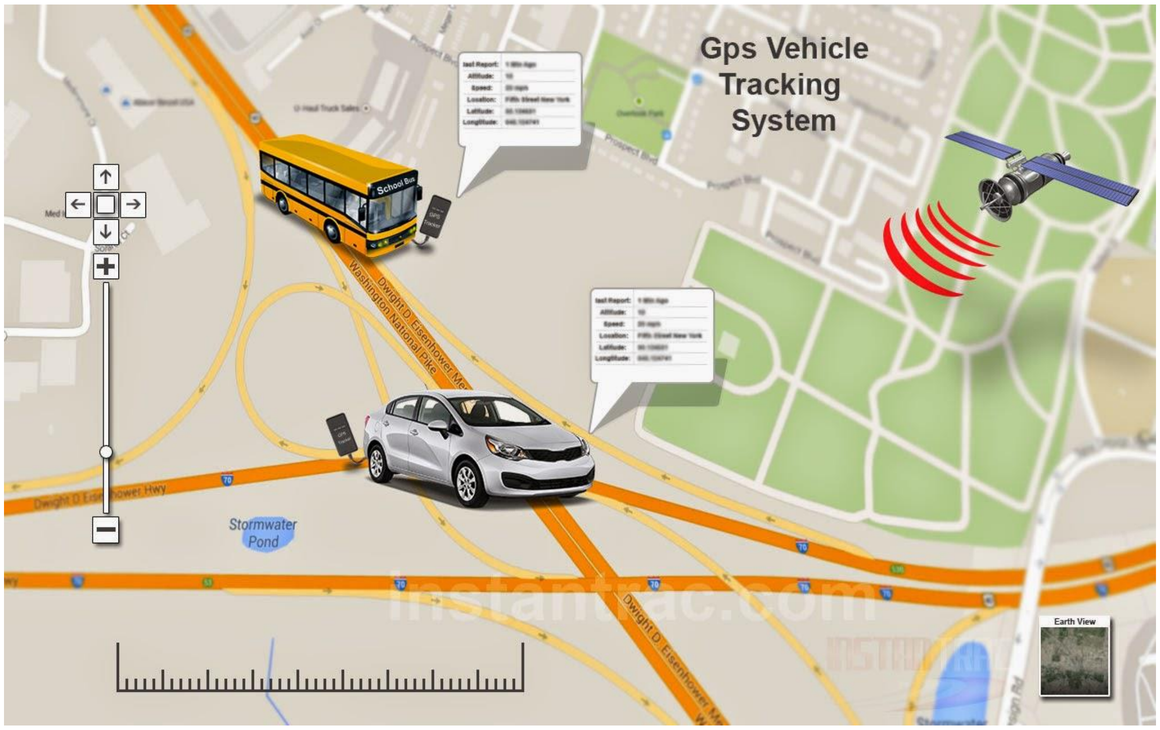
# GPS Vehicle Tracking System

(GPS MONITORING WITH SERVER CONTROL)



MIKAIL BISWAS MRIDU | ROLL NO: 1507081
MD. RIFAT ANWAR | ROLL NO: 1507066
MD. RUBEL HASAN | ROLL NO: 1507074

SEC: B

CSE3104: PERIPHERALS AND INTERFACING LABORATORY | 01 JULY, 2018

## Project Overview

There are various Advanced Arduino Projects, which solves the problems in our life. This is one of those Arduino projects. The advanced vehicle tracking system is an enhanced system that allows a user to track the vehicle using GPS along with GSM modem. Using this vehicle tracking system user gets the location details of the vehicle where it is currently on his mobile and it can track it on Google map. For this purpose, we are using Arduino Uno R3 as the main processing unit. The whole system is controlled by Arduino Uno R3. This Arduino UNO R3 is interfaced to GSM/GPRS/GPS Shield (B). Once you start system it starts sending location details. The user can track it on Google map using specific webpage containing details of GPS location. Once the user goes to that webpage, he/she can see the location on Google Map with a marker. Nowadays every smartphone has Google Maps application preinstalled. So this webpage opens in the Google Maps app. GPS tracker system constantly keeps on sending GPS data of location details of where the vehicle is located. Using the map we can see the places around the vehicle also we can see the road on the map. So by this way the user can get the location of the vehicle and real-time vehicle location details.

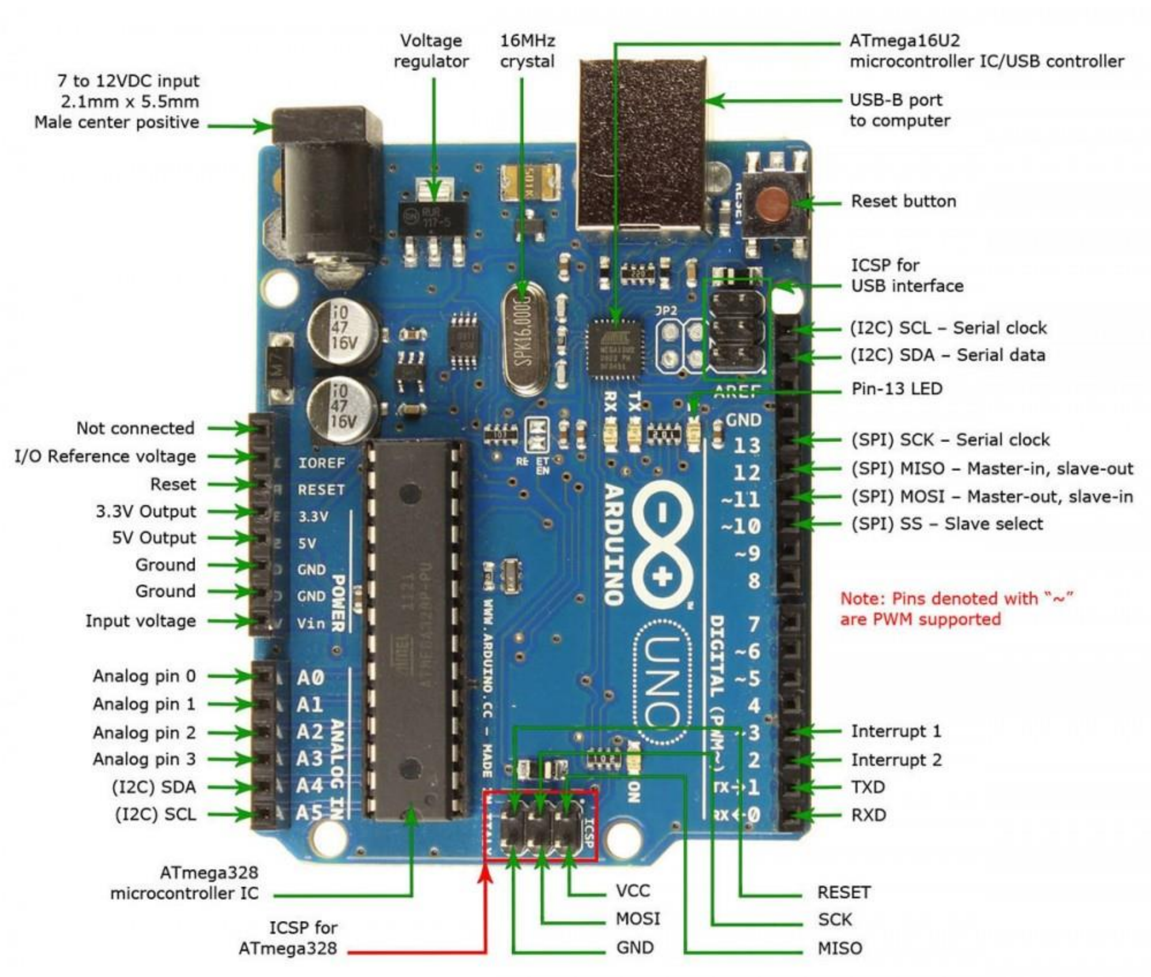
Real time vehicle tracking system using Arduino, GSM and GPS is an innovative and user-friendly system. Sometimes company's transportation vehicles consume more and more fuel which results in loss of money. The solution for this is to install GPS tracking device in the vehicle. It sends real-time updates of the vehicle coordinates. It also improves safety and security of our car. We can also see the history specific webpage. By using tracker system can see what time car was at which place.

## Total equipment for Project

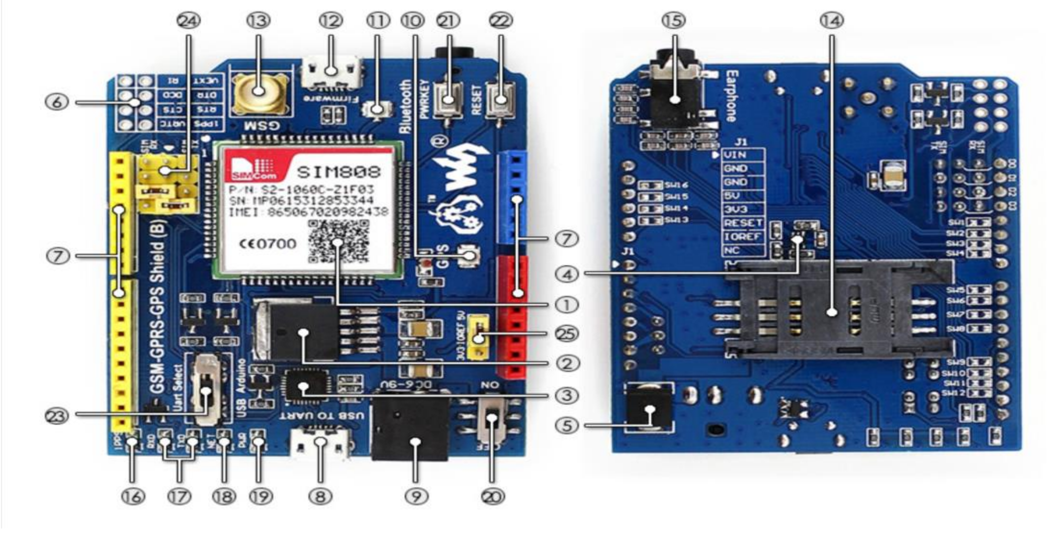
1. Arduino Uno R3
2. GSM/GPRS/GPS Shield (B) SIM808 Module
3. GPS Antenna
4. USB cables

## Circuit Diagram

1. Arduino UNO R3:



2. GSM/GPRS/GPS Shield (B) SIM808 Module:



1. **SIM808 module**
2. **MIC29302 power chip**
3. **CP2102:** USB TO UART converter
4. **SMF05C:** TVS diode
5. **1N5408:** onboard rectifier
6. **SIM808 functional pins**
7. **Arduino expansion connector**
8. **USB TO UART interface**
9. **DC power jack**
10. **GPS antenna connector**
11. **Bluetooth antenna connector**
12. **Firmware upgrade interface**
13. **GSM antenna connector**
14. **SIM card slot**
15. **3.5mm earphone/mic jack**
16. **GPS status indicator**
17. **CP2102 UART Tx/Rx indicator**
18. **NET indicator:**
    - flashes fast when the module starts up
    - flashes slowly after GSM register succeed
19. **Power indicator**
20. **Power switch**
21. **SIM808 control button:** press the button and hold for 1s, to startup/shutdown the SIM808
22. **Reset button**
23. **UART selection switch,** select controlling the SIM808 via:
    - CP2102
    - UART pins of Arduino interface
24. **SIM808 UART configuration:**
    - SIM_TX: SIM808 UART TX
    - SIM_RX: SIM808 UART RX
25. **IOREF power selection:** configure the UART voltage level

3. GPS Antenna:

## Connection Details

At first Arduino UNO R3 is connected with GSM/GPRS/GPS Shield (B).Arduino digital pin 7 and 8 are connected with SIM808 module's USB to UART interface RX and TX connector and digital GND connector connected with module's Arduino expansion GND connector. Then GPS and GSM antenna are connected with GSM/GPRS/GPS Shield (B) SIM808 Module. Next everything is powered up using USB power connectors. GPS stands for Global Positioning System. The satellites give exact position which means GPS modem should receive the signal from all available satellites. GPS receives data from Antenna. GPS antenna should be outside of vehicles. GSM module is used to communicate through server. We need to insert sim card into the GSM/GPRS/GPS Shield (B) SIM808 Module. GPS tracking device should be fitted inside the vehicle where it is not visible. This project can be operated on battery of the vehicle.

## GPS Debugging

1. Send following AT commands:

- AT+ CGNSPWR =1 (GPS power up)
- AT+ CGNSTST =1 (GPS reset)

Return OK.

2. GPS signal output:

   Set  baud rate to  11250 in serial monitor.

3. Click serial monitor to check the GPS information using following AT commands:

- AT+CGNSPWR(GPS power control) =1, GPS power up
- AT+CGNSINF(Get current GPS location info)= often 32
- AT+CGPSSTATUS(GPS status)

## SIM Debugging

- Use following AT commands:
  1. AT+SAPBR=3,1,"APN","gpinternet"
  2. AT+SAPBR=1,1
  3. AT+HTTPINIT
  4. AT+HTTPPARA="CID",1
  5. AT+HTTPPARA="URL", http://www.iforce2d.net/test.php(this is our used server's web link to sync time zone)
  6. AT+HTTPACTION=0

(Please see SIM808_AT+Command+Manual_V1.01 for more details of AT commands)

## Source Code

### sim808GPSTracker.ino:

```
#include <SoftwareSerial.h>

SoftwareSerial ss(7,8); //(RX,TX)

#define GSM_PORT ss

#include "sim808.h"

#define JOURNEY "abcdefghi2"


void setup() {

  // setup code here, to run once:

  ss.begin(9600);

  Serial.begin(115200);

  Serial.println("Starting...");

  sim808_setup();

}


void sendPositionReport(unsigned long now) {

  GSM_PORT.print("AT+HTTPPARA=\"URL\",\"http://www.iforce2d.net/gt/gt2.php?");

// gps data will be saved in this web link

  GSM_PORT.print("&jn=");

  GSM_PORT.print(JOURNEY);

  GSM_PORT.print("&tm=");

  GSM_PORT.print( utc );

  GSM_PORT.print("&fx=");

  GSM_PORT.print(fixStatus);

  GSM_PORT.print("&lt=");

  GSM_PORT.print(lat);
  GSM_PORT.print("&ln=");
  GSM_PORT.print(lon);
  GSM_PORT.print("&sv=");
  GSM_PORT.print(sats);
  GSM_PORT.print("&ha=");
  GSM_PORT.print(hdop);
  GSM_PORT.print("&gs=");
  GSM_PORT.print(sog);
  GSM_PORT.print("&hd=");
  GSM_PORT.print(cog);
  GSM_PORT.println("\"");
  flushGSM(now);
  delay(500);
  sendGSM("AT+HTTPACTION=0");
}

void loop() {
  unsigned long now = millis();
  boolean gotGPS = false;
  if ( actionState == AS_IDLE ) {
    if ( fixStatus > 0 && now > lastActionTime + 10000 ) {
      sendPositionReport(now);
      lastActionTime = now;
      httpResult = 0;
      actionState = AS_WAITING_FOR_RESPONSE;
    }
  }
  else {
    // waiting on response - abort if taking too long
    if ( now > lastActionTime + 15000 ) {
      actionState = AS_IDLE;
      parseState = PS_DETECT_MSG_TYPE;
      resetBuffer();
    }
  }
  sim808_loop();
}
```

## Sim808.h:

```
#define DEBUG_SERIAL
void updateScreen();
enum _parseState {
  PS_DETECT_MSG_TYPE,
 PS_IGNORING_COMMAND_ECHO,
  PS_HTTPACTION_TYPE,
  PS_HTTPACTION_RESULT,
  PS_HTTPACTION_LENGTH,
  PS_HTTPREAD_LENGTH,
  PS_HTTPREAD_CONTENT,
  PS_CGNSINF_RUN_STATUS,
  PS_CGNSINF_FIX_STATUS,
  PS_CGNSINF_UTC,
  PS_CGNSINF_LAT,
  PS_CGNSINF_LON,
  PS_CGNSINF_MSL,
  PS_CGNSINF_SOG,
  PS_CGNSINF_COG,
  PS_CGNSINF_FIX_MODE,
  PS_CGNSINF_RESERVED1,
  PS_CGNSINF_HDOP,
  PS_CGNSINF_PDOP,
  PS_CGNSINF_VDOP,
  PS_CGNSINF_RESERVED2,
  PS_CGNSINF_GPS_SATS_IN_VIEW,
  PS_CGNSINF_GNSS_SATS_USED,
  PS_CGNSINF_GLONASS_SATS_IN_VIEW,
  PS_CGNSINF_RESERVED3,
  PS_CGNSINF_CN0,
  PS_CGNSINF_HPA,
  PS_CGNSINF_VPA
};

enum _actionState {
  AS_IDLE,
  AS_WAITING_FOR_RESPONSE
```

```
};

byte actionState = AS_IDLE;
unsigned long lastActionTime = 0;
byte parseState = PS_DETECT_MSG_TYPE;
char buffer[20];
byte pos = 0;

int httpResult = 0;
int contentLength = 0;
byte fixStatus = 0;
char utc[24];
char lat[16];
char lon[16];
char sog[8];
char cog[8];
char hdop[8];
byte sats = 0;

void resetBuffer() {
  memset(buffer, 0, sizeof(buffer));
  pos = 0;

void parseATText(byte b) {
#ifdef DEBUG_SERIAL
  Serial.write(b);
#endif
```

```
buffer[pos++] = b;
if ( pos >= sizeof(buffer) )
  resetBuffer(); // just to be safe


  #ifdef DEBUG_SERIAL
 // Detailed debugging
 /*Serial.println();
 Serial.print("state = ");
 Serial.println(parseStat);
 Serial.print("b = ");
 Serial.println(b);
 Serial.print("pos = ");
 Serial.println(pos);
 Serial.print("buffer = ");
 Serial.println(buffer);*/


 #endif
switch (parseState) {
case PS_DETECT_MSG_TYPE:
  {
    if ( b == '\n' )
      resetBuffer();
    else {
      if ( pos == 3 && strcmp(buffer, "AT+") == 0 ) {
        parseState = PS_IGNORING_COMMAND_ECHO;
      }
      else if ( b == ':' ) {
```

```
#ifdef DEBUG_SERIAL
     Serial.print("Checking message type: ");
     Serial.println(buffer);
#endif
     if ( strcmp(buffer, "+HTTPACTION:") == 0 ) {
#ifdef DEBUG_SERIAL
      Serial.println("Received HTTPACTION");
#endif
      parseState = PS_HTTPACTION_TYPE;
     }
     else if ( strcmp(buffer, "+HTTPREAD:") == 0 ) {
#ifdef DEBUG_SERIAL
      Serial.println("Received HTTPREAD");
#endif
      parseState = PS_HTTPREAD_LENGTH;
     }
     else if ( strcmp(buffer, "+CGNSINF:") == 0 ) {
#ifdef DEBUG_SERIAL
      Serial.println("Received CGNSINF");
#endif
      parseState = PS_CGNSINF_RUN_STATUS;
     }
     resetBuffer();
    }
   }
  }
  break;
  case PS_IGNORING_COMMAND_ECHO:
    {
      if ( b == '\n' ) {
#ifdef DEBUG_SERIAL
        Serial.print("Ignoring echo: ");
        Serial.println(buffer);
#endif
        parseState = PS_DETECT_MSG_TYPE;
        resetBuffer();
      }
    }
    break;
  case PS_HTTPACTION_TYPE:
    {
      if ( b == ',' ) {
#ifdef DEBUG_SERIAL
        Serial.print("HTTPACTION type is ");
        Serial.println(buffer);
#endif
        parseState = PS_HTTPACTION_RESULT;
        resetBuffer();
      }
    }
    break;
  case PS_HTTPACTION_RESULT:
    {
      if ( b == ',' ) {
#ifdef DEBUG_SERIAL
        Serial.print("HTTPACTION result is ");
        Serial.println(buffer);
#endif
        httpResult = atoi(buffer);
        parseState = PS_HTTPACTION_LENGTH;
        resetBuffer();
      }
    }
    break;
  case PS_HTTPACTION_LENGTH:
    {
      if ( b == '\n' ) {
#ifdef DEBUG_SERIAL
        Serial.print("HTTPACTION length is ");
        Serial.println(buffer);
#endif
        contentLength = atoi(buffer);
        // now request content
        if ( contentLength > 0 ) {
          GSM_PORT.print("AT+HTTPREAD=0,");
          GSM_PORT.println(buffer);
        }
        else
          actionState = AS_IDLE;
        parseState = PS_DETECT_MSG_TYPE;
        resetBuffer();
      }
    }
    break;
  case PS_HTTPREAD_LENGTH:
    {
      if ( b == '\n' ) {
        contentLength = atoi(buffer);
#ifdef DEBUG_SERIAL
        Serial.print("HTTPREAD length is ");
        Serial.println(contentLength);

        Serial.print("HTTPREAD content: ");
#endif
        parseState = PS_HTTPREAD_CONTENT;
        resetBuffer();
      }
    }
    break;
  case PS_HTTPREAD_CONTENT:
    {
      // for this demo I'm just showing the content bytes in the serial monitor
#ifdef DEBUG_SERIAL
      Serial.write(b);
#endif
      contentLength--;
      if ( contentLength <= 0 ) {
        // all content bytes have now been read
        parseState = PS_DETECT_MSG_TYPE;
        resetBuffer();
#ifdef DEBUG_SERIAL
        Serial.print("\n\n\n\n");
#endif
        actionState = AS_IDLE;
      }
    }
    break;
  case PS_CGNSINF_RUN_STATUS:
  case PS_CGNSINF_FIX_STATUS:
  case PS_CGNSINF_UTC:
  case PS_CGNSINF_LAT:
  case PS_CGNSINF_LON:
  case PS_CGNSINF_MSL:
  case PS_CGNSINF_SOG:
  case PS_CGNSINF_COG:
  case PS_CGNSINF_FIX_MODE:
  case PS_CGNSINF_RESERVED1:
  case PS_CGNSINF_HDOP:
  case PS_CGNSINF_PDOP:
  case PS_CGNSINF_VDOP:
  case PS_CGNSINF_RESERVED2:
  case PS_CGNSINF_GPS_SATS_IN_VIEW:
  case PS_CGNSINF_GNSS_SATS_USED:
  case PS_CGNSINF_GLONASS_SATS_IN_VIEW:
```

```
  case PS_CGNSINF_RESERVED3:
  case PS_CGNSINF_CNo:
  case PS_CGNSINF_HPA:
   {
     if ( b == ',' ) {
#ifdef DEBUG_SERIAL
        Serial.print("CGNSINF result for is ");
        Serial.print( parseState );
        Serial.print(" is ");
        Serial.println(buffer);
#endif
        if ( parseState == PS_CGNSINF_FIX_STATUS )
          fixStatus = atoi( buffer );
        else if ( parseState == PS_CGNSINF_GNSS_SATS_USED )
          sats = atoi( buffer );
        else if ( parseState == PS_CGNSINF_LAT )
          strncpy( lat, buffer, min(15,strlen(buffer)-1));
        else if ( parseState == PS_CGNSINF_LON )
          strncpy( lon, buffer, min(15,strlen(buffer)-1));
        else if ( parseState == PS_CGNSINF_SOG )
          strncpy( sog, buffer, min(7,strlen(buffer)-1));
        else if ( parseState == PS_CGNSINF_COG )
          strncpy( cog, buffer, min(7,strlen(buffer)-1));
        else if ( parseState == PS_CGNSINF_HDOP )
          strncpy( hdop, buffer, min(7,strlen(buffer)-1));
        else if ( parseState == PS_CGNSINF_UTC )
          strncpy( utc, buffer, min(14,strlen(buffer)-1));
        parseState += 1;

        resetBuffer();

      }

    }

    break;

  case PS_CGNSINF_VPA:

    {

      if ( b == '\n' ) {

#ifdef DEBUG_SERIAL

        Serial.print("PS_CGNSINF_VPA is ");

        Serial.println(buffer);

#endif

        actionState = AS_IDLE;

        parseState = PS_DETECT_MSG_TYPE;

        resetBuffer();



        Serial.print( "GPS state: " );

        Serial.print( utc );

        Serial.print( ", " );

        Serial.print( fixStatus );

        Serial.print( ", " );

        Serial.print( sats );

        Serial.print( " sats, hdop " );

        Serial.print( hdop );

        Serial.print( ", " );

        Serial.print( lat );
        Serial.print( ", " );
        Serial.print( lon );
        Serial.print( ", " );
        Serial.print( sog );
        Serial.print( " km/h, " );
        Serial.print( cog );
        Serial.println( " degrees" );
        //updateScreen();
      }
    }
    break;
  }
}
void sendGSM(const char* msg, int waitMs = 500) {
  GSM_PORT.println(msg);
  while(GSM_PORT.available()) {
    parseATText(GSM_PORT.read());
  }
  delay(waitMs);
}
void sim808_setup() {
  delay(500);
  ss.println("AT+CGNSPWR=1\n");
  delay(500);
  // wait ten seconds for GSM module to connect to mobile network
  Serial.print( "Waiting for SIM startup..." );
  delay(10000);
  sendGSM("AT+SAPBR=3,1,\"APN\",\"gpinternet\"");  // change this for your cell provider
  sendGSM("AT+SAPBR=1,1",3000);
  sendGSM("AT+HTTPINIT", 500);
  sendGSM("AT+HTTPPARA=\"CID\",1", 500);
  delay(500);
  ss.println("AT+CGNSPWR=1\n");
  delay(500);
}

void flushGSM(unsigned long now) {
  while(GSM_PORT.available()) {
    //lastActionTime = now;
    parseATText(GSM_PORT.read());
  }
}

void sim808_loop() {
  unsigned long now = millis();
  flushGSM(now);
  static unsigned long lastLocCheck = 0;
  if ( now - lastLocCheck > 5000 ) {
    ss.println("AT+CGNSINF\n");
    lastLocCheck = now;
  }
}
```

## Server link for GPS result

http://www.iforce2d.net/gt/index.php?jn=abcdefghi2&tz=6&os=1&lt=300

Here, “jn=abcdefghi2” is our journey id, “tz=6” means time zone GMT+6, “os=1” is first position point and “lt=300” is last position point. The first 100-200 position points for a journey id is garbage data if satellite signals are low, after that it’ll give correct position and os(>250 preffered) and lt can be changed to see different points.

## Applications of GPS Vehicle Tracker

- Vehicle tracker can be used in a motor cycle, car, school bus, truck, transport vehicles.
- We can also use this system for vehicle accident detection. And we can use it for woman tracking, child tracking system.

## Advantages of GSM and GPS vehicle tracking system

- We don’t have to call the driver to know their location. We can track the route.
- Fleet management system is fastest, easiest and reliable. So user can stay stress-free by installing this system to track his/her vehicle.

## Future Development

- We can send multiple SMS to multiple mobile numbers. Also, we can dial a call to the mobile numbers.
- Over-speed detection and hard breaking detection can be done in a future enhancement.