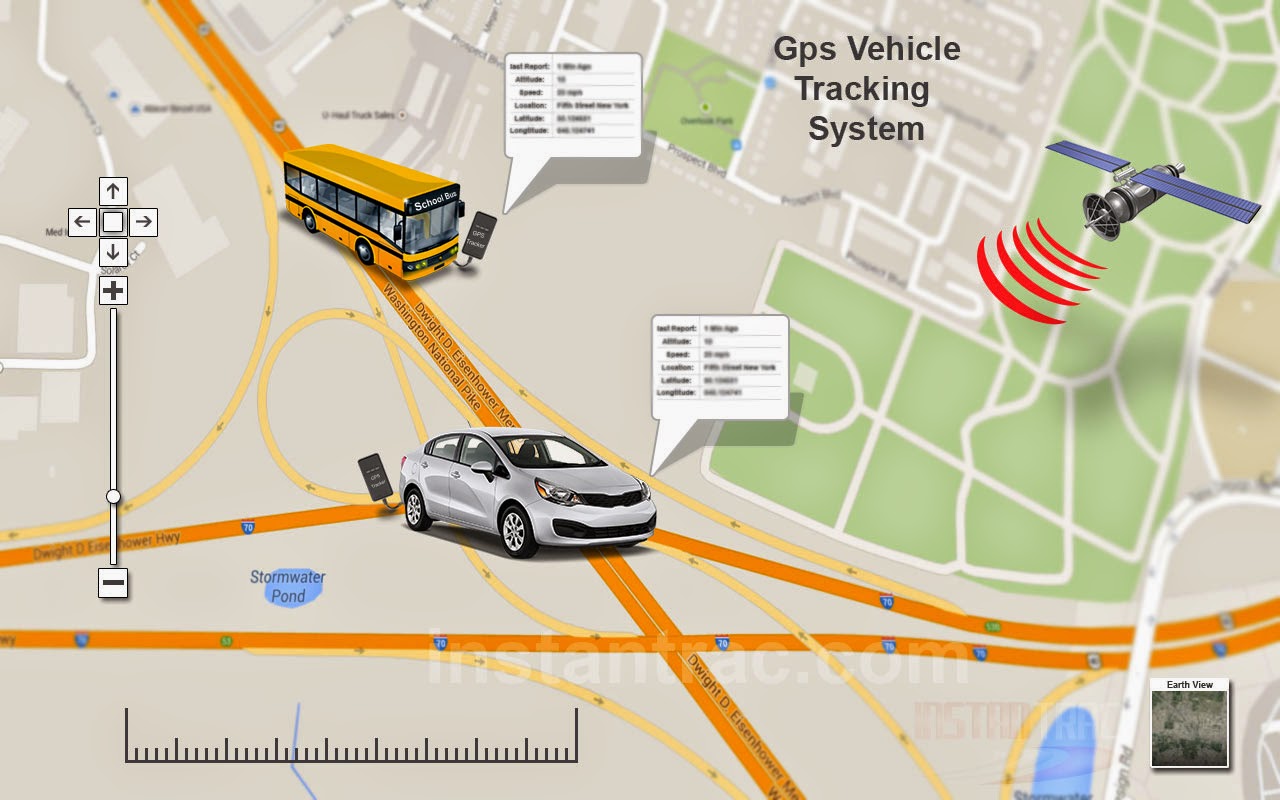
GPS Vehicle Tracking System

(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.

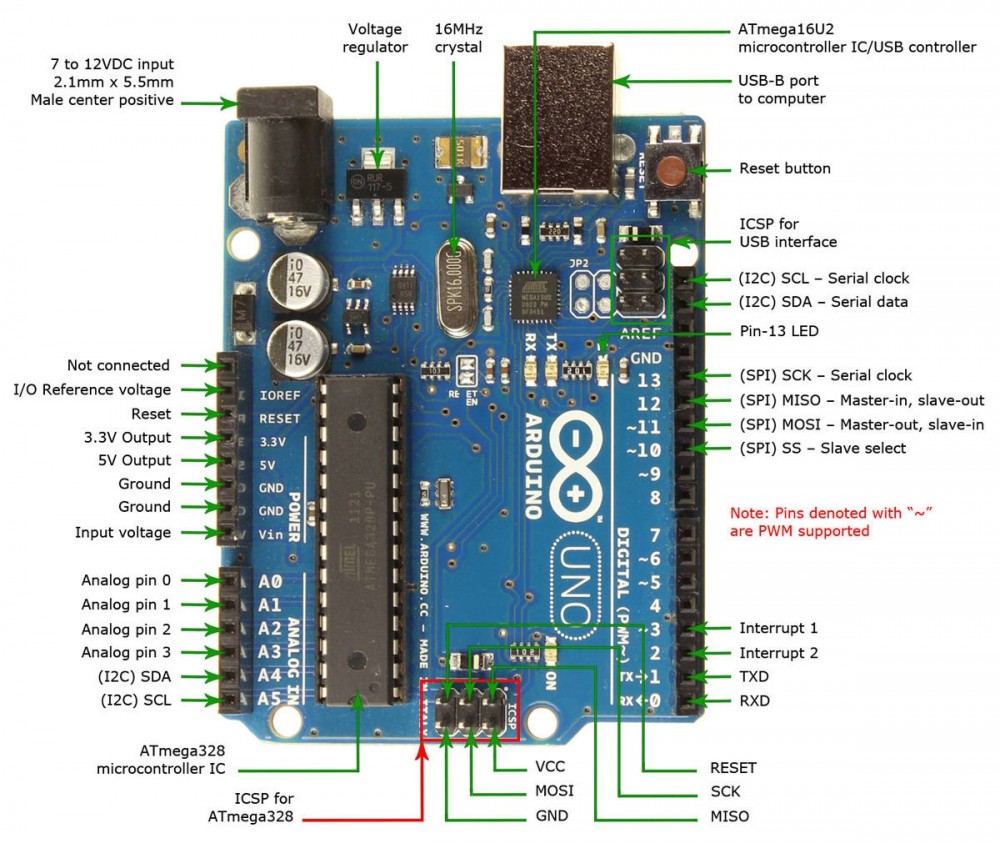
Real time vehicle tracking system using Arduino, GSM and GPS is an innovative and user-friendly system. Sometimes company’s transportation vehicles consume more and more fuel which results in loss of money. The solution for this is to install GPS tracking device in the vehicle. It sends real-time updates 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

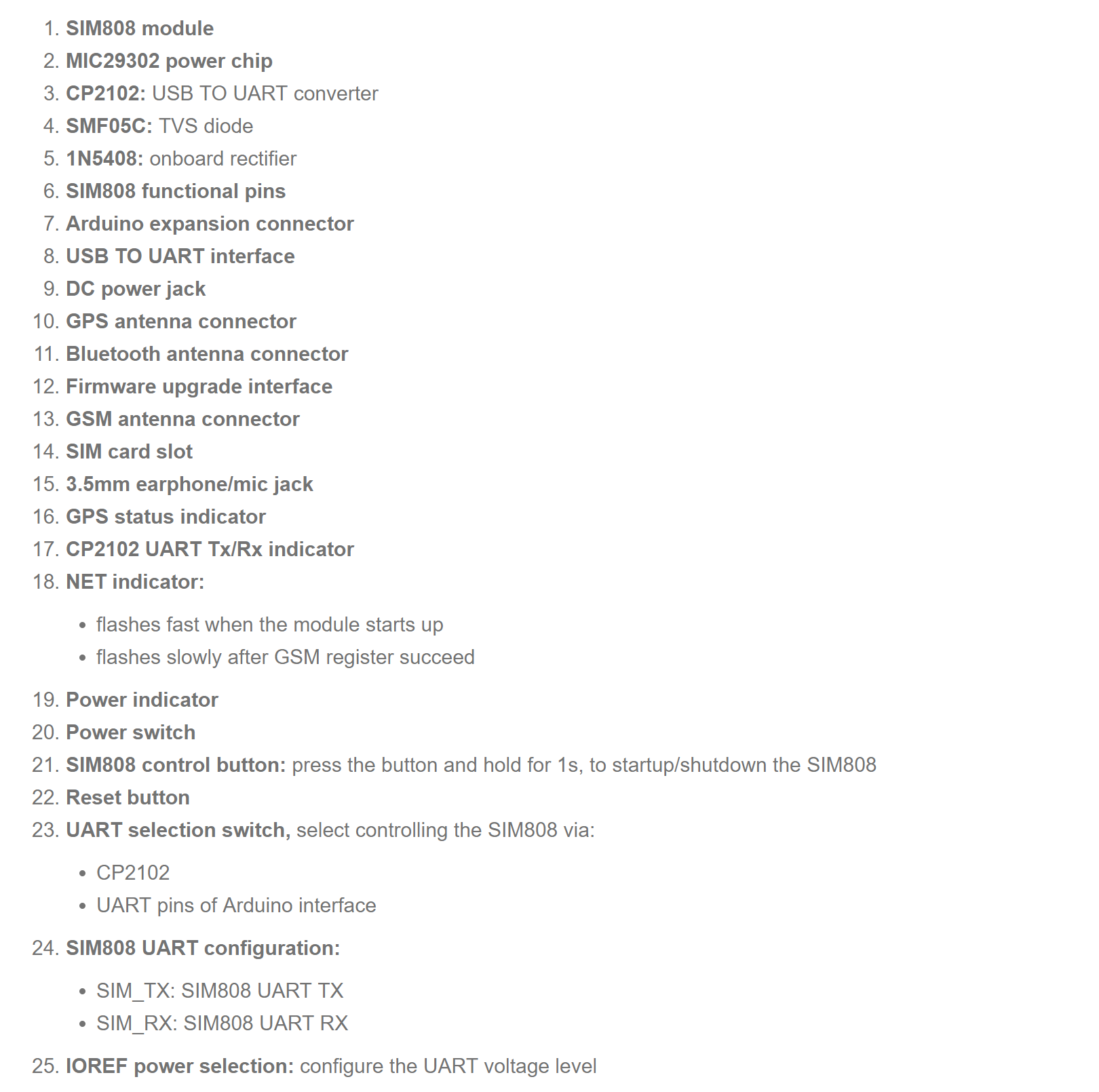
1. Arduino Uno R3
2. GSM/GPRS/GPS Shield (B) SIM808 Module
3. GPS Antenna
4. USB cables

# Circuit Diagram

1. Arduino UNO R3:



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



1. 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.

1. GPS signal output:

Set baud rate to 11250 in serial monitor.

1. 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\_CN0:

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.