

**JOMO KENYATTA UNIVERSITY**  
**OF**  
**AGRICULTURE AND TECHNOLOGY**

**PROJECT TITLE: OVERSPEED MONITORING SYSTEM**

**PROJECT MEMBERS**

**JOEL MIGWI                      EN271-0330/2007**

**KARIUKI S. KAIRU              EN271-0317/2007**

**SUPERVISOR:**

**MR BAARIU**

Presented on 4<sup>th</sup> Dec 2012

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILMENT OF REQUIREMENT FOR THE AWARD  
OF DEGREE IN ELECTRICAL AND ELECTRONICS ENGINEERING.

**DECLARATION.**

We do hereby declare that this project has never been presented or published anywhere or in any institution of learning. Therefore, it is our own original work.

Signed :.....

Date:.....

Joel Migwi

Signed:.....

Date.....

Kariuki S. Kairu

## **CERTIFICATION**

This is to certify that the above named student has so far carried out the project work, detailed in this report, under my supervision.

**Signed.....Date.....**

**Mr. Baariu**

**Project supervisor,**

**Department of Electrical and Electronic Engineering,**

**JKUAT.**

**DEDICATION.**

To those from whom we have learned with:

Family, Teachers and Colleagues.

**ACKNOWLEDGEMENT.**

We would like to acknowledge the following; the almighty God for bringing us this far, our supervisor Mr. Baariuwho has been guiding us, our lecturers, classmates and friends.

## ABSTRACT

This project will entail a GPS speed sensing mechanism which automatically updates a database with the details of an over speeding vehicle using the GSM system. Once the details are updated, the driver is charged for over speeding.

A GPS module is connected to a microcontroller which detects when the critical speed is exceeded and triggers the sending of different text messages, using GSM technology, to a police database and the driver. The text sent to the database contains the details of the car. The text sent to the owner informs them that they have over sped and the fine they are supposed to pay.

It is innovative because it uses both GPS and GSM technology. It also helps in the advancement of science and technology by diversifying the usage of GSM technology.

## Table of Contents

PROJECT TITLE: OVERSPEED MONITORING SYSTEM.....	i
DECLARATION.....	ii
CERTIFICATION.....	iii
DEDICATION.....	iv
ACKNOWLEDGEMENT.....	v
ABSTRACT.....	vi
TABLE OF FIGURES.....	ix
LIST OF EQUATIONS.....	ix
1.0 INTRODUCTION.....	10
1.1 Problem statement:.....	10
1.2 Justification.....	10
1.3 Objectives.....	2
1.3.1 General objectives.....	2
1.3.2 Specific objectives.....	2
2.0 LITERATURE REVIEW:.....	2
2.1 Speed cameras.....	3
2.1.1 Fixed instantaneous speed cameras.....	3
.....	3
2.1.2 Average speed cameras.....	4
2.2 Police operated equipment.....	5
2.2.1 Lidar speed gun.....	5
2.2.2 Radar speed gun.....	7
2.3 Speed governor.....	8
2.4 GPS system.....	9
2.5 GSM Network.....	11
3.0: METHODOLOGY.....	14
3.1 A brief description of the over speed monitoring system.....	14
3.1.1 Block diagram of the over speed monitoring system.....	14
3.1.2 Advantages of the system.....	15
3.1.3 Disadvantages of the system.....	15
3.4 Hardware Implementation.....	16

3.4.1 Arduino GPS receiver-EB365.....	16
3.4.2 Microcontroller-Arduino mega .....	17
3.4.3 LCD Display.....	20
3.4.4 GSM Interface (SIM900).....	21
3.5 Software design .....	23
3.5.1 GPS system.....	23
3.5.2 Arduino program cycle.....	28
.....	34
3.5.4 SIM 900 .....	34
3.5.5 Database .....	36
4.0 RESULTS.....	38
5.0 CHALLENGES RECOMMENDATIONS AND CONCLUSION .....	41
5.1 Challenges .....	41
5.2 Recommendations .....	41
5.3 conclusion .....	41
6.0 APPENDIX AND REFERENCES .....	42
6.1 Appendix .....	42
6.1.1 Time frame .....	58
6.1.2 Budget .....	59
6.2 References .....	59



## TABLE OF FIGURES

Figure 1: simplified block diagram of how fixed camera work .....	3
Figure 2: simplified block diagram of how a Radar gun works .....	8
Figure 3:Speed governor .....	9
Figure 4: GPS system .....	11
Figure 8:GSM architecture .....	12
Figure 10: Block diagram of the over speed monitoring system .....	15
Figure 5: GPS receiver EB365 .....	16
Figure 6: Arduino Mega .....	17
Figure 7: LCD display .....	20
Figure 9: SIM 900 .....	22
Figure 11: block flow of the system .....	23
Figure 12: Mapping of the polygons .....	28
Figure 13:Arduino program cycle.....	29
Figure 14:LCD Arduino mega interfacing .....	33
Figure 15: speed results .....	39
Figure 16: Database results.....	40

---

## LIST OF EQUATIONS

Equation 1 .....	6
Equation 2 .....	7

---

---

## CHAPTER 1

---

### 1.0 INTRODUCTION

#### 1.1 Problem statement:

According to the 2009 World Health Organization (WHO) global status report, Kenya recorded 3,760 traffic deaths, the highest in East Africa region. By 2015, WHO predicts the increase of road deaths to rise from 1.2 to 1.8 million, and 2.4 million by 2030. The key causes of road accidents are careless driving, drunken driving, over speeding, poorly serviced vehicles among others. Of these, over speeding accounts for about 51.9% according to African Health Sciences (AFR) organization.

Mitigation of loss of lives due to these avoidable mistakes is therefore imperative if economic sustainability is to be maintained. To solve the major problem of over speeding, several methods have been adopted but most of them are largely ineffective or manually operated and depend on the user's ability to be alert when using them. As such, an automatic speed alert and reporting system is needed which can inform a driver the speed limit as well as his/her speed limit and also alert him/her if he/she exceeds the limit.

The project will help to curtail over speeding mostly by public service vehicles due to the automatic nature of speed detection and the levies that will be imposed thereof.

#### 1.2 Justification

Taking into account that Kenya is a growing economy in terms of research, innovations and industry, there is a need for home grown solutions to our problems in order to achieve vision 2030.

Most drivers are not aware of the set speed limits on various roads and hence they end up exceeding the limit. Most over speed detection methods such as radar, speed cameras and ticketing system have been found not to be totally effective in curtailing over speeding on our roads.

Our project will provide a continuous and reliable 24 hour over speed monitoring system. Our project will also show the driver his current speed and the speed limit of the road. This will make the drivers be more careful which will in turn reduce road accidents and improve life

expectancy. Other benefits include reduction of highway bribery incidents by traffic police as well as less expense since the service will not be costly.

## 1.3 Objectives

### 1.3.1 General objectives

- To curb over speeding on Kenyan roads
- To automate speed checking , alert and fine imposition
- To reduce highway bribery.

### 1.3.2 Specific objectives

- To obtain speed parameters using the GPS system.
- To interface the GPS receiver with a microcontroller
- To develop a program that extracts the speed from satellites and compares it with a critical speed.
- To interface the microcontroller with an LCD to alert the driver of their current speed, the speed limit and a warning if they are over speeding.
- To send text messages to the driver and a database using the GSM network.
- To keep a database of vehicles that over speed.

## CHAPTER 2

---

### 2.0 LITERATURE REVIEW:

There are many methods used by authorities all over the world to monitor motor vehicle speeds on roads. These methods generally fall into one of two categories:

- To attempt to identify drivers or vehicles that are breaking the speed limit for the purposes of prosecution,
- To remind vehicle users what the speed limit is, and that it should be obeyed.

Some of the methods include:

## 2.1 Speed cameras

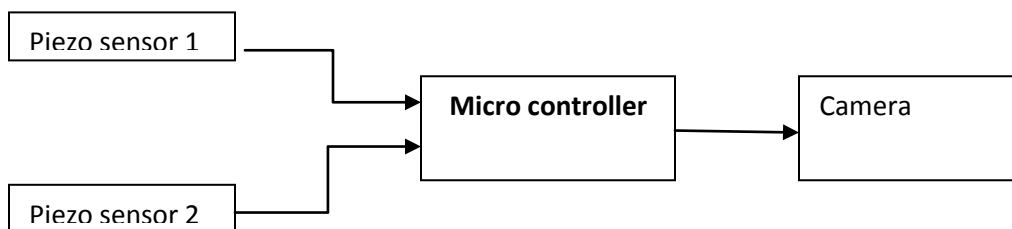
There are two types of speed camera in use:

### 2.1.1 Fixed instantaneous speed cameras

These cameras are installed beside a road and record the instantaneous speed of vehicles and a photograph of vehicles that have been identified as breaking the speed limit.

#### Principle of operation

Two piezo-electric strips are embedded in the road; two such strips at a known distance apart will detect your wheels over them. The pressure impressed on the strips will result to a proportional voltage. This voltages act as the signals that will be processed by the microprocessor. The time it takes between the two strips gives the speed. (Speed equals distance divided by time). A microprocessor records the time taken by the vehicle between the two strips and calculates the speed of the vehicle. The camera takes a photograph if the vehicle is over speeding. A simple illustration is as shown below



**Figure 1:simplified block diagram of how fixed camera work**

There are two types commonly in use:

#### 2.1.1.1 Gatso speed cameras

These were the first to be introduced. Their principle of operation is as explained above. The camera takes two photographs a normal and an infra red. The infra-red will 'see' the registration mark when the vehicle is caked in dirt. Also, to collect the proof, two photographs are taken in quick succession, so that the distance travelled can be seen. The "quick succession" is a known time: Again speed can be calculated. To determine the distance, white square marks are usually painted in the gutter of the road and again on the crown of the road. These are a known distance and the two photographs will show the distance travelled.

## Disadvantages

- They have no mechanism of sending the captured images. The authorities have to make periodic stops to collect the films.
- Gatsos can very easily be stopped. Burning them works, as does filling them with expandable foam.

### 2.1.1.2 Truvelo speed cameras

These cameras also use the above principle of operation. The Truvelo camera is usually a front-facing camera taking pictures using digital cameras which use infra-red to take a picture of a vehicle as it approaches, which includes an image of the driver. These cameras then transmit the image and speed to the authorities virtually instantly.

## Advantage

- They have the advantage over the Gatso, as they can transmit the image taken to a database instantly.

## Disadvantages

- They are expensive to roll out and maintain.
- They have no means of instantaneously notifying the driver they are over speeding. Police send the driver a ticket afterward through their mail.
- Speed cameras in general monitor over speeding but cannot stop the driver from over speeding like speed limiters

### 2.1.2 Average speed cameras

SPECS average speed camera systems utilize state of the art video system with Automatic Number Plate Reading (ANPR) digital technology. ANPR is a mass surveillance method that uses optical character recognition on images to read vehicle registration plates. ANPR can be used to store the images captured by the cameras as well as the text from the license plate, with some configurable to store a photograph of the driver. There are six primary algorithms that the software requires for identifying a license plate:

- Plate localization – responsible for finding and isolating the plate on the picture.
- Plate orientation and sizing – compensates for the skew of the plate and adjusts the dimensions to the required size.
- Normalization – adjusts the brightness and contrast of the image.
- Character segmentation – finds the individual characters on the plates.
- Optical character recognition.

- Syntactical/Geometrical analysis – check characters and positions against country-specific rules

## **Operation**

SPECS consists of a minimum of two cameras each fitted with infra red illuminators fitted on gantries above the road, so they can work day or night. SPECS speed cameras work out the vehicles average speed, given the time it takes to drive between the two camera positions. SPECS average speed cameras are fitted either at the roadside or in the central reservation (as pictured below) a set distance apart to create a speed controlled zone, or where appropriate, groups of cameras can be linked to create a speed controlled network. As vehicles pass between the entry and exit camera points their number plates are digitally recorded, whether speeding or not. Then, by ANPR recognition, the images on the video of matching number plates are paired up, and because each image carries a date and time stamp, the computer can then work out your average speed between the cameras. There is no film used for SPECS. SPECS are commonly used to enforce speed limits on dual carriageways and motorways. This is because one SPECS gantry can monitor up to four lanes of traffic at any one time. These cameras measure average speeds over a known or measured distance.

## **Advantages**

- They are more accurate than radar and other hand held equipment.
- They are less prone to sabotage as they are mounted far from reach.

## **Disadvantages**

- They are very expensive to roll out and maintain.
- They can only be fitted on highways and not connector roads as the cameras are far between.
- They have no means of instantaneously notifying the driver they are over speeding. Police send the driver a ticket afterward through their mail.
- Speed cameras in general monitor over speeding but cannot stop the driver from over speeding like speed limiters.

## **2.2 Police operated equipment**

Police officers can use LIDAR speed guns or the older and less accurate radar speed guns to monitor vehicle speeds on road. These may be handheld from temporary static sites or mounted on vehicles.

### **2.2.1 Lidar speed gun**

LIDAR(light detection and ranging) speed guns use a more direct method that relies on the reflection time of light. A laser speed gun shoots a very short burst of infrared laser light and

then waits for it to reflect off the vehicle. The gun counts the number of nanoseconds it takes for the round trip, and by dividing by 2 it can calculate the distance to the car. If the gun takes 1,000 samples per second, it can compare the change in distance between samples and calculate the speed of the car.

The use of many reflections and an averaging technique in the speed measurement process increase the integrity of the speed reading. Vehicles are usually equipped with a vertically oriented registration plate that, when illuminated, causes a high integrity reflection to be returned to the LIDAR - despite the shape of the vehicle.

The basic LIDAR principles discussed can be expressed in the Single-scattering LIDAR equation:

$$P_r(R) = P_0 \left( \frac{c\tau}{2} \right) \beta(R) A_r R^{-2} \exp \left[ -2 \int_0^R \alpha(r) dr \right]$$

### Equation 1

#### Where:-

$P_r$  is the instantaneous received power at time  $t$ ,

$P_0$  is the transmitted power at time  $t_0$ ,

$c$  is the velocity of light,

$\tau$  is the pulse duration,

$\beta$  is the volume backscattering coefficient of the atmosphere,

$R$  is range,

$A_r$  is the receiving area and  $\alpha$  is the volume extinction coefficient of the atmosphere (cross section per unit volume).

#### Advantages

- It's more accurate than the radar speed gun.
- It's cheaper to roll out than speed cameras

## Disadvantages

- Being mostly a handheld device requires police presence.
- Lidar does not automatically alert the driver if he is over speeding.
- User training and certification are required so that a radar operator can use the equipment effectively.
- On very hot days with low humidity a visible mirage/reflection of the target vehicle is created. In many cases, when the laser is aimed at the target vehicle the infrared beam also receives readings from both the target vehicle and the mirage causing a Sweep error.

### 2.2.2 Radar speed gun

A radar speed gun is a Doppler radar unit that may be hand-held, vehicle-mounted or static. It measures the speed of the objects at which it is pointed by detecting a change in frequency of the returned radar signal caused by the Doppler Effect, whereby the frequency of the returned signal is increased in proportion to the object's speed of approach if the object is approaching, and lowered if the object is receding. Such devices are frequently used for speed limit enforcement.

From that difference, the radar speed gun can calculate the speed of the object from which the waves have been bounced. This speed is given by the following equation:

$$v = \frac{\Delta f}{2f}c$$

#### Equation 2

Where:-

$c$  is the speed of light,

$f$  is the emitted frequency of the radio waves

$\Delta f$  is the difference in frequency between the radio waves that are emitted and those received back by the gun.

## Advantages

- It's the cheapest method of over speed monitoring.

## Disadvantages



- It's a stationary road enforcement tool.
- User training and certification are required so that a radar operator can use the equipment effectively.
- Reliable only when the location has been certified to be free of environmental influences that will cause false readings.
- Antenna Positioning Error-The radar beam travels in a straight line, neither bending around curves nor following the contour of hilly terrain. If the antenna is not properly positioned, it may seem to clock an approaching car when, in fact, it's clocking another car in the background.
- Look-Past Error-Even if the operator aims his antenna properly, radar is still subject to "look-past" error. This is caused by the radar looking past a small reflection in the foreground to read a larger reflection behind. This error is all the more insidious because poorly-trained operators assume it can't happen.

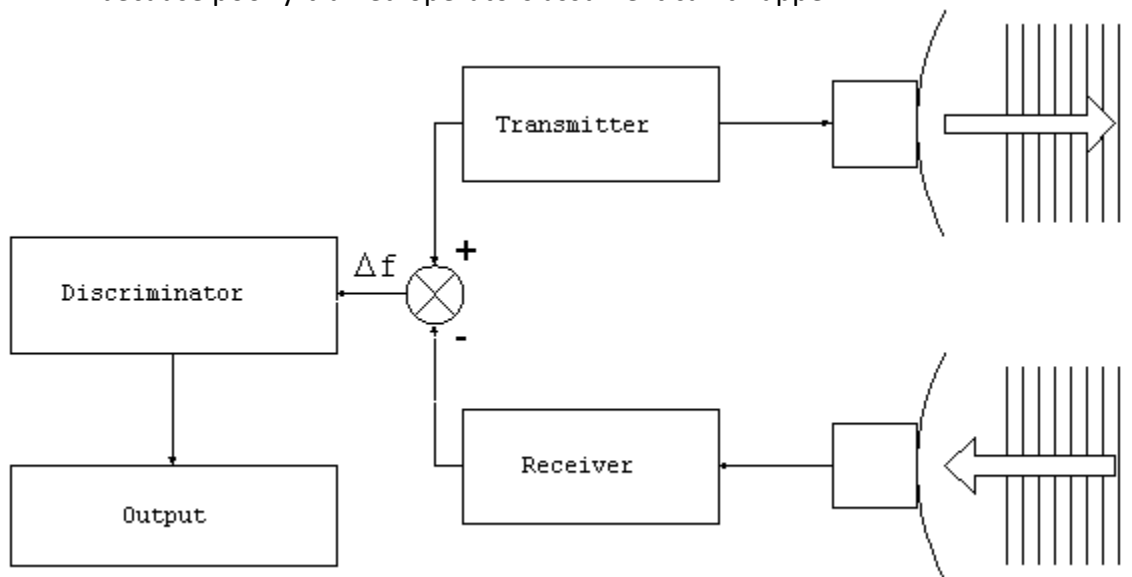
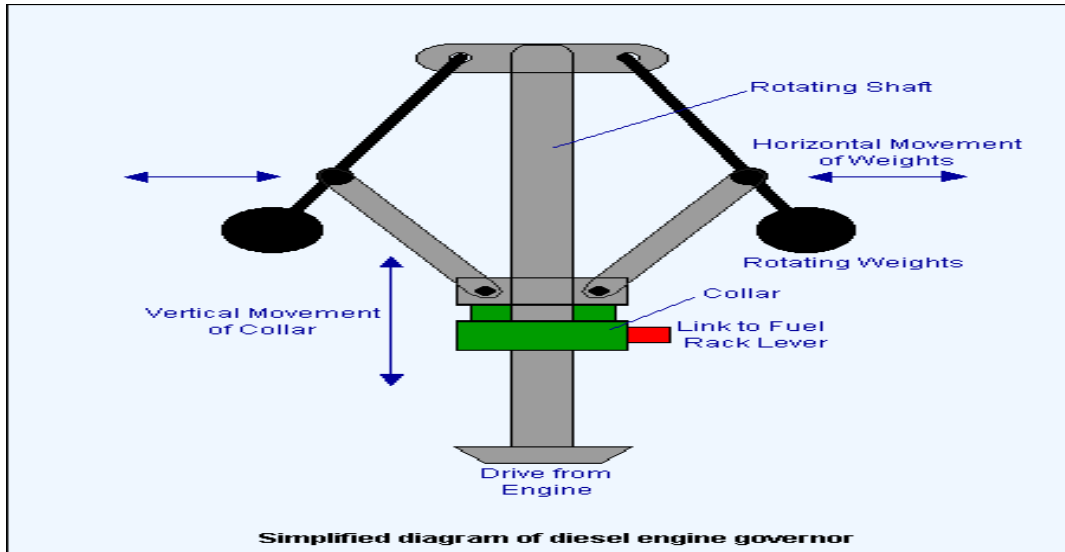


Figure 2:simplified block diagram of how a Radar gun works

### 2.3 Speed governor

The governor is a simple mechanical device which first appeared on steam engines. It operates on a diesel engine as shown in the diagram below.



**Figure 3:Speed governor**

### Operation

The governor consists of a rotating shaft, which is driven by the diesel engine. A pair of flyweights is linked to the shaft and they rotate as it rotates. The centrifugal force caused by the rotation causes the weights to be thrown outwards as the speed of the shaft rises. If the speed falls the weights move inwards.

The flyweights are linked to a collar fitted around the shaft by a pair of arms. As the weights move out, so the collar rises on the shaft. If the weights move inwards, the collar moves down the shaft. The movement of the collar is used to operate the fuel rack lever controlling the amount of fuel supplied to the engine by the injectors.

### Advantages

- It directly limits vehicle over speed by cutting the fuel into the engine.
- It doesn't require human operation and monitoring like radar and LIDAR.

### Disadvantages

- Requires a government law to enforce vehicles to fit speed governors.
- They are prone to malfunction and interference.

### 2.4 GPS system

Gps stands for global system positioning. It was first developed by the United States department of defense for military purposes such as target location, espionage and intelligence collection. It was later sanctioned for use by common civilians for free. The Gps system

comprises of a fleet of satellites that orbit the earth in a geo synchronous manner. In order to determine a location on the earth's surface, at least three satellites are required to accurately triangulate the exact position based on the latitude-longitude imaginary lines.

In addition to giving the location of a gps receiver on the earth's surface, GPS gives other parameters such as altitude, speed, course, number of satellites communicating and so on. . It is for these additional features that we adopted GPS to work alongside GSM in our project.

Using GPS, a device is able to calculate a lot of information about a moving object. Using even basic time and location data, a GPS unit can quickly calculate the relative speed of the object, based on how much distance it covered in a given time.

GPS devices are positional speedometers, based on how far the device has moved since the last measurement. The algorithm also uses the **Doppler shift** in the pseudo range signals from the satellites. The speed reading is normalized, and is not an instant speed.

Speeds are updated at short intervals to maintain accuracy at all times. It uses frequent calculations to determine the vehicle's speed. For example, using a standard movement per time calculation, if you have covered 80 feet in one second, the GPS device works out and converts that to MPH, which in this case is 55MPH.

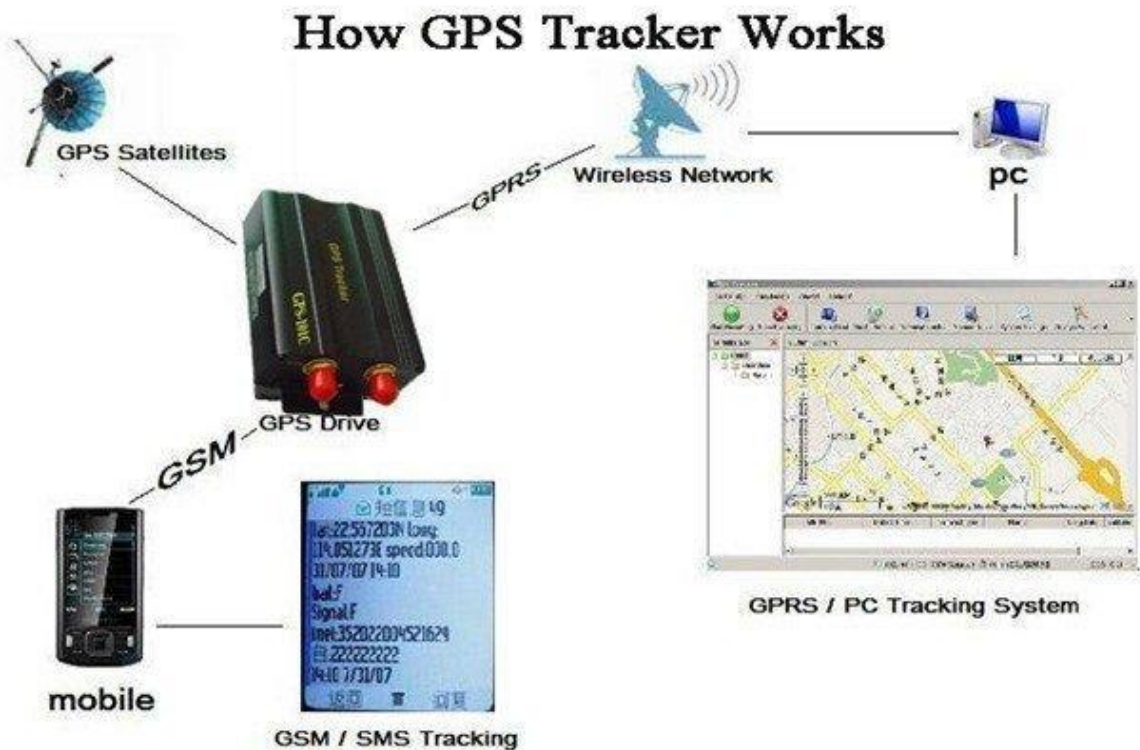


Figure 4:GPS system

## 2.5GSM Network

The GSM system was designed as a second generation (2G) cellular phone technology. One of the basic aims was to provide a system that would enable greater capacity to be achieved than the previous first generation analogue systems. GSM achieved this by using a digital TDMA (time division multiple access approach).

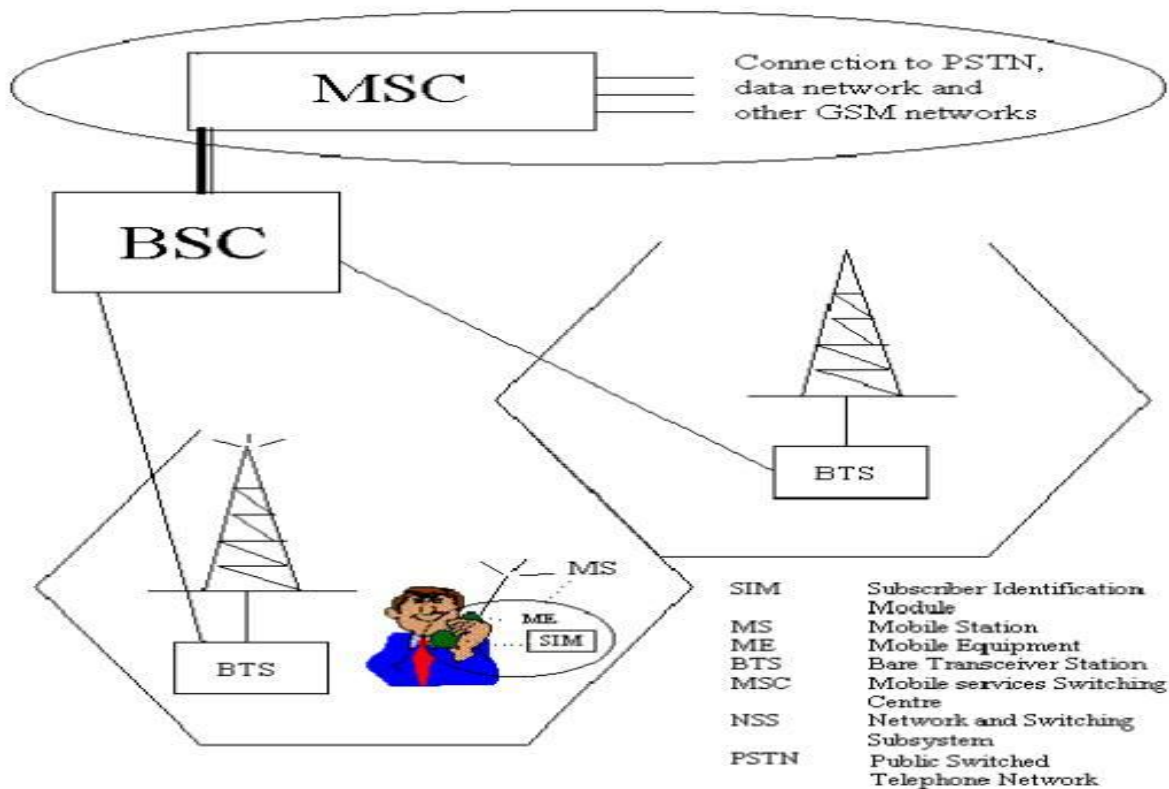
Speech or voice calls are the primary function for the GSM cellular system. In addition to the voice services; GSM cellular technology supports a variety of other data services like data and short text messages.

One service that has grown enormously is the short message service. Developed as part of the GSM specification, it has also been incorporated into other cellular technologies. It can be thought of as being similar to the paging service but is far more comprehensive allowing bi-directional messaging, store and forward delivery, and it also allows alphanumeric messages of a reasonable length. This service has become particularly popular as it provided a simple, low fixed cost.

The base transceiver stations (BTS) are organized into small groups, controlled by a base station controller (BSC) which is typically co-located with one of the BTSs. The BSC with its associated BTSs is termed the base station subsystem (BSS).

Further into the core network is the main switching area. This is known as the mobile switching Centre (MSC). Associated with it is the location registers, namely the home location register (HLR) and the visitor location register (VLR) which track the location of mobiles and enable calls to be routed to them. Additionally there is the Authentication Centre (AuC), and the Equipment Identify Register (EIR) that is used in authenticating the mobile before it is allowed onto the network and for billing.

The ME or mobile equipment, is the item that the end user sees. One important feature implemented on GSM is the use of a Subscriber Identity Module. This card carried with it the users identity and other information to allow the user to upgrade a phone very easily, while retaining the same identity on the network. It is also used to store other information such as "phone book" and other items.



**Figure 5:GSM architecture**

In a GSM network, the following areas are defined:

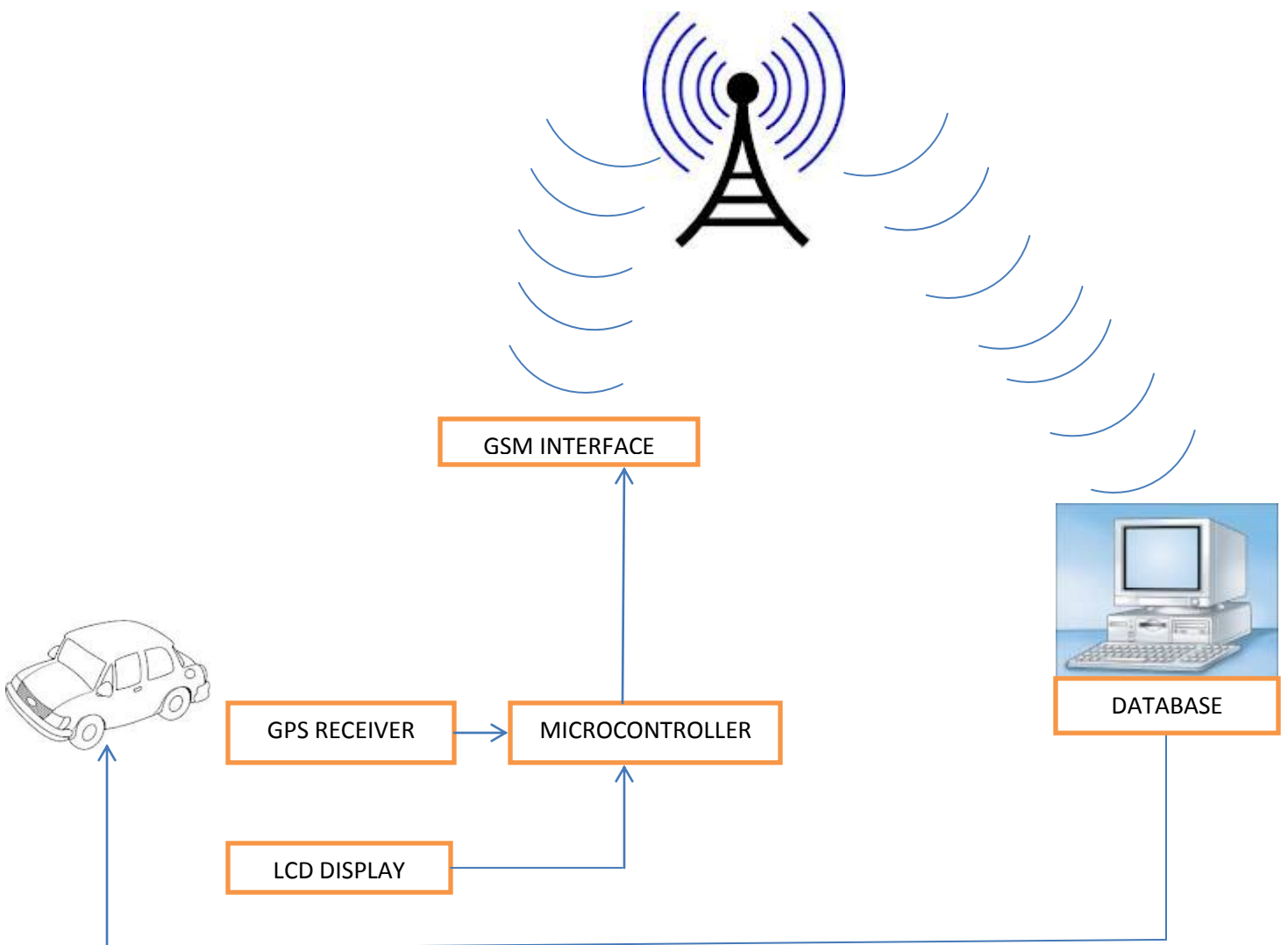
- **Cell:** Cell is the basic service area: one BTS covers one cell. Each cell is given a Cell Global Identity (CGI), a number that uniquely identifies the cell.
- **Location Area:** A group of cells form a Location Area. This is the area that is paged when a subscriber gets an incoming call. Each Location Area is assigned a Location Area Identity (LAI). Each Location Area is served by one or more BSCs.
- **MSC/VLR Service Area:** The area covered by one MSC is called the MSC/VLR service area.
- **PLMN:** The area covered by one network operator is called PLMN. A PLMN can contain one or more MSCs.

**3.0: METHODOLOGY**

**3.1 A brief description of the over speed monitoring system**

Our system incorporates the use of GPS system and GSM technology to build a system that monitors over speeding vehicles.

**3.1.1 Block diagram of the over speed monitoring system**



**Figure 6:Block diagram of the over speed monitoring system**

### OPERATION

The system employs a GPS receiver that obtains the speed of the vehicle, a microcontroller that compares the speed with the set speed limit and an LCD display that shows the speed limit and the current speed. If the speed of the vehicle exceeds the speedlimit, the LCD displays a warning message andif the vehicle continues to over speed, through the use of a GSM shield, :-

- a) An SMS is sent to a police database that the vehicle has exceeded the speed limit.
- b) The owner of the vehicle will be alerted also through an SMS that his vehicle is over speeding and also the amount they have to pay for the offence.

### **3.1.2 Advantages of the system**

- It is an automatic 24hr system unlike the radar and LIDAR devices which must be hand held.
- The system updates a database that can be reviewed in future
- our system informs the driver of the offence and fine instantly unlike the speed camera tickets that take time to be sent.
- Its cheaper and easier to implement than the speed cameras.
- The system will help curb corruption as there will be less police bribery incident

### **3. 1.3 Disadvantages of the system**

- It can be sabotaged like the speed governors
- In areas of poor network coverage,the information might take time to be relayed.
- The SMS sent notifies the owner of the vehicle and not necessarily the driver



## 3.4 Hardware Implementation

### 3.4.1 Arduino GPS receiver-EB365



**Figure 7: GPS receiver EB365**

Arduino GPS shield is a GPS module bread out board designed for Global Positioning System receiver with SD interface. The GPS system comprises of a fleet of satellites that orbit the earth in a geo synchronous manner. In order to determine a location on the earth's surface, at least three satellites are required to accurately triangulate the exact position based on the latitude-longitude imaginary lines. The Arduino GPS shield will track to a satellite to give us the speed of the vehicle and the speed limit of the particular area.

#### FEATURES

1. Active antenna design with high receives sensitivity, compatible normal antenna.
2. Extremely fast time to first fix at low signal level.
3. UART (universal asynchronous receive transmit) interface.
4. 5V/3.3V compatible operation voltage level.

### 3.4.1.1 Advantages of using the GPS system

- It's a convenient system
- it's a global system
- it gives you more parameters(speed, location and time)
- It can also be used for tracking the vehicle incase it's stolen

### 3.4.1.2 Disadvantages of the GPS system

- Indicated speed can fluctuate more than with traditional speedometers, and can be slower to respond to changes in speed.
- GPS can lose track if there is poor reception from GPS satellites: under trees, in tunnels, in canyons (including urban canyons between tall buildings).

## 3.4.2 Microcontroller-Arduino mega

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

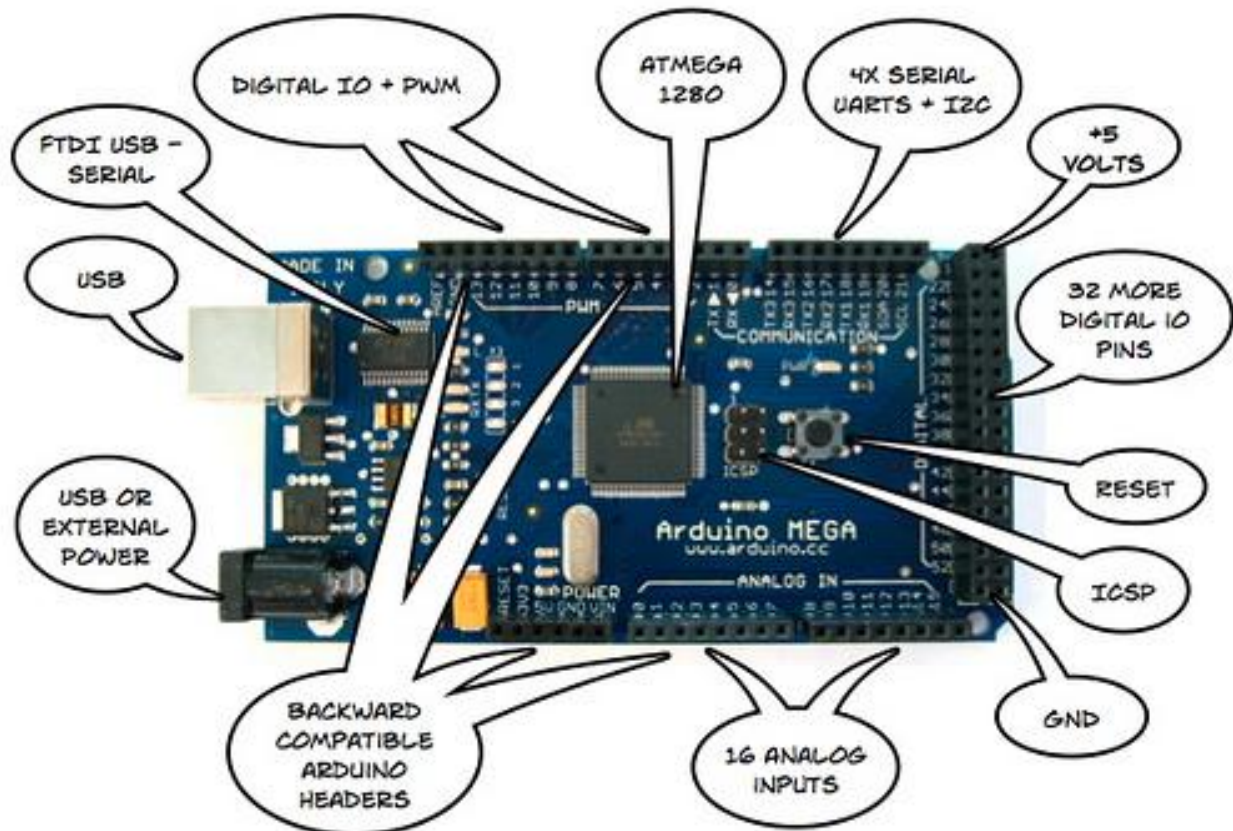


Figure 8: Arduino Mega

## Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. The recommended range is 7 to 12 volts.

The power pins are as follows

1. VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
2. 5V. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
3. 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
4. GND. Ground pins.

## Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the boot loader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

### Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

1. Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15
2. (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL (transistor transistor logic) serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
3. External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.
4. PWM: 0 to 13. Provide 8-bit PWM output with the `analogWrite()` function.
5. SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI (serial peripheral interface) communication using the SPI library.

6. LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and analog Reference() function.

### Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer.

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Mega2560's digital pins.

### Programming

The Arduino Mega can be programmed with the Arduino software. The ATmega2560 on the Arduino Mega comes preburned with a boot loader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. An Arduino sketch is used to write Arduino programs, compile and debug them then load them into the Arduino.

### Shield Compatibility

The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove.

Digital

pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively).

#### *3.4.2.1 Advantages of using the Arduino Mega*

- Arduino simplifies the amount of hardware and software development needed to get the system running
- The Arduino hardware platform already has the power and reset circuitry setup.
- It already has circuitry to program and communicate with the microcontroller over the USB port.

- The I/O pins are typically already fed out to sockets for easy access
- Arduino provides a number of libraries to make programming easier

### 3.4.3 LCD Display

LCD (Liquid Crystal Display) screen is an electronic display module. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven-segment and other multi-segment LEDs. The reasons being A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.



Figure 9: LCD display

#### 16x2 LCD PIN ASSIGNMENTS

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	V <sub>CC</sub>
3	Contrast adjustment; through a variable resistor	V <sub>EE</sub>
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2

10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V <sub>CC</sub> (5V)	Led+
16	Backlight Ground (0V)	Led-

#### *3.4.3.1 Advantages of using the 6 segment LCD display*

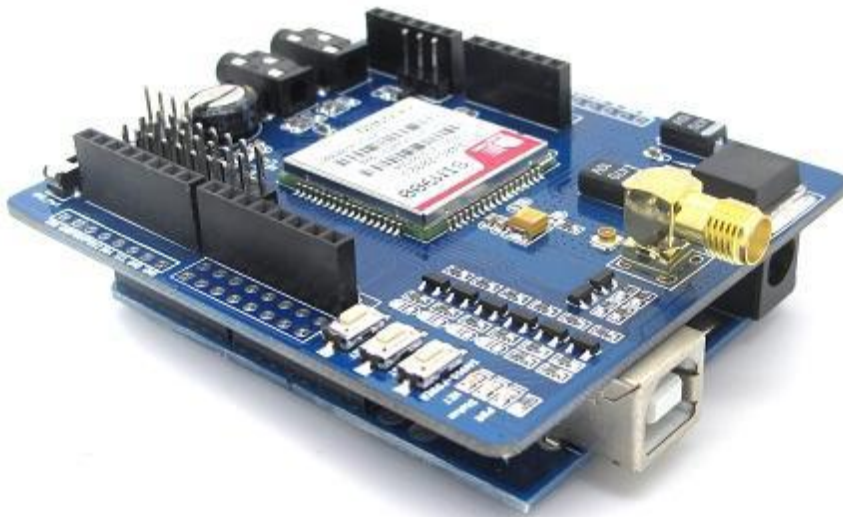
1. It's more economical than 7 segment displays.
2. It's easily programmable.

### **3.4.4 GSM Interface (SIM900)**

The Arduino GSM shield (SIM900) was used as our GSM interface. SIM 900 was used to send short message text to the server and the driver. The GPRS/GSM Shield provides you a way to use the GSM cell phone network to receive data from a remote location. The shield allows you to achieve this via any of the three methods:

- Short Message Service
- Audio
- GPRS

The GPRS/GSM Shield is compatible with all boards which have the same form factor (and pin out) as a standard Arduino Board. The GPRS/GSM Shield is configured and controlled via its UART using simple AT commands. The GSM Shield is like a cell phone without the Human Machine Interface.



**Figure 10: SIM 900**

### FEATURES

1. Quad-Band 850 / 900/ 1800 / 1900 MHz - would work on GSM networks in all countries across the world.
2. Control via AT commands - Standard Commands: GSM 07.07 & 07.05 | Enhanced Commands: • SIMCOM AT Commands.
3. SIM Card holder and GSM Antenna - present onboard.
4. Low power consumption - 1.5mA (sleep mode).
5. The factory default setting for the GPRS Shield UART is 19200 bps

The GSM Shield comes with all the accessories that we would need to get started with sending data over the GSM network. First we install the SIM card in the GSM Shield. Connect the antenna to the GSM Shield. Install the GSM Shield over the Arduino mega, being a shield design for arduinos, it plugs in fairly easily. The GSM\_TX & GSM\_RX jumpers on the GSM Shield are mounted in SW Serial position - that is GSM\_TX is connected to D7(RX) and GPRS\_RX to D8(TX). The Arduino is then connected to the computer using a USB cable. The ATmega328P microcontroller on board mega has only one UART which is used for communicating with the PC. What we need is an Arduino Sketch (a set of instructions written for Arduino boards for a specific task) running inside the ATmega328P that would emulate a second serial port (UART) using software on the digital pins D8 and D7 and patch through all the communication between

this second software serial port and the actual hardware serial port. By doing this, all the data coming from the computer (connected to the actual hardware UART) would be relayed as is to the GPRS Shield (connected to software UART) and we would be able to issue AT commands to control the GPRS Shield or load a sketch program to send messages.

#### 3.4.4.1 Advantages of using GSM communication.

- It is a cheaper method of communication.
- It's has a wide coverage compared to wireless communication.
- It is easily implementable since it is already rolled out.
- The larger public already understands the basics of GSM communication

### 3.5 Software design

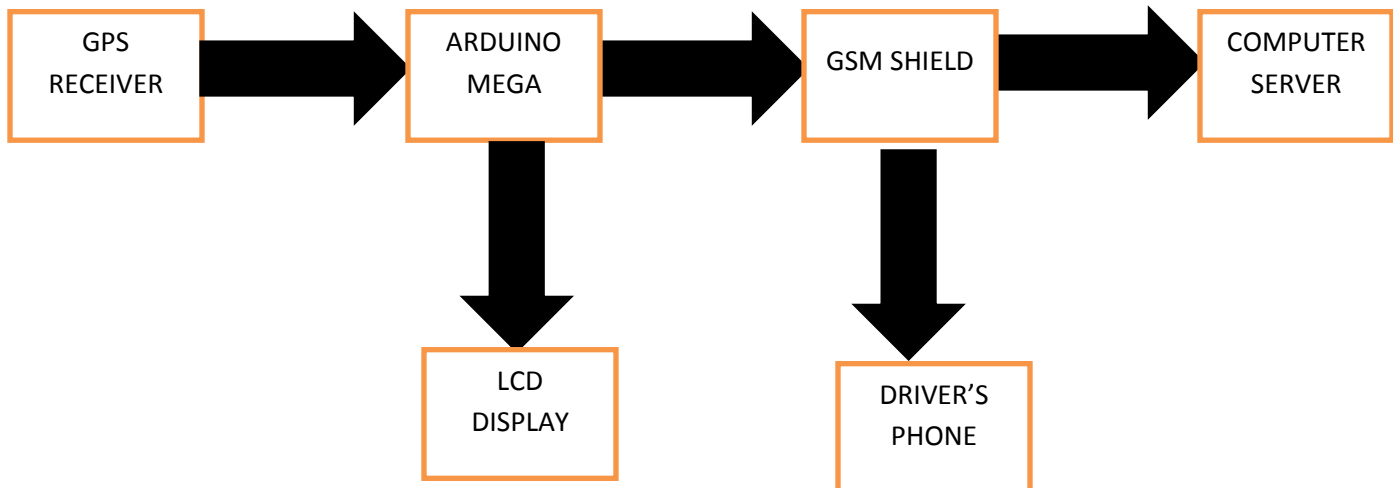


Figure 11: block flow of the system

#### 3.5.1 GPS system

In addition to giving the location of a GPS receiver on the earth's surface, GPS gives other parameters such as altitude, speed, course, number of satellites communicating and so on .In order to establish the speed zones complete with speed limits, GPS system was the most



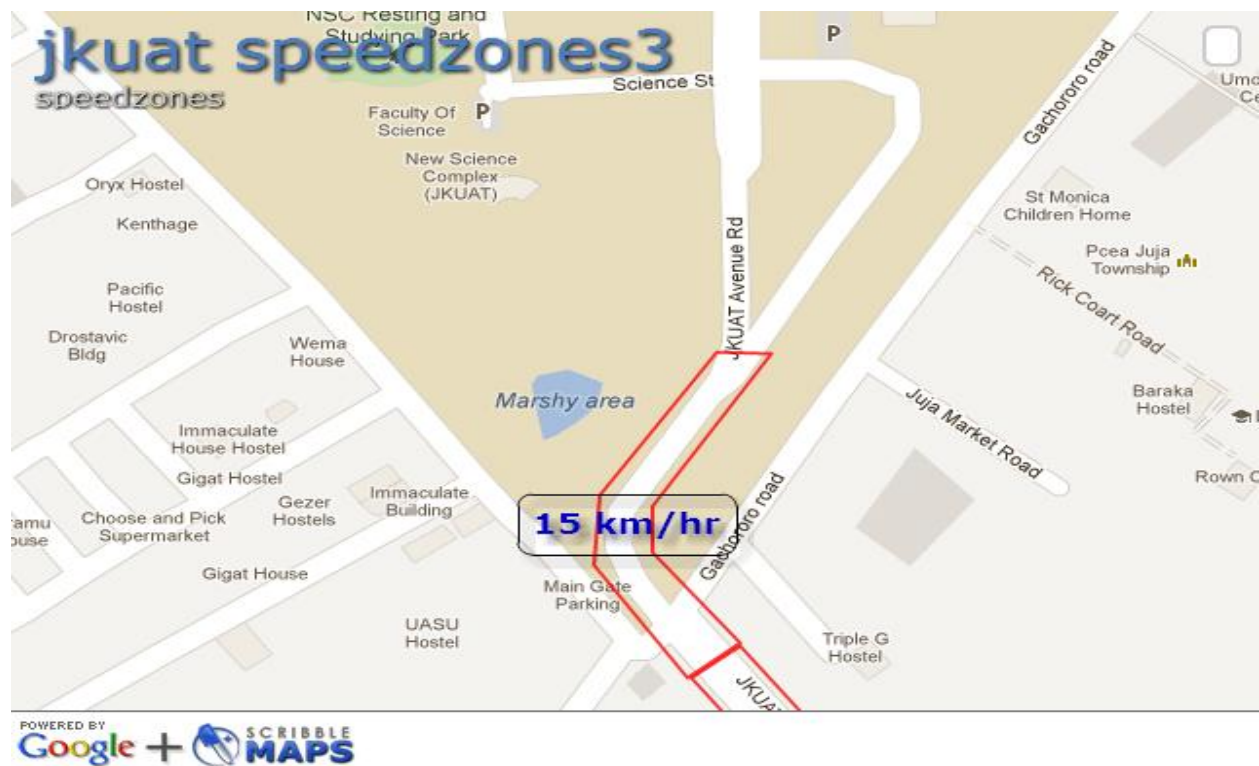
appropriate because of its capabilities. First, polygons were drawn on the Google map of the particular area of interest, which in this case is the JKUAT main campus area. The polygons were allocated speed limits which would govern how fast drivers could drive in them.

The microcontroller program will entail the declaration of these polygons' vertices (corners) in terms of latitude-longitude coordinate system. When the GPS receiver receives the coordinates of its current location and the speed at which the vehicle is cruising, the program checks for the polygon in which the vehicle is located and displays the speed limit of that particular polygon on a LCD screen. If the driver exceeds the speed limit, a message is sent to the owner of the vehicle as well as the server.

The JPEG images below are speed zones demarcations of the JKUAT area which is our area of interest. They are made from Google maps with the help of a drawing tool called scribble maps tool. A catch-all polygon encloses all the polygons so that all areas, especially residential are also included. Below are the maps.

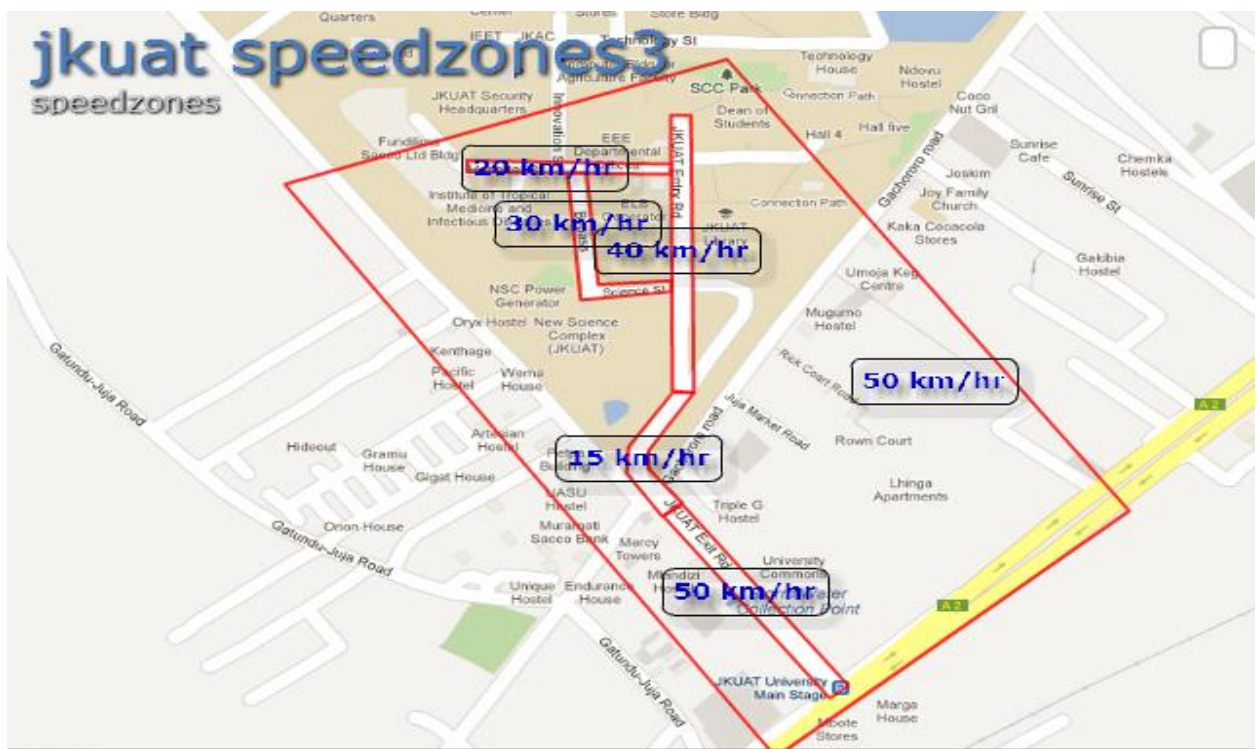


]

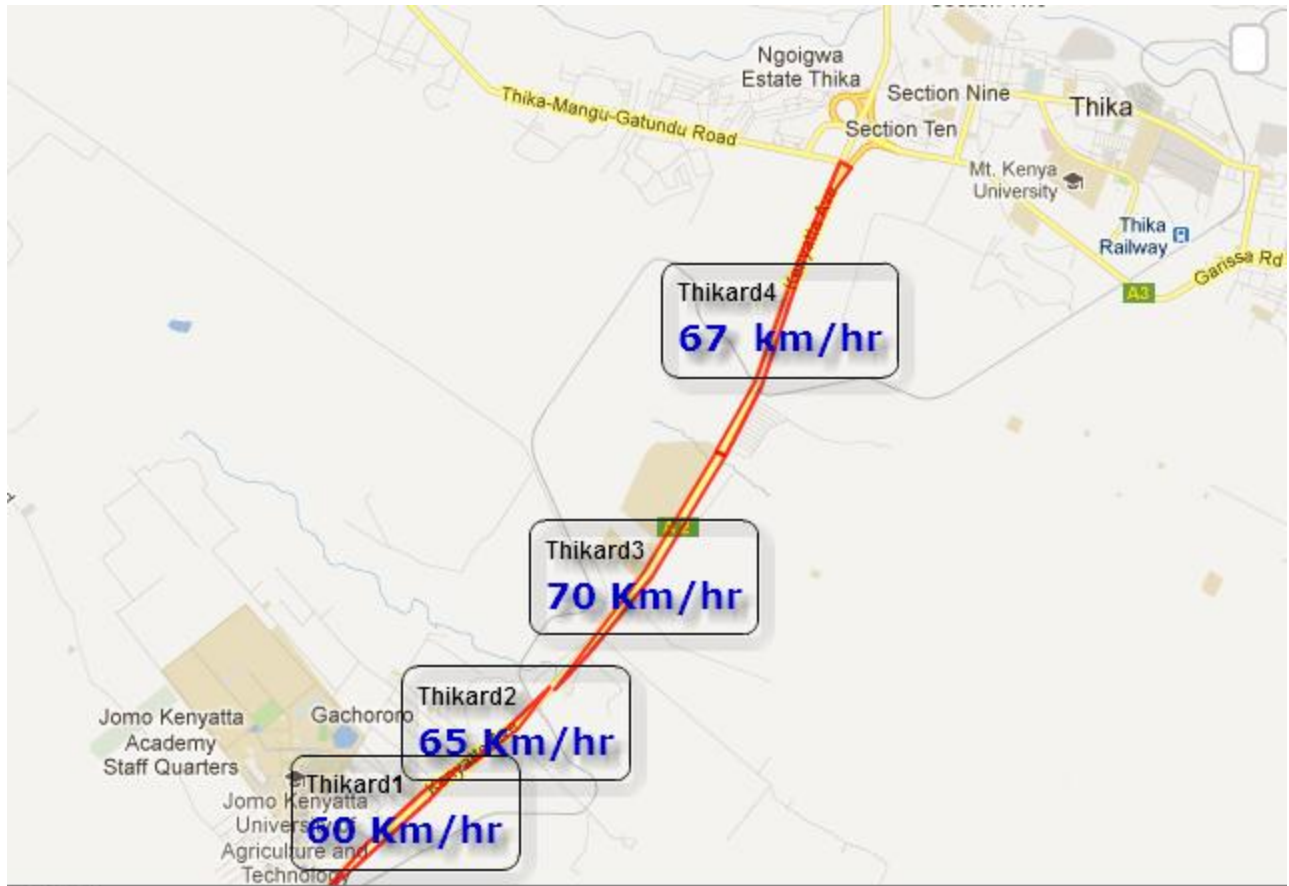




POWERED BY Google + SCRIBBLE MAPS



POWERED BY Google + SCRIBBLE MAPS



POWERED BY  
Google + SCRIBBLE  
MAPS



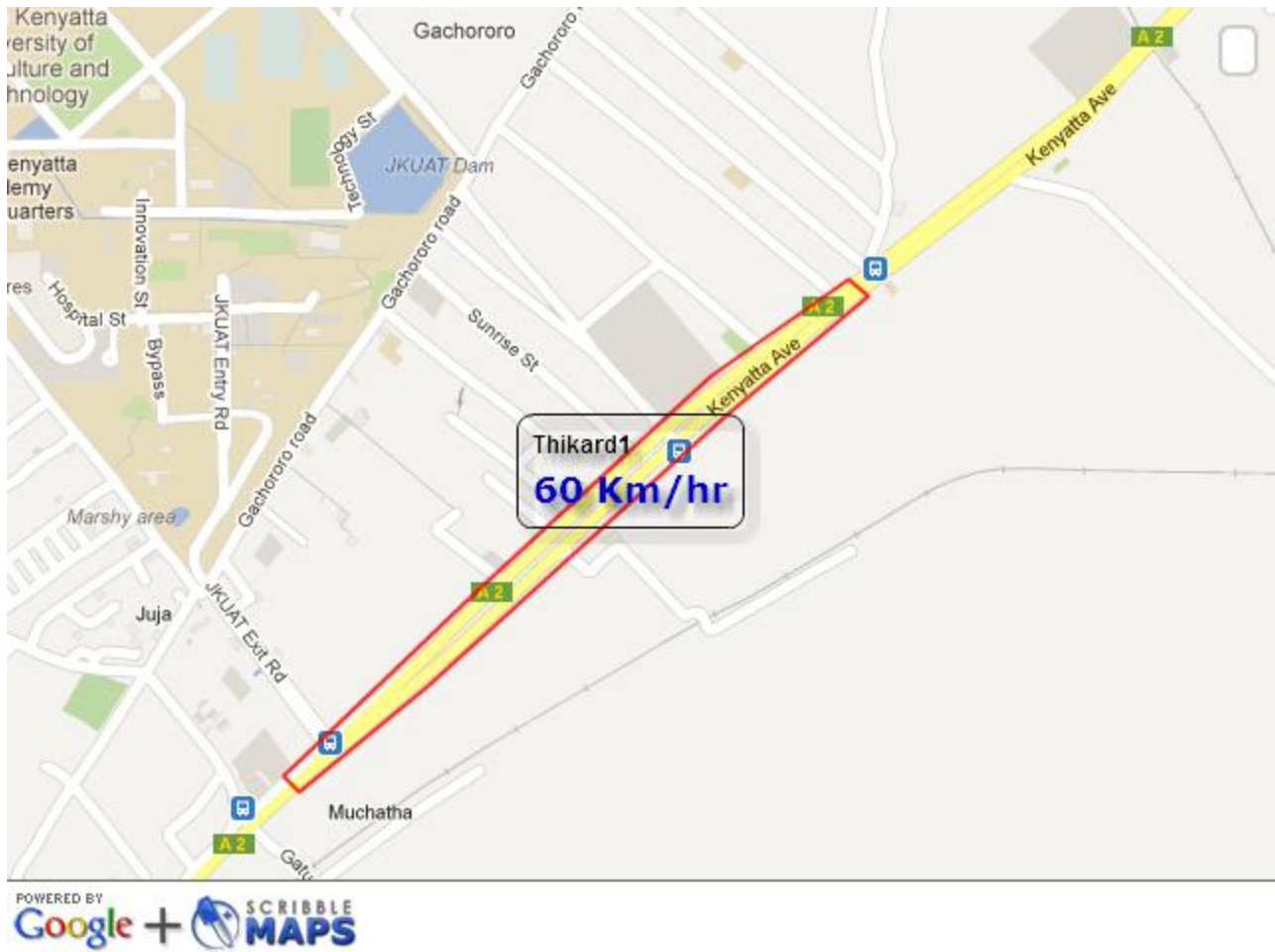


Figure 12: Mapping of the polygons

### 3.5.2 Arduino program cycle

When the GPS receiver tracks a satellite, it receives the longitude, latitude and speed from the satellite. The program algorithm uses the latitude and longitude to identify which polygon the vehicle is in and hence the speed limit of the particular polygon.

The microcontroller Extracts and outputs the speed limit assigned to the particular polygon and displays it on the LCD display. If the speed received exceeds the speed limit, a warning message is displayed on the LCD screen. Messages are also sent as described later.

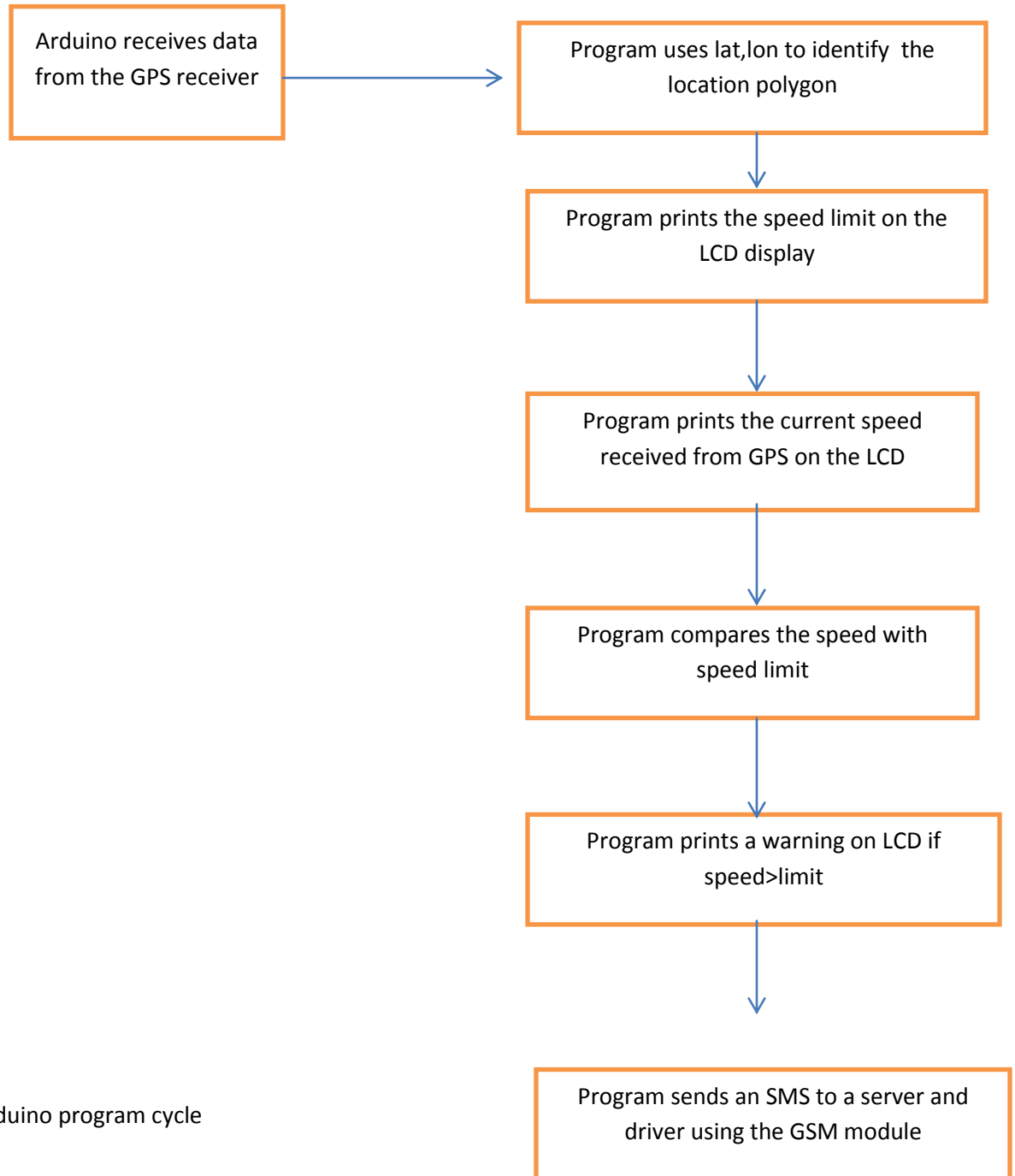


Figure 13:Arduino program cycle

### **Arduino Programming**

Arduino microcontroller is easy to program since it has its own software that does not even need to be installed. The software used in this project was the arduino 0023 software version due to its compatibility with both GPS and GSM libraries.

To begin with, important libraries were imported. These libraries were the GPS library and LCD library. Next, functions were declared and 'called' appropriately.

```
#include <TinyGPS.h>

#include <LiquidCrystal.h>
```

A typical arduino program has two default functions, that is, void setup and void loop. Void setup is used for pin initialization, hardware baud rate initialization and so on. This function was used to initialize the LCD and serial port baud rates. GSM initialization was also done here where the message center and SIM card number were initialized. The set up function is shown below.

```
void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);           //to be used by gsm module
  Serial1.print("\r");
  delay(1000);                  //Wait for a second while the modem sends an "OK"
  Serial1.print("AT+CMGF=1\r"); //Because we want to send the SMS in text mode
  delay(1000);

  Serial1.print("AT+CSCA="+254733000810+"\r"); //Setting for the SMS Message center number,
  delay(1000);

  lcd.begin(16, 2);             //16 by 2 LCD initialization

  //Print a message to the LCD.
  lcd.print("gps search....");

  setupSpeedZones();

  // Allow Eb-365 to power up
  delay(2000);

}
```

Void loop is a function that repeats a set of commands repetitively as assigned to it. In this case, void loop was used to call upon the speed reading, printing and comparison with speed limit functions.

```
void loop() {

  boolnewdata = false;

  if (feedgps()) {
    newdata = true;
  }
  if (newdata)
  {

  gpsdump(gps);
```

As can be seen above, functions called `gpsdump(gps)`, `feedgps`, and `speeding` are called upon in loop function. As will be explained later, `'gpsdump'` and `'feedgps'` functions work hand in hand to obtain gps data while `'speeding'` function is used to compare the current speed with the speed limit.

After this, the rest is simply functions declaration and . Some of the functions we have include Void `'setupspeedzones'`, Boolean `'isinspeedzone'`, `'intgetspeedlimit'`, `'booleanisspeeding'` and so on. These functions have self-explanatory names as can be seen. for example, the function is speeding `'Boolean isinspeedzone'` is useful for checking which polygon the vehicle is located and hence determine the speed limit.

Functions help a program to flow in an organized manner and prevent unnecessary events taking place when not desired. When a function is `'called'`, the program flow jumps to the specific function called and executes from there.

A function can return a value or not. Other functions are used for determining `'true'` or `'false'` so that an appropriate action can be taken. These functions are initialized as Boolean followed by the function name. For example, the function `'Boolean is speeding'` in this program is used to compare the current speed the vehicle is moving at and the set speed limit. If the speed limit is exceeded, the function returns `'true'` while if it is not, the function returns `'false'`. If the returned value is true, another function `'void sendtexts'` is called within the loop function. This function is responsible for sending texts. Below is a snippet of the speed comparison function, `'Boolean isinspeedzone'`.

```
booleanisSpeeding() {  
int speed = (int)(gps.f_speed_kmph() + 0.5);  
int speedLimit = getSpeedLimit();  
  
if (speed > speedLimit) {  
lcd.print("Slow down!!!");  
}  
delay(2000); //give the driver time to slow down  
  
if (speed > speedLimit) { //failure to slow down msg is sent  
return true;  
}
```



In order to retrieve data from GPS, the Tinygps library was used. This is an open source library developed for using with Arduino microcontroller for GPS data acquisition. To begin with, we downloaded the library and incorporated it into the arduino software. The function 'gpsdump(gps)' is used to obtain the data as shown below.

```
static void gpsdump(TinyGPS&gps)
{
longlat, lon;
unsigned long fix_age, time, date, speed, course;

// retrieves +/- lat/long in 100000ths of a degree
gps.get_position(&lat, &lon, &fix_age);

Serial.println(getSpeedLimit());

Serial.print("lat: ");
Serial.print(lat);
Serial.print(" lng: ");
Serial.print(lon);
Serial.print(" speed: ");
Serial.println(gps.f_speed_kmph());
lcd.clear();
lcd.print("LIMIT: ");
lcd.print(getSpeedLimit(),DEC);
lcd.setCursor(0,1);
lcd.print("SPEED: ");
lcd.print(gps.f_speed_kmph(),2);
}
```

### 3.5.3 LCD Display

The LCD is interfaced to the microcontroller with four data pins, register, enable, ground and read/write pin. The LCD displays the speed limit of the particular area and the current speed from the GPS receiver. If the speed is exceeded, the LCD will display a warning message.

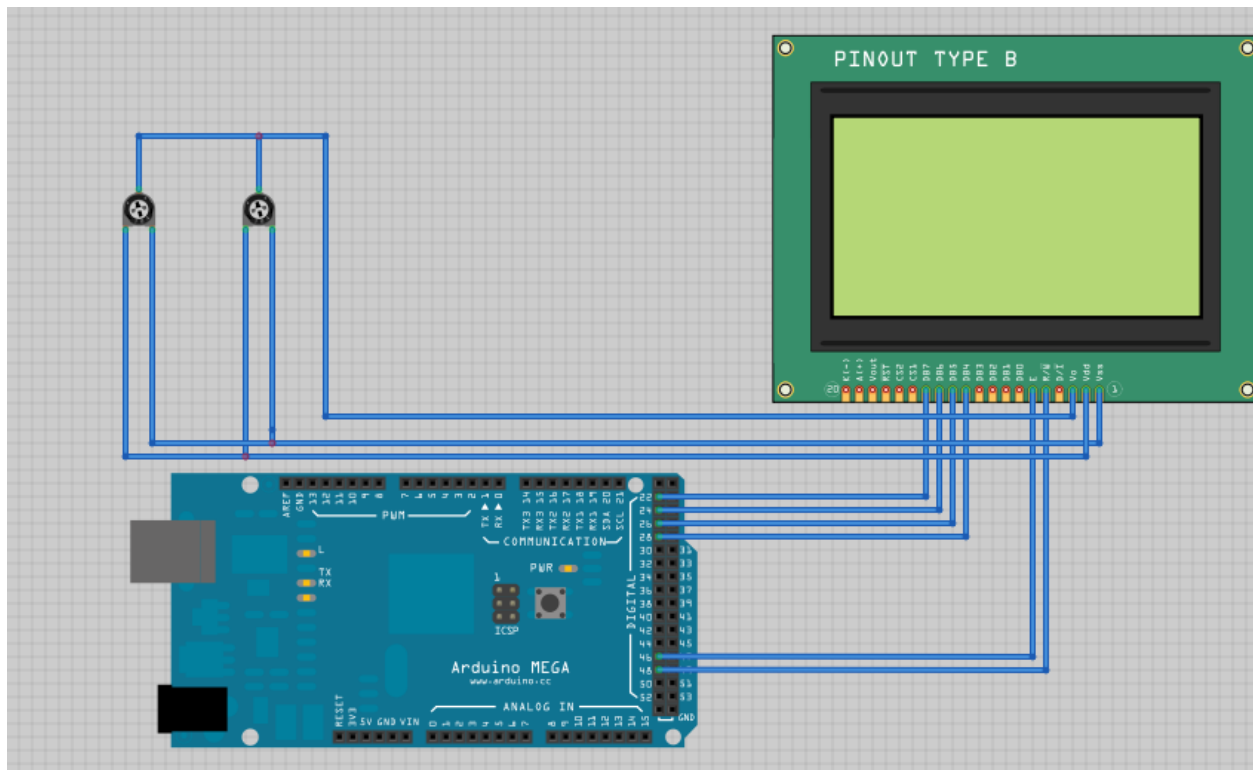


Figure 14:LCD Arduino mega interfacing

The LCD library was imported and LCD pins to be interfaced were initialized. `lcdbegin` is used to set up the LCD number of rows and columns. `setcursor` is used to set the cursor on the preferred row and column.

```
#include <LiquidCrystal.h>//this includes the LCD library

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);// initialize the library with the numbers of the interface pins

void setup() {

  // set up the LCD's number of columns and rows:

  lcd.begin(16, 2);

  // Print a message to the LCD.

  lcd.print("searching GPS...");//

}

void loop() {

  // set the cursor to column 0, line 1

  // (note: line 1 is the second row, since counting begins with 0):

  lcd.setCursor(0, 1);

  lcd.print("speed limit");//prints the speed limit of the area

}
```

### 3.5.4 SIM 900

When the speed limit is exceeded, the Arduino SIM 900 automatically sends an SMS to a police database. The message sent contains the registration number of the vehicle, the name of the owner and their phone number.

The GSM Shield can send SMSs in two modes: Text mode and PDU (or binary) mode. Since we wanted to send out a human readable message, we selected the text mode i.e. AT+CMGF=1.

AT+CMGS="+254727348186" This will instruct the GPRS Shield to start accepting text for a new message meant for the phone number specified (replace the number with the phone number of the target phone). Serial1 is used because GPS shield uses serial0.both shield cannot share the same serial.

### General SMS Commands

AT+CMGF – Set SMS Text Mode or SMS PDU Mode

AT+CMGD – Delete a Received Message

AT+CMGL – List Received Messages

AT+CMGS – Send SMS Message

AT+CMGL – List Received Messages

```
void setup()
{
  Serial.begin(19200); //Default serial port setting for the GPRS modem is 19200bps 8-N-1
  Serial.print1("\r");
  delay(1000); //Wait for a second while the modem sends an "OK"
  Serial.print("AT+CMGF=1\r"); //Because we want to send the SMS in text mode
  delay(1000);
  //Serial.print1("AT+CSCA="+254755500029"\r"); //Setting for the SMS Message center number,
  //delay(1000); //uncomment only if required and replace with
  //the message center number obtained from
  //your GSM service provider.
  //Note that when specifying a string of characters
  // " is entered as \"
  Serial.print1("AT+CMGS="+918446043032"\r"); //Start accepting the text for the message
  //to be sent to the number specified.
  //Replace this number with the target mobile number.
  delay(1000);
  Serial.print("car details"); //The text for the message
  delay(1000);
  Serial.print(26,BYTE); //Equivalent to sending Ctrl+Z
}
void loop()
```

### 3.5.5 Database

The database receives the details of the over speeding vehicle and stores then for future reference. The application was written in java programming language and it automatically receives the sent SMS from SIM900, stores the message in a database, retrieves the phone number of the owner from the SMS and automatically sends another SMS containing the charges for over speeding to the owner.

The application was divided into three parts, receiving the SMS, retrieving and sending the SMS to s database and automatically sending a text message.

Initially we import a collection class that is used to hold a collection of object(*java.util.ArrayList*). Class defines type of instances while void means the method has no return value. Themethod public void (do it) public means that the program, do it,can be accessed from anywhere and has no restrictions.

SerialModemGateway gateway = new SerialModemGateway("modem.com4", "COM8", 9600, "Huawei", "E173");this is used to set the modem settings.

```
msgList = new ArrayList<InboundMessage>();

        Service.getInstance().readMessages(msgList, MessageClasses.ALL)

        for (InboundMessagemsg : msgList)

        {

            System.out.println(msg);
```

This reads all the messages in the SIM card and displays them.It also waits to receive any incoming message.The message received (inbound message) should then be converted to an array so that it can be split for easier manipulation in the database.

```
public static void main(String args[])

    {

ReadMessages app = new ReadMessages();

        try

        {

            app.dolt();

        }

        catch (Exception e)

        {
```

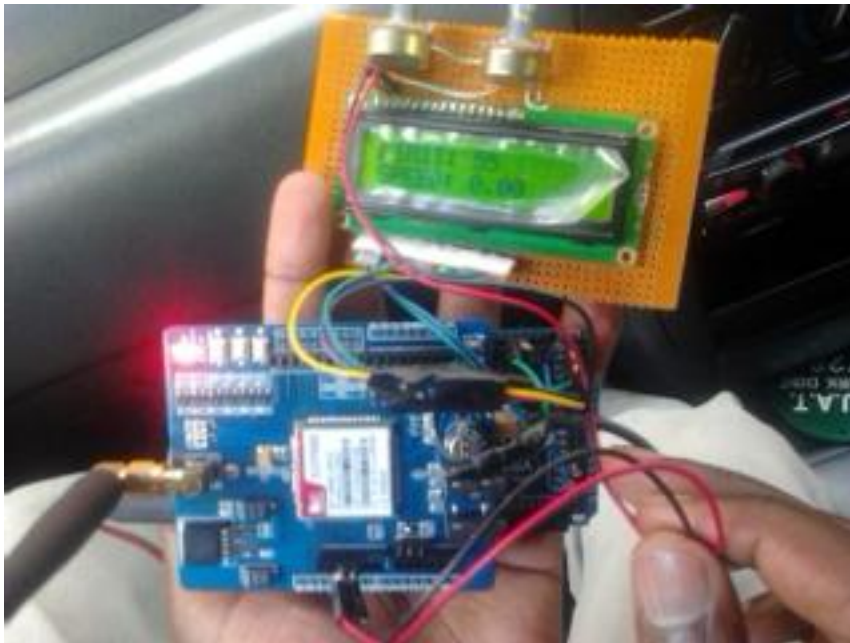
This is the main part of any java program, which is responsible for execution of the written instructions. catch (Exception e) will catch all the errors while e.printStackTrace() will show where the errors are.

```
OutboundMessage msg = new OutboundMessage("+254727348186", "pay fine ksh10,000 for  
overspeeding");  
  
    Service.getInstance().sendMessage(msg);  
  
    System.out.println(msg);
```

In the sending application, this instruction above sends the message to the driver of the vehicle. Service in java an operation offered as an interface that stands alone in the model. instance defines the behaviors of objects and getInstance() in this case will call on the sendMessage instruction to send the message that is “pay fine ksh10,000 for over speeding”.

## 4.0 RESULTS

In the testing of the system, the components were connected as follows. The LCD first displays “searching GPS...” as GPS shield receives data from the satellites. Then it prints out the speed limit of the area and the current speed of the vehicle.



The speed limit of the area is shown as 55kph and the current speed of the vehicle is zero.



This shows the speed limit on the first row and the current speed of the vehicle on the second line. The particular polygon has a speed limit of 50kph. The current speed of the vehicle is 39.39kph.



Figure 15: speed results

As we increased the speed up to 61kph, which is above 50kph, a text message is sent to a database and another received on the owners phone.



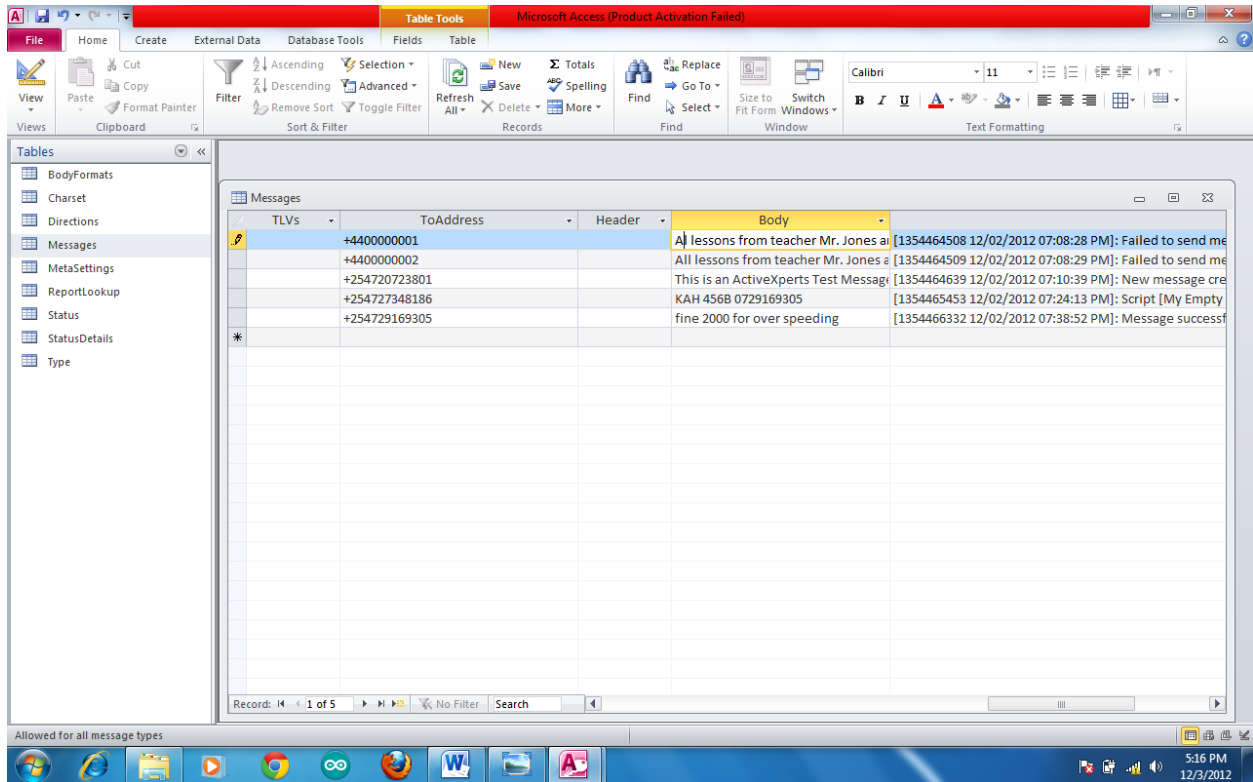


Figure 16: Database results

The print screen above shows the logging of the messages on the database created. The text message sent is stored in a database with details of the vehicle and the number of the owner. Another message is also sent to the owner of the vehicle.

## **5.0 CHALLENGES RECOMMENDATIONS AND CONCLUSION**

### **5.1 Challenges**

- The components making up for the system were not readily available.
- High cost of the components resulted in the use of less costly components which are not as effective.
- Obstructions e.g. cloud cover and trees affect the receiving capability of the GPS.
- Mapping into polygons of a huge area like a country may be quite challenging.

### **5.2 Recommendations**

In case of the system being rolled out in a country, mapping polygons may be replaced with wireless antennas placed in different speed zones. Such that the critical speed adjusts according to the signal received. Also other of the capabilities of the GPS receiver like vehicle tracking could also be incorporated.

### **5.3 conclusion**

The GPS receiver obtained the speed of the vehicle and assigned the speed limits according to the mapped areas. The LCD displayed the speed limits and the current speed of the vehicle. If the speed limit was exceeded, the LCD would print a message warning the driver to slow down and if the vehicle did not slow down, a text message with the details of the vehicle would be sent via the GSM shield to a database. Also an SMS containing the fine charged was sent to the owner of the vehicle.

## 6.0 APPENDIX AND REFERENCES

### 6.1 Appendix

#### ARDUINO PROGRAM CODE

```
#include <TinyGPS.h>
```

```
#include <LiquidCrystal.h>
```

```
LiquidCrystal lcd(48, 46, 28, 26, 24, 22);
```

```
#define DEFAULT_SPEED_LIMIT 55
```

```
#define NSPEEDZONES 8
```

```
boolean debugMode = true;
```

```
boolean speeding = false;
```

```
class Vertex {
```

```
public:
```

```
Vertex(float llat, float llng) {
```

```
lat = llat;
```

```
lng = llng;
```

```
}
```

```
float lat;
```

```
float lng;
```

```

};

classSpeedZone {
public:
SpeedZone(int s) {
speedLimit = s;
}
voidsetVertices(int n, Vertex *v) {
nVertices = n;
vertices = v;
}
intnVertices;
Vertex *vertices;
intspeedLimit;
};

SpeedZone *speedZones[NSPEEDZONES];
TinyGPSgps;

void setup() {
Serial.begin(9600);
Serial1.begin(9600); //to be used by gsm module
Serial1.print("\r");
delay(1000); //Wait for a second while the modem sends an "OK"
Serial1.print("AT+CMGF=1\r"); //Because we want to send the SMS in text mode

```

```

delay(1000);

Serial1.print("AT+CSCA=\"+254733000810\"\\r"); //Setting for the SMS Message center number,
delay(1000);

lcd.begin(16, 2);

//Print a message to the LCD.
lcd.print("gps search....");

setupSpeedZones();

    // Allow Eb-365 to power up
delay(2000);

}

voidsetupSpeedZones() {
    // jkuat avenue
    speedZones[0] = &SpeedZone(40);
    speedZones[0]->setVertices(4, (Vertex[4]){
    Vertex(-1.10007761, 37.01458699),
    Vertex(-1.09518616, 37.01458699),
    Vertex(-1.09518616, 37.01486594),
    Vertex(-1.10012052, 37.01490885)});
}

```

```
// main gate
speedZones[1] = &SpeedZone(15);
speedZones[1]->setVertices(8, (Vertex[8]){
Vertex(-1.10228734, 37.01444751),
Vertex(-1.10152573, 37.01392180),
Vertex(-1.10104302, 37.01395399),
Vertex(-1.10009906, 37.01459772),
Vertex(-1.10012052, 37.01490885),
Vertex(-1.10113957, 37.01425439),
Vertex(-1.10143992, 37.01425439),
Vertex(-1.10206208, 37.01472646)});
```

```
// jkuat exit road
speedZones[2] = &SpeedZone(50);
speedZones[2]->setVertices(4, (Vertex[4]){
Vertex(-1.10553757, 37.01688296),
Vertex(-1.10230879, 37.01446897),
Vertex(-1.10208353, 37.01473719),
Vertex(-1.10529086, 37.01714045)});
```

```
// science street/ NSC bypass
```

```
speedZones[3] = &SpeedZone(30);
speedZones[3]->setVertices(6, (Vertex[6]){
Vertex(-1.09829695, 37.01456553),
Vertex(-1.09845785, 37.01325661),
Vertex(-1.09623739, 37.01308495),
Vertex(-1.09621594, 37.01335317),
Vertex(-1.09818968, 37.01351410),
Vertex(-1.09812532, 37.01456553}});
```

```
//hospital ave
```

```
speedZones[4] = &SpeedZone(20);
speedZones[4]->setVertices(4, (Vertex[4]){
Vertex(-1.09619448, 37.01160437),
Vertex(-1.09600140, 37.01161510),
Vertex(-1.09603358, 37.01455480),
Vertex(-1.09623739, 37.01454407}});
```

```
// Thikard1
```

```
speedZones[5] = &SpeedZone(60);
speedZones[5]->setVertices(8, (Vertex[8]){
Vertex(-1.10645301, 37.01634485),
Vertex(-1.10595958, 37.01598007),
```

```

Vertex(-1.10184047, 37.02031452),
Vertex(-1.09733519, 37.02522832),
Vertex(-1.09527563, 37.02825386),
Vertex(-1.09561889, 37.02868301),
Vertex(-1.09823625, 37.02557165),
Vertex(-1.10402875, 37.01913435));

    // Thikard2
speedZones[6] = &SpeedZone(65);
speedZones[6]->setVertices(6, (Vertex[6]){

Vertex(-1.09538290, 37.02829677),
Vertex(-1.08853914, 37.03593570),
Vertex(-1.08879659, 37.03627903),
Vertex(-1.09210047, 37.03318912),
Vertex(-1.09403131, 37.03063566),
Vertex(-1.09557598, 37.02861864));

// Thikard3
speedZones[7] = &SpeedZone(70);
speedZones[7]->setVertices(7, (Vertex[7]){

Vertex(-1.08878586, 37.03638631),
Vertex(-1.08835679, 37.03591425),
Vertex(-1.07853093, 37.04393941),
Vertex(-1.06827596, 37.05011922),

```



```
Vertex(-1.06870504, 37.05093462),  
Vertex(-1.07896001, 37.04466898),  
Vertex(-1.08569651, 37.03908998}});
```

```
// Thikard1
```

```
speedZones[8] = &SpeedZone(67);  
speedZones[8]->setVertices(9, (Vertex[9]){  
Vertex(-1.06868359, 37.05104190),  
Vertex(-1.06825451, 37.05018360),  
Vertex(-1.06220449, 37.05348808),  
Vertex(-1.05152039, 37.05730754),  
Vertex(-1.04341074, 37.06082660),  
Vertex(-1.04409728, 37.06185657),  
Vertex(-1.04607105, 37.06031162),  
Vertex(-1.05177784, 37.05773670),  
Vertex(-1.06276230, 37.05404598}});
```

```
if (debugMode) {  
printSpeedZones();  
}
```

```
}
```

```
/*
```

```
* This is the point-in-polygon algorithm adapted from
```

```

* http://www.ecse.rpi.edu/Homepages/wrf/Research/Short\_Notes/pnpoly.html
*/
booleaninSpeedZone(intspeedZone, float lat, float lng) {
SpeedZone *s = speedZones[speedZone];

int i, j;

boolean inside = false;

for (i = 0, j = s->nVertices-1; i < s->nVertices; j = i++) {
if ( ((s->vertices[i].lat>lat) != (s->vertices[j].lat>lat)) &&
      (lng < (s->vertices[j].lng - s->vertices[i].lng) * (lat - s->vertices[i].lat) / (s->vertices[j].lat - s->vertices[i].lat)
+ s->vertices[i].lng) )
inside = !inside;
}

return inside;
}

booleaninSpeedZone(intspeedZone) {
floatlat, lng;

unsigned long fix_age;

// retrieves +/- lat/long in 100,000ths of a degree
gps.f_get_position(&lat, &lng, &fix_age);

returninSpeedZone(speedZone, lat, lng);
}

void loop() {

```

```
boolnewdata = false;

//unsigned long start = millis();

// Every second we print an update
//while (millis() - start < 1000)
//{
if (feedgps())
newdata = true;
//}
if (newdata)
{

gpsdump(gps);

}

speeding = isSpeeding();

if (speeding )
{
sendtexts();
}
```

```

}

static bool feedgps()
{
while (Serial.available())
{
if (gps.encode(Serial.read())) {
return true;
}
}
return false;
}

int getSpeedLimit() {
bool isInSpeedZone;

for(int i=0; i<NSPEEDZONES; i++) {
isInSpeedZone = isInSpeedZone(i);
if (isInSpeedZone) {
return speedZones[i]->speedLimit;
}
}

return DEFAULT_SPEED_LIMIT;
}

bool isInSpeeding() {

```

```

int speed = (int)(gps.f_speed_kmph() + 0.5);
intspeedLimit = getSpeedLimit();

if (speed >speedLimit) {
lcd.print("Slow down!!!");
}

delay(2000); //give the driver time to slow down

if (speed >speedLimit) { //failure to slow down msg is sent
return true;
}

return false;
}

booleansendtexts()
{

Serial1.print("AT+CMGS=\"+254729169305\"\\r"); //Start accepting the text for the message
//to be sent to the number specified.

//Replace this number with the target mobile number.

delay(1000);

Serial1.print("KBC      238C\\r"); //The text for the message

delay(1000);

Serial1.print(26,BYTE); //Equivalent to sending Ctrl+Z

```

```

}

void printSpeedZones() {

for(int i=0;i<NSPEEDZONES;i++) {

SpeedZone *s = speedZones[i];

Serial.println(s->speedLimit);

for(int v=0;v<s->nVertices;v++) {

Serial.print("");

Serial.print(s->vertices[v].lat);

Serial.print(" ");

Serial.print(s->vertices[v].lng);

Serial.println("");

}

}

}

static void gpsdump(TinyGPS&gps)

{

long lat, lon;

unsigned long fix_age, time, date, speed, course;

// retrieves +/- lat/long in 100000ths of a degree

gps.get_position(&lat, &lon, &fix_age);

```

```
Serial.println(getSpeedLimit());

Serial.print("lat: ");
Serial.print(lat);
Serial.print(" lng: ");
Serial.print(lon);
Serial.print(" speed: ");
Serial.println(gps.f_speed_kmph());
lcd.clear();
lcd.print("LIMIT: ");
lcd.print(getSpeedLimit(),DEC);
lcd.setCursor(0,1);
lcd.print("SPEED: ");
lcd.print(gps.f_speed_kmph(),2);

}
```

Database program code

```
// SendMessage.java - Sample application.
```

```
//
```

```
// This application shows you the basic procedure for sending messages.
```

```
//
```

```
package examples;
```

```
import org.smslib.ATGateway;
```

```
import org.smslib.IOutboundMessageNotification;
```

```
import org.smslib.Library;
```

```
import org.smslib.OutboundMessage;
```

```
import org.smslib.Service;
```

```
import org.smslib.modem.SerialModemGateway;
```

```
public class SendMessage
```

```
{
```

```
    public void doIt() throws Exception
```

```
    {
```

```
        OutboundNotification outboundNotification = new OutboundNotification();
```

```
        System.out.println("Example: Send message from a serial gsm modem.");
```

```
        System.out.println(Library.getLibraryDescription());
```

```
        System.out.println("Version: " + Library.getLibraryVersion());
```

```
        SerialModemGateway gateway = new SerialModemGateway("modem.com5", "COM40",  
9600, "Huawei", "E1550");
```

```
        gateway.setInbound(true);
```



```

gateway.setOutbound(true);

gateway.setSimPin("");

// Explicit SMSC address set is required for some modems.
gateway.setSmscNumber("+254727348186");

Service.getInstance().setOutboundMessageNotification(outboundNotification);

Service.getInstance().addGateway(gateway);

Service.getInstance().startService();

System.out.println();

System.out.println("Modem Information:");

System.out.println(" Manufacturer: " + gateway.getManufacturer());

System.out.println(" Model: " + gateway.getModel());

System.out.println(" Serial No: " + gateway.getSerialNo());

System.out.println(" SIM IMSI: " + gateway.getImsi());

System.out.println(" Signal Level: " + gateway.getSignalLevel() + " dBm");

System.out.println(" Battery Level: " + gateway.getBatteryLevel() + "%");

System.out.println();

// Send a message synchronously.

OutboundMessage msg = new OutboundMessage("+254729169305", "pay fine of ksh
10,000");

Service.getInstance().sendMessage(msg);

System.out.println(msg);

System.out.println("Now Sleeping - Hit <enter> to terminate.");

System.in.read();

Service.getInstance().stopService();

}

```

```

public class OutboundNotification implements IOutboundMessageNotification
{
    public void process(AGateway gateway, OutboundMessage msg)
    {
        System.out.println("Outbound handler called from Gateway: " +
gateway.getGatewayId());
        System.out.println(msg);
    }
}

public static void main(String args[])
{
    SendMessage app = new SendMessage();
    try
    {
        app.doIt();
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
}
}

```

### 6.1.1 Time frame

TASK	JUNE	JULY	AUGUST	SEPTEMBER	OCTOBER	NOVEMBER	DECEMBER
Research							
Literature review							
Mini presentation							

Hardware assembly and software design						
Final documentation						

### 6.1.2 Budget

ITEM	COST
GPS receiver	6,200
GPS antenna	1,500
Arduino mega	3,760
SIM 900	9,500
TOTAL	20,960

### 6.2 References

1. [www.ecse.rpi.edu/Homepages/wrf/Research/Short\\_Notes/pnpoly](http://www.ecse.rpi.edu/Homepages/wrf/Research/Short_Notes/pnpoly).
2. [www.arduinoforum.com](http://www.arduinoforum.com)
3. [www.java/forum/tutorials](http://www.java/forum/tutorials)
4. SIM900\_AT Command Manual\_V1.03
5. EB-365 user manual V1\_2010719
6. gprs-shield-sld33149p manual