Programming Arduino with GSM Module

A **GSM Module** is basically a GSM Modem (like SIM 900) connected to a PCB with different types of output taken from the board – say TTL Output (for Arduino, 8051 and other microcontrollers) and RS232 Output to interface directly with a PC (personal computer). The board will also have pins or provisions to attach mic and speaker, to take out +5V or other values of power and ground connections. These type of provisions vary with different modules.

1. We use SIM900 GSM Module – This means the module supports communication in 900MHz band. We are from India and most of the mobile network providers in this country operate in the 900Mhz band. If you are from another country, you have to check the mobile network band in your area. A majority of **United States** mobile networks operate in 850Mhz band (the band is either 850Mhz or 1900Mhz). **Canada** operates primarily on 1900 Mhz band.

2. Check the power requirements of GSM module – GSM modules are manufactured by different companies. They all have different input power supply specs. You need to double check your GSM modules power requirements. In this tutorial, our gsm module requires a 12 volts input. So we feed it using a 12V,1A DC power supply. I have seen gsm modules which require 15 volts and some other types which needs only 5 volts input. They differ with manufacturers. If you are having a 5V module, you can power it directly from Arduino's 5V out.

Steps for BOOTING GSM Module:

- Insert SIM in allocated slot
- Power on GSM modem with 12V power supply
- Check the status LED, It should blink once in 3 seconds
- Call to the number of the SIM inserted, you should get callback tone

Steps for connecting GSM Module to arduino:

There are two ways of connecting GSM module to arduino. In any case, the communication between Arduino and GSM module is serial. So we are supposed to use serial pins of Arduino (Rx and Tx). So if you are going with this method, you may connect the Tx pin of GSM module to Rx pin of Arduino and Rx pin of GSM module to Tx pin of Arduino. You read it right ? **GSM Tx -> Arduino Rx** and **GSM Rx -> Arduino Tx**. Now connect the ground pin of arduino to ground pin of gsm module! So that's all! You made 3 connections and the wiring is over! Now you can load different programs to communicate with gsm module and make it work.

Note:- The problem with this connection is that, while programming Arduino uses serial ports to load program from the Arduino IDE. If these pins are used in wiring, the program will not be loaded successfully to Arduino. So you have to disconnect wiring in Rx and Tx each time you burn the program to arduino. Once the program is loaded successfully, you can reconnect these pins and have the system working!

To avoid this difficulty, We are using an alternate method in which two digital pins of arduino are used for serial communication. We need to select two **PWM enabled pins of arduino** for this method. So I choose pins **2** and **3** (which are PWM enabled pins). This method is made possible with the SoftwareSerial Library of Ardunio. SoftwareSerial is a library of Arduino which enables serial data communication through other digital pins of Arduino. The library replicates hardware functions and handles the task of serial communication

- **GSM Tx –>Arduino Rx (Here pin 2)**
- **GSM Rx –>ArduinoTx. (Here pin 3)**
- **Make the ground common between Arduino and GSM modem.**

Programs

1. GSM Module: Call to a particular number

Aim: Call using Arduino and GSM Module – to a specified mobile number inside the program.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);  // (Rx, Tx)

void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
```

```
Serial.println("CALLING.....");
cell.println("ATD+9538433364;");   // ATD – Attention Dial
delay(20000);
}
void loop() {

}
```

2. Call to a particular numberon an alert

Aim: Call a specified mobile number mentioned in the program using Arduino and GSM Module when a flame sensor detects “fire”.

Connections for flame sensor:

Arduino	**Flame Sensor**
5V	VCC
GND	GND
A0	A0

Program:

```
#include <SoftwareSerial.h>
SoftwareSerialcell(2,3);
void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
}
void loop() {
intval=analogRead(A0);
Serial.println(val);
delay(1000);
if (val<50)
  {
Serial.println("CALLING.....");
cell.println("ATD+919742980606;");
delay(10000);
cell.println("ATH");  // Attention Hook Control
   }
}
```

3. Sending and Receiving Message

Aim:

1) Send SMS using Arduino and GSM Module – to a specified mobile number inside the program

2) Receive SMS using Arduino and GSM Module – to the SIM card loaded in the GSM Module.

Program:

Note: According to the code, message will be sent and received when 's' and 'r' are pressed through serial monitor respectively.

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

void setup()
{
mySerial.begin(9600);   // Setting the baud rate of GSM Module
Serial.begin(9600);    // Setting the baud rate of Serial Monitor (Arduino)
delay(100);
}


void loop()
{
if (Serial.available()>0)
switch(Serial.read())
  {
case 's':
SendMessage();
break;
case 'r':
RecieveMessage();
break;
  }

if (mySerial.available()>0)
Serial.write(mySerial.read());
}

voidSendMessage()
{
```

```
mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode //AT+CMGF, SMS Format
delay(1000);  // Delay of 1000 milli seconds or 1 second
mySerial.println("AT+CMGS=\"+919742980606\"\r");  // AT+CMGS, Send Message
// Replace  withyour mobile number
delay(1000);
mySerial.println("I am SMS from GSM Module");
// The SMS text you want to send
delay(100);
mySerial.println((char)26);// ASCII code of CTRL+Z , to terminate the message
delay(1000);
}

voidRecieveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0");  // AT+CNMI, New Message Indications
// AT Command to recieve a live SMS
delay(1000);
 }
```

4. Controlling LED through received messages:

Aim:

Use received message throughArduino and GSM Module to control Switching ON / OFF the LED.

Connection: Attach LED to pin 13 and GND.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);

voidreadfn()
  {
if (cell.available()) {
while (cell.available()) {
Serial.write(cell.read());
    }
     }
  }

void setup() {
pinMode(13,OUTPUT);
Serial.begin(9600);
```

```
cell.begin(9600);
cell.println("AT");
delay(1000);
readfn();
  //New SMS alert
cell.println("AT+CNMI=1,2,0,0,0");
 }


void loop() {
if(cell.available())
 {
  String message =cell.readString();
Serial.println(message);
if(message.indexOf("SWITCH ON")>0)
  {
digitalWrite(13,HIGH);
  }
else if(message.indexOf("SWITCH OFF")>0)
  {
digitalWrite(13,LOW);
  }
else
  {
Serial.println ("Nothing to do...");
  }
}
}
```