ISLAMIC UNIVERSITY OF TECHNOLOGY, IUT

ORGANIZATION OF THE ISLAMIC COOPERATION, OIC

HIGHER DIPLOMA IN ELECTRICAL AND ELECTRONICS ENGINEERING (HDEE)

GSM BASED HOME AUTOMATION

A THESIS PRESENTED TO

ELECTRICAL AND ELECTRONIC ENGINEERING DEPARTMENT

BY

AHMADOU KARIMOU (122317)
FARHAD (122309)
NAJIBULLAH (122307)

SUPERVISED
BY
ASSISTANT PROF: GOLAM SAROWAR
ELECTRICAL AND ELECTRONIC ENGINEERING DEPARTMENT, IUT.

NOVEMBER 2015

Certification of Approval

Certified that the project entitled

"GSM BASED HOME AUTOMATION"

Is submitted to the department of Electrical and Electronics Engineering

Islamic University of Technology (IUT) Dhaka

(Bangladesh)

In the partial fulfillment of the requirements for the degree of

Higher Diploma in Electrical and Electronics Engineering (HDEE)

SUPERVISED BY:

Sir Golam sarowar assistant professor

Project supervisor

Department of Electrical and Electronics Engineering (EEE)

…………………………………………………………………………………………………….

Date ………………………………………………………………………………………..

Signature of the head of department

…………………………………………………………………………………………………….

Prof. Dr. Md. Shahidullah

Head:

Date ……………………………………………………………………………

Ahmadou karimou (122317)…………………………………………………………..

Farhad (122309)…………………………………………………………………………………

Najibullah (122307) ……………………………………………………………………….

Acknowledgement

First and foremost, we would like to thank the all MIGHTY ALLAH for making it possible for us to be able to complete this project. We would like to thank our supervisor of this project, assistant prof**. Golam Sarowar** for the valuable guidance and advice. He inspired us greatly to work in this project. His willingness to motivate us contributed tremendously to our project. We also would like to thank him for showing us some example that related to the topic of our project. Besides, we would like to thank the authority of Islamic University of Technology Dhaka Bangladesh for providing us with a good environment and facilities for our studies. Also, we would like to take this opportunity to thank the head of the department of electrical and electronics engineering (EEE) for letting us do this project. It gave us an opportunity to participate and learn about the operation of microcontrollers and the GSM communication. Finally, an honorable mention goes to our families and for their understandings of letting us study abroad for more knowledge. Without helps of the particular that mentioned above, we would face many difficulties while doing this project

Dedicated to my dear parents for the love and support they have given me throughout my studies

ABSTRACT

With the increase in consumption of energy and population, there is a grave need to conserve energy in every way possible. The inability to access and control the appliances from remote locations is one of the major reasons for energy loss. A web or an android application is used by the users to give instructions to these systems. This system can make use of a host of communication methods such as Wi-Fi, GSM, Bluetooth, ZigBee. Different controlling devices and configurations can be found in existing systems. Such systems have been found already in many places for a wide variety of applications. This paper presents a survey of all such systems.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Electronic and Electrical environment with respect to this context is any environment which consists of appliances such as fans, television sets, air conditioners, motors, heater, lighting systems, etc. A remotely accessible environment is an environment in which each appliance can be remotely accessed and controlled using software as an interface, which includes an Android application and a Web application. Such remotely accessible systems are already available in the market, but have a number of drawbacks as well. This paper aims to perform a survey of all the existing such systems and compare the available features.

The paper will also compare and contrast all the systems and look at their various features and disadvantages. A wide variety of options are available for the home automation systems. All of these will be examined at length.

METHODOLOGIES

GSM based Home Automation System

The system proposed provides the means to control the home: the GSM network through SMS. The real time monitoring has been an important feature that can be used in the home automation systems. As a change in the status of the devices occurs, the user can be informed in real time. The user commands are transferred to a server which is usually done by a PC. The server processes the user commands and sends them to the relevant units. This can help control the appliances. GSM is used as a communication medium to help establish connection in places where there may not be proper internet connectivity. The server uses AT commands to communicate with the GSM modem. The mobile interface is developed using J2ME. The server has 4 engines running. The system can be controlled using SMS. It can send confirmation messages. Each application node has four parts – the transmitter, receiver, I/O device from the Arduino mega2560 to the GSM. The main control program in the server takes status information from the devices transceiver in real time.

The system makes use of the command pins of the Arduino pins according to the interface for home appliances control. It makes use of GSM for control of the appliances. This is an SMS based system. GSM has been used due to its high availability, coverage and security. The control of home appliances is done primarily through SMS codes. AT commands can be sent through the GSM network and this controls the home devices. Messages are sent by the device to the user through SMS as well most probably the android SMS applications like that which we used for this project. This system can however incur additional costs for the SMS. There is no UI that the user can use to control the device. This system has the drawback of not being able to program the devices. Also SMS depends on the networks and there is a possibility of delayed delivery. The system does not have any state information related to the devices and expects the user to keep track of it.

The system is described as an M2M system. It uses GSM for communication. GSM offers options for M2M which include Dual Tone Multi Frequency (DTMF), SMS and General Packet Radio Service (GPRS). This system chooses to use the SMS along with AT (attention) commands. It has a PC as a center of commands. A GSM dial-up and communication system is embedded in the PC. Visual C++ is used for implementation. The PC decodes the received messages through SMS and performs the commands required. It is a system that can be programmed for the required application as per requirements. The system also has the ability to control mechanical appliances, through sensors that convert electrical to mechanical signals. However, this system is not designed to provide feedback to the user which is seen as one of its drawback. It cannot be used as a real time control system.

The home server is built upon a SMS/GPRS mobile cell module and a microcontroller. This allows the user to monitor and control any appliances at home using any Java enabled cell phone. The project presents the design and implementation of AT modem driver, text based command processing software and power failure resilient output from the microcontroller. The mobile phones are used to provide a user friendly interface. They also serve to send commands and receive feedback from the system as SMS strings. The hardware components include an Atmel microcontroller, which is connected with a 4-channel relay module. A message format based authentication system has to be used. The text messages sent will contain the format which is used to ensure the message is sent from a valid source. The main drawback of this system is that it relies heavily on the SMS, which is not very fast and dependable. There can be delays in delivery. Also security of the system is compromised since message commands are sent freely over the network. A system based on GSM network through SMS is used to control the home appliances as shown in the subsequent figure. An Arduino board is the controller used to interface the appliances. It uses certain peripheral drivers and relays to achieve this interfacing. The smart phone is the user interface device. The system uses the ‘App Inventor’ visual programming tool to develop the interface and other tools to deploy the app. The app generates SMS messages based on the user commands and sends it to the GSM modem attached to the Arduino. This allows the user to control the home appliances. The system suffers from the same drawbacks of cost and reliability of network strength. Also the interface is preprogrammed and cannot be customized based on devices.

The primary communication means is through GSM. An Android mobile phone is used to get the message commands and converts them into text. This is sent through SMS to the GSM network. This module is connected to a microchip controller. This controller interprets the commands and performs the appropriate actions. The control of electrical circuits is done with a separated system, to isolate the load from the control circuitry. However the usage of SMS makes it unreliable.

A system uses the GSM network along with a microcontroller. The user enters the commands .These are sent through SMS. However, this system uses codes that are easily interpreted by the microcontroller. This will receive the commands that are sent via SMS. AT commands are used to communicate with the modem. This system has remote access capabilities from all over the world. However it cannot function in real time. A microcontroller along with GSM can form a home automation system. The commands are sent from the mobile phone through GSM to the GSM modem. This command is converted to text and sent to the controller through commanded pins of the microcontroller. These commands are interpreted by the microcontroller and the corresponding action is performed.

CHAPTER 2

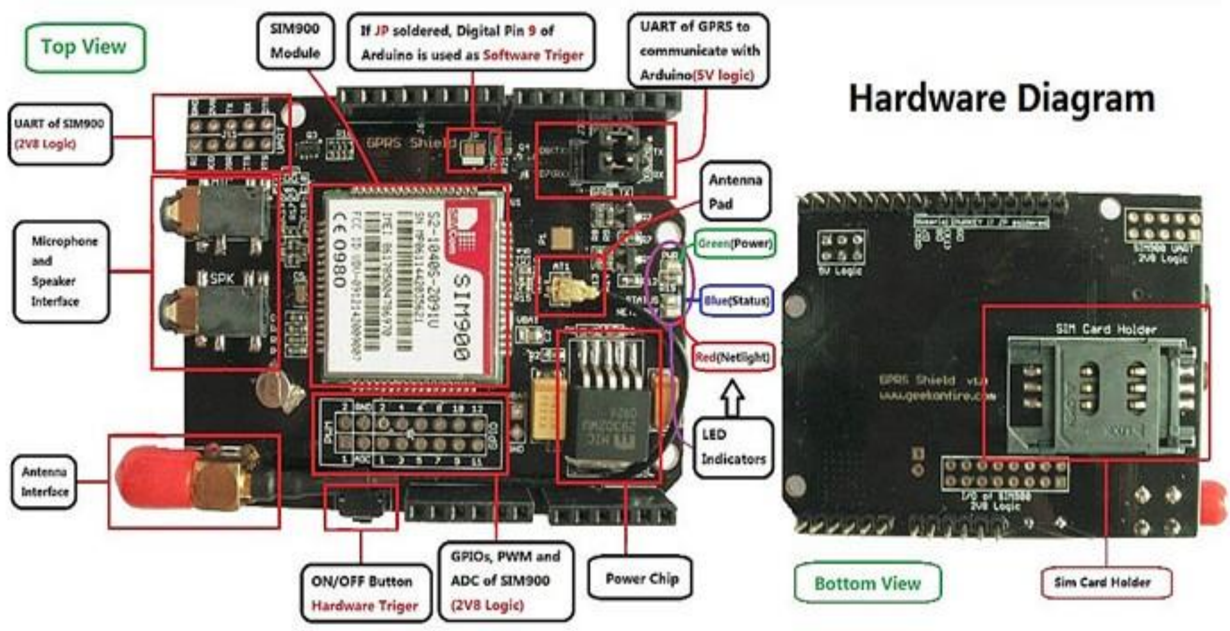
COMPONENTS AND DESIGN

COMPONENTS:-

GSM MODULE:

INTRODUCTION:

A GSM (Global System for Mobile) modem is a special type of modem which accepts a SIM card, and operates over a subscription to a mobile operator, just like a mobile phone. From the mobile operator perspective, a GSM modem is like a mobile phone. A GSM modem is subject to an interface that allows applications to send and receive messages over the modem interface. The mobile operator charges for this message sending and receiving as if it was performed directly on a mobile phone. To perform these tasks, a GSM modem must support an extended AT command set for sending and receiving SMS messages, as defined in the ETSI GSM 07.05 and 3GPP TS 27.005 specifications. GSM modems can be a quick and efficient way to get started with SMS, because a special subscription to an SMS service provider is not required. In most parts of the world, GSM modems are a cost effective solution for receiving SMS messages, because the sender is paying for the message delivery recommended as GSM MODEM.


Top View
SIM900 Module
If JP soldered, Digital Pin 9 of Arduino is used as Software Triger
UART of GPRS to communicate with Arduino(5V logic)
Hardware Diagram
UART of SIM900 (2V8 Logic)
Microphone and Speaker Interface
Antenna Pad
Green(Power)
Blue(Status)
Red(Netlight)
LED Indicators
Antenna Interface
ON/OFF Button Hardware Triger
GPIOs, PWM and ADC of SIM900 (2V8 Logic)
Power Chip
Bottom View
SIM Card Holder
Sim Card Holder

GSM (Global System for Mobile) / GPRS (General Packet Radio Service) TTL –Modem is SIM900 Quad-band GSM / GPRS device, works on frequencies 850 MHZ, 900 MHZ, 1800

MHZ and 1900 MHZ. It is very compact in size and easy to use as plug in GSM Modem. The

Modem is designed with 3V3 and 5V DC TTL interfacing circuitry, which allows User to directly interface with 5V Microcontrollers (PIC, AVR, Arduino, 8051, etc.) as well as 3V3

Microcontrollers (ARM, ARM Cortex XX, etc.). The baud rate can be configurable from 9600-

115200 bps through AT (Attention) commands. This GSM/GPRS TTL Modem has internal

TCP/IP stack to enable User to connect with internet through GPRS feature. It is suitable for SMS as well as DATA transfer application in mobile phone to mobile phone interface.

The modem can be interfaced with a Microcontroller using USART (Universal Synchronous Asynchronous Receiver and Transmitter) feature (serial communication).We shall discuss the various parts of the GSM modem below.

Parts of the GSM modem:

SIMCom SIM900A GSM Module:

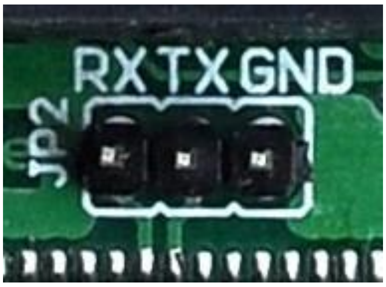
This is actual SIM900 GSM module which is manufactured by SIMCom. Designed for global market, SIM900 is a quad-band GSM/GPRS engine that works on frequencies GSM
850MHz, EGSM 900MHz, DCS 1800MHz and PCS 1900MHz. SIM900 features GPRS multi slot class 10/ class 8 (optional) and supports the GPRS coding schemes CS-1, CS-2, CS-3 and CS-4. With a tiny configuration of 24mm x 24mm x 3mm, SIM900 can meet almost all the space requirements in User's applications, such as M2M, smart phone, PDA and other mobile devices.

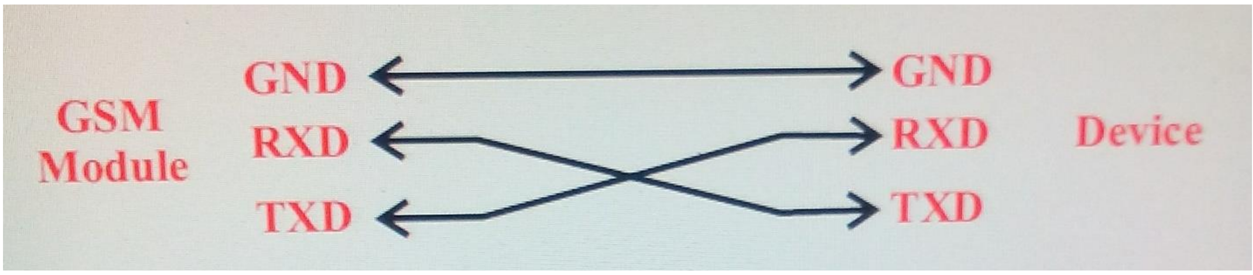
Power Supply Socket:

This power supply socket which actually named as AC/DC Socket provides the functionality to user to connect external power supply from Transformer, Battery or Adapter through DC jack. User can provide maximum of 12V AC/DC power supply through AC/DC socket. This is power supply designed into maximum protection consideration so that it can even prevent reverse polarity DC power supply as well as DC conversion from AC power Supply. It also includes LM317 Voltage Regulator which provides an output voltage adjustable over a1.2V to 37v

RXD, TXD and GND pins (JP2):

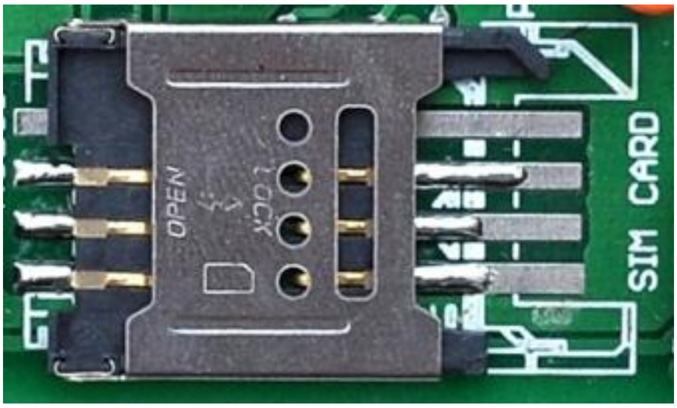
These pins are used to connect devices which needs to be connected to GSM module through USART (Universal Synchronous Asynchronous Receiver and Transmitter) communication. Devices may be like Desktop or Laptop Computer System, Microcontrollers, etc. RXD (Receive Data) should be connected to TXD (Transmit Data) of other device and
Vice versa, whereas GND (Ground) should be connected to other device's GND pin to make ground common for both systems.





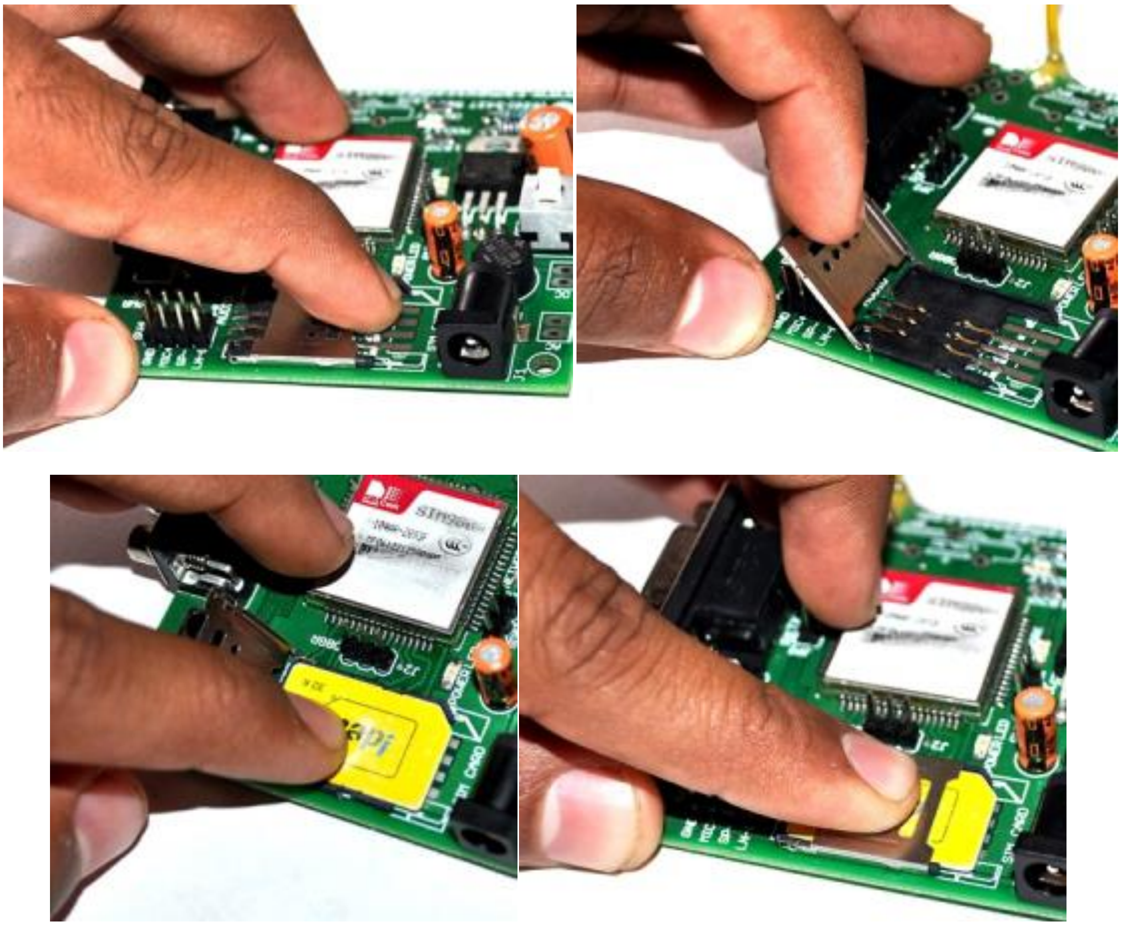
SIM (Subscriber Identity Module) Card Slot:

This onboard SIM card slot provide User functionality of insert a SIM (GSM only) card of any service provider. Process of inserting and locking SIM card into SIM card slot is given in this manual. While inserting in and removing out SIM card from SIM card slot, User needs to take precaution that power supply should be OFF so that after making Power supply ON it will be easy to reinitialize with SIM for this module



Inserting SIM card into SIM card Slot/Holder:

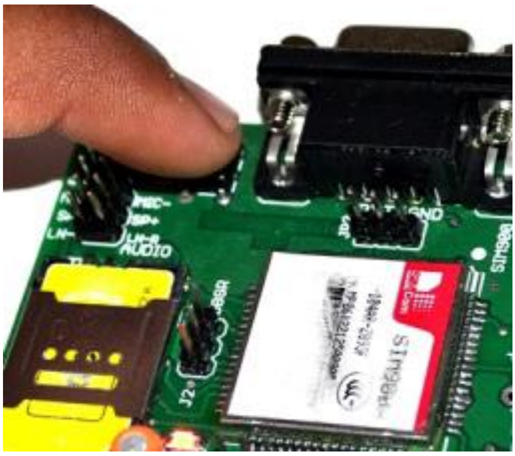
Here is the process how to insert SIM card into SIM card slot. User just need to unlock SIM card cover by sliding back. Then user need to open this cover and insert SIM card according to slot. Put down cover on SIM card and then lock by sliding forward.



Warning: *Be careful about SIM card slot as it is too delicate one while inserting in and removing out SIM card.*

Power On/Off and Module On/Off process:

Here is the process how User should make power supply on/off and module on/off. First of all User need to connect external power supply by using Battery / Adapter / Transformer. Now user needs to press Power On/Off switch (It is push-on push-off switch, thus User need to push it to make power on and push it again to make power supply off). Two LEDs will glow, one is Power On/Off indicator LED and another one is Network Status LED (which glows continuous to indicate no network or searching for network). After this User needs to press Module on switch (denoted as PWR) for at least 2 seconds. As soon as Module On/Off LED will glow User can release this switch, Network LED will blink to indicate signal strength.



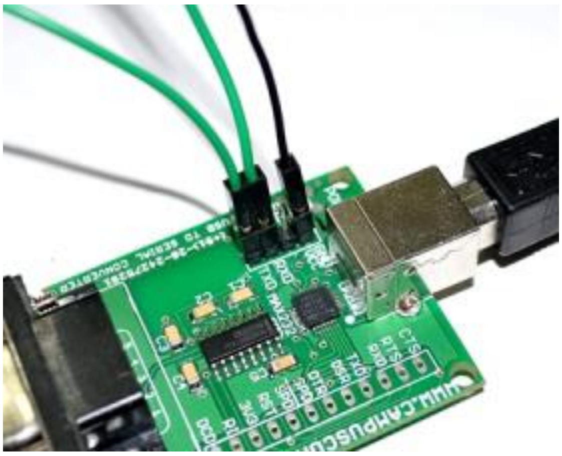
***Note:** If message sending, receiving, calling functions are failed due to weak signal strength then user needs to check SIM card's service provider's availability in coverage area it those not mean that GSM Module is not working properly.*

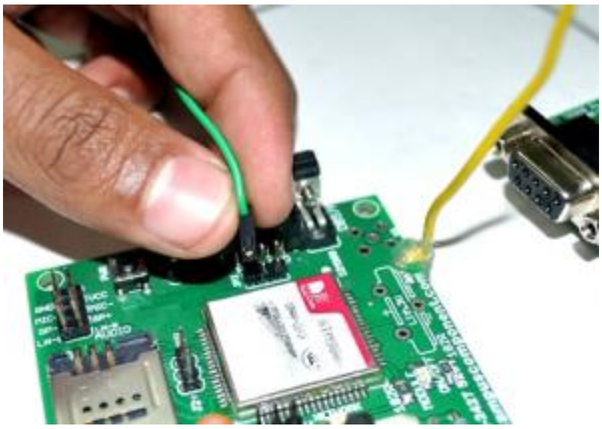
Connecting GSM Module with Serial to USB converter through RXD, TXD and GND:

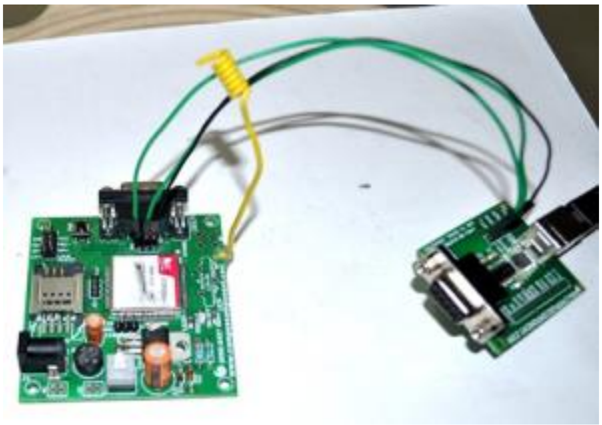
This module is designed in a way so that User can connect this module without Serial cable, this module can be connected to any of Serial to USB converter module or cable. Here we have shown demo how to connect this interfacing board with CP2102 Serial to USB converter
Module through RXD, TXD and GND.

Connect CP2102 Serial to USB converter module to PC through USB cable, connect one end of USB cable to PC's USB connector and connect another end of USB to CP2102 modules USB connector.

Connect three Single Berg Wires to CP2102 module's RXD, TXD and GND pin. Then connect RXD wire to TXD of GSM module and TXD wire to RXD of GSM module. Make GND common by connecting GND wire to GND pin of GSM module.

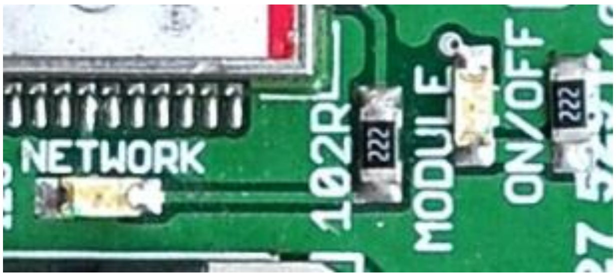






Indicator LEDs:

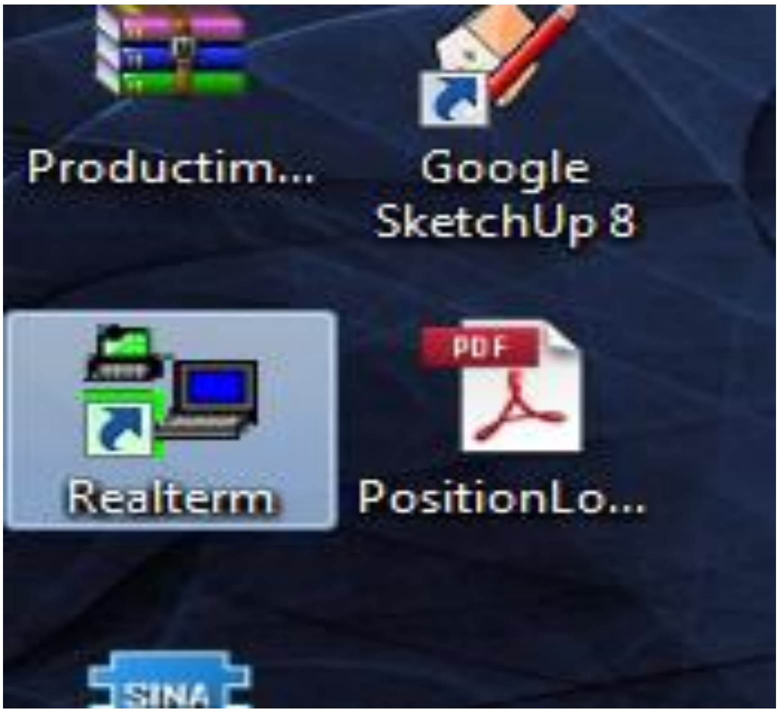
Indicator LEDs just used to indicate status accordingly. These are three LEDs represents Power On/Off Status, Network Status and Module On/Off Status respectively. Power LED will keep on until the power supply is enable to this board by using *push-on push-off* switch. Network Status LED will show whether inserted SIM card successfully connected to service provider's network or not, in short signal strength. Module On/Off indicator LED will show status of GSM module's power on/off.



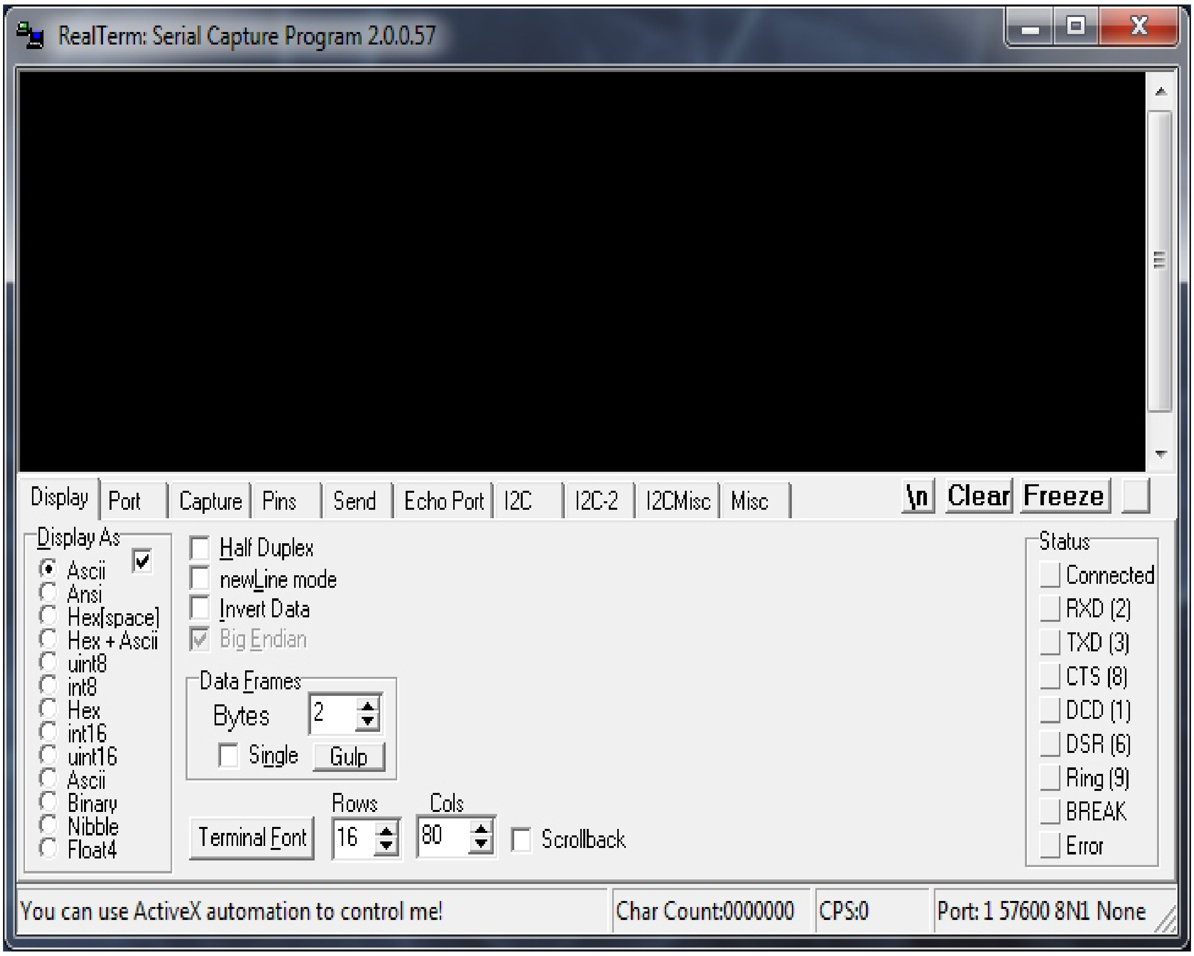
Testing GSM Module on Terminal Software:

Terminal software used share the data through Serial port. Hence here also User need this software to test GSM module. For demo purpose we are going to show demo for how to send

SMS and how to dial a call through *'Real term'* software. First of all install Real term, it will create its own shortcut on Desktop. Then double click on Real term icon.



Windows will appear



Display As
● Ascii ☑
○ Ansi
○ Hex[space]
○ Hex + Ascii
○ uint8
○ int8
○ Hex
○ int16
○ uint16
○ Ascii
○ Binary
○ Nibble
○ Float4

□ Half Duplex
□ newLine mode
□ Invert Data
☑ Big Endian

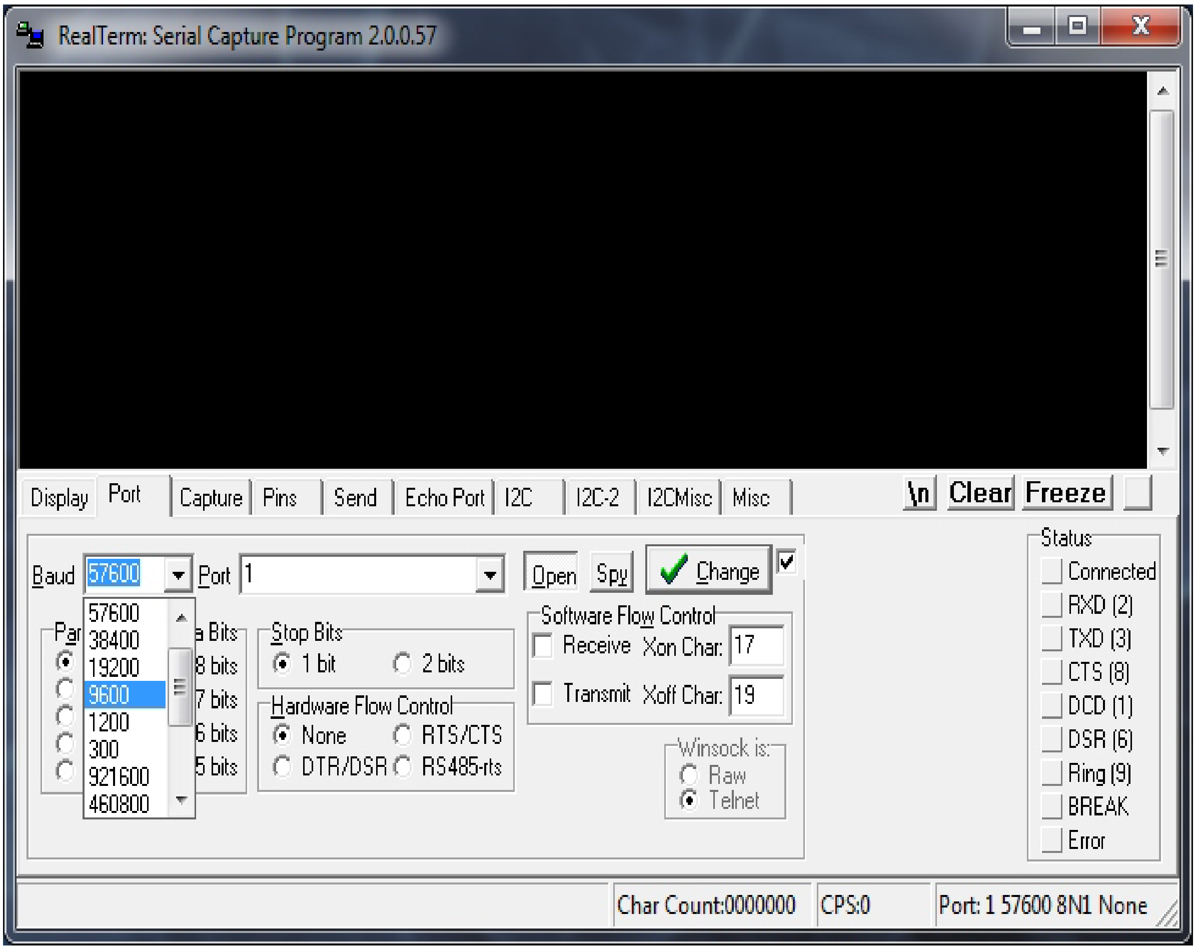
Data Frames
Bytes 2
□ Single

Rows 16
Cols 80
□ Scrollback

Status
□ Connected
□ RXD (2)
□ TXD (3)
□ CTS (8)
□ DCD (1)
□ DSR (6)
□ Ring (9)
□ BREAK
□ Error

You can use ActiveX automation to control me!
Char Count:0000000
CPS:0
Port: 1 57600 8N1 None

Select baud rate as 9600



Baud 57600 Port 1

Par ● ○ ○ ○ ○

Data Bits: 8 bits, 7 bits, 6 bits, 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

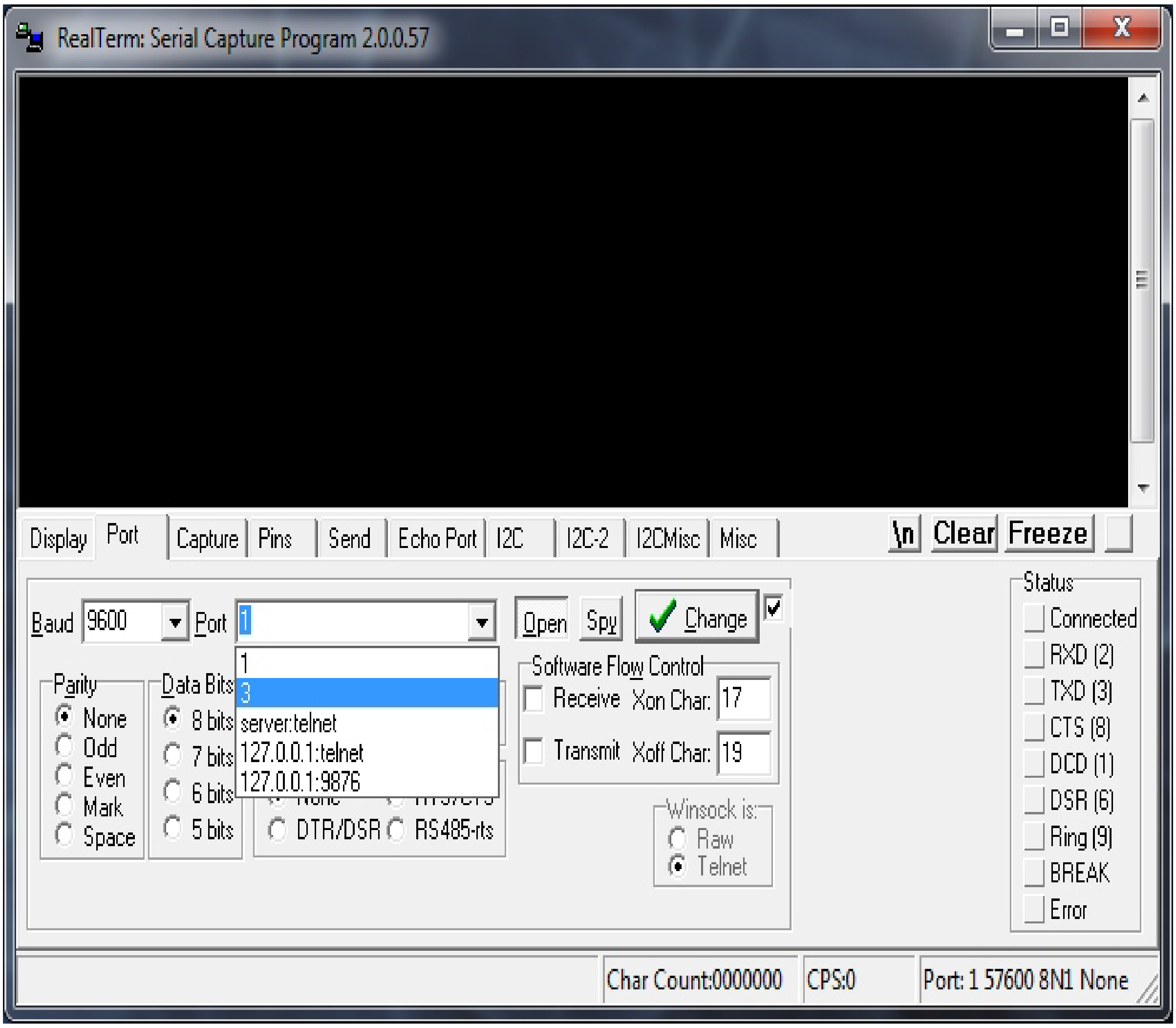
Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

Char Count:0000000 CPS:0 Port: 1 57600 8N1 None

Select COM port through which GSM module is connected. (This can be known from ‘Computer Management > Device Manager’)

Click on ‘Open’ to open the port or to start communication. Click on Change button.



RealTerm: Serial Capture Program 2.0.0.57

Baud 9600 Port 1

1
3
server:telnet
127.0.0.1:telnet
127.0.0.1:9876

Parity
● None
○ Odd
○ Even
○ Mark
○ Space

Data Bits
● 8 bits
○ 7 bits
○ 6 bits
○ 5 bits

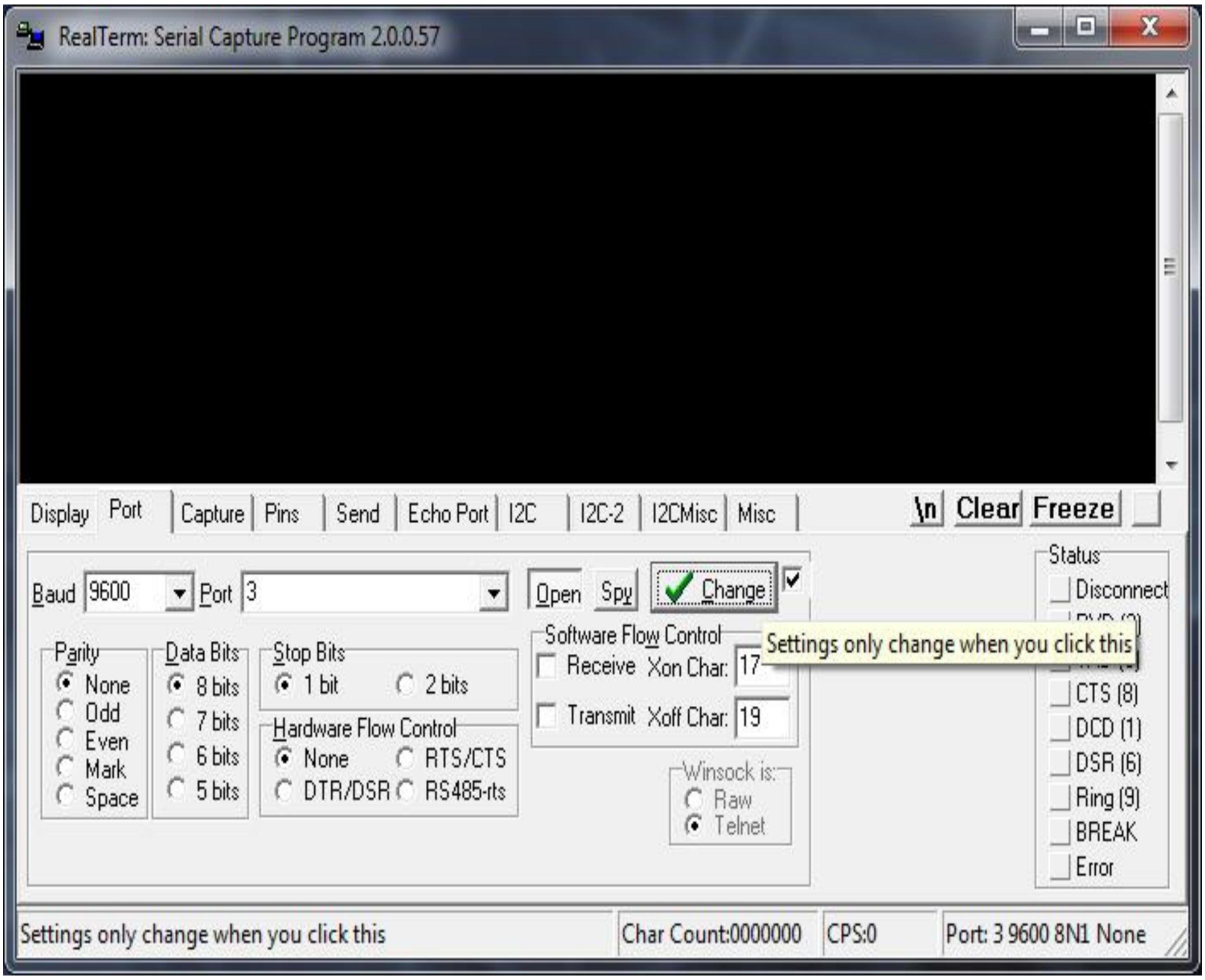
○ DTR/DSR ○ RS485-rts

Software Flow Control
☐ Receive Xon Char: 17
☐ Transmit Xoff Char: 19

Winsock is:
○ Raw
● Telnet

Status
☐ Connected
☐ RXD (2)
☐ TXD (3)
☐ CTS (8)
☐ DCD (1)
☐ DSR (6)
☐ Ring (9)
☐ BREAK
☐ Error

Char Count:0000000 CPS:0 Port: 1 57600 8N1 None



Baud 9600 Port 3

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

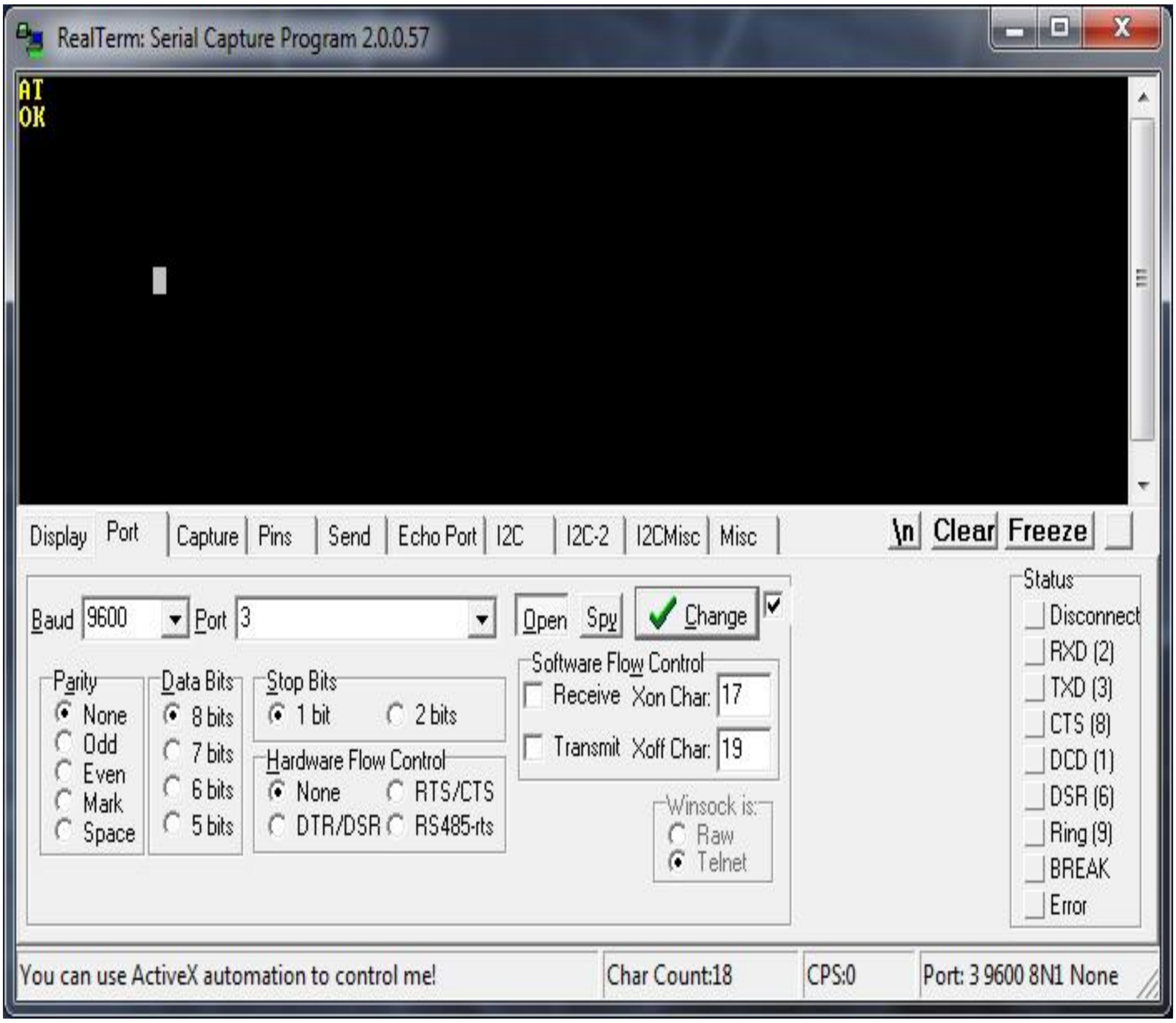
Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

Settings only change when you click this

Settings only change when you click this | Char Count:0000000 | CPS:0 | Port: 3 9600 8N1 None

To initialize GSM module type ‘AT’ in capital letters and then press enter key on keyboard. If there are no errors occur in this communication then it will send ‘OK’.



```
AT
OK
```

Baud 9600 Port 3

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

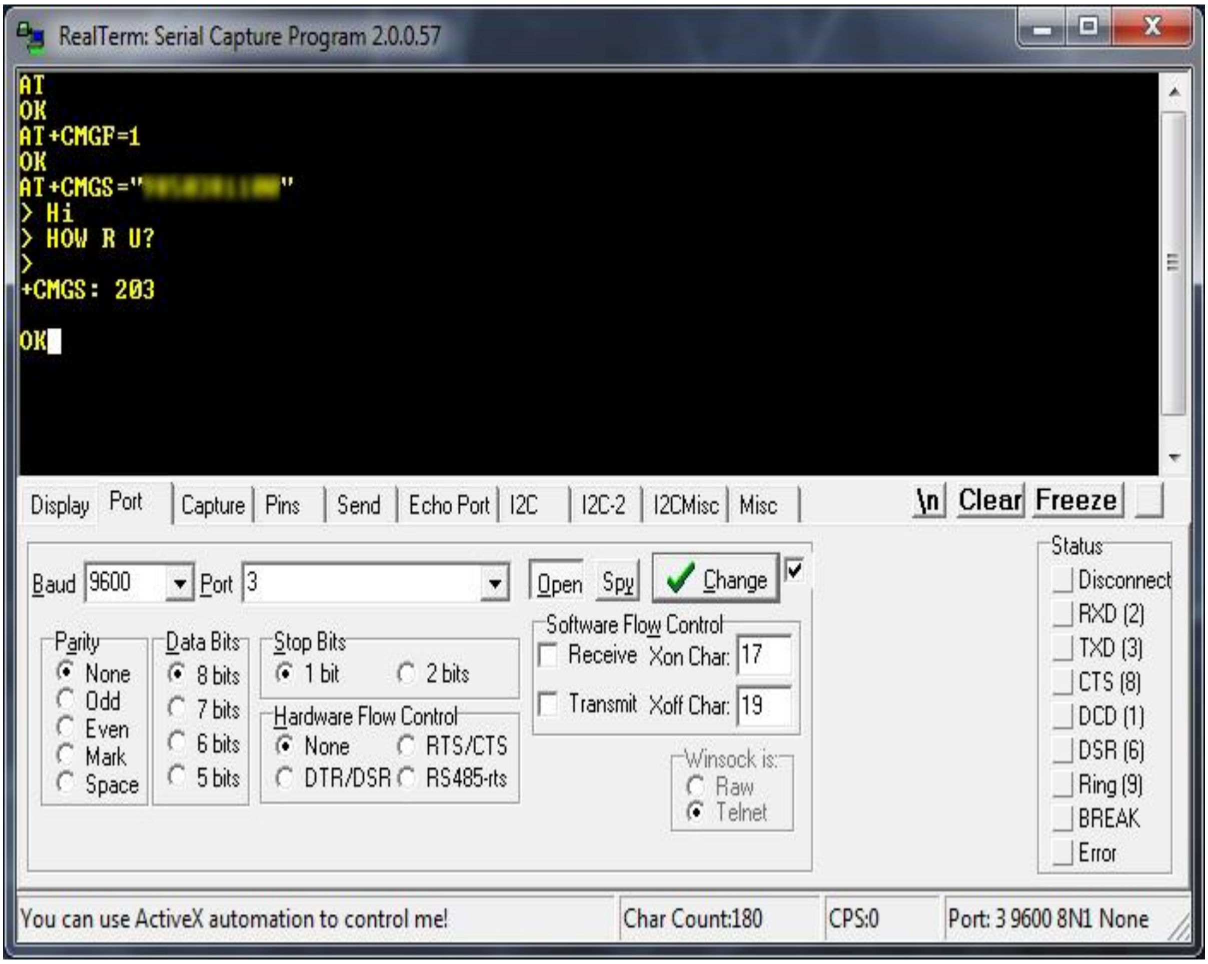
Software Flow Control: ☐ Receive Xon Char: 17 ☐ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

You can use ActiveX automation to control me! | Char Count:18 | CPS:0 | Port: 3 9600 8N1 None

To send SMS type following commands

AT+CMGF=1, press enter key it will return ‘OK’ then proceed, AT+CMGS=“9876543210” <user can put any 10 digit number> type message after getting letter ‘>’ after end of SMS press Ctrl+z on keyboard If it returns ‘+CMGS: 203’ (or any other number) ‘OK’ then SMS sent successfully.



```
AT
OK
AT+CMGF=1
OK
AT+CMGS="[redacted]"
> Hi
> HOW R U?
>
+CMGS: 203

OK
```

Baud 9600 Port 3

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

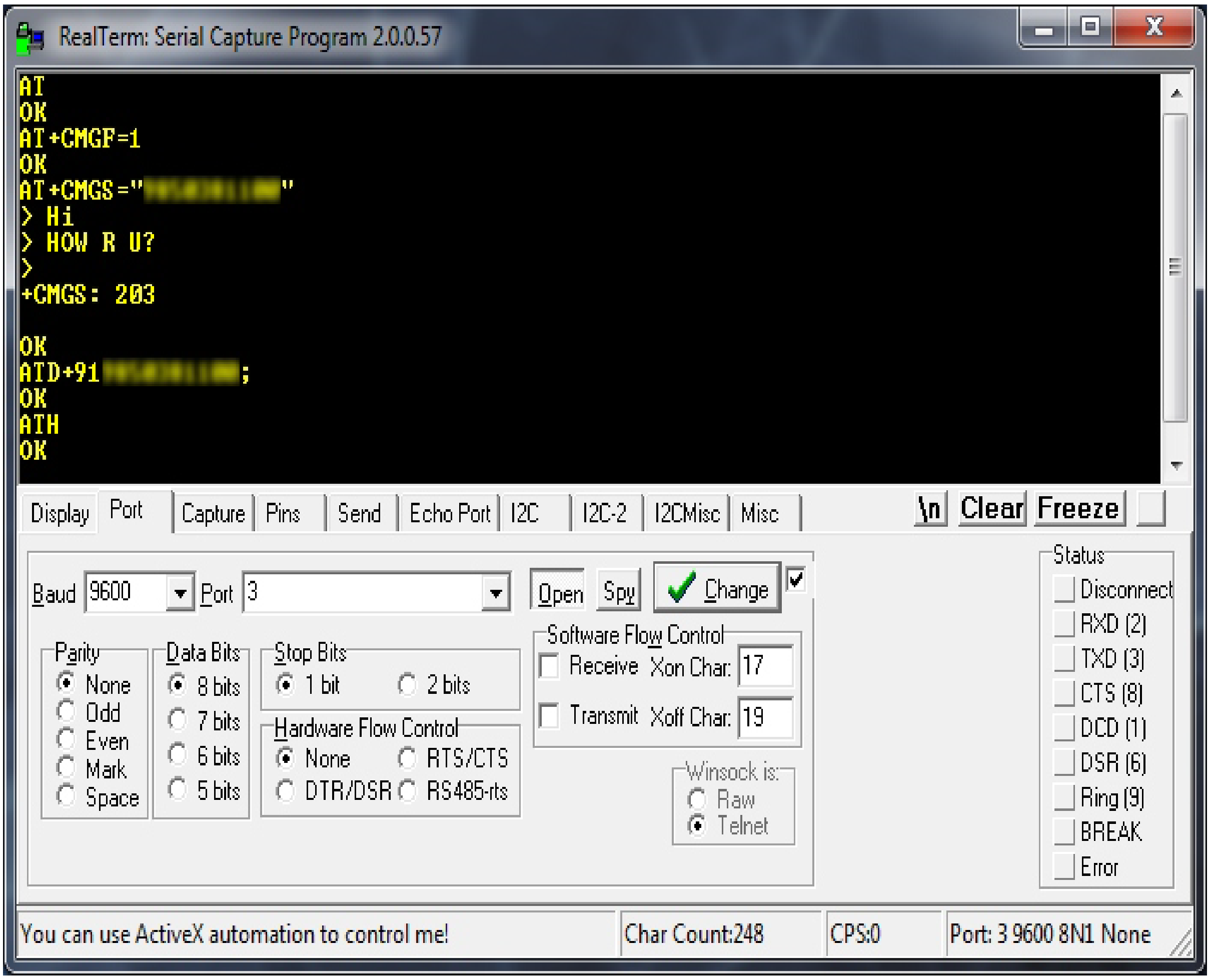
Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

You can use ActiveX automation to control me! Char Count:180 CPS:0 Port: 3 9600 8N1 None

To dial a call type following commands

ATD+919876543210; <user can provide any 10 digit number with country code> it will return ‘OK’ if there are no errors occurs in communication to disconnect type ATH if it returns ‘OK’ it means call disconnected successfully.



RealTerm: Serial Capture Program 2.0.0.57

```
AT
OK
AT+CMGF=1
OK
AT+CMGS="[redacted]"
> Hi
> HOW R U?
>
+CMGS: 203

OK
ATD+91[redacted];
OK
ATH
OK
```

Baud 9600 Port 3

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

You can use ActiveX automation to control me! | Char Count:248 | CPS:0 | Port: 3 9600 8N1 None

Features:

· Quad Band GSM/GPRS: 850 / 900 / 1800 / 1900 MHz

· Built in RS232 to TTL or vice versa Logic Converter (MAX232)

· Configurable Baud Rate

· SMA (Sub Miniature version A) connector with GSM L Type Antenna

· Built in SIM (Subscriber Identity Module) Card holder

· Built in Network Status LED

· Inbuilt Powerful TCP / IP (Transfer Control Protocol / Internet Protocol) stack for internet data transfer through GPRS (General Packet Radio Service)

· Audio Interface Connectors (Audio in and Audio out)

· Most Status and controlling pins are available

· Normal Operation temperature: -20 °C to +55 °C

· Input voltage: 5V to 12V DC

· LDB9 connector (Serial Port) provided for easy interfacing

LIQIUD CRYSTAL DISPLAY (LCD):

Electronic visual display, or video display that uses the light modulate. LCDs do not emit light directly LCDs are used in a wide range of applications including computer monitors, television, instrument panels, aircraft cockpit displays, signage, etc. They are common in consumer devices such as video players, gaming LCD devices, clocks, watches, calculators, and telephones. This LCD's typically have 14 data pins and 2 for the LED backlight. Character LCDs use a standard 14-pin interface and those with backlights have 16 pins. There is also be a single backlight pin, with the other connection via Ground or VCC pin. The two backlight pins may precede the pin 1. The nominal backlight voltage is around 4.2V at 25˚C using a VDD 5V capable model.

Features:

a) 5 ´ 8 and 5 ´ 10 dot matrix possible

b) Low power operation support: 2.7 to 5.5V

c) Wide range of liquid crystal display driver power 3.0 to 11V

d) 80 ´ 8-bit display RAM (80 characters max.)

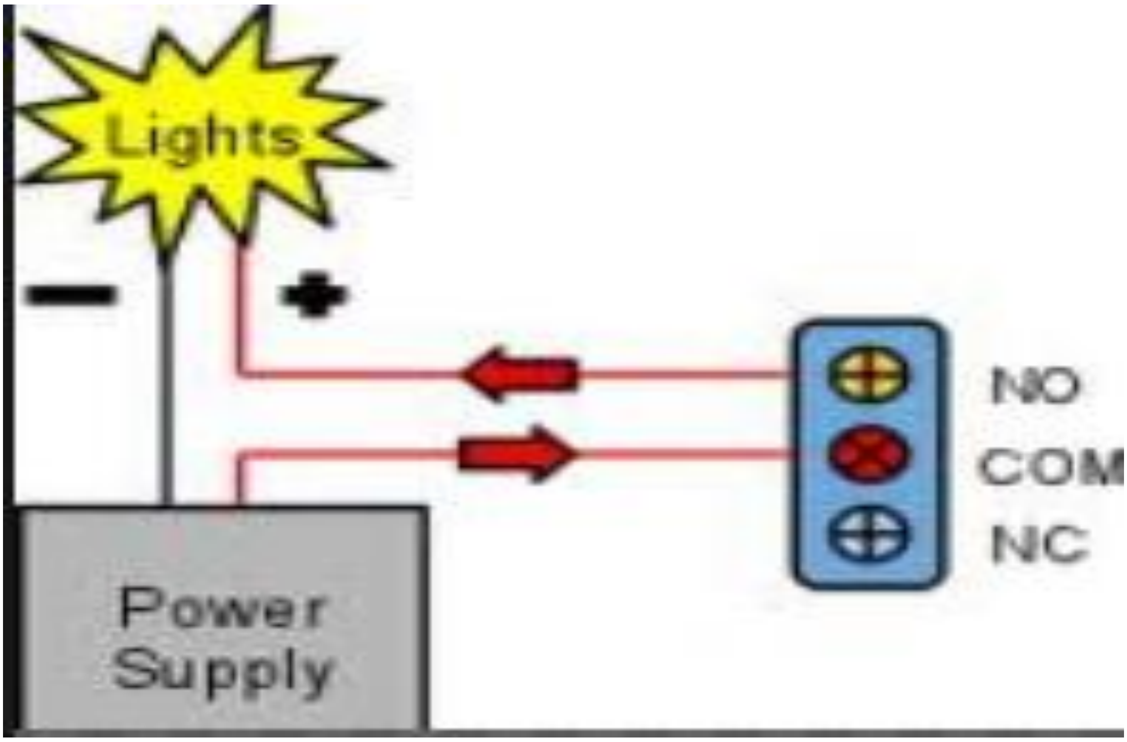
4-CHANNEL 5V DC RELAY MODULE

Relays are electromechanical devices, which are used as a switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The main advantages of using relay as a switch is that, they provide electrical isolation between inputs with the help of magnetic coupling and also we can handle a large power using a relay. There are different types of relays are available and they can be categorized in various ways. For example relays can be SPST, SPDT, DPST, DPDT etc. SPDT stands for Single

Pole A relay will switch one or more poles, each of whose contacts can be thrown by energizing the coil in one of three ways: Normally-open (NO) contacts connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive. It is also called a form

A contact or "make" contact. NO contacts can also be distinguished as "early-make" or NOEM, which means that the contacts will close before the button or switch is fully engaged. Normally-closed (NC) contacts disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive. It is also called a Form B contact or "break" contact. NC contacts can also be distinguished as "late-break" or NCLB, which means that the contacts will stay closed until the button or switch is fully disengaged. Change-over (CO), or double-throw (DT), contacts control two circuits: one normally-open contact and one normally-closed contact with a common terminal. It is also called a Form C contact or "transfer" contact ("break before make"). If this type of contact utilizes a "make before break" functionality, then it is called a Form D contact. The following designations are commonly encountered: SPST – Single Pole

Single throw. These have two terminals which can be connected or Including two for the coil, such a relay has four terminals in total. It is ambiguous whether the pole is normally open or normally closed.



The terminology "SPNO" and "SPNC" is sometimes used to resolve the ambiguity.

A 4-channel relay module if capable of the following

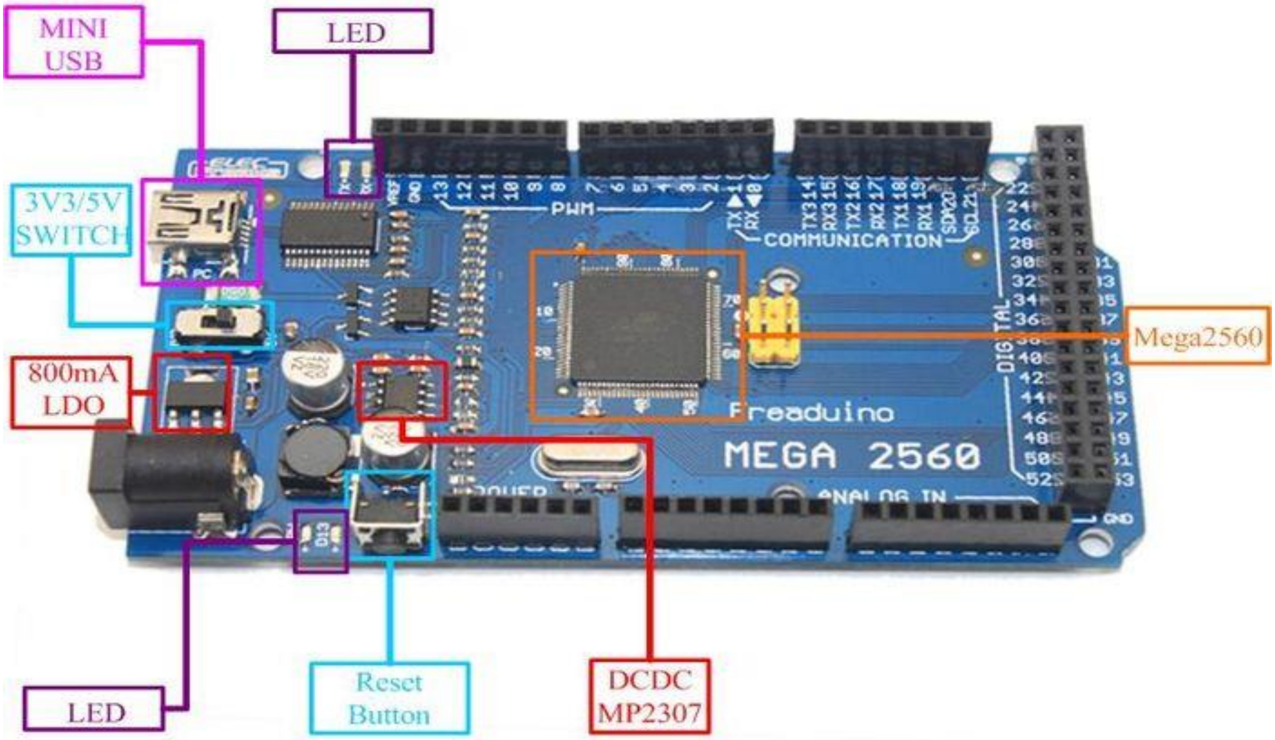
- Capable of switching four appliances at the same time.
- Input operating voltage is 5V DC.
- Output switching voltage: 220vac (10A) & 25vdc (10A)
- Easy to install and user friendly.

ARDUINO MEGA

INTRODUCTION:

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.

This is what the Arduino board look like



The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions.

An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program.

What you need for a working system:

1. Arduino mega board
2. USB programming cable (A to B)
3. 9V battery or external power supply (for stand-alone operation)
4. Solderless breadboard for external circuits, and 22 g solid wire for connections
5. Host PC running the Arduino development environment. Versions exist for Windows, Mac and Linux.

Warning:

Watch the polarity as you connect your battery to the snap as reverse orientation could blow out your board.

Do not put your board down on a conductive surface; you will short out the pins on the back!

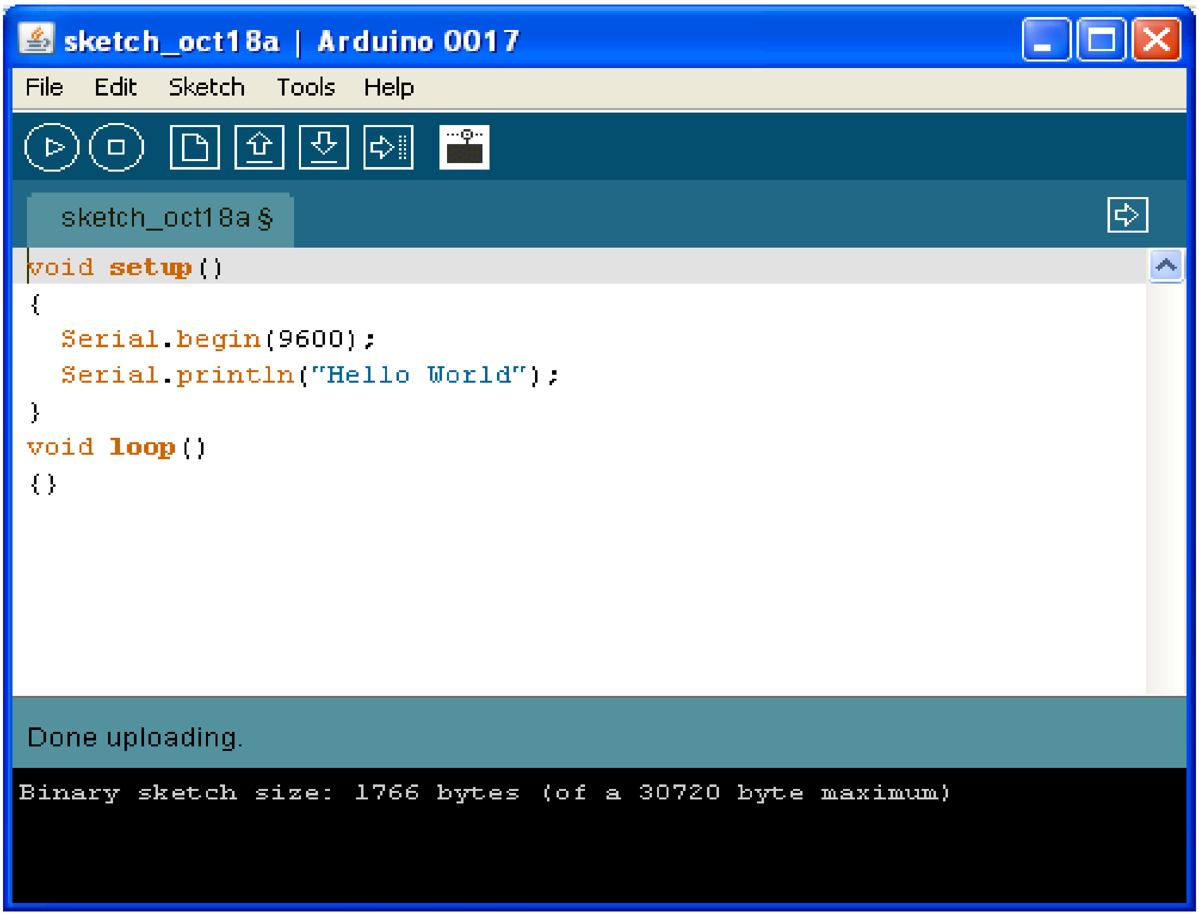
Disconnect your Arduino from the computer. Connect a 9 V battery to the Arduino power jack using the battery snap adapter. Confirm that the blinking program runs. This shows that you can power the Arduino from a battery and that the program you download runs without needing a connection to the host PC

Start the Arduino development environment. In Arduino-speak, programs are called “sketches”, but here we will just call them programs.

In the editing window that comes up, enter the following program, paying attention to where semi-colons appear at the end of command lines.

```
void setup()
{
Serial.begin(9600);
Serial.println("Hello World");
}
void loop()
{}
```

Your window will look something like this



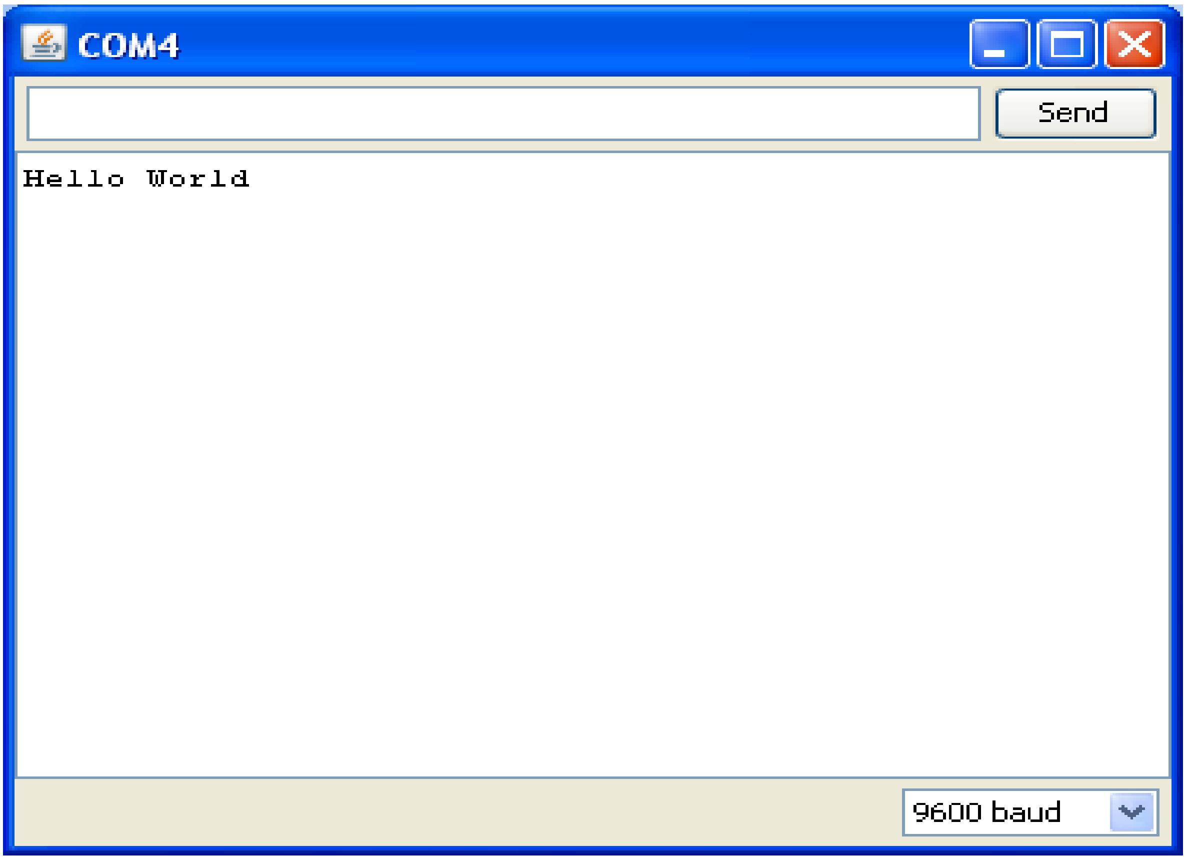
```
void setup()
{
  Serial.begin(9600);
  Serial.println("Hello World");
}
void loop()
{}
```

Done uploading.

```
Binary sketch size: 1766 bytes (of a 30720 byte maximum)
```

Click the Upload button or Ctrl-U to compile the program and load on the Arduino board.

Click the Serial Monitor button. If all has gone well, the monitor window will show your message and look something like this



```
Hello World
```

Congratulations; you have created and run your first Arduino program

Push the Arduino reset button a few times and see what happens.

Hint: If you want to check code syntax without an Arduino board connected, click the Verify button or Ctrl-R.

Hint: If you want to see how much memory your program takes up, Verify then look at the message at the bottom of the programming window.

Troubleshooting:

If there is a syntax error in the program caused by a mistake in typing, an error message will appear in the bottom of the program window. Generally, staring at the error will reveal the problem. If you continue to have problems, try these ideas

- Run the Arduino program again
- Check that the USB cable is secure at both ends.
- Reboot your PC because sometimes the serial port can lock up
- If a “Serial port…already in use” error appears when uploading

Programming Concepts:

This chapter covers some basic concepts of computer programming, going under the assumption that the reader is a complete novice.

A computer program is a sequence of step-by-step instructions for the computer to follow. The computer will do exactly what you tell it to do, no more no less. The computer only knows what's in the program, not what you intended. Thus the origin of the phrase, "Garbage in, garbage out".

The set of valid instructions comes from the particular programming language used. There are many languages, including C, C++, Java, Ada, Lisp, Fortran, Basic, Pascal, Perl, and a thousand others. The Arduino uses a simplified variation of the C programming language.

For any programming language, the instructions must be entered in a specific syntax in order for the computer to interpret them properly. Typically, the interpretation is a tow step process. A compiler takes the language specific text you enter for the program and converts it into a machine readable form that is downloaded into the processor. When the program executes, the processor executes the machine code line by line.

Arduino Programming Language:

The Arduino runs a simplified version of the C programming language, with some extensions for accessing the hardware. In this guide, we will cover the subset of the programming language that is most useful to the novice Arduino designer.

All Arduino instructions are one line. The board can hold a program hundreds of lines long and has space for about 1,000 two-byte variables. The Arduino executes programs at about 300,000 source code lines per sec.

Creating a Program:

Programs are created in the Arduino development environment and then downloaded to the
Arduino board. Code must be entered in the proper syntax which means using valid command names and a valid grammar for each code line. The compiler will catch and flag syntax errors before download. Sometimes the error message can be cryptic and you have to do a bit of hunting because the actual error occurred before what was flagged.

Although your program may pass cleanly through the syntax checker, it still might not do what you wanted it to. Here is where you have to hone your skills at code debugging. The Arduino did what you told it to do rather than what you wanted it to do. The best way to catch these errors is to read the code line by line and be the computer. Having another person go through your code also helps. Skilled debugging takes practice.

Program Structure:

All Arduino programs have two functions, setup () and loop (). The instructions you place in the startup () function are executed once when the program begins and are used to initialize. Use it to set directions of pins or to initialize variables. The instructions placed in loop are executed repeatedly and form the main tasks of the program. Therefore every program has this structure

```
void setup()
{
// commands to initialize go here
}
void loop()
{
// commands to run your machine go here
}
```

The absolute, bare-minimum, do-nothing program that you can compile and run is

```
void setup() {} void loop() {}
```

The program performs no function, but is useful for clearing out any old program. Note that the compiler does not care about line returns, which is why this program works if typed all on one line.

Common Coding Errors:

- Forgetting the semi-colon at the end of a statement
- Misspelling a command
- Omitting opening or closing braces

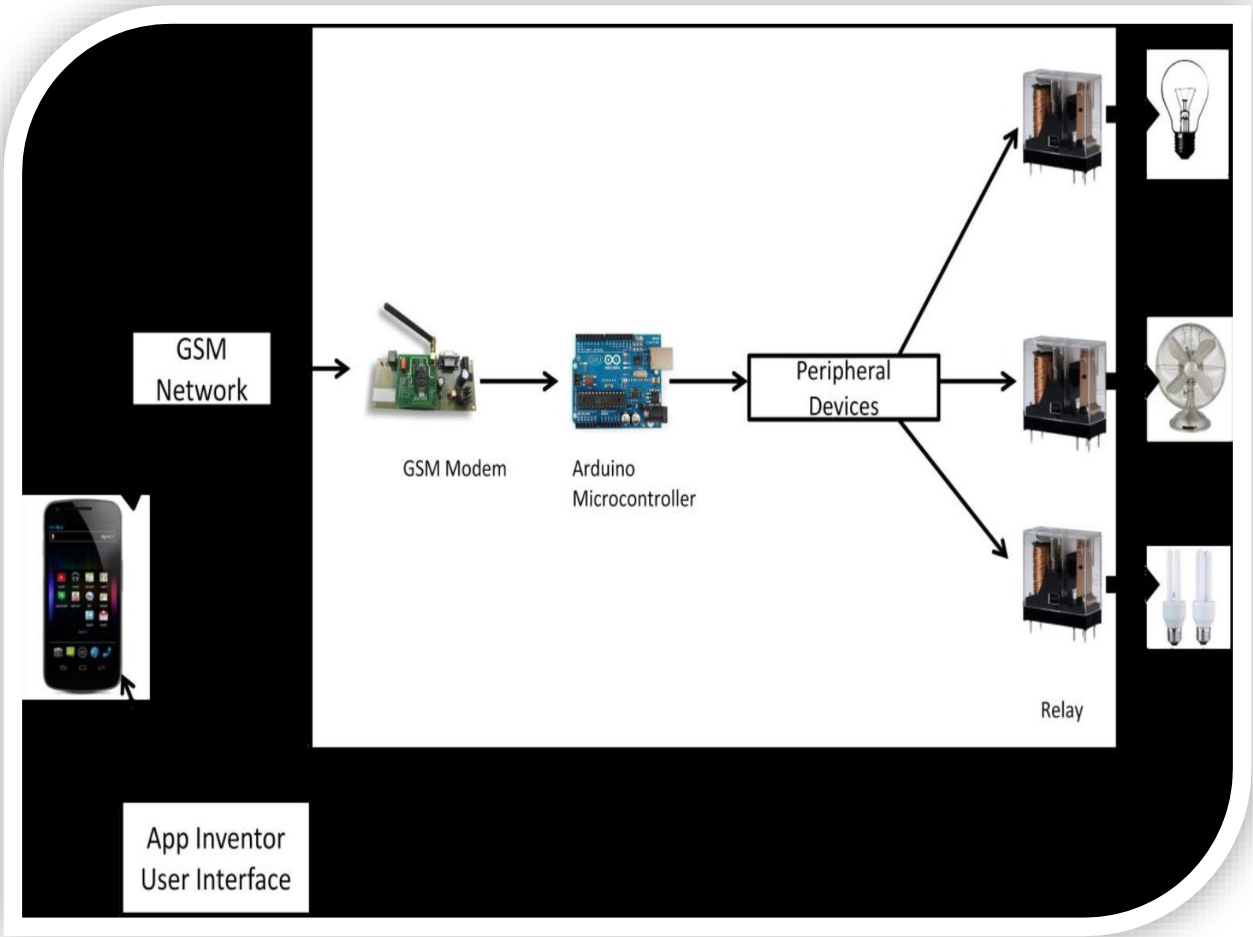
Arduino Programming Basics

COMMAND	**DESCRIPTION**
Pin Mode (n, INPUT)	Set pin *n* to act as an input. One-time command at top of program.
Pin Mode (n, OUTPUT)	Set pin *n* to act as an output
Digital Write (n, HIGH)	Set pin *n* to 5V
Digital Write (n, LOW)	Set pin *n* to 0V
Delay (x)	Pause program for *x* millisec, *x* = 0 to 65,535
Tone (n, f ,d)	Play tone of frequency *f* Hz for *d* millisec on speaker attached to pin *n*
For ( )	Loop. Example: for (i=0;i<3;i++){} Do the instructions

	enclosed by {} three times
If (expr) {}	Conditional branch. If *expr* true, do instructions enclosed by {}
While (expr) {}	While *expr* is true, repeat instructions in {} indefinitely

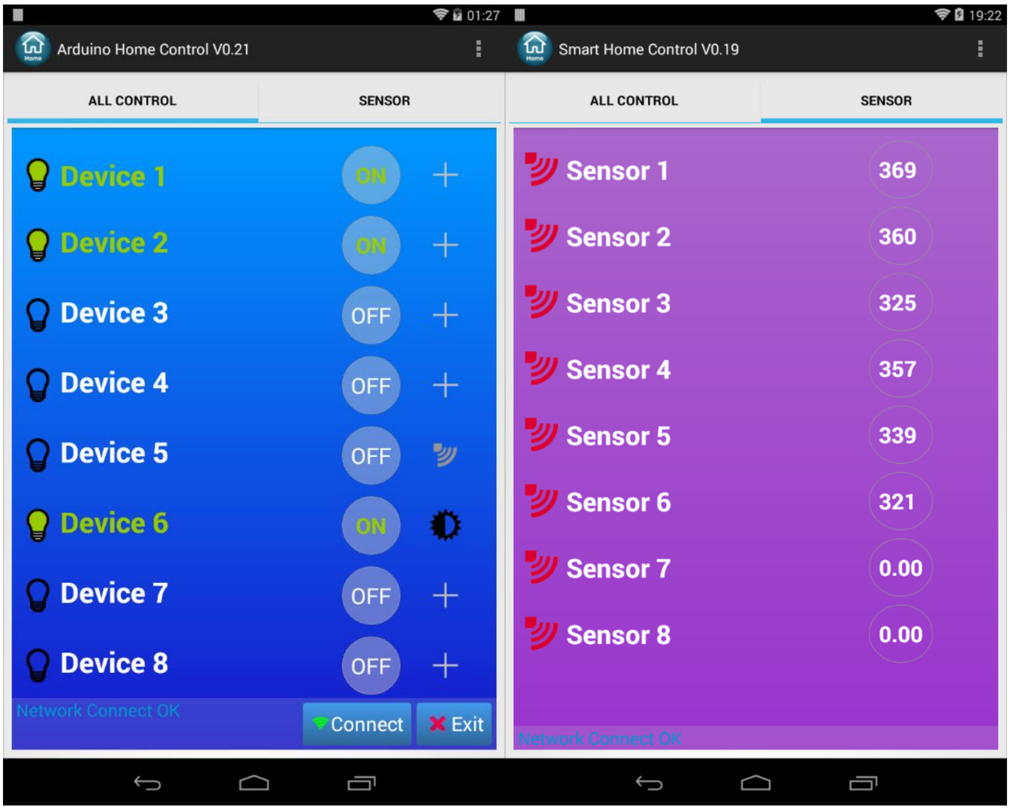
SYSTEM DESIGN:

The core of the home automation system consists of two main hardware components: the GSM sim900 modem and the Arduino mega 2560 microcontroller board which is flexible, inexpensive, offers a variety of digital and analog inputs, serial interface and digital and PWM outputs. It is easy to use, connects to computer via USB and communicates using standard serial protocol as well as the android applications, runs in standalone mode and as interface connected to the GSM modem. Also, it comes with free authoring software. It is an open-source project, software/hardware is extremely accessible and very flexible to be customized and extended. The architecture of the system developed is shown in figure. With the GSM modem platform management and Arduino mega control algorithm that enables the user to access the home appliances through cellular phone using SMS communication. It communicates with the Arduino mega microcontroller board through SMS data transfer mains. A number of appliances and sensors are connected to ports of the Arduino mega board. The home Appliances can be monitored and accessed remotely by user cellular phone.



Android App:

In today's era almost everyone is using Android software which is very helpful and also we can find and use most of the applications for our purpose. For this project we have found one Android app where we can save the command (#axbxcxdx) and other particulars to communicate with GSM modem easily and also in short time. The name of that app. Is ("SMS_RC") which can be easily downloaded from 'Google play store'



01:27

Arduino Home Control V0.21

ALL CONTROL | SENSOR

Device 1 ON

Device 2 ON

Device 3 OFF

Device 4 OFF

Device 5 OFF

Device 6 ON

Device 7 OFF

Device 8 OFF

Network Connect OK

Connect | Exit

19:22

Smart Home Control V0.19

ALL CONTROL | SENSOR

Sensor 1 369

Sensor 2 360

Sensor 3 325

Sensor 4 357

Sensor 5 339

Sensor 6 321

Sensor 7 0.00

Sensor 8 0.00

Network Connect OK

CHAPTER 3

COMPONENTS PART LIST ACTUAL IMAGE AND COST

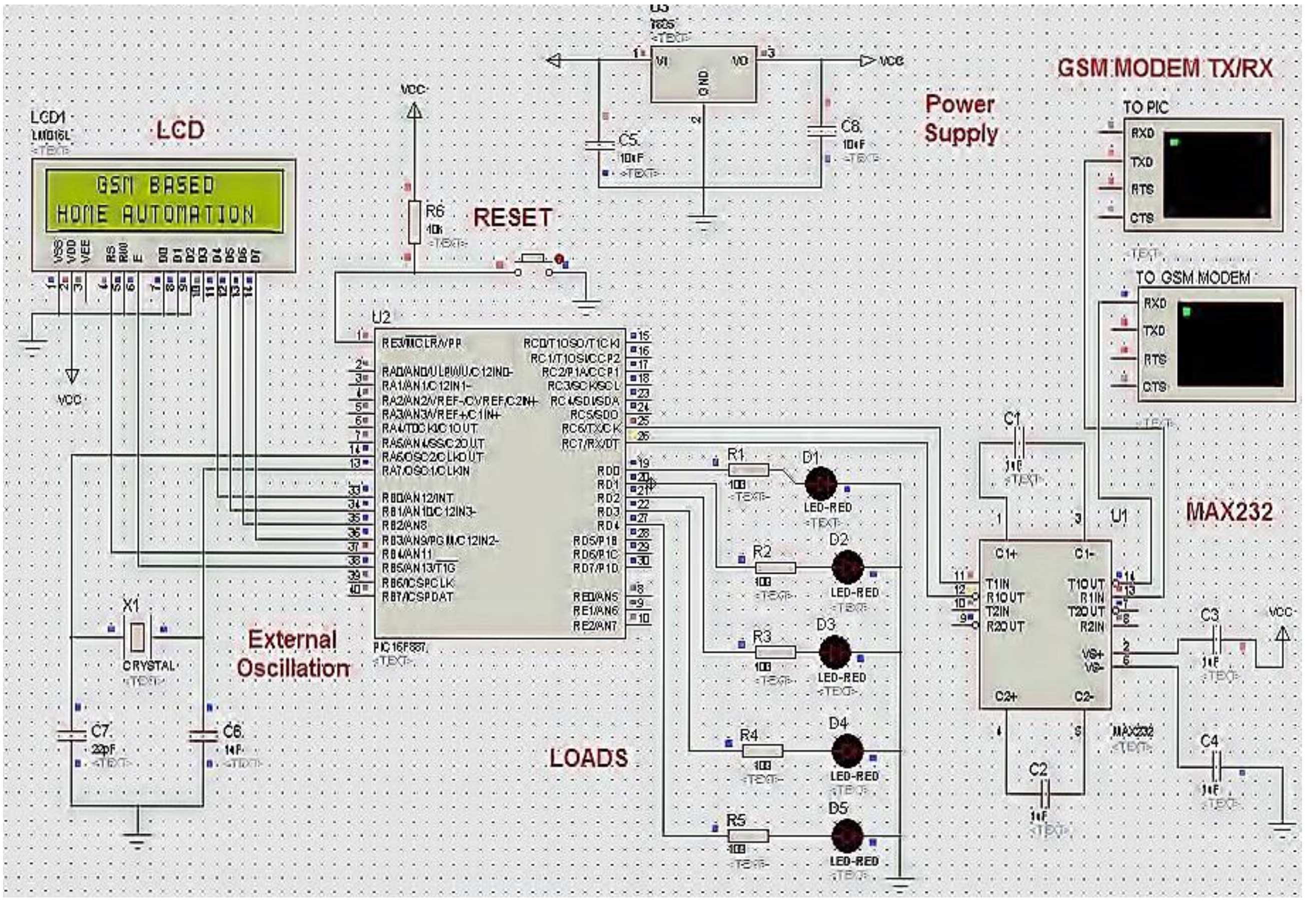
COMPONENT	MODEL	QUANTITY
GSM modem	Sim900A	1
AR DUINO	Mega2560	1
LCD	Shield(16*2)	1
4-CHANNEL REALAY MODULE	JQC-3FC	1
3-12V ADAPTOR	NA-380S	1
SIMCARD		1
BULBS		4

TOTAL COST OF THE PROJECT:

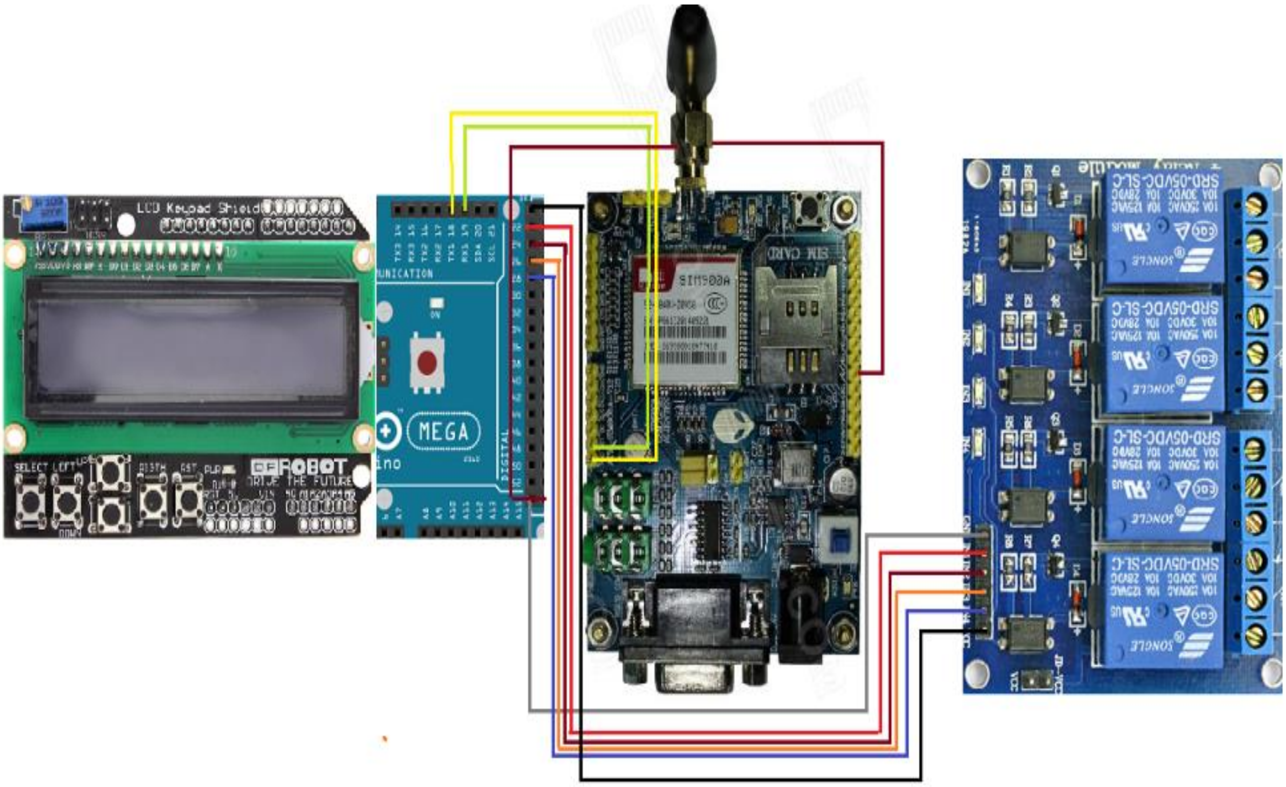
COMPONENTS:	PRICE: (TK)
1) GSM Modem (SIM900A)	4500
2) Arduino Mega 2560	1300
3) LCD shield (16*2)	350
4) Relay module	350
5) Jumper wires	260
6) Bread board	170
7) SIM card	150

	TOTAL PRICE = 7080 TK

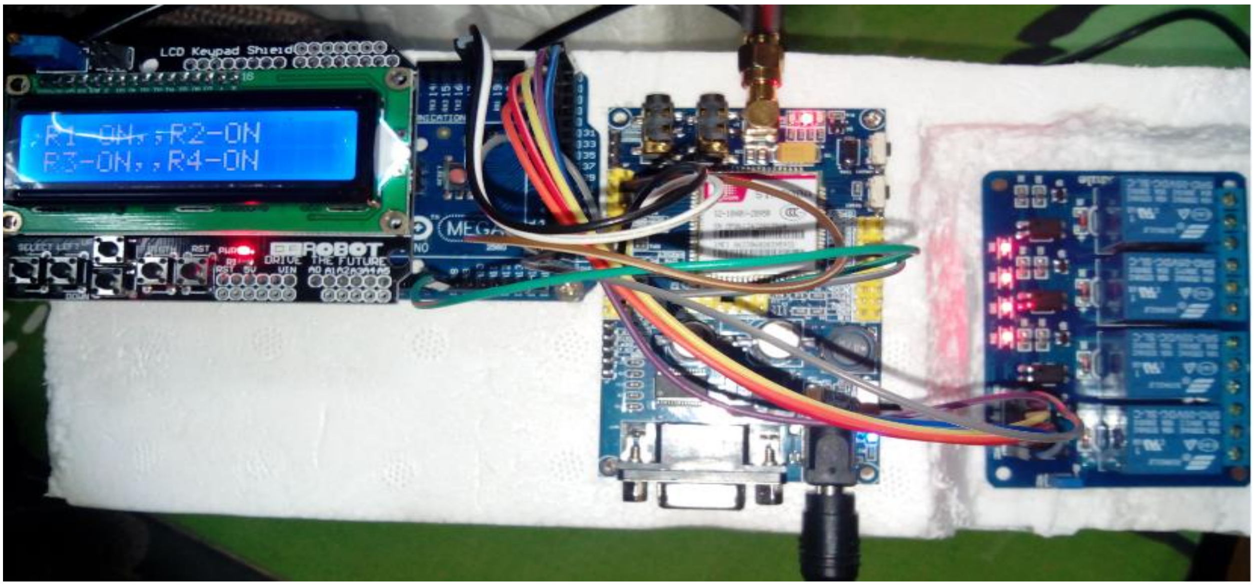
Simulated circuit:



BLOCK DIAGRAM



ACTUAL IMAGE OF THE PROJECT:



CHAPTER 4

WORKING OPERATION & PROGRAMMING CODE

This chapter deals with the working principles of SMS based control system using GSM MODEM. This chapter also introduces programming used in Arduino mega 2560

WORKING:

The idea behind this project is to use the existing GSM infrastructure. So, all the operations involve the GSM system also. As we send any SMS, it goes through the GSM system. Any sent SMS can be received if we use a SIM card and GSM module. To operate any GSM modem, we have to use the AT commands to operate them. For example, if any SMS arrives the GSM modem sends the serial data in ASCII format. We can read these data if we connect the modem with the serial port of the microcontroller at the baud rate of 9600. As the microcontroller comes to know that a SMS has been arrived, it can sent a proper AT command to read the SMS. The reading of SMS returns the mobile no of sender, the time and much more information. We have to select the SMS part of the message. The starting string of the SMS is used as the password. As the password is matched, then the SMS arrival is assumed to be valid by the microcontroller otherwise, it ignores the SMS. Working of SMS based device control system using GSM MODEM is very simple. It can be simply understood by its block diagram &circuit diagram.

PROGRAMMING CODE:

MESSAGE FORMAT:

- To operate the appliances we have to send SMS command from mobile to arduino mega.
- The message format of the SMS command is as follow:

#axbxcxdx

- Here a, b, c &d are four appliances respectively.
- X is integer value which is either “0” or “1”.

0= OFF & 1=ON

ARDUINO CODE:

```
//#include <SoftwareSerial.h>      // standard Arduino library
#include <LiquidCrystal.h>       // standard Arduino library

char incoming_char = 0;

char inchar;   // Will hold the incoming character from the GSM shield
//SoftwareSerial Serial1(19, 18);Rx &Tx pins connected to pins 18&19
of arduino
LiquidCrystal lcd(48, 49, 44, 45, 46, 47);     //lcd pin arrangement to
show operations
//LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
int led1 = 22;       // digital output pin no. of arduino mega to which
relay
int led2 = 24;     //          module is connected
int led3 = 26;
int led4 = 28;
//const int pin = A8; //analog pin A8 used for initializing the GSM.
```

```
const int value = 0;          // initial analog input equal to zero
void setup()    // setup to execute when command will receive
{
  Serial.begin(19600);    //Buad rate to show the operations on serial
monitor
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);         //declaring digital pins as an output
  pinMode(led4, OUTPUT);
  //  pinMode(pin, INPUT);   // declaring analog input pin to read the
voltage from GSM
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);    // declaring digitals output pins initially
//HIGH i.e (5v output)
  digitalWrite(led3, HIGH);

  digitalWrite(led4, HIGH);
  lcd.begin(16, 2);          // adding LCD feature in the program
  Serial1.begin(19600);   // declaring buad rate for serial 1 ports
  //
  Serial.println("started"); //  will show the â€œstartedâ€� message on
serial monitor
  lcd.setCursor(0, 0);     //(column,row) setting the position  of the
cursor
  lcd.print("INITIALIZING");     //will  show  the  â€œINITIALIZINGâ€�
message on LCD shield
  delay(1000);
  Serial1.print("AT+CMGF=1\r");     // set SMS mode to text for //GSM
modem (AT Commands)
  delay(1000);
  Serial1.print("AT+CNMI=2,2,0,0,0\r");     // blurt out contents of new
SMS and immediately send the command to arduino
  delay(10);
  Serial.println("Ready...");              // will show the â€œReadyâ€¦â€�
//message on serial monitor
```

```
 lcd.setCursor(2, 1);      //(column,row)  // setting the cursor position
 lcd.print("READY...");      // will print â€œREADYâ€¦â€⍰ message on
LCD
 /*
 {
  int analogValue = analogRead(pin);     // These lines will initialize
//the GSM modem to turn on the GSM modem
  if (analogValue > value)   //.. automatically from //arduino mega
without pressing the â€œpower onâ€⍰
  {
   //button on GSM modem which is normally used to turn on
modem.
   digitalWrite(9, LOW);    //when the voltage is available //from GSM
modem the arduino will make
   delay(7000);    // D 52 pin LOW otherwise if voltage //value is not
available then arduino
  }   // will make D 52 HIGH to turn on GSM modem/*
  else
  {
   digitalWrite(52, HIGH);
   delay(1000);
  }
 }
 */
 SIM900power();
 lcd.setCursor(0, 0); //(column,row)     // setting cursor position  on
LCD shield
 lcd.print(" GSM - MODEM");                    // will print â€œGSM-
//MODEMâ€⍰ message on LCD SHIELD
 lcd.setCursor(2, 1);                  //(column,row)
 lcd.print("ACTIVATED");
}

void SIM900power()
// software equivalent of pressing the GSM shield "power" button
```

```
{
  digitalWrite(9, HIGH);
  delay(1000);
  digitalWrite(9, LOW);
  delay(7000);
}


void loop()                    // void loop setup
{
  /*
    if(Serial1.available() >0)
  {
    incoming_char=Serial1.read(); //Get the character from the cellular
serial port.
    Serial.print(incoming_char); //Print the incoming character to the
terminal.
  }
  */

    if (Serial1.available() > 0) // reading command from serial 1
  {
    inchar = Serial1.read();
    if (inchar == '#')                // # character for the format
    {
      delay(10);

      inchar = Serial1.read();
      if (inchar == 'a')          // if value comes â€œa0â€� then make D 22
HIGH
      {
        delay(10);
        inchar = Serial1.read();
        if (inchar == '0')
        {
```

```
    digitalWrite(led1, HIGH);
    Serial.println("A low");
    lcd.setCursor(0, 0);                    //(column,row)
    lcd.print("R1-0FF,");
  }
  else if (inchar == '1') // if value comes â€œa1â€�then make D22
  { //LOW to turn on relay 1
    digitalWrite(led1, LOW);
    Serial.println("A HIGH");
    lcd.setCursor(0, 0);                //(column,row)
    lcd.print("R1-ON,,");
  }
}else if (inchar == 'b')
{
  inchar = Serial1.read();
  if (inchar == '0')
  {
    digitalWrite(led2, HIGH);
    Serial.println("b low");
    lcd.setCursor(7, 0);           //(column,row)
    lcd.print("R2-OFF");
  }
  else if (inchar == '1')
  {
    digitalWrite(led2, LOW);               // for relay 2
    Serial.println("B high");
    lcd.setCursor(7, 0);                  //(column,row)
    lcd.print("R2-ON");
  }
}
else if (inchar == 'c')
{ // Similar process to operate relay 3
  inchar = Serial1.read();
  if (inchar == '0')
  {
    digitalWrite(led3, HIGH);
    Serial.println("c low");
    lcd.setCursor(0, 1);            //(column,row)
    lcd.print("R3-OFF,");
  }
  else if (inchar == '1')
  {
    digitalWrite(led3, LOW);
    Serial.println("c HIGH");
    lcd.setCursor(0, 1);             //(column,row)
    lcd.print("R3-ON,,");
  }
}else if (inchar == 'd')
  { // commands to operate relay 4 v
    delay(10);           //similar process
    inchar = Serial1.read();
    if (inchar == '0')
    {
      digitalWrite(led4, HIGH);
      Serial.println("d low");
      lcd.setCursor(7, 1);                  //(column,row)
      lcd.print("R4-OFF");
    }
    else if (inchar == '1')
    {
      digitalWrite(led4, LOW);
      Serial.println("d high");
      lcd.setCursor(7, 1); //(column,row)
      lcd.print("R4-ON");     // for relay 4
    }
    delay(10);
  }

Serial1.println("AT+CMGD=1,4"); // delete all SMS
```

```
/*
 if (Serial1.available() > 0) // reading command from serial 1
 {
  inchar = Serial1.read();
  if (inchar == '#')                // # character for the format
  {
   delay(10);

   inchar = Serial1.read();
   if (inchar == 'a')         // if value comes â€œa0â€� then make D 22
HIGH
   {
    delay(10);
    inchar = Serial1.read();
    if (inchar == '0')
    {
     digitalWrite(led1, HIGH);
     Serial.println("A low");
     lcd.setCursor(0, 0);                      //(column,row)
     lcd.print("R1-0FF,");
    }
    else if (inchar == '1') // if value comes â€œa1â€�then make D22
    { //LOW to turn on relay 1
     digitalWrite(led1, LOW);
     Serial.println("A HIGH");
     lcd.setCursor(0, 0);                 //(column,row)
     lcd.print("R1-ON,,");
    }
   }
   delay(10);
   inchar = Serial1.read();   // similar procedure as for relay 1 to
operate the relay 2
   if (inchar == 'b')
   {
    inchar = Serial1.read();
  if (inchar == '0')
  {
    digitalWrite(led2, HIGH);
    Serial.println("b low");
    lcd.setCursor(7, 0);          //(column,row)
    lcd.print("R2-OFF");
  }
  else if (inchar == '1')
  {
    digitalWrite(led2, LOW);              // for relay 2
    Serial.println("B high");
    lcd.setCursor(7, 0);                  //(column,row)
    lcd.print("R2-ON");
  }
}
delay(10);
inchar = Serial1.read();
if (inchar == 'c')
{ // Similar process to operate relay 3
  inchar = Serial1.read();
  if (inchar == '0')
  {
    digitalWrite(led3, HIGH);
    Serial.println("c low");
    lcd.setCursor(0, 1);           //(column,row)
    lcd.print("R3-OFF,");
  }
  else if (inchar == '1')
  {
    digitalWrite(led3, LOW);
    Serial.println("c HIGH");
    lcd.setCursor(0, 1);            //(column,row)
    lcd.print("R3-ON,,");
  }
}
```

```
        delay(10);
        inchar = Serial1.read();
        if (inchar == 'd')
        { // commands to operate relay 4 v
          delay(10);           //similar process
          inchar = Serial1.read();
          if (inchar == '0')
          {
            digitalWrite(led4, HIGH);
            Serial.println("d low");
            lcd.setCursor(7, 1);                    //(column,row)
            lcd.print("R4-OFF");
          }
          else if (inchar == '1')
          {
            digitalWrite(led4, LOW);
            Serial.println("d high");
            lcd.setCursor(7, 1); //(column,row)
            lcd.print("R4-ON");     // for relay 4
          }
          delay(10);
        }

      Serial1.println("AT+CMGD=1,4"); // delete all SMS

*/
    }
  }

}
```

CHAPTER 5

ADVANTAGES & DISADVANTAGES:

This project is made with best of capabilities and dedication. Details were taken care of in preparing it. The problems encountered in various steps were taken into account and eliminated to an extent so that they may not harm the project functioning. Necessary conditions were laid out so as to prevent limitation for the project. But, as it is that every system is not perfect in all the aspects. They have some associated limitations. Here are some advantages, in general and comparative of the counterparts, and disadvantages of the project.

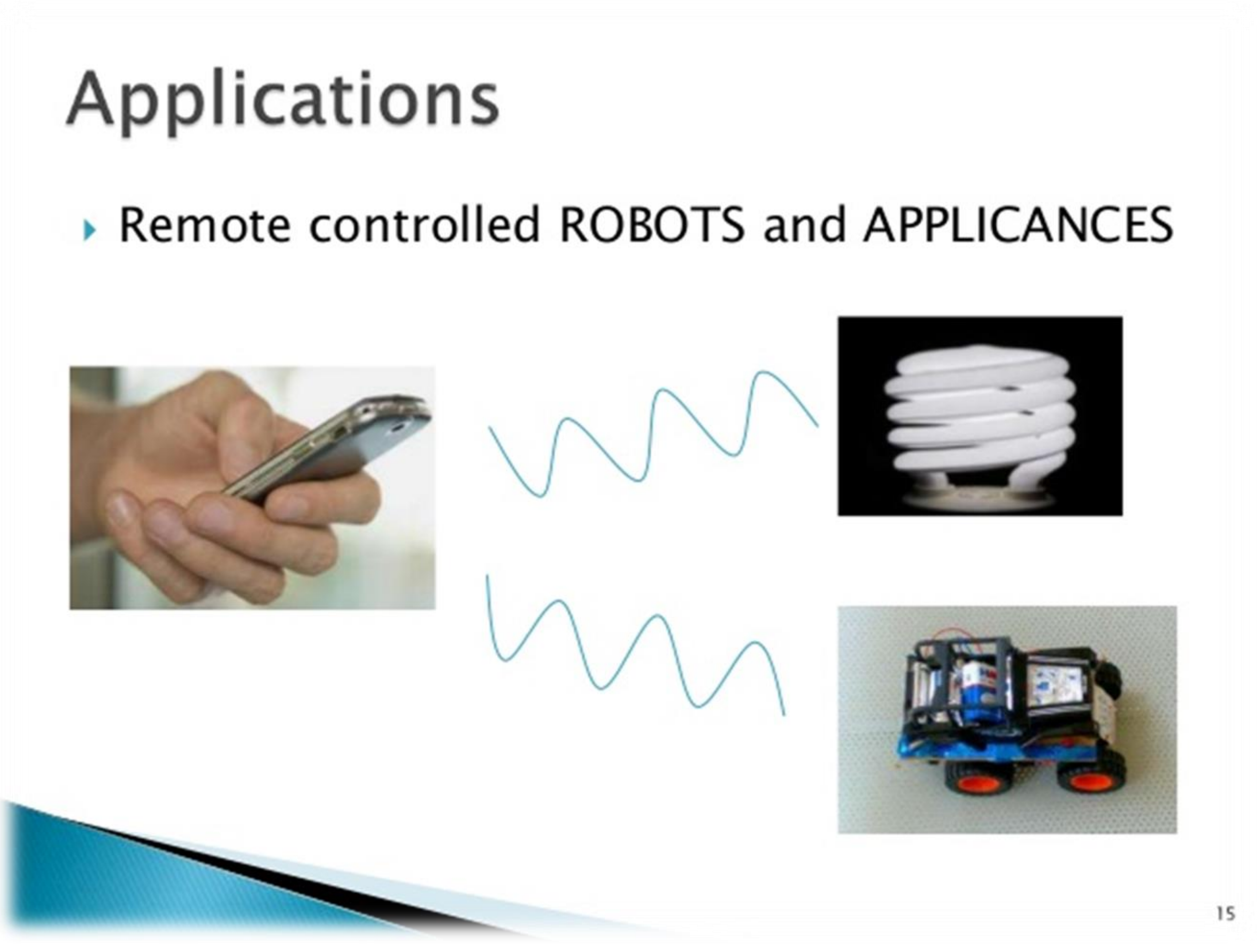
ADVANTAGES:

1. Devices can be controlled from long distances.
2. Economical design
3. Can be easily implemented in homes
4. Can be used by everyone with just the knowledge of text SMS
5. Format of the SMS is simple to understand and write.

DISADVANTAGES:

1. The system is network dependent. Hence, network congestion can reduce the reliability of the system
2. There is no feed back
3. Cost in sending the SMS to the GSM

CHAPTER 6



Applications

- Remote controlled ROBOTS and APPLICANCES

The previous chapter deal with advantages and disadvantages of the project, the project can be put for use in following fields

1. Home automation, which was the seed for developing the idea of project.
2. Remote device control.
3. This will help to eliminate need of human personnel attending the device till it has to be switched off/on. Based on experience of approximate time to switch on/off
4. He or she can control the device, saving time. It will also help to control device while attending to other work.
5. User can control device in case of forgetting to do so while leaving for some other place.

CONCLUSION

Based on all the system surveyed and their advantages and drawbacks, this project presents the features to be possessed by an ideal system for home automation with remote access. An ideal system should be available from all over the world to a user and in real time. A GSM network is identified as a candidate for this. However, the data channel of GSM must be used, to provide internet access. Only the Internet can ensure that access can be made available at all times. This will give rise to a standard access method for the home appliances using the Internet protocol. The user interface should be a web application that has an associated mobile application. So that people of all kinds can access the system. Such a system should also have the feature of being easy to install. Only then can automated homes become commercially viable. There should be a lot of thought put into the design of the user interface for these apps. Plug and play capabilities will be an added bonus for the system. Ease of adding a new device to an automated house will play an important role in taking forward the systems commercially.

Future scope for the home automation systems involves making homes even smarter. Homes can be interfaced with sensors including motion sensors, light sensors and temperature sensors and provide automated toggling of devices based on conditions. More energy can be conserved by ensuring occupation of the house before turning on devices and checking brightness and turning off lights if not necessary. The system can be integrated closely with home security solutions to allow greater control and safety for home owners. The next step would be to extend this system to automate a large scale environment, such as offices and factories

REFERENCES

1: Baris Yuksekkaya, A. Alper Kayalar, M. Bilgehan Tosun, M. Kaan Ozcan, and Ali Ziya Alkar "A GSM, Internet and Speech Controlled Wireless Interactive Home Automation System", 2006, IEEE Transactions on Consumer Electronics, Vol. 52(3) , pp. 837 - 843.