Ex. No. 1 EXPLORE DIFFERENT COMMUNICATION METHODS WITH IOT DEVICES.

OBJECTIVE:

To explore different communication methods with IoT devices such as interfacing Arduino to Bluetooth module and interfacing Arduino to GSM module.

REQUIRED COMPONENTS:

- Arduino UNO
- Arduino IDE Software
- SIM 800C GSM Module
- Full-Size SIM Card (Unlocked)
- HC-05 Bluetooth Module
- 12V 2Amp Power Supply
- Breadboard
- USB cable for uploading code into Arduino UNO
- LED
- Jumper Wires
- Android App -Serial Terminal Bluetooth App

BACKGROUND THEORY:

GSM SIM 900 Module

- The GSM SIM 900 Module is a type of Arduino Shield, which means it can also be mounted on top of Arduino UNO.
- This is a type of modem, used for long-distance data transmission with the use of GSM technology where there is no internet connectivity.
- This makes it useful in projects which require remote data transmission. Many projects can be made using this module such as call or text message-based triggers which can be used in the day-to-day life.
- Farmers can extensively use this technology in the day-to-day life, such as controlling water pumps with a simple silent call or a text message sitting at the home.
- For example, a gsm based agriculture project. This can also be used to send the data from the module to the mobile phones without the use of the internet.

- Other project ideas include intruder alert notification, Receive timely updates from the sensor to the mobile, etc.

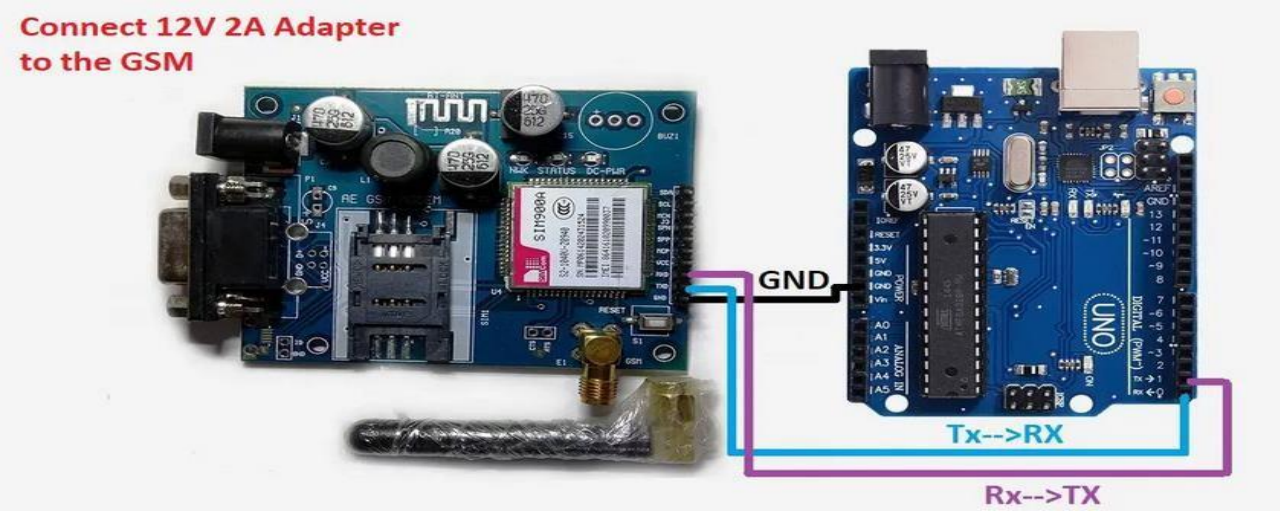
Working of GSM SIM 900 Module

The GSM module uses GSM and GPRS technology to communicate with another device wirelessly. It uses a 2G network to connect with the internet and supports Quad-band (EGSM 900, GSM 850, DCS 1800, PCS1900). Because of this, one can send or receive messages from this or make or receive voice calls using the module by connecting the microphone and speakers to the respective ports given on it. This module can be used for security purposes like gsm-based forest fire alerts and control systems. This Modu

also houses an inbuilt RTC to keep track of time, which is very helpful for timer-based applications. The versatility of this module is very high due to its ability to read and send messages without any hassle. Because of the use of AT Commands, it is very easy to configure. because of the presence of an external antenna, the module can also be used in areas with low signal areas.

GSM SIM 900 Module.

Interfacing GSM Module with Arduino Circuit Diagram.


Connect 12V 2A Adapter
to the GSM
GND
Tx-->RX
Rx-->TX

Connection Table

Hardware Setup

1. Connect the SIM800C module to the Arduino:
    - SIM800C RX to Arduino pin 4
    - SIM800C TX to Arduino pin 5
    - SIM800C GND to Arduino GND
    - SIM800C VCC to a suitable power source (usually 4.2V to 4.4V)
2. Insert a Mobitel or Hutch SIM card into the SIM800C module.

Commands Used for Operation

Command	Description
AT+CSMS	To Select message service
AT+CPMS	To Preferred message storage
AT+CMGF	select Message format
AT+CSCA	Service center address
AT+CSMP	Set text mode parameters in sim
AT+CSDH	To Show text mode parameters in Sim
AT+CSCB	Select cell broadcast message types
AT+CSAS	Save settings in the gsm module
AT+CRES	Restore all settings
AT+CNMI	Message indications to TE
AT+CMGL	To make the list of messages
AT+CMGR	Read new message
AT+CMGS	Send a new message
AT+CMSS	Send message from sim storage
AT+CMGW	Write a message to gsm memory
AT+CMGD	Delete message

CODING:

Interfacing GSM Module with Arduino Code

SIM800C Setup

```
#include <SoftwareSerial.h>
SoftwareSerial SIM900A(10,11); // RX | TX
// Connect the SIM900A TX to Arduino pin 10 RX.
// Connect the SIM900A RX to Arduino pin 11 TX.
char c = ' ';
void setup()
{
// start th serial communication with the host computer
Serial.begin(9600);
while(!Serial);
Serial.println("Arduino with SIM900A is ready");

// start communication with the SIM900A in 9600
SIM900A.begin(9600);
Serial.println("SIM900A started at 9600");
delay(1000);
Serial.println("Setup Complete! SIM900A is Ready!");
}

void loop()
{

// Keep reading from SIM800 and send to Arduino Serial Monitor
if (SIM900A.available())
{ c = SIM900A.read();
Serial.write(c);}

// Keep reading from Arduino Serial Monitor and send to SIM900A
if (Serial.available())
{ c = Serial.read();
SIM900A.write(c);
}

}
```

Message Response

```
#include <SoftwareSerial.h>
SoftwareSerial SIM900A(10,11);
void setup()
{
  SIM900A.begin(9600);   // Setting the baud rate of GSM Module
  Serial.begin(9600);    // Setting the baud rate of Serial Monitor (Arduino)
  Serial.println ("SIM900A Ready");
  delay(100);
```

```
  Serial.println ("Type s to send message or r to receive message");
}
void loop()
{
  if (Serial.available()>0)
   switch(Serial.read())
  {
    case 's':
      SendMessage();
      break;
    case 'r':
      RecieveMessage();
      break;
  }
 if (SIM900A.available()>0)
   Serial.write(SIM900A.read());
}
 void SendMessage()
{
  Serial.println ("Sending Message");
  SIM900A.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);
  Serial.println ("Set SMS Number");
  SIM900A.println("AT+CMGS=\"+917904329949\"\r"); //Mobile phone number to send
message
  delay(1000);
  Serial.println ("Set SMS Content");
  SIM900A.println("Good morning, how are you doing?");// Messsage content
  delay(100);
  Serial.println ("Finish");
  SIM900A.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
  Serial.println ("Message has been sent ->SMS Selesai dikirim");
}
 void RecieveMessage()
{
  Serial.println ("SIM900A Membaca SMS");
  delay (1000);
  SIM900A.println("AT+CNMI=2,2,0,0,0"); // AT Command to receive a live SMS
  delay(1000);
  Serial.write ("Unread Message done");
 }
```

Bluetooth Module (HC-05)

The Bluetooth module is a device which is used for short range wireless communicatio n to the respective connected device. This module uses serial port protocol for the wireless communication and comes with two configurations that are master and slave. In the master mode the module searches for the other devices to connect and can connect to

the other devices. However, in the slave mode the module cannot connect to the devices

by itself. In short the master more the device controls other devices and in slave mode the device is being controlled by some other device. To change the master slave configuration, it is possible to use the AT commands of the Bluetooth module. Moreover, to use the AT mode it is possible to set the baud rate of 38400 and for serial communication use the baud rate of 9600.

- Operating Voltage: 4 V to 6V (have internal 3.3V regulator).
- Operating Current: 30mA
- Integrated antenna and an edge connector.
- Range about 10 meters.
- Configurable in both master and slave modes.
- Pins: STATE, RXD, TXD, GND, VCC, KEY/ENABLE

HC-05 Bluetooth Module.

Pin Out

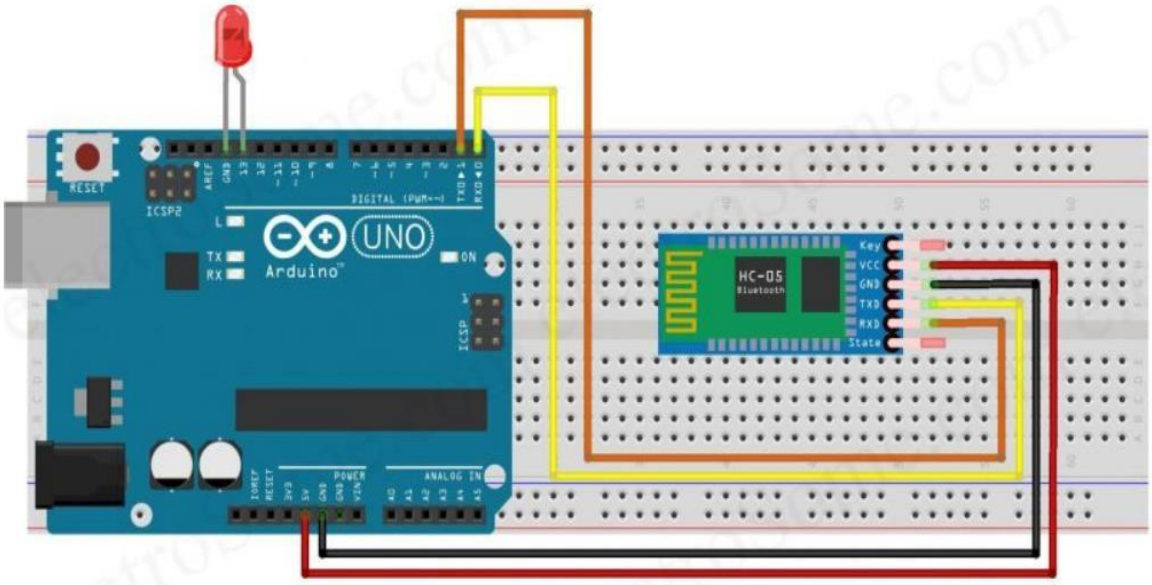
- **STATE:** State pin indicates whether the module is connected or paired with a device. When the module is not connected, this pin will be in LOW state and the on-board LED will be flashing fast. But when the module is paired or connected to

a device, the state pin will be in HIGH state and the on-board LED will be flashing with a delay.

- **RXD:** This is a UART RX pin. This pin is used to send AT commands when the module is in command mode. And it is used to send data to the connected device when the module is in data mode.
- **TXD:** This is a UART TX pin. This pin is used to push out responses to AT command when the module is in command mode. And it is used to push out data sent by the connected device when the module is in data mode.
- **GND:** Power supply -ive.
- **VCC:** Power supply +ive.
- **EN/KEY:** This input is used to switch between command and data mode. If this pin is set HIGH, the module will be in command mode. Similarly, if this pin is set LOW, the module will be in data mode.

Circuit Diagram

Interfacing HC-05 Bluetooth Module with Arduino Uno.



Description

- RXD pin of HC-05 Bluetooth – TXD pin of Arduino Uno
- TXD pin of HC-05 Bluetooth – RXD pin of Arduino Uno

- GND pin of HC-05 Bluetooth – GND pin of Arduino Uno
- VCC pin of HC-05 Bluetooth – 5V output pin of Arduino Uno
- Positive pin of LED – Pin 13 of Arduino Uno
- Negative pin of LED – GND pin of Arduino Uno

Coding for Bluetooth

```
#include <SoftwareSerial.h>

//Create software serial object to communicate with HC-05
SoftwareSerial mySerial(3, 2); //HC-05 Tx & Rx is connected to Arduino #3 & #2

void setup()
{
  //Begin serial communication with Arduino and Arduino IDE (Serial Monitor)
  Serial.begin(9600);

  //Begin serial communication with Arduino and HC-05
  mySerial.begin(9600);

  Serial.println("Initializing...");
  Serial.println("The device started, now you can pair it with bluetooth!");
}

void loop()
{
  if(Serial.available())
  {
    mySerial.write(Serial.read());//Forward what Serial received to Software Serial
Port
  }
  if(mySerial.available())
  {
    Serial.write(mySerial.read());//Forward what Software Serial received to Serial
Port
  }
  delay(20);
}
```

Arduino Bluetooth Controller

Installed Serial Terminal Bluetooth app from Google Play Store. This app will act as a Bluetooth remote controller for Arduino. It is very easy to use this app. Open the app and connect to the HC-05 device.

Working of Bluetooth Module (HC-05)

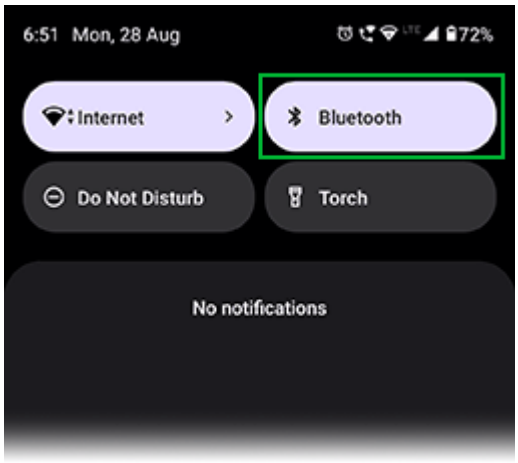
Once you have uploaded the sketch, open the serial monitor at baud rate 9600. You should see a message saying: “The device started, now you can pair it with bluetooth!”.

Connecting to the Android Phone

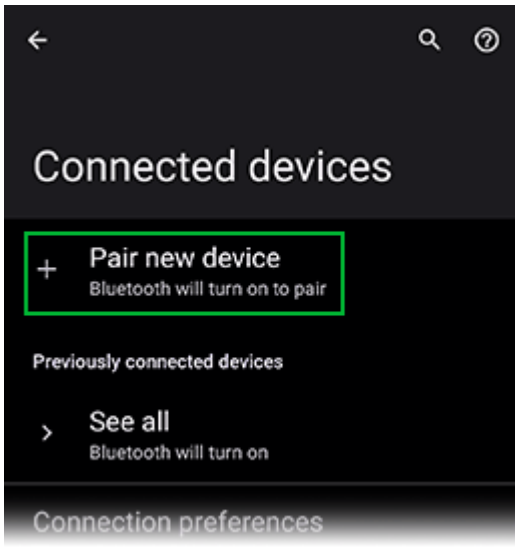
Let’s set up a wireless connection between the HC-05 module and an Android phone. The process may vary depending on the device, but the general steps are quite similar.

1. Make sure the HC-05 module is powered up and ready to establish a connection. The onboard LED should be blinking rapidly at about 2 Hz, indicating that it is discoverable.

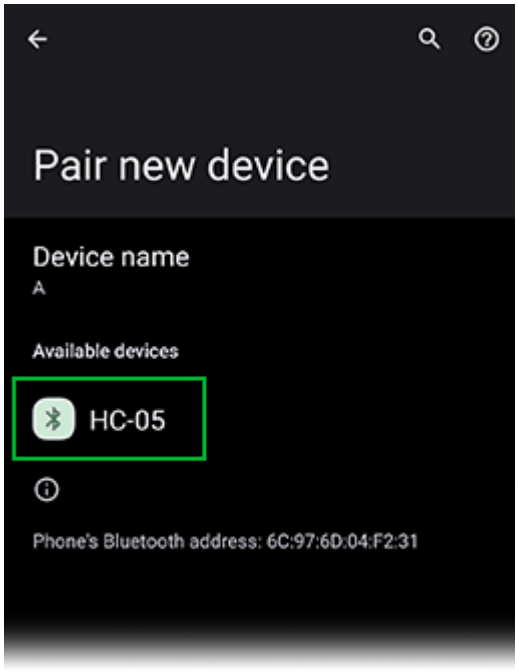
2. Now, swipe down from the top of your Android phone’s screen and make sure Bluetooth is turned on.



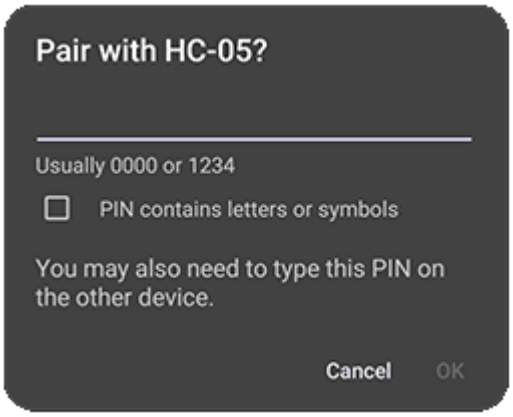
3. Touch and hold the Bluetooth icon, then tap “Pair new device” and wait a few seconds.



4. Tap the name of the Bluetooth device you want to pair with your device (in our case, HC-05). Follow any on-screen instructions.

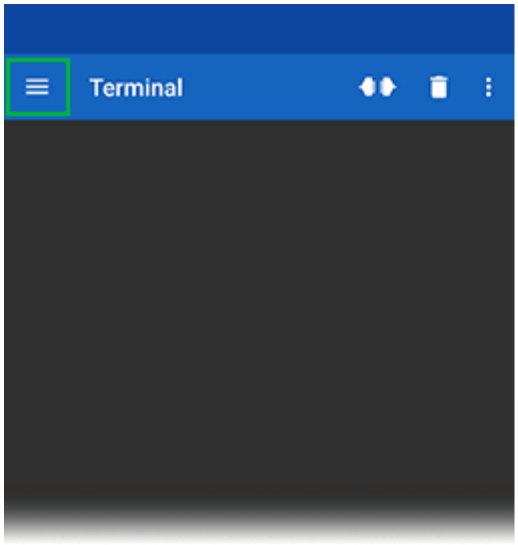


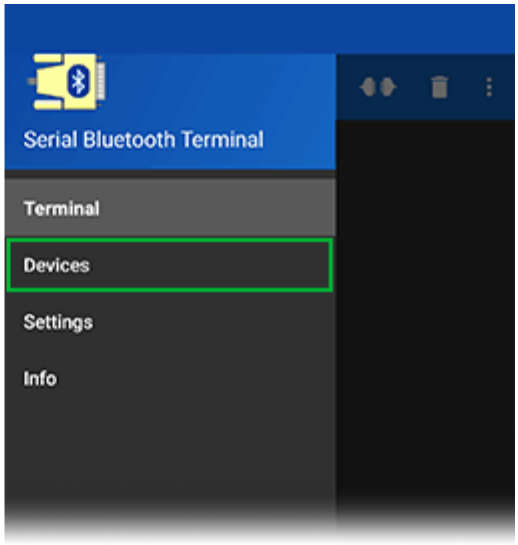
5. When asked, enter 1234 as the PIN code. This is the default PIN for every HC-05 module.



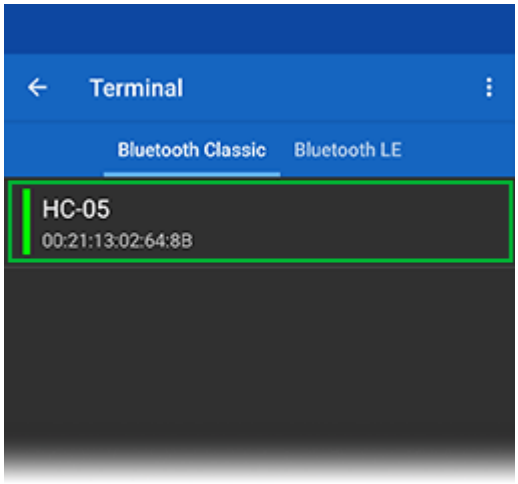
6. For the next steps in this tutorial, you need a Bluetooth Terminal application installed on your smartphone. We recommend using the Android app “Serial Bluetooth Terminal,” available in the Play Store.

7. After installing, launch the “Serial Bluetooth Terminal” app. Click on the icon in the top left corner and choose “Devices”.

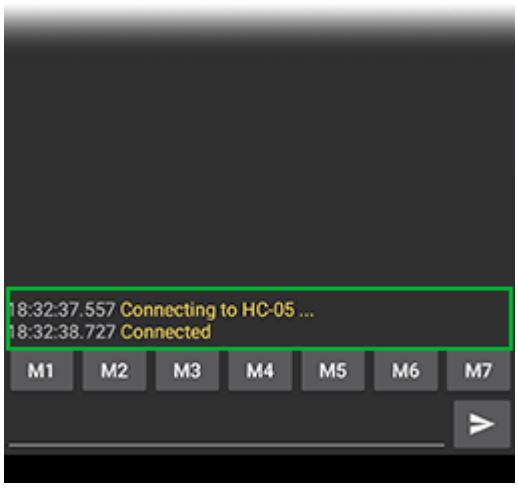




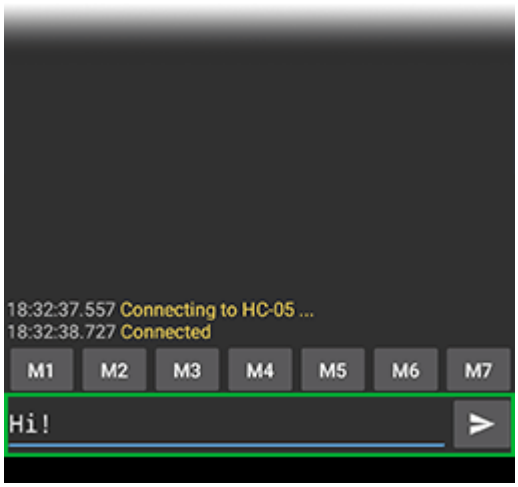
8. You should see a list of devices you've previously paired with. Select "HC-05" from this list.



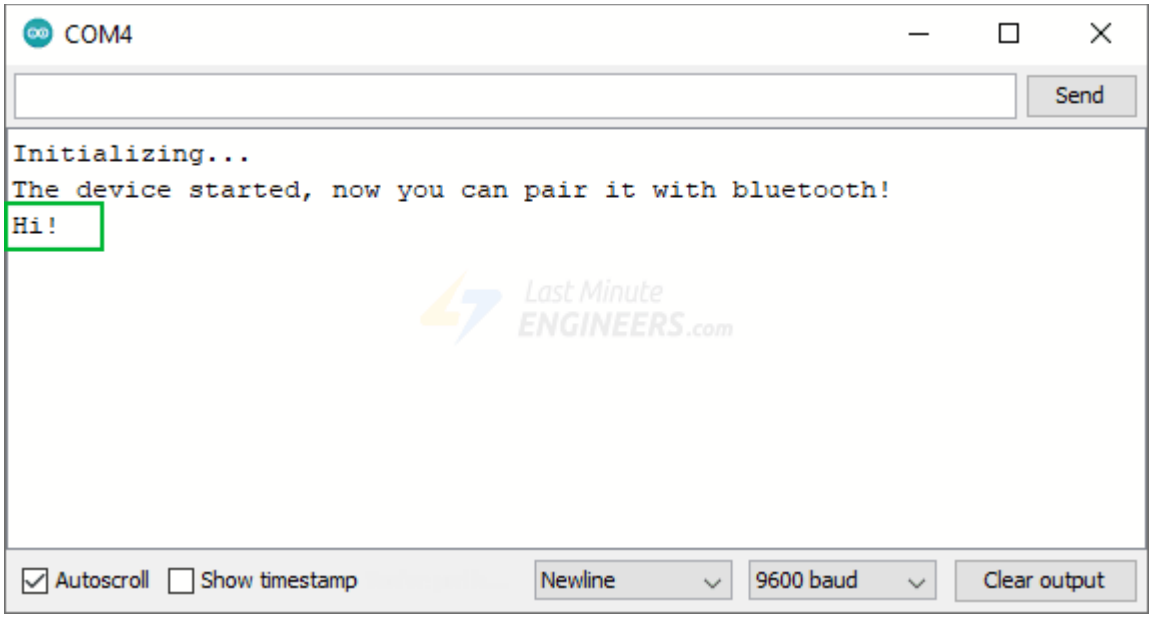
9. You should get a "Connected" message. The flash pattern on your HC-05's onboard LED should now change to two quick flashes followed by a pause. That's it! Your smartphone is now successfully paired with the HC-05 Bluetooth module and ready to communicate.



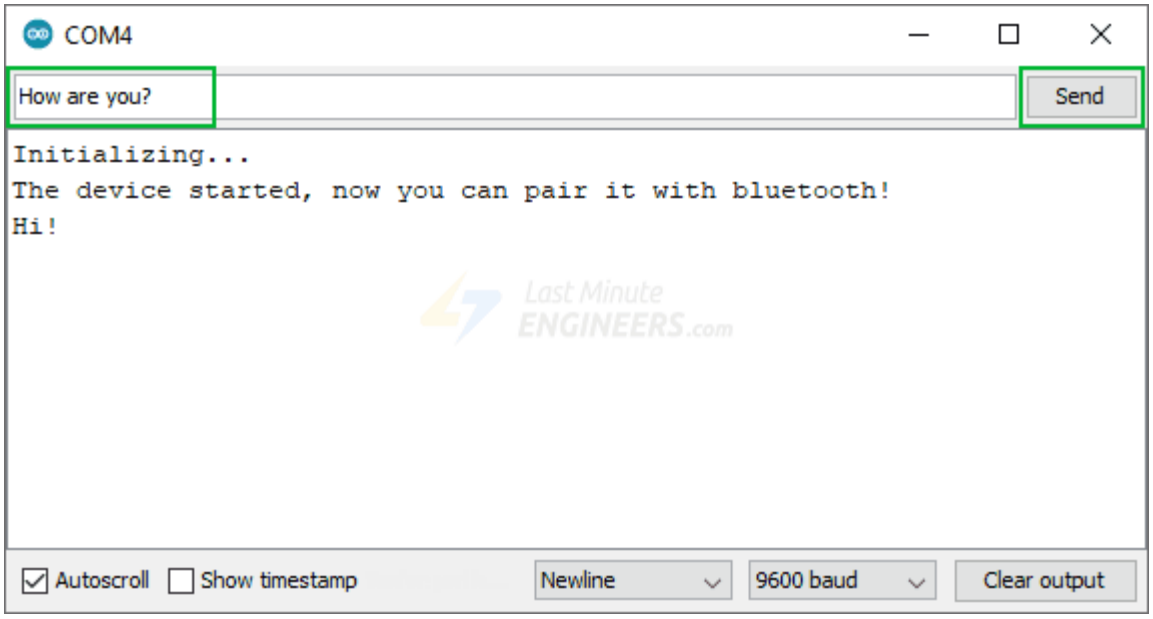
10. Now, type something in the input box located at the bottom of the app, for example, “Hi!”



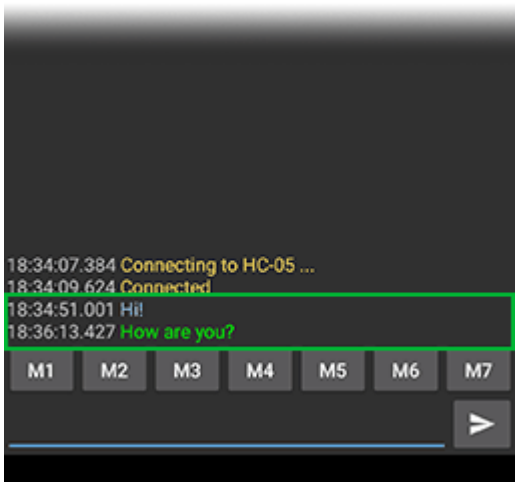
11. You should instantly receive that message in the Arduino IDE Serial Monitor.



12. You can also exchange data between your Serial Monitor and your smartphone. Type something in the Serial Monitor’s top input box and press the “Send” button.



13. You should instantly receive that message in the Serial Bluetooth Terminal App.



CONCLUSION:

Thus the different communication methods with IoT devices such as interfacing Arduino to GSM module and interfacing Arduino to Bluetooth module were explored successfully.