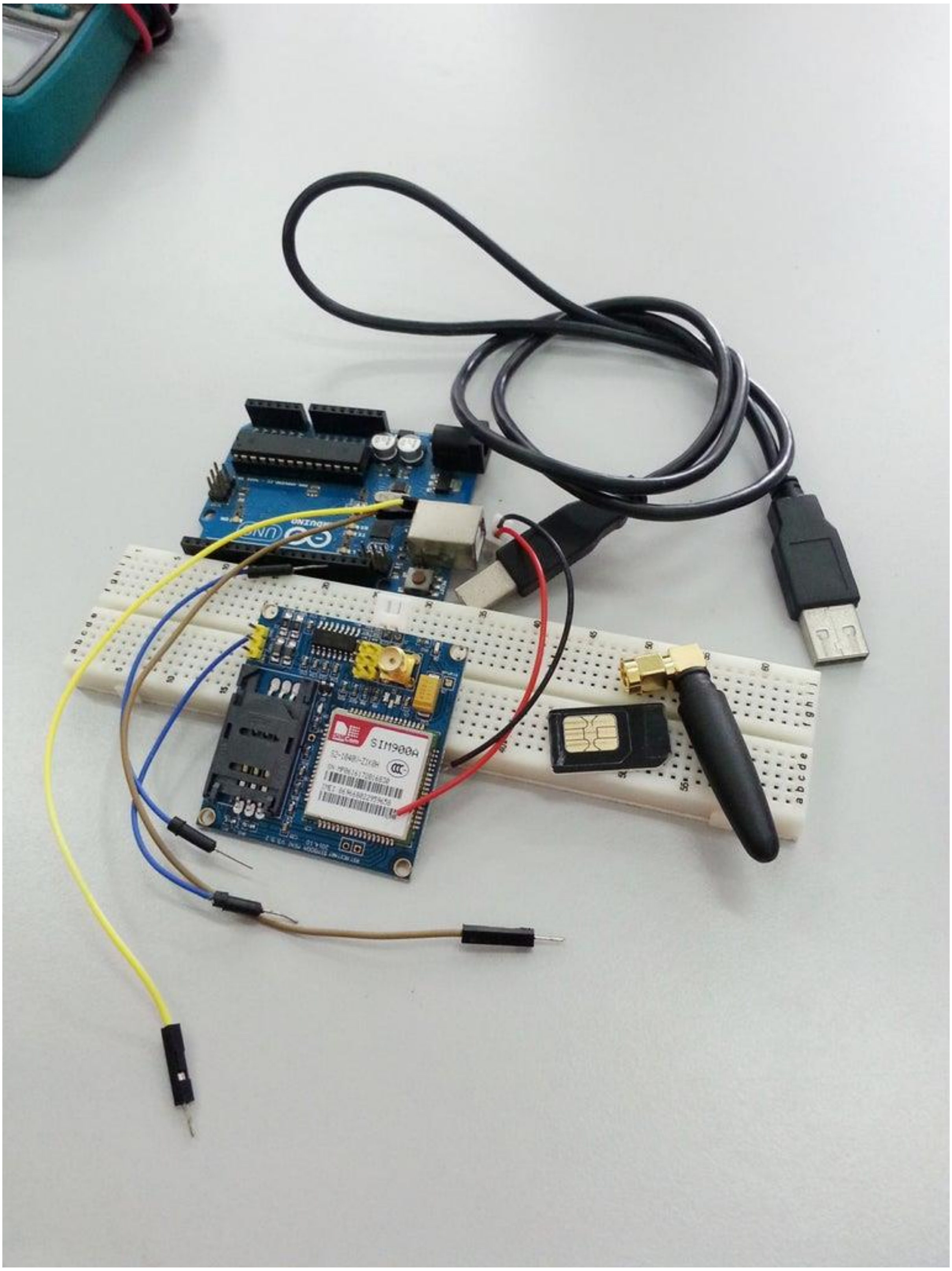
Description

SIM900A Modem is built with Dual Band GSM/GPRS based SIM900A modem from SIMCOM. It works on frequencies 900/ 1800 MHz. SIM900A can search these two bands automatically. The frequency bands can also be set by AT Commands. The baud rate is configurable from 1200-115200 through AT command. The GSM/GPRS Modem is having internal TCP/IP stack to enable you to connect with internet via GPRS. SIM900A is an ultra compact and reliable wireless module. This is a complete GSM/GPRS module in a SMT type and designed with a very powerful single-chip processor integrating AMR926EJ-S core, allowing you to benefit from small dimensions and cost-effective solutions.

Specification

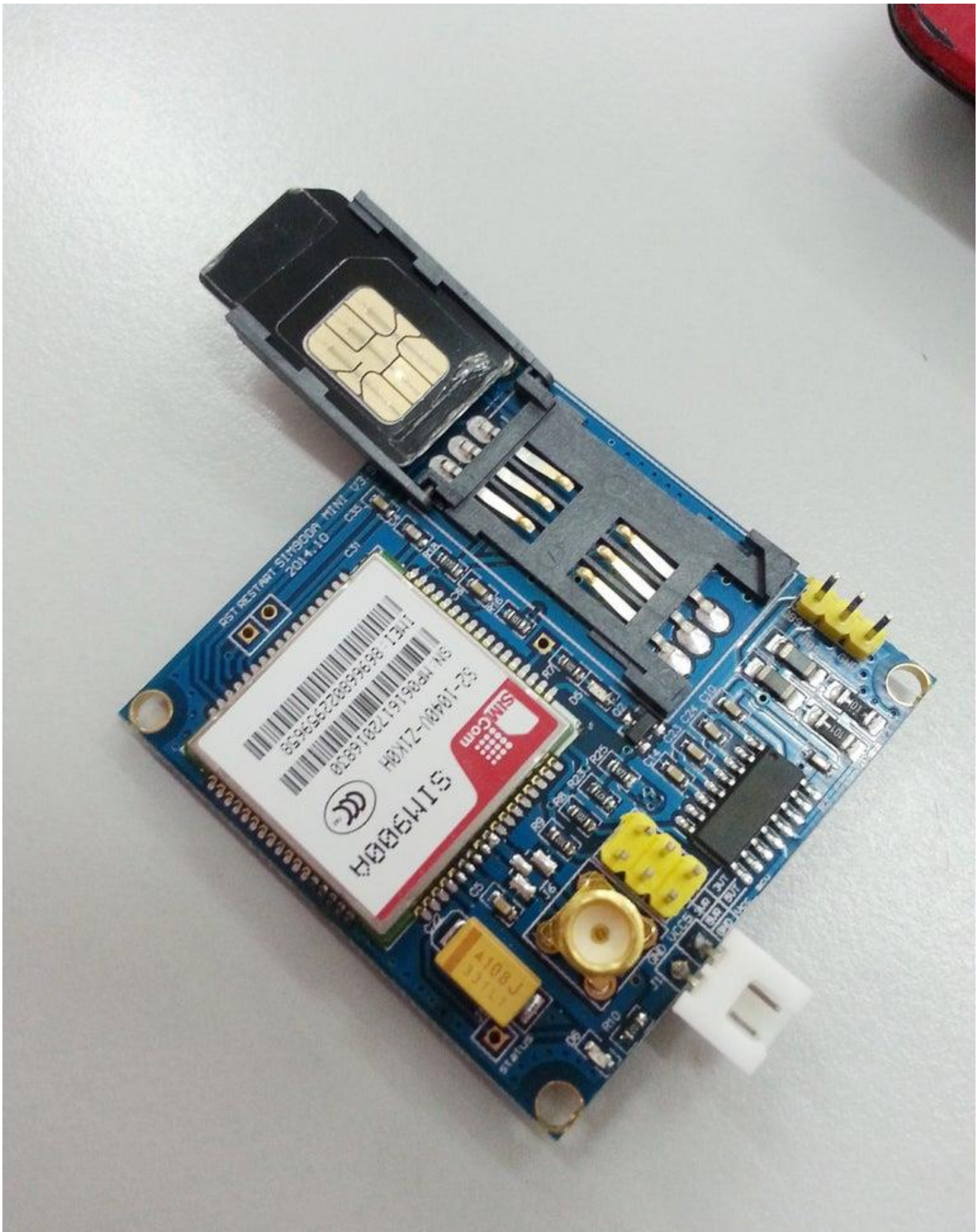
- Dual-Band 900/ 1800 MHz
- GPRS multi-slot class 10/8GPRS mobile station class B
- Compliant to GSM phase 2/2+
- Dimensions: 24*24*3 mm
- Weight: 3.4g
- Control via AT commands (GSM 07.07 ,07.05 and SIMCOM enhanced AT Commands)
- Supply voltage range : 5V
- Low power consumption: 1.5mA (sleep mode)
- Operation temperature: -40°C to +85 °

Step 1: Material Preparation

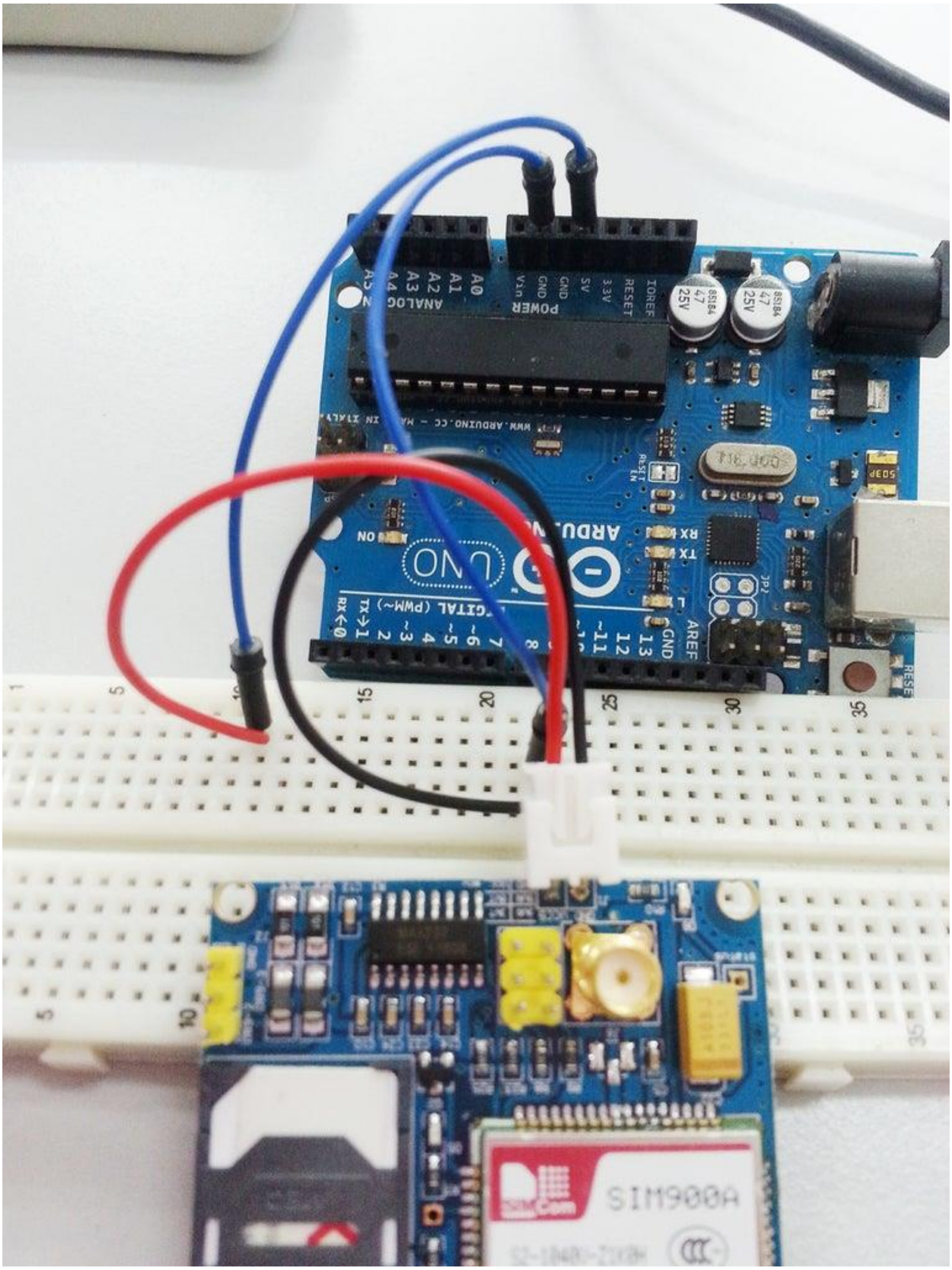


In this tutorial, you will need :

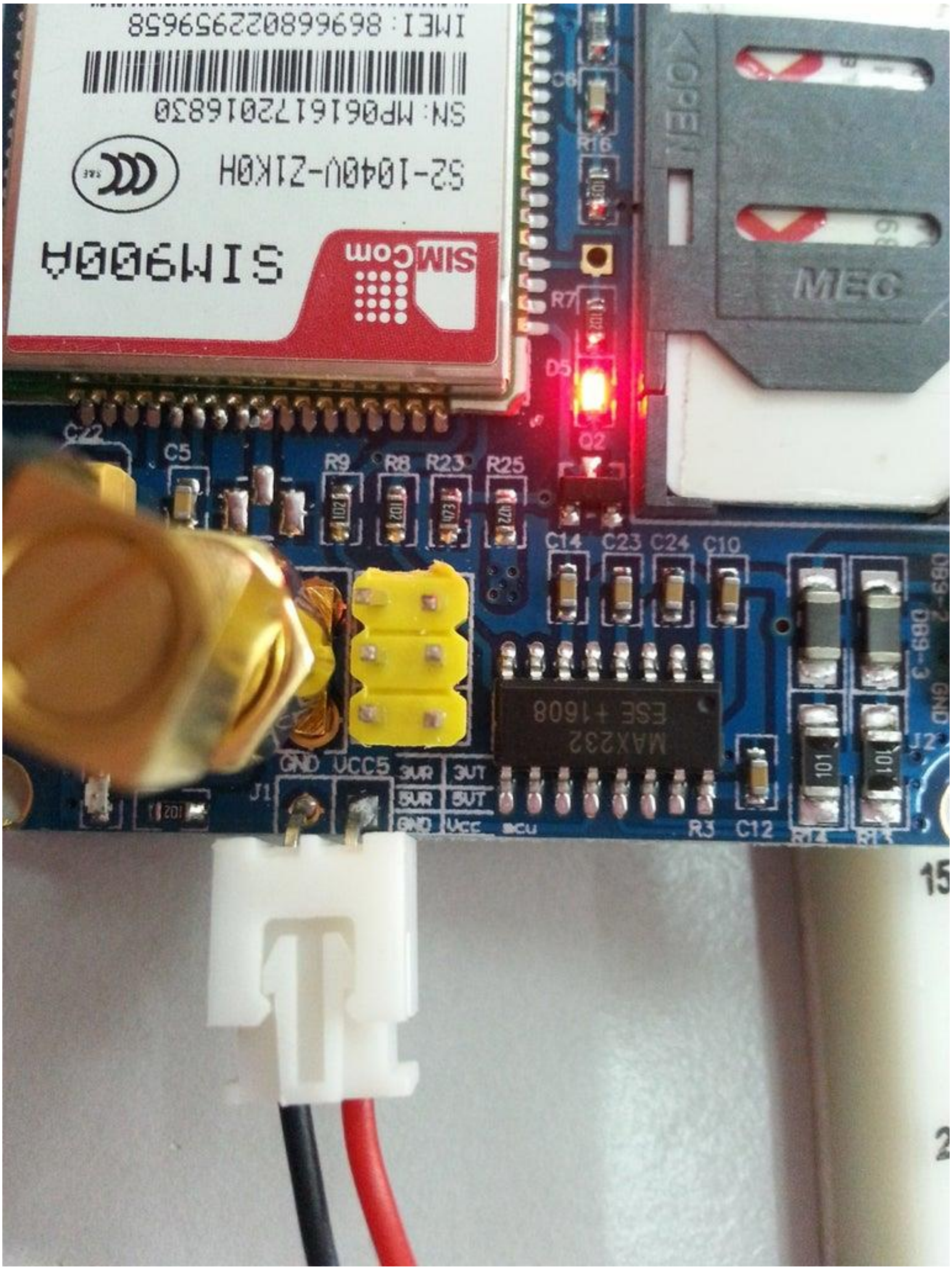
1. GSM SIM900A (MINI V3.9.2)
2. Arduino Uno Board and USB
3. Jumper Wire
4. Power adapter 5V
5. SIM card
6. Breadboard

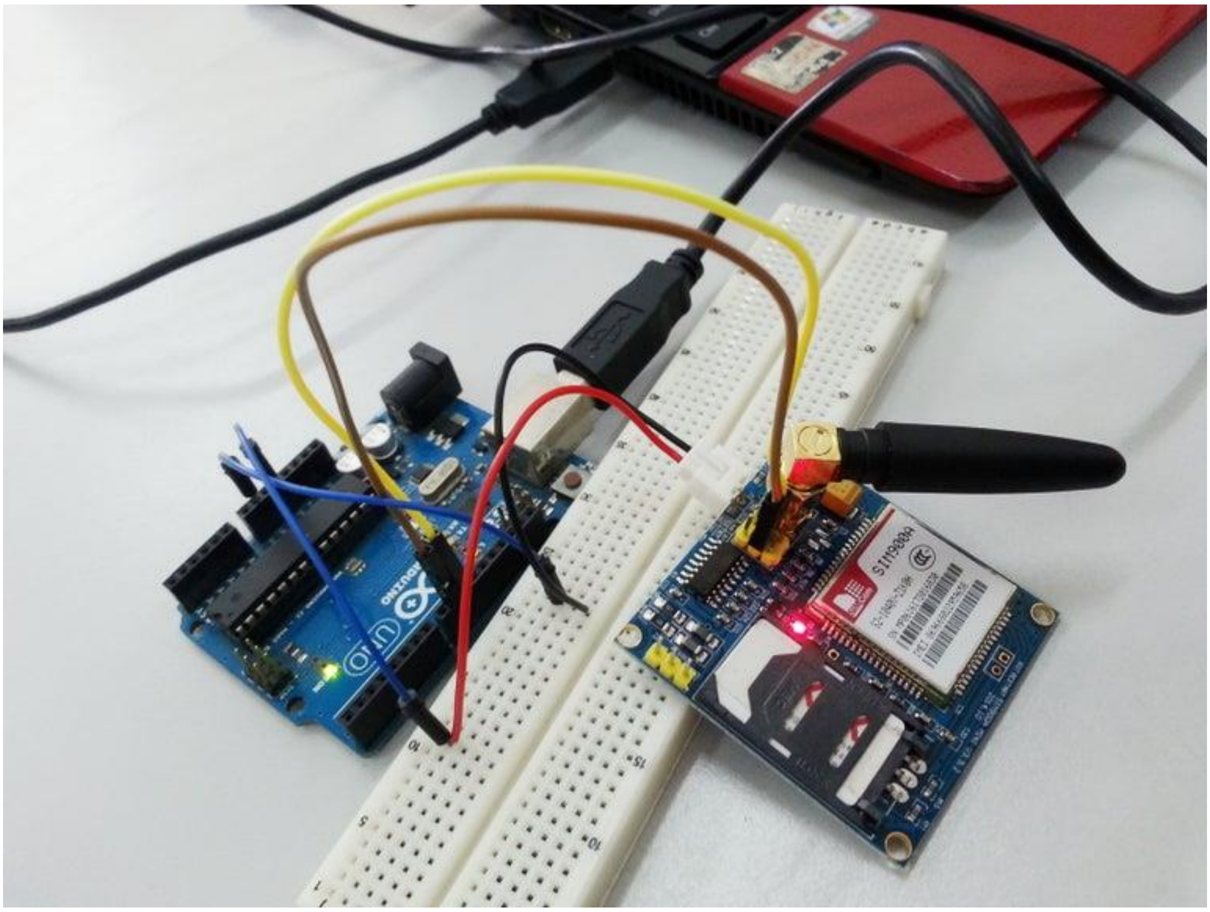

RST RESTART SIM900A MINI V3
2014.10
SIMCom
SIM900A
STATUS
GND
VCC5

SIM900A
SIMCom
RST RESTART SIM900A MINI V3.8.2
2014.10
status
GND
VCC


ANALOG IN
A0
A1
A2
A3
A4
A5
POWER
Vin
GND
GND
5V
3.3V
RESET
IOREF
ARDUINO
UNO
DIGITAL (PWM~)
AREF
GND
13
12
~11
RX
TX
ON
SIM900A

1. Insert your SIM card to GSM module and lock it. (picture 1 and 2)
2. power up your gsm by connecting it to Arduino's 5V and GND (picture 3)
3. Connect the Antenna (picture 4)
4. Now wait for some time (say 1 minute) and see the blinking rate of ‘status LED’ or ‘network LED’ (D6, refer picture 5) //GSM module will take some time to establish connection with mobile network//
5. Once the connection is established successfully, the status/network LED will blink continuously every 3 seconds. You may try making a call to the mobile number of the sim card inside GSM module. If you hear a ring back, the gsm module has successfully established network connection.


SIM900A
SIMCom
OPEN
MEC
C22
C5
R9
R8
R23
R25
R7
R16
C6
D5
Q2
C14
C23
C24
C10
MAX232
GND
VCC5
3VR
3VT
5VR
5VT
GND
Vcc
mcu
J1
R3
C12
R14
R13
J2
DB9-2
DB9-3
GND



You can see a TTL pin with 3VR, 3VT, 5Vr, 5VT, VCC and GND on your sim900a near your power supply pin. You have to connect **GSM's 5VT to Arduino D9** and **GSM's 5VR to Arduino's D10** for serial communication between arduino and sim900a module.

Step 4: Basic AT Command

1. To change sms sending mode : **AT+CMGF=1**

```
mySerial.println("AT+CMGF=1");
```

2. To read SMS in text mode : **AT+CNMI=2,2,0,0,0**

```
mySerial.println("AT+CNMI=2,2,0,0,0");
```

3. To make a call : **ATD+60XXXXXXXXX;** //replace X with number you want to call, change +60 to your country code

```
mySerial.println("ATD+60XXXXXXXXX;");
```

4. To disconnect / hangup call : **ATH**

```
mySerial.println("ATH");
```

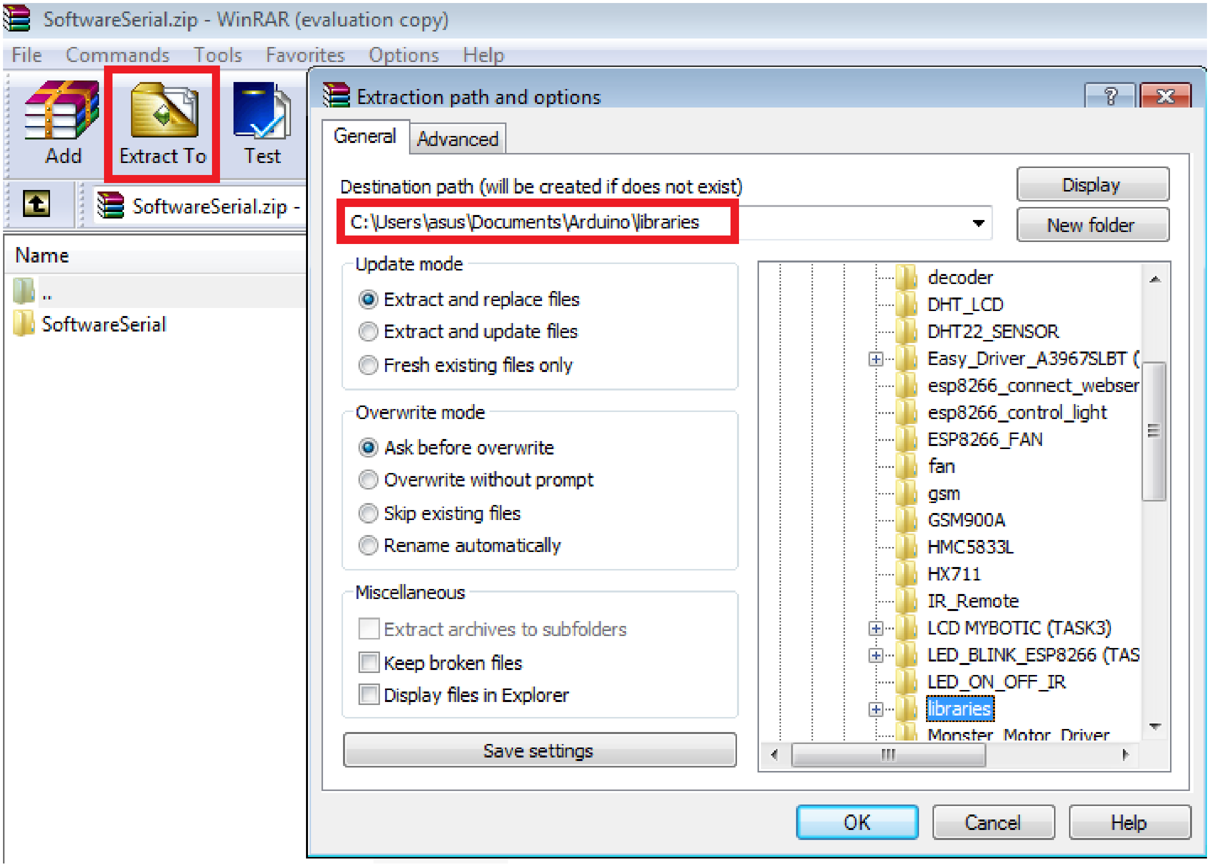
5. To redial : **ATDL**

```
mySerial.println("ATDL");
```

6. To receive a phone call : **ATA**

```
mySerial.println("ATA");
```

Step 5: Library



Destination path (will be created if does not exist)

C:\Users\asus\Documents\Arduino\libraries

Update mode

- ● Extract and replace files
- ○ Extract and update files
- ○ Fresh existing files only

Overwrite mode

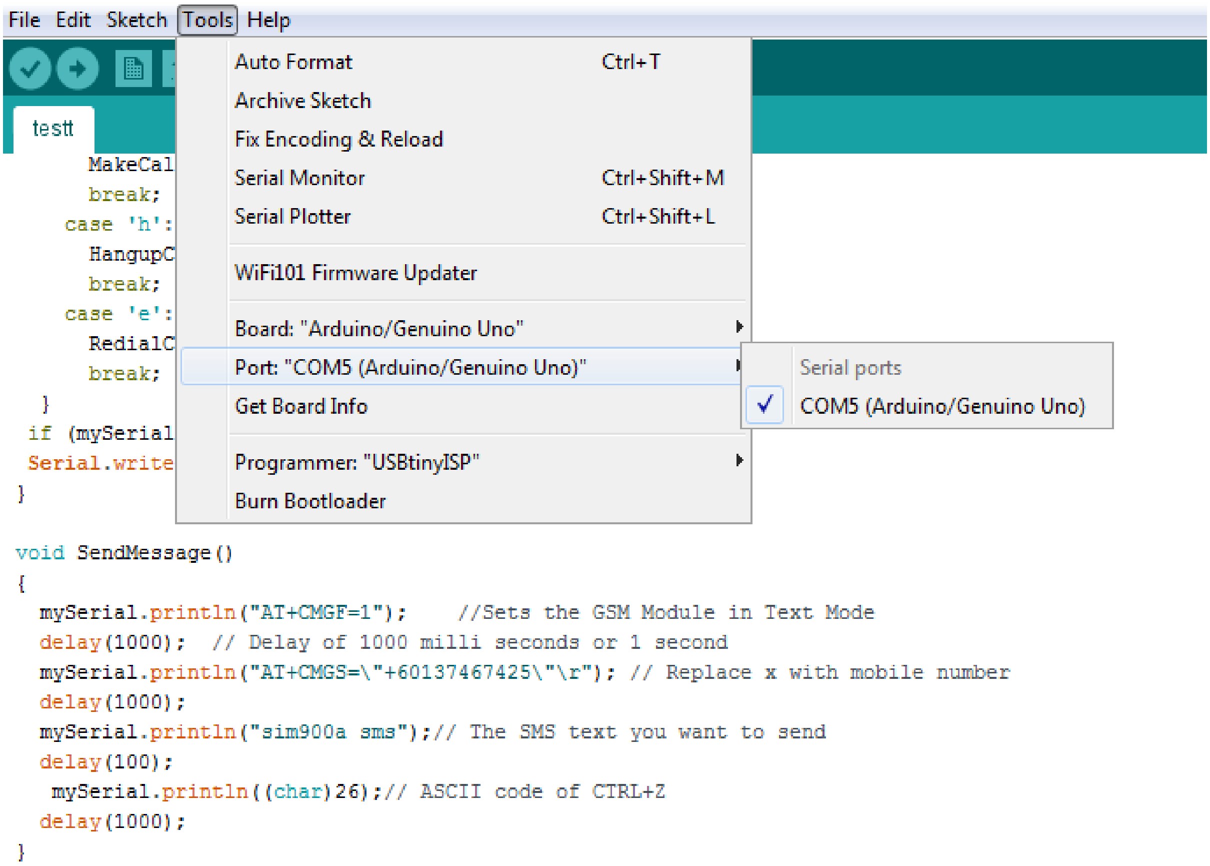
- ● Ask before overwrite
- ○ Overwrite without prompt
- ○ Skip existing files
- ○ Rename automatically

Miscellaneous

- □ Extract archives to subfolders
- □ Keep broken files
- □ Display files in Explorer

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. To be able to interface gsm module with arduino, you will have to download this library and extract it into your Arduino's libraries.

Software Serial.zip attached



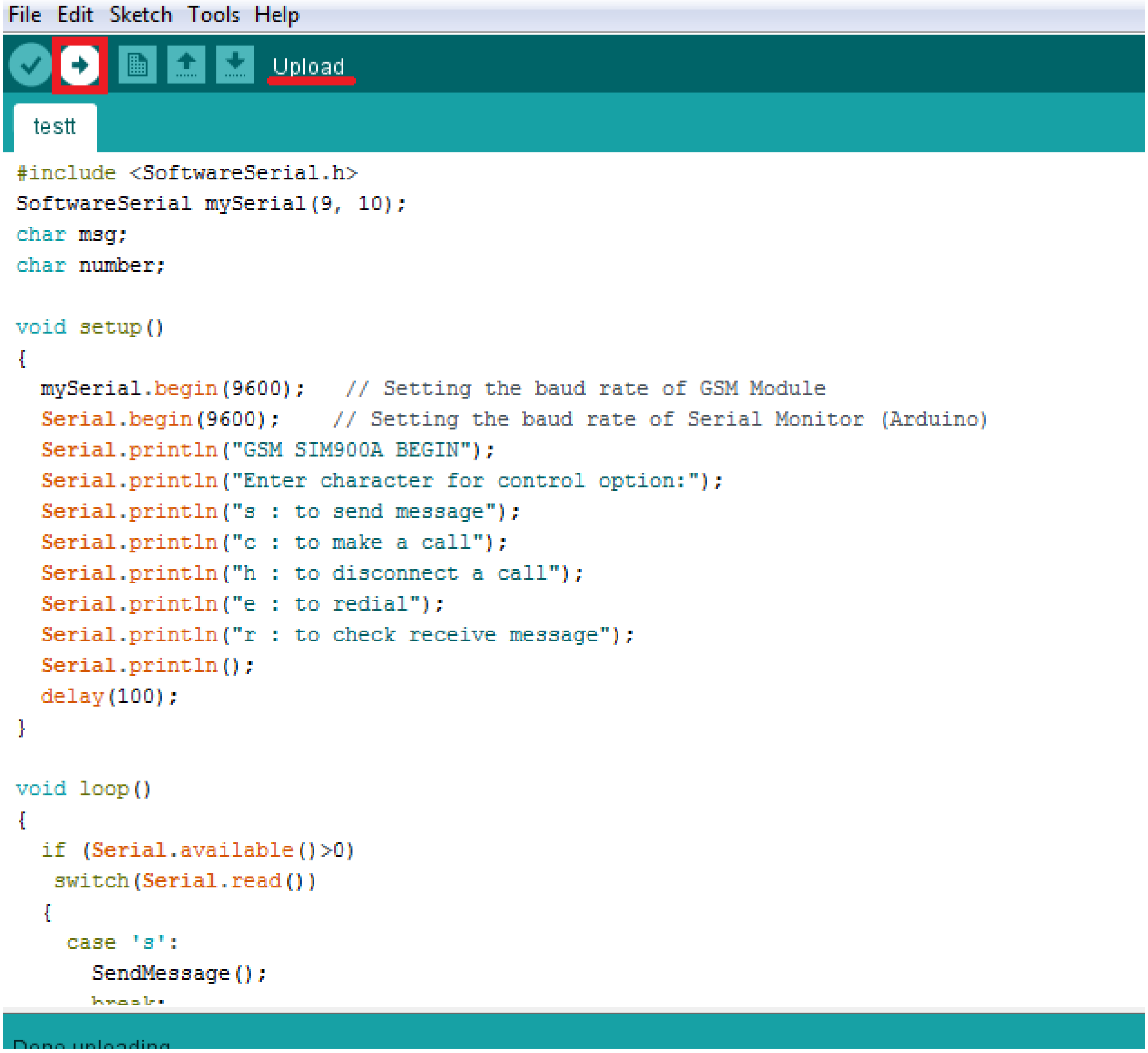
Board: "Arduino/Genuino Uno"

Port: "COM5 (Arduino/Genuino Uno)"

✓ COM5 (Arduino/Genuino Uno)

Programmer: "USBtinyISP"

```
        MakeCal
        break;
      case 'h':
        HangupC
        break;
      case 'e':
        RedialC
        break;
    }
  if (mySerial
  Serial.write
}

void SendMessage()
{
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+60137467425\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("sim900a sms");// The SMS text you want to send
  delay(100);
   mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}
```

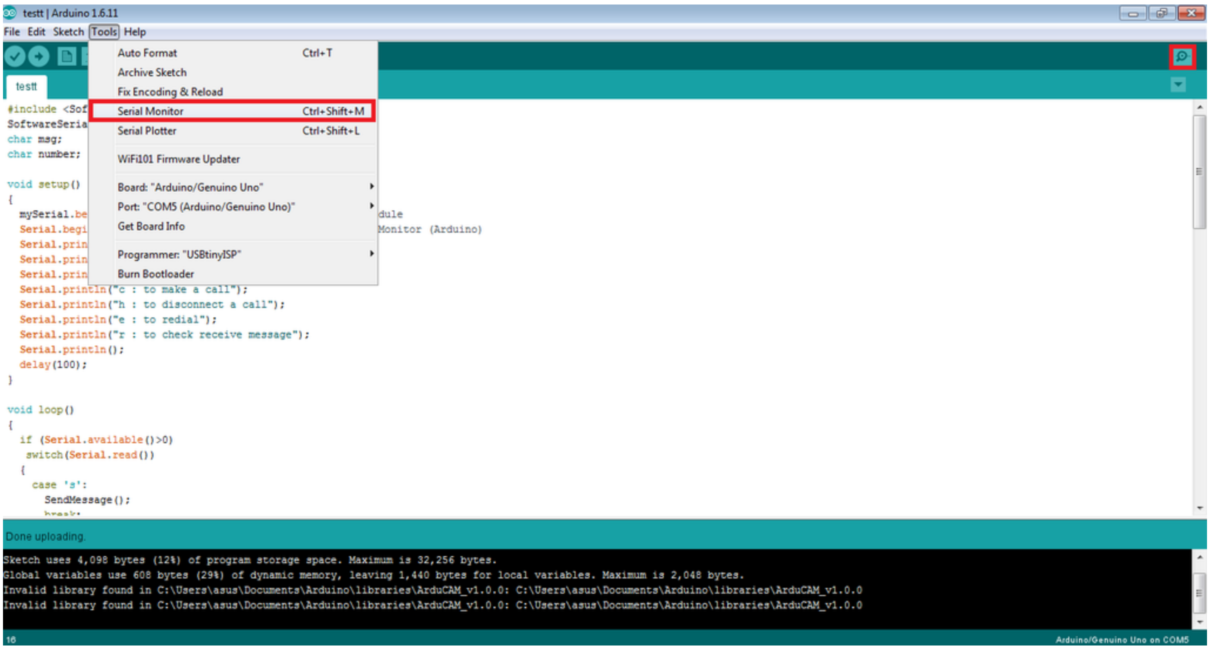


```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(9, 10);
char msg;
char number;

void setup()
{
  mySerial.begin(9600);   // Setting the baud rate of GSM Module
  Serial.begin(9600);    // Setting the baud rate of Serial Monitor (Arduino)
  Serial.println("GSM SIM900A BEGIN");
  Serial.println("Enter character for control option:");
  Serial.println("s : to send message");
  Serial.println("c : to make a call");
  Serial.println("h : to disconnect a call");
  Serial.println("e : to redial");
  Serial.println("r : to check receive message");
  Serial.println();
  delay(100);
}

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

Done uploading.

Download the sample source code below and open it on your Arduino IDE. Select the correct board and port and upload it into your Arduino Uno Board.

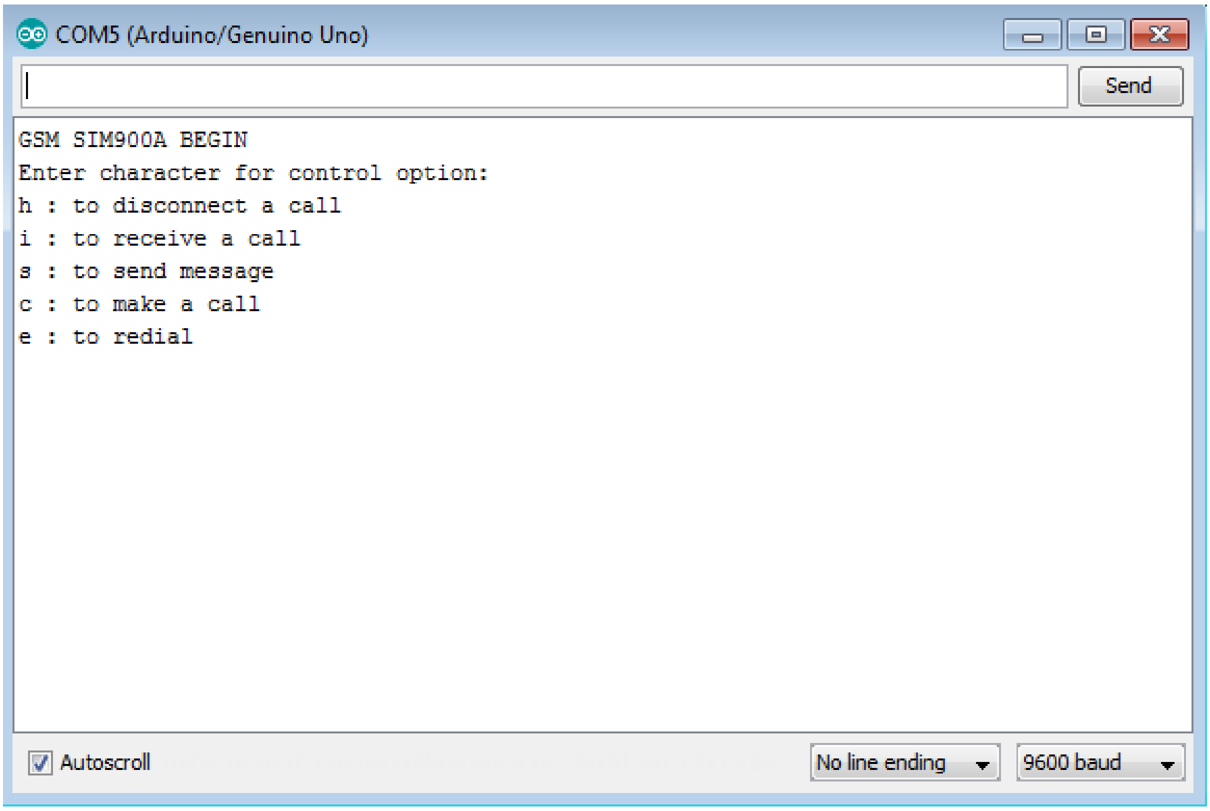
sim900amini.rar


testt | Arduino 1.6.11
File Edit Sketch Tools Help
testt
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM5 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "USBtinyISP"
Burn Bootloader
16
Arduino/Genuino Uno on COM5

```
#include <Sof
SoftwareSeria
char msg;
char number;

void setup()
{
  mySerial.be                                                dule
  Serial.begi                                                Monitor (Arduino)
  Serial.prin
  Serial.prin
  Serial.prin
  Serial.println("c : to make a call");
  Serial.println("h : to disconnect a call");
  Serial.println("e : to redial");
  Serial.println("r : to check receive message");
  Serial.println();
  delay(100);
}

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

Done uploading.

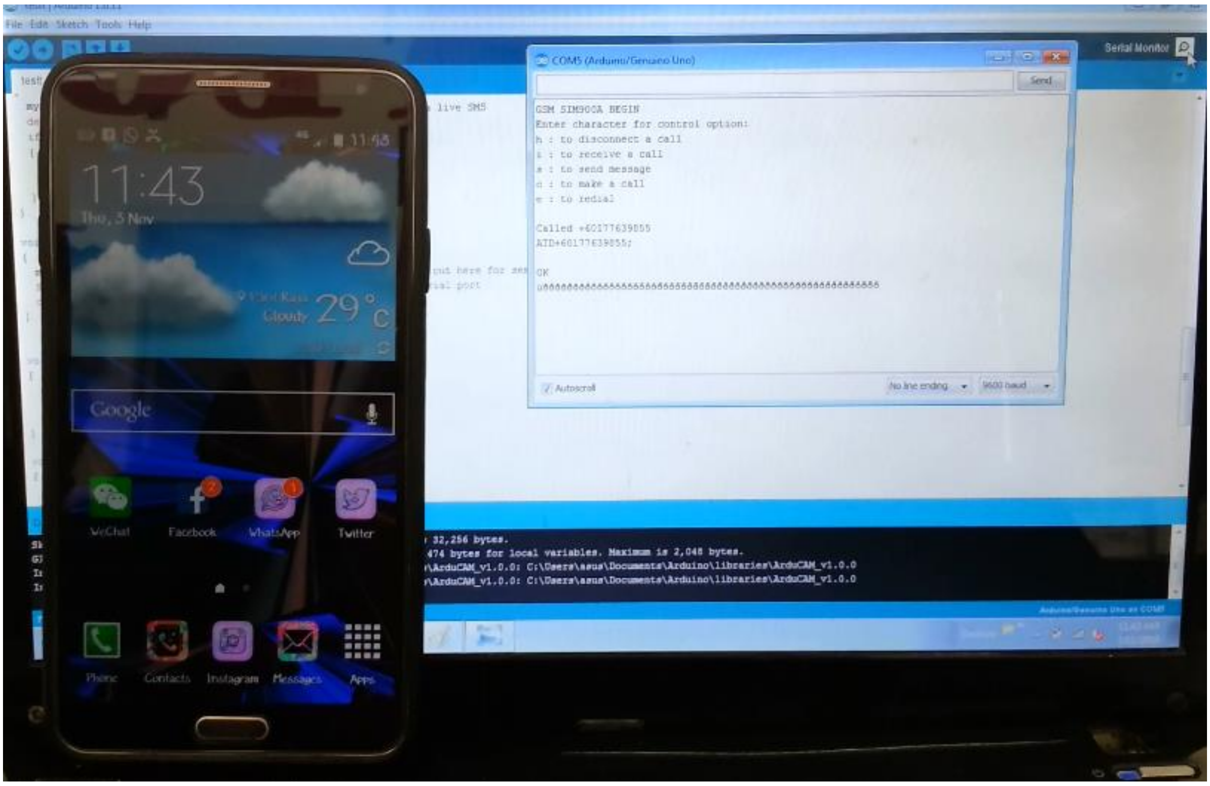
```
Sketch uses 4,098 bytes (12%) of program storage space. Maximum is 32,256 bytes.
Global variables use 608 bytes (29%) of dynamic memory, leaving 1,440 bytes for local variables. Maximum is 2,048 bytes.
Invalid library found in C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
Invalid library found in C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
```



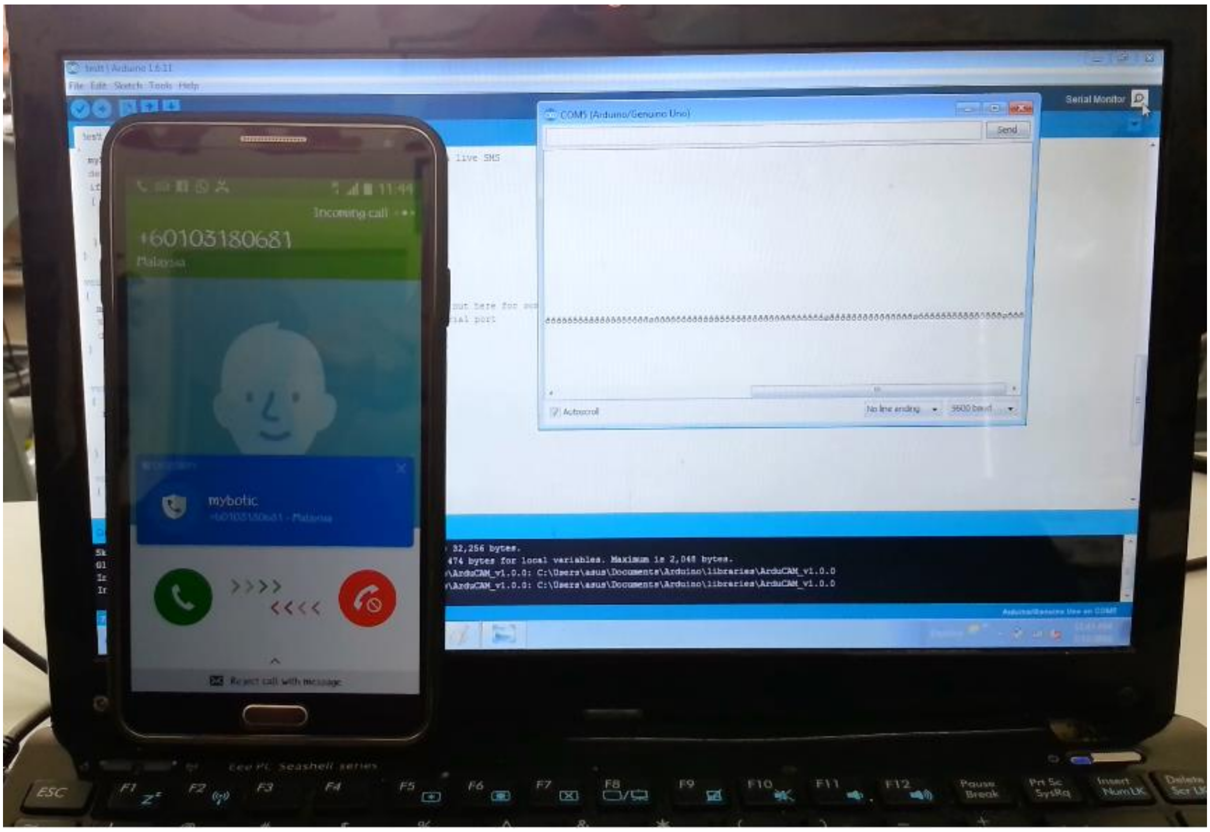
```
GSM SIM900A BEGIN
Enter character for control option:
h : to disconnect a call
i : to receive a call
s : to send message
c : to make a call
e : to redial
```

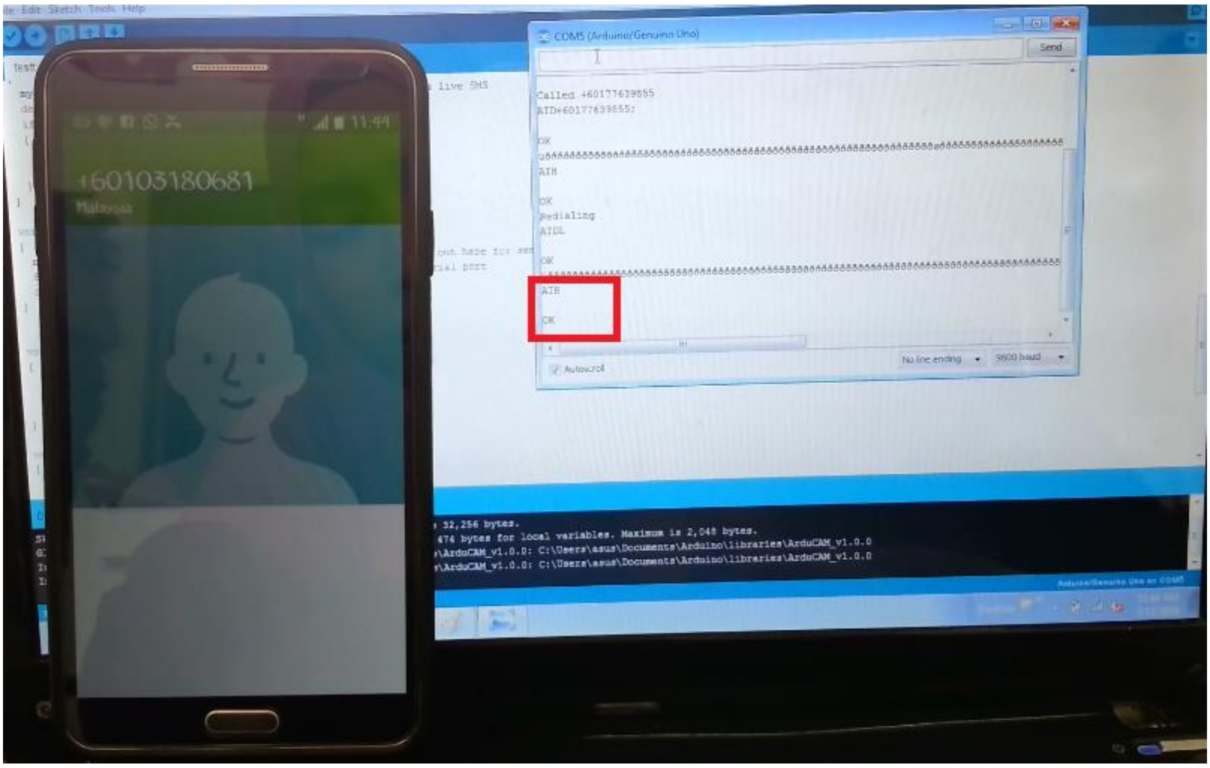
After you has succesfully uploaded your source code, open your serial monitor. Serial monitor will display as shown in the picture above.

Step 8: Result : Call / Redial



```
GSM SIM900A BEGIN
Enter character for control option:
h : to disconnect a call
i : to receive a call
s : to send message
c : to make a call
e : to redial

Called +60177639855
ATD+60177639855;

OK
```





```
Called +60177639855
ATD+60177639655;

OK

ATH

OK
Redialing
ATDL

OK

ATH

OK
```

No line ending
9600 baud

```
32,256 bytes.
474 bytes for local variables. Maximum is 2,048 bytes.
\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
```

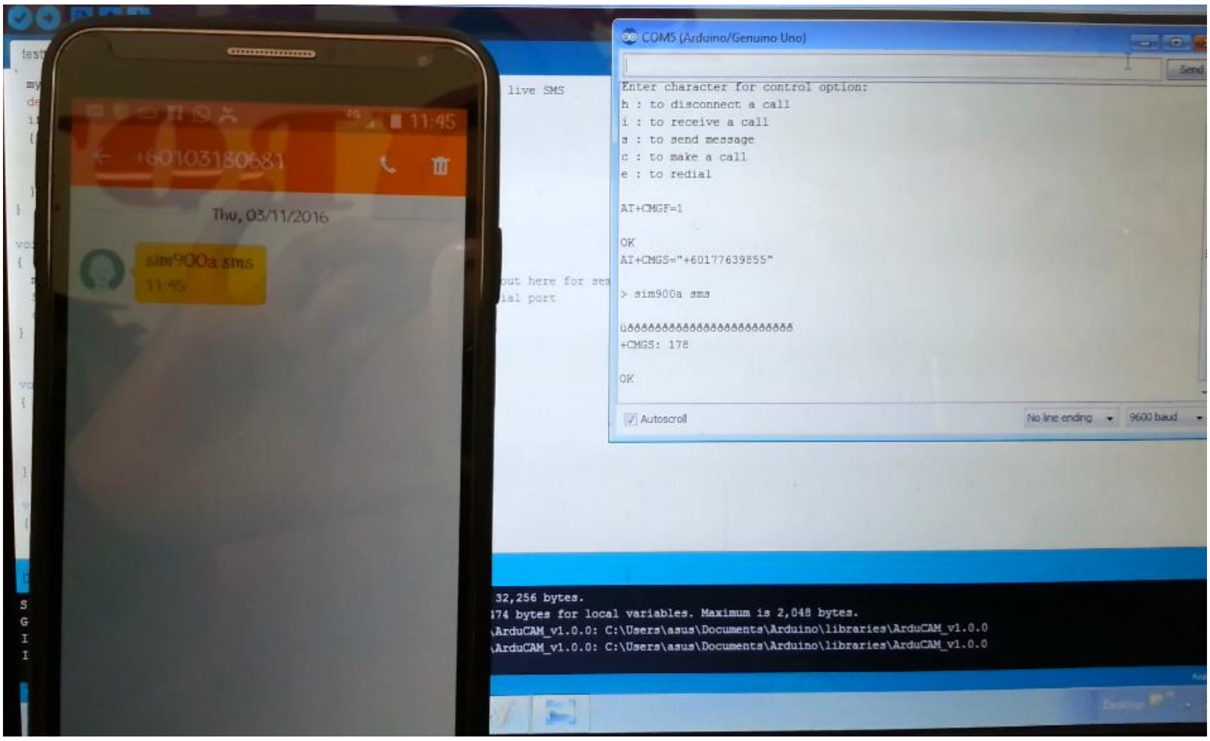
1. As you key-in **c : to make a call**, gsm will read the ATD command and make a call to a phone number you have upload in your source code. (Picture 1 and 2)

2. When you key-in **h : to disconnect/hangup call**, gsm will read the ATH command and disconnect the connection. (Picture 3)

3. When you key-in **e : to redial**, gsm will read the ATDL command and redialing previous number it has called.(Picture 4)

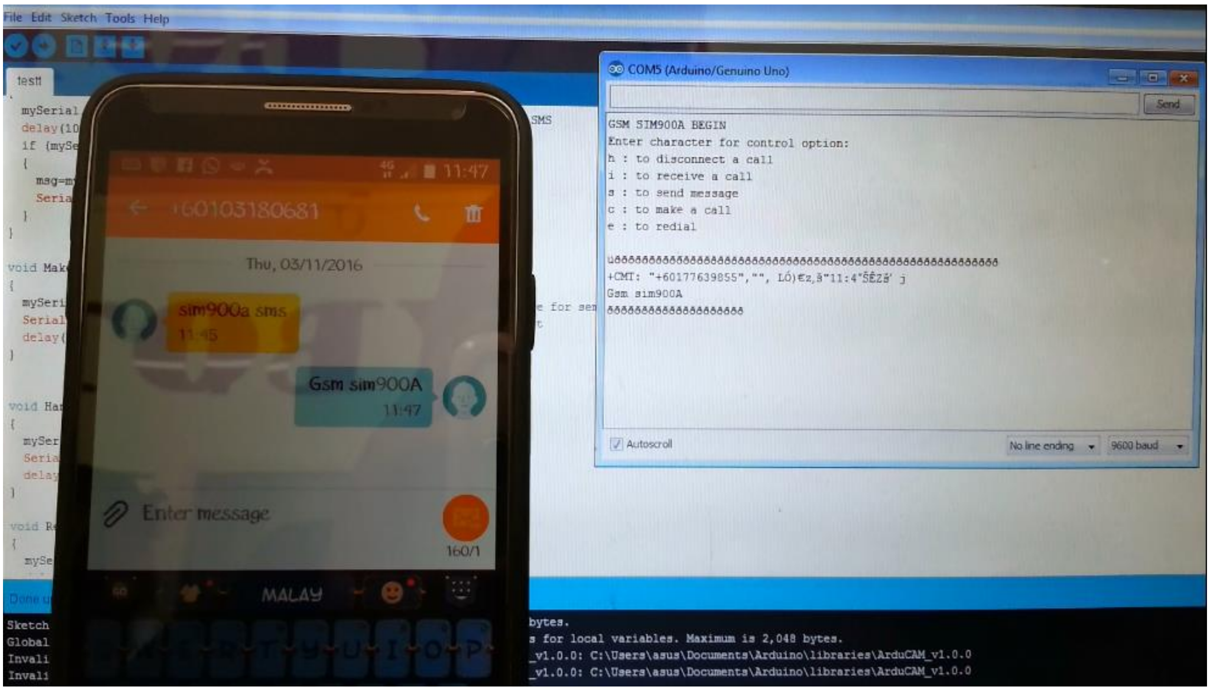
4. When there is an incoming call, you can see RING printed on serial monitor and you can click **i : to receive a call** and GSM's ATA command will be carried out and you will be connected to a call connection. (Picture 5)

Step 9: Result : Send and Receive SMS



```
Enter character for control option:
h : to disconnect a call
i : to receive a call
s : to send message
c : to make a call
e : to redial

AT+CMGF=1

OK
AT+CMGS="+60177639855"

> sim900a sms

uðððððððððððððððððððððððð
+CMGS: 178

OK
```

```
32,256 bytes.
174 bytes for local variables. Maximum is 2,048 bytes.
\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
\ArduCAM_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
```



```
GSM SIM900A BEGIN
Enter character for control option:
h : to disconnect a call
i : to receive a call
s : to send message
c : to make a call
e : to redial

uððððððððððððððððððððððððððððððððððððððððððððððððððð
+CMT: "+60177639855","", LÓ)€z,š"11:4'ŠÊZš' j
Gsm sim900A
ðððððððððððððððððððð
```

```
bytes.
s for local variables. Maximum is 2,048 bytes.
_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
_v1.0.0: C:\Users\asus\Documents\Arduino\libraries\ArduCAM_v1.0.0
```

1. Key-in s to send SMS. Recepient's number and text message printed on serial monitor. **NOTE : You can edit the recepient's phone number and text message on your source code.**

2. When gsm receive a message, text message and number will be printed on serial monitor.