Research
Design Lab

GSM GPRS SIM800C Module

Contents

1. Overview

GSM/GPRS Modem-RS232 is built with the Quad-Band GSM/GPRS engine-SIM800C, works on frequencies 850/900/1800/1900MHz. The module comes with RS232 interface, which allows you to connect PC as well as microcontroller with RS232 Chip(MAX232). The baud rate is configurable from 9600-115200 through AT command. The GSM/GPRS modem is having internal TCP/IP stack to enable you to connect with internet via GPRS. This module is suitable for SMS, Voice as well as DATA transfer application in M2M interface. The onboard Regulated Power supply allows you to connect wide range of unregulated power supply. Using this modem, you can make audio calls, SMS, Read SMS, attend the incoming calls and internet etc., through simple AT commands.

2. Features

- Quad-Band GSM/GPRS 850/900/1800/1900MHz.
- RS232 interface for the direct communication with computer or MCU kit.
- Configurable Baud Rate.
- Power controlled using 29302WU IC.
- ESD Compliance.
- Enable with MIC and Speaker socket.
- Enable with Audio Jack
- With push push sim card holder.
- With Stub antenna and SMA connector.
- Input Voltage: 12V DC.
- High quality PCB FR4 Grade with FPT certified

3. SIM800C General Features

- Quad-Band GSM/GPRS 850/900/1800/1900MHz.
- GPRS Multi-Slot Class 12/10
- GPRS Mobile Station Class B
- Compliant to GSM Phase 2/2+ : Class 4(2W @ 850/900MHz)
  Class 1(1W @ 1800/1900MHz)
- Dimensions: 17.6*15.7*2.3mm
- Weight: 1.3g
- Control via AT Commands: (3GPP TS 27.007,27.006 and SIM Com enhanced AT Commands)
- Supply Voltage Range 3.4 ~ 4.4V
- Low Power Consumption.
- Operation Temperature: -40℃ ~85℃.

4. Software Features

- 0710 MUX Protocol
- Embedded TCP/UDP Protocol
- FTP/HTTP
- MMS
- POP3/SMTP
- DTMF
- Jamming Detection
- Audio Record
- SSL
- Bluetooth 3.0(Optional)

5. Applications

- Industrial automation.
- GPRS based data logging.
- GPRS and GPS application.
- Home automation.
- Health monitoring.
- Agriculture automation
- Vehicle tracking.
- Remote monitoring and controlling.
- GPRS based Weather report logging
- GSM GPRS based Security alert.
- GPRS based remote terminal for file transfer.
- IVRS.
- Bulksms sending

6. Specification

6.1. Specifications for GPRS Data

- GPRS Class 12: max.85.6 kbps (downlink/uplink)
- PBCCH Support
- Coding Scheme CS 1,2,3,4
- PPP-Stack
- USSD

6.2. Specifications for SMS via GSM/GPRS

- Point to Point MO and MT
- SMS Cell Broad Cast
- Text and PDU Mode

6.3. Specifications for Voice

- Tricodec: Half Rate(HR)

  Full Rate(FR)

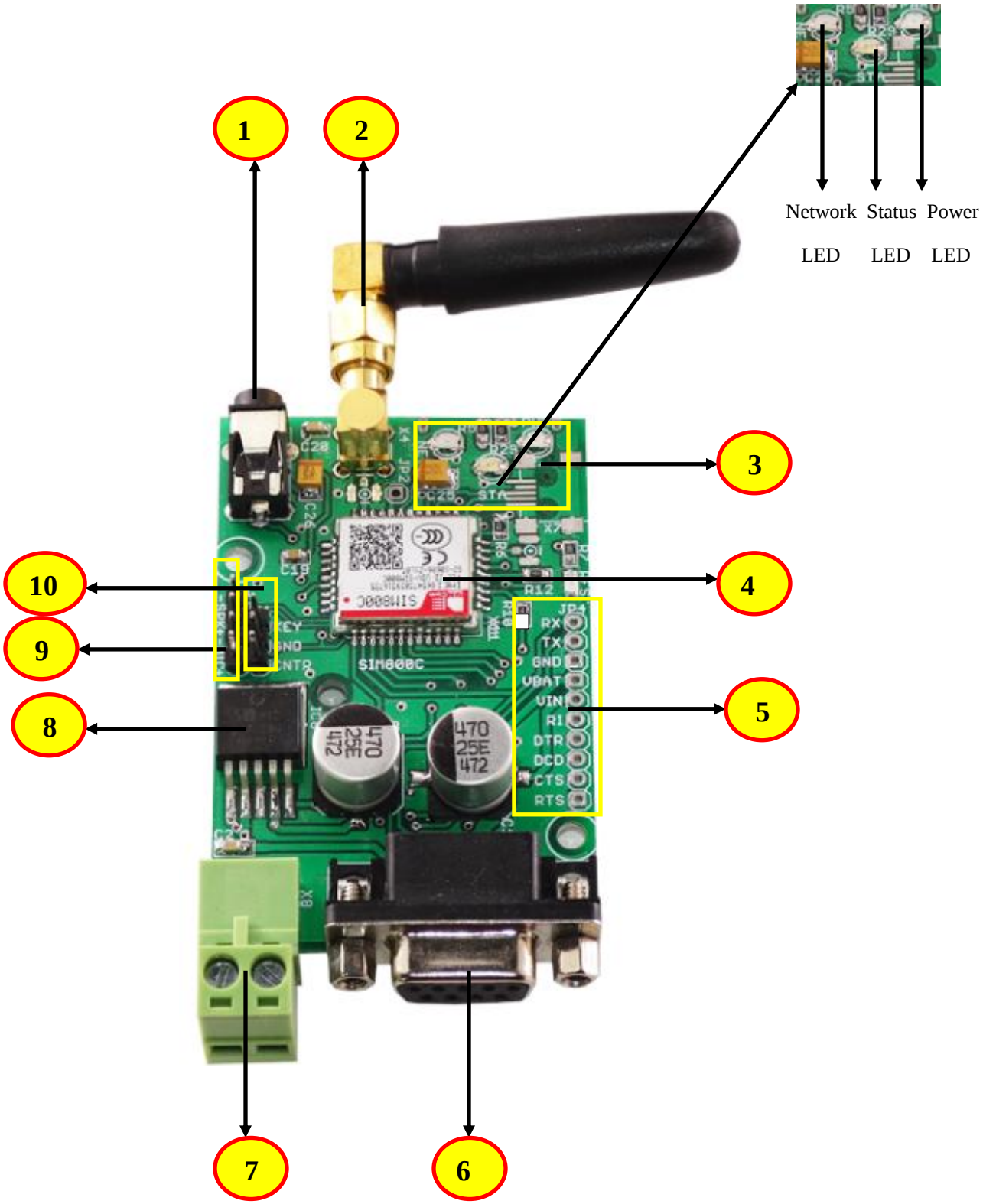
  Enhanced Full Rate(EFR)
- AMR: Half Rate(HR)

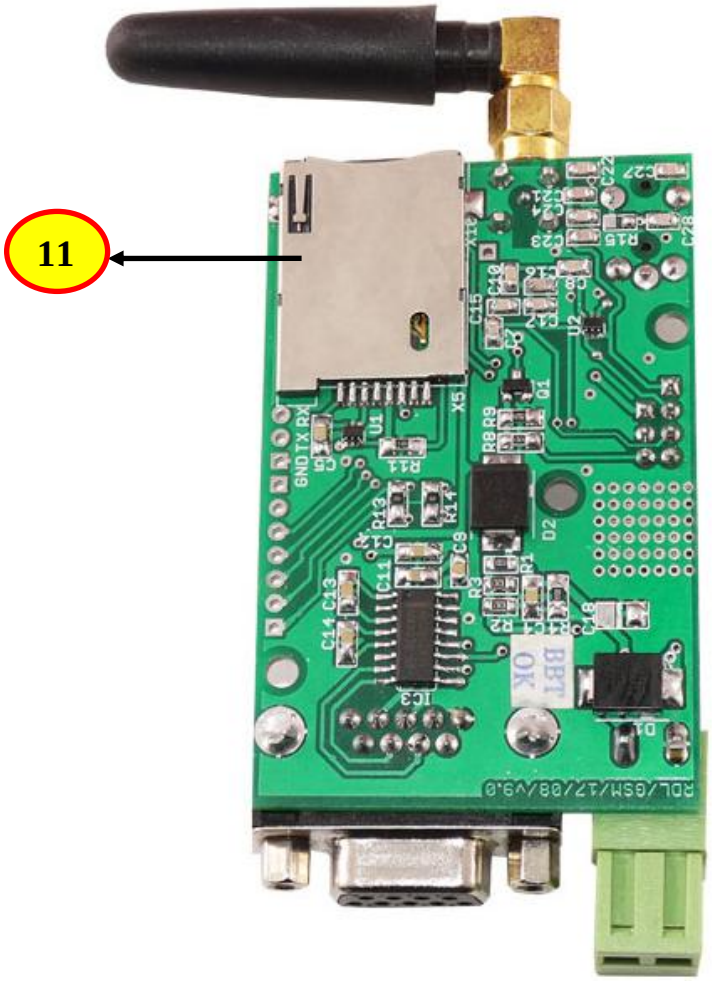
  Full Rate(FR)
- Hands-Free Operation (Echo Suppression).

7. Interfaces

- 42 SMT Pins Including
- Analog Audio Interface
- RTC Backup
- USB Interface
- Serial Interface
- Interface to External SIM 3V/1.8V
- GPIO
- ADC
- GSM Antenna Pad
- Bluetooth Antenna Pad

8. Narration


1
2
3
4
5
6
7
8
9
10
Network LED
Status LED
Power LED
RX
TX
GND
VBAT
VIN
RI
DTR
DCD
CTS
RTS
SIM800C



1. Audio Jack
2. Stub antenna with SMA connector.
3. Network, Status and Power indicator.
4. SIM800C Module
5. General GPIO SIM800C
6. DB9 Connector
7. Power Supply 12V/2A
8. DC to DC Converter (29302WU IC)
9. Pins for Speaker and Mic
10. Power On Selection Pins
11.SIM Card Slot

9. Basic AT Commands for Testing

9.1. GSM AT Commands :

- To Check Modem

  AT↲

  OK

- To Change SMS Sending Mode

  AT+CMGF=1↲

  OK

- To Send New SMS

  AT+CMGS=”Mobile Number.”↲

  <Message

  {CTRL+Z}

- To Receive SMS

  AT+CMGD=1↲ {To delete the message in buffer}

  AT+CMGR=1↲ {To receive the first messge AT+CMGR=1}

  {To receive the second messge AT+CMGR=2 and so on}

  +CMGL: 1,”REC READ”,”+85291234567”,”07/05/01,08:00:15+32”145,37

  <Message

- Preferred SMS Message Storage

  AT+CPMS=?↲

  +CPMS: (”SM”),(“SM”),(“SM”)

  OK

  AT+CPMS=?↲

  +CPMS: ”SM”,19,30,“SM”,19,30,”SM”,19,30

- To Make Voice Call

  ATD9876543210; ↲

- To Redial Last Number

  ATDL ↲

- To Receive Incoming Call

  ATA ↲

- To Hangup Or Disconnect Call

  ATH ↲

- To Set A Particular Baud Rate

  AT+IPR=?↲ {To view the baud rate value}

  AT+IPR=0↲ {To set the modem to autobauding mode}

- Operator Selection

  AT+COPS=?↲

  OK

  AT+COPS?↲

  +COPS:0,0,"AirTel"

  OK

- To Set Cellular Result Codes For Incoming Call Indication

  AT+CRC=?↲

  +CRC: (0-1)

  OK

  AT+CRC?↲

  +CRC: 0

  OK

  AT+CRC=1↲

  +CRC: 1

  OK

  +CRING: VOICE

- Read Operator Names

AT+COPN=? ↲

OK

AT+COPN ↲

+COPN: “472001″,”DHIMOBILE”

+COPN: “60500 +COPN: “502012″,”maxis mobile”

+COPN:

+COPN: “502013″,”TMTOUCH”

+COPN

+COPN: “502016″,”DiGi”

+COPN: “502017″,”TIMECel”"

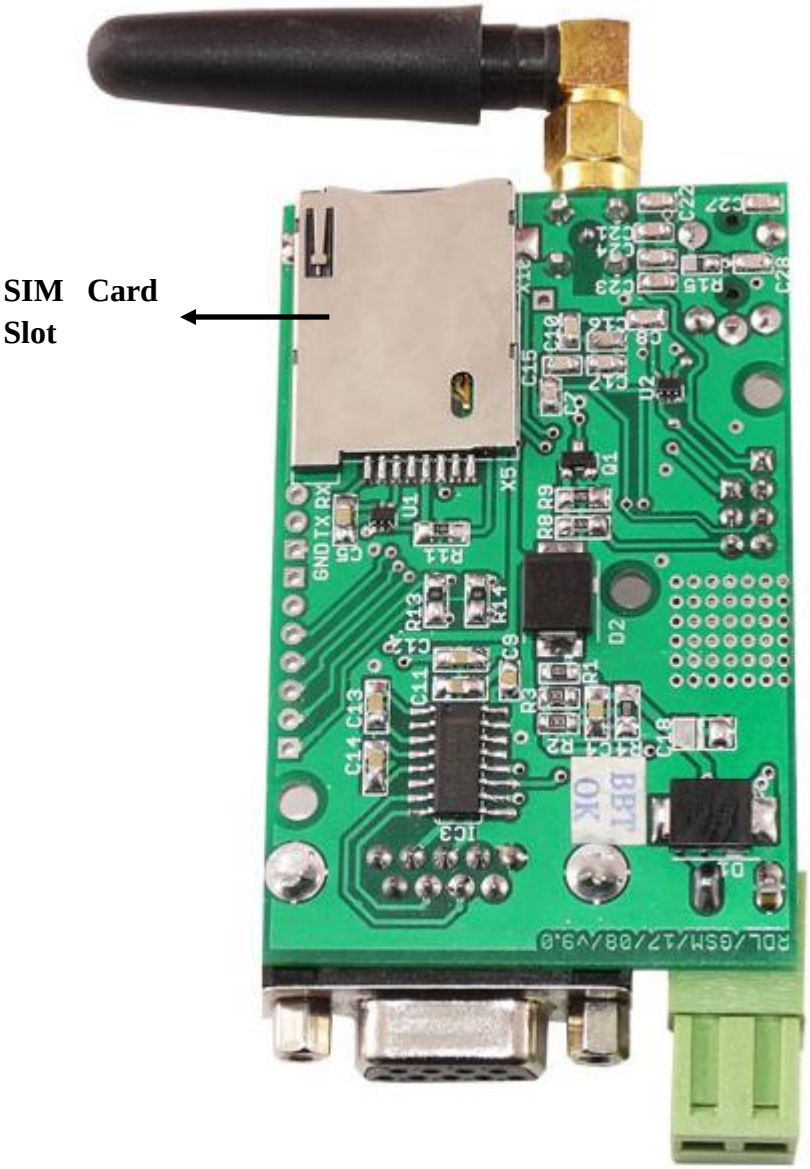
+COPN: “502019″,”CELCOM GSM”

9.2. GPRS AT Commands :

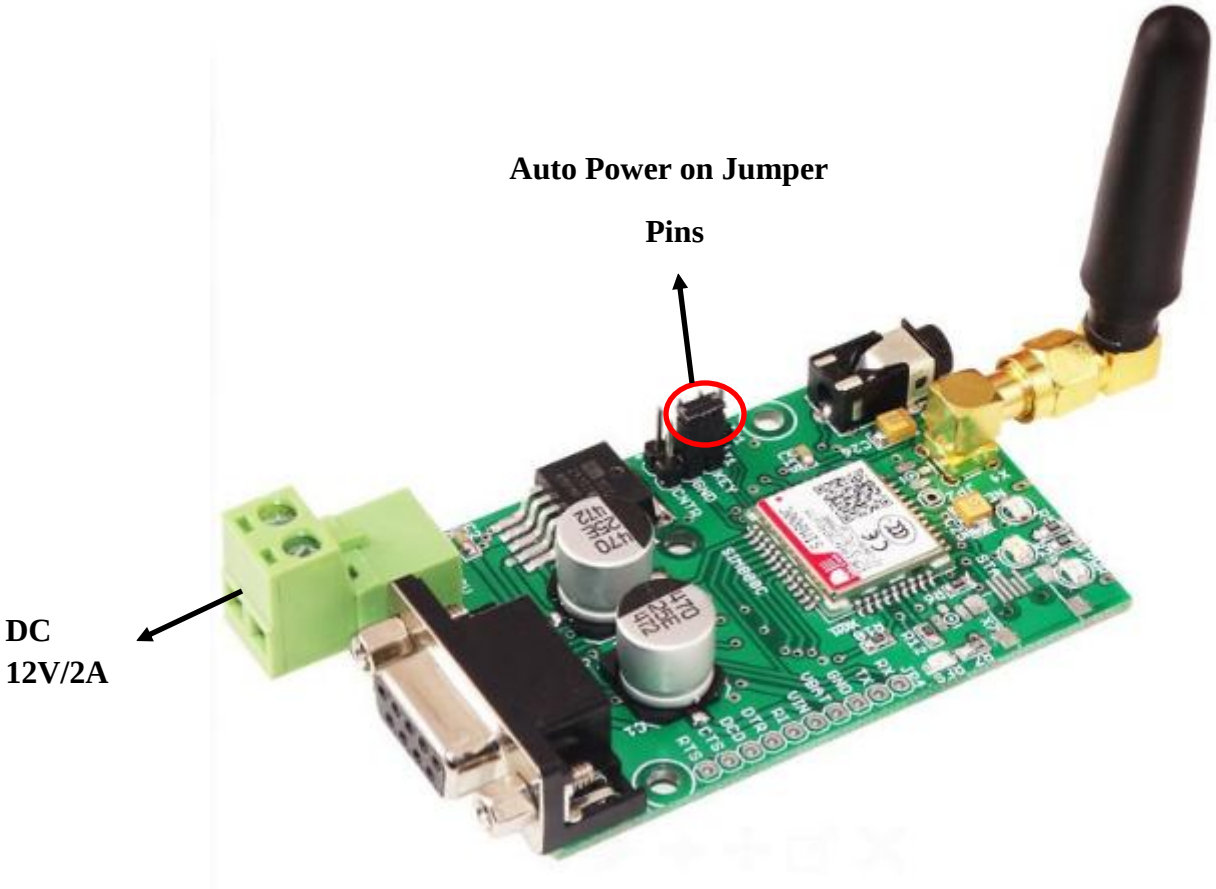
Commands	Description
AT+CGATT ↲	ATTACH/DETACH FROM GPRS SERVICE
AT+CGDCONT ↲	DEFINE PDP CONTEXT
AT+CGQMIN ↲	QUALITY OF SERVICE PROFILE (MINIMUM ACCEPTABLE)
AT+CGQREQ↲	QUALITY OF SERVICE PROFILE (REQUESTED)
AT+CGACT↲PDP	CONTEXT ACTIVATE OR DEACTIVATE
AT+CGDATA↲	ENTER DATA STATE
AT+CGPADDR↲	SHOW PDP ADRESS
AT+CGCLASS↲	GPRS MOBILE STATION CLASS
AT+CGEREP↲	CONTROL UNSOLICITED GPRS EVENT REPORTING
AT+CGREG↲	NETWORK REGISTRATION STATUS
AT+CGSMS↲	SELECT SERVICE FOR MO SMS MESSAGES
AT+CGCOUNT↲	GPRS PACKET COUNTERS

10. Module Setup

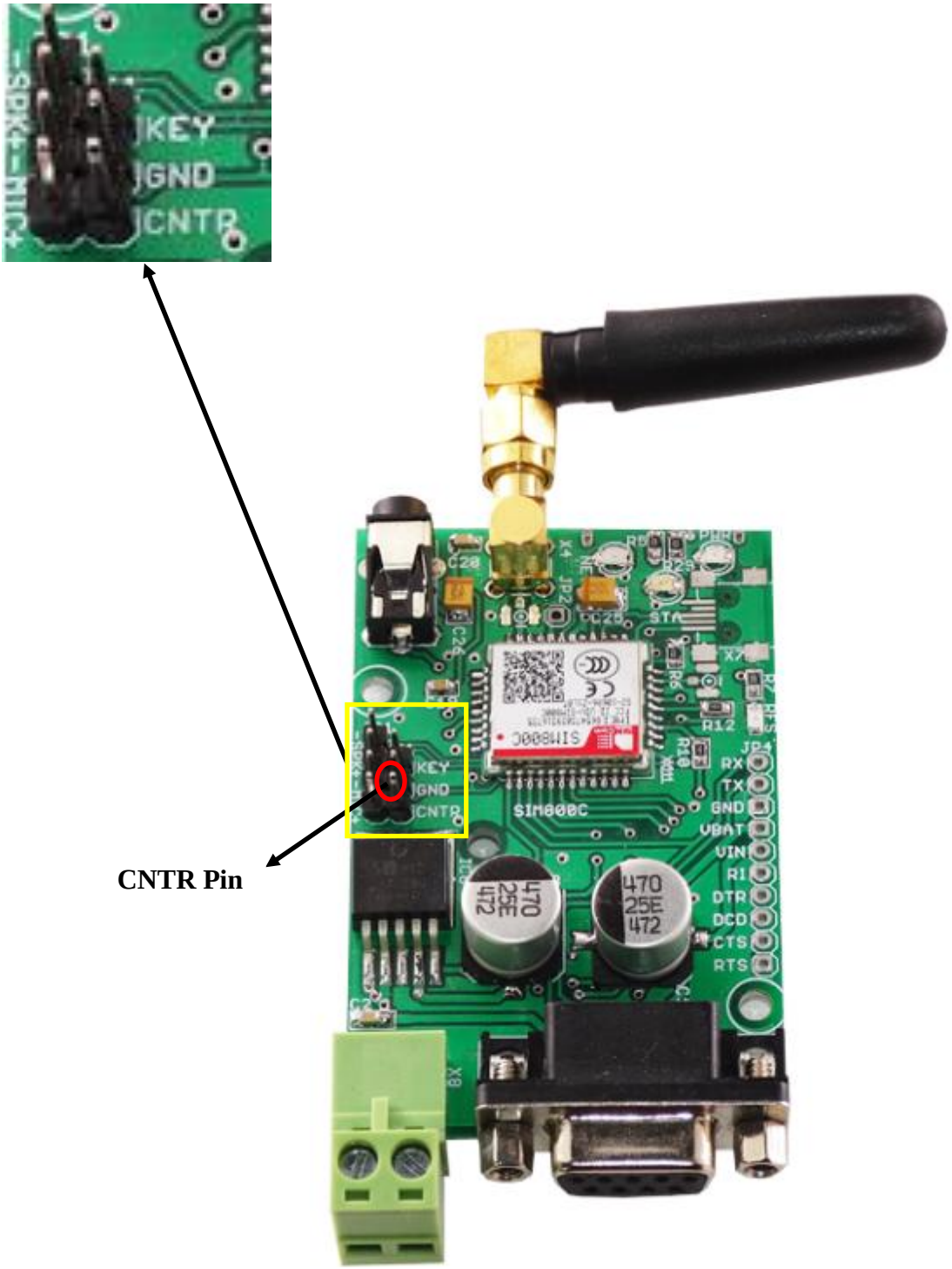
Step 1 : Insert SIM Card into the SIM Card Slot



Step2: Plug in12V-2A DC power adapter, power led is lit(place jumper between KEY and GND pin for only to turn ON automatically).



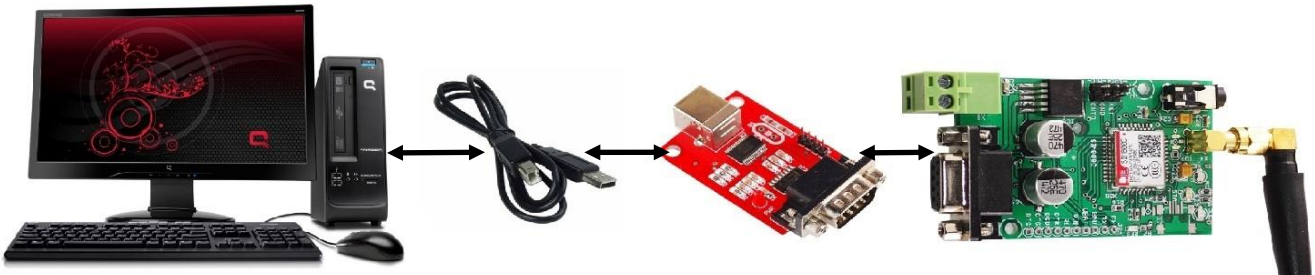
Step 3: Give Rising Edge Pulse to CNTR pin (External Power on Key) using any external controller (To turn on manually without jumper)



Step 4: Connect the PC to DB9 Connector of the GSM GPRS SIM800C Module via USB To RS232 Converter Module given in below link and USB Cable A to B.

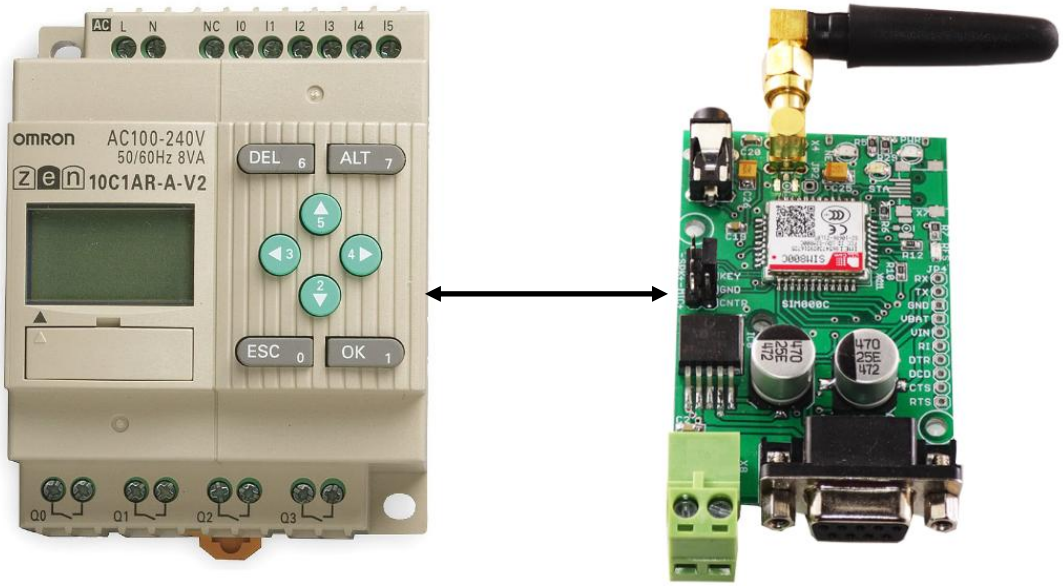
https://researchdesignlab.com/usb-to-rs232-converter.html

Interfacing PC with GSM GPRS SIM800C Module



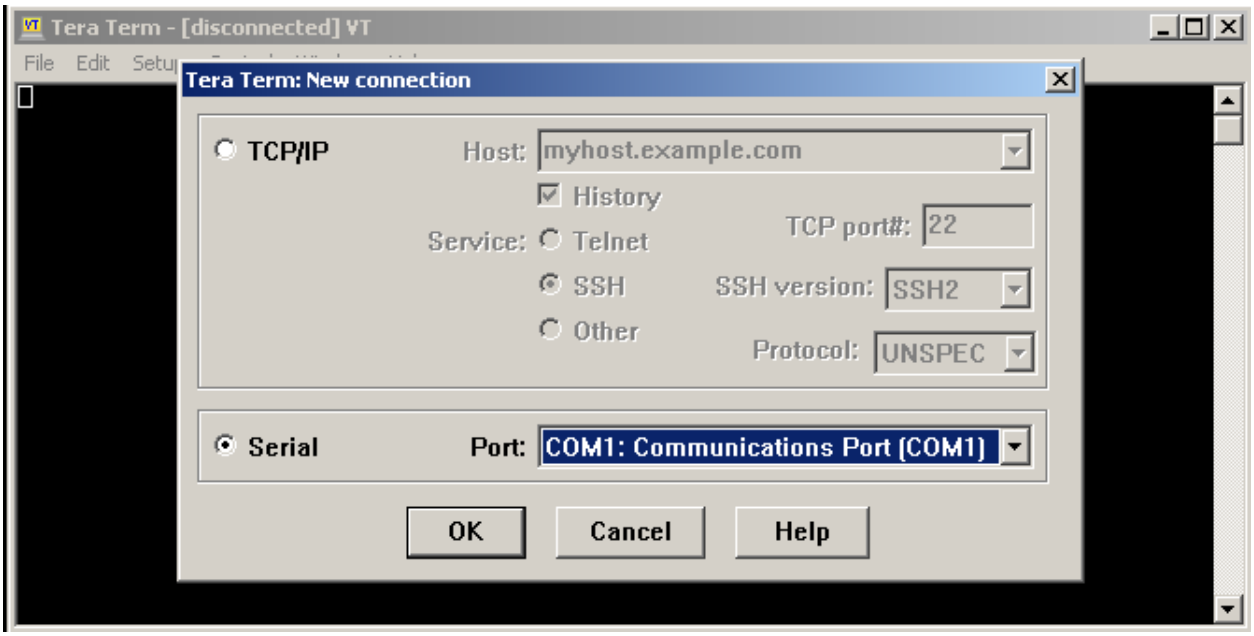
PC **USB Cable A to B** **USB To RS232 Converter Module** **GSM/GPRS SIM800C Module**

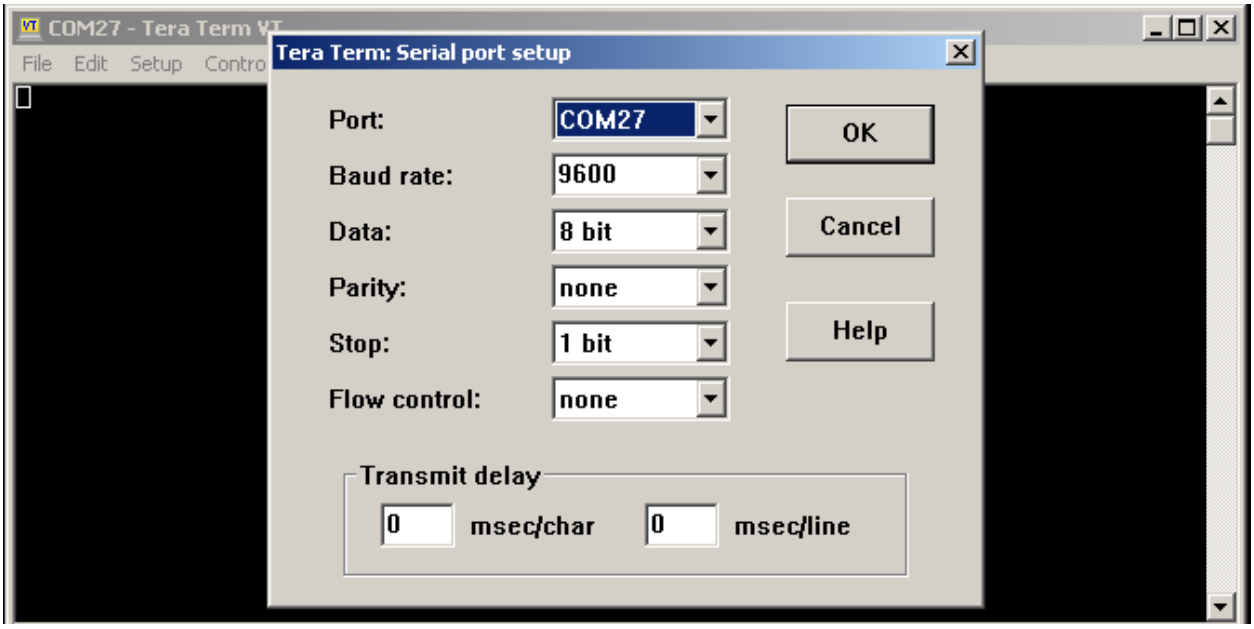
Interfacing PLC with GSM GPRS SIM800C Module

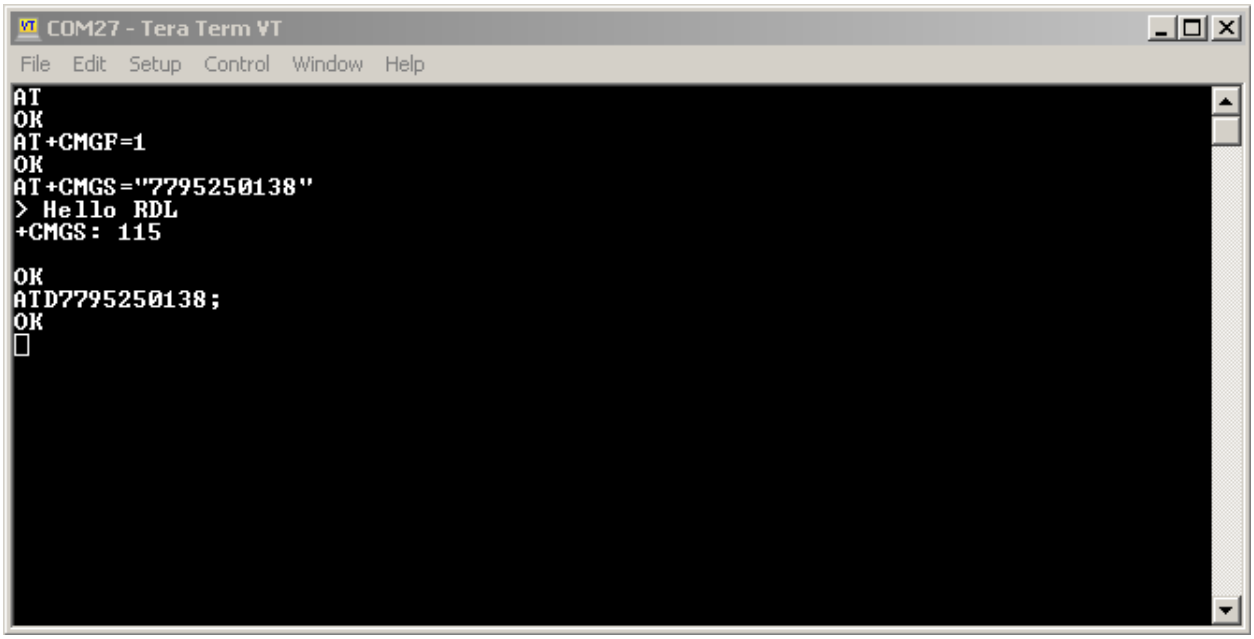


PLC **GSM/GPRS SIM800C Module**

Step 6: Open any Serial Terminal software(eg: Tera Term, HyperTerminal), choose appropriate COM port and then use AT commands listed in this manual for basic testing GPRS GSM messaging and voice calling.



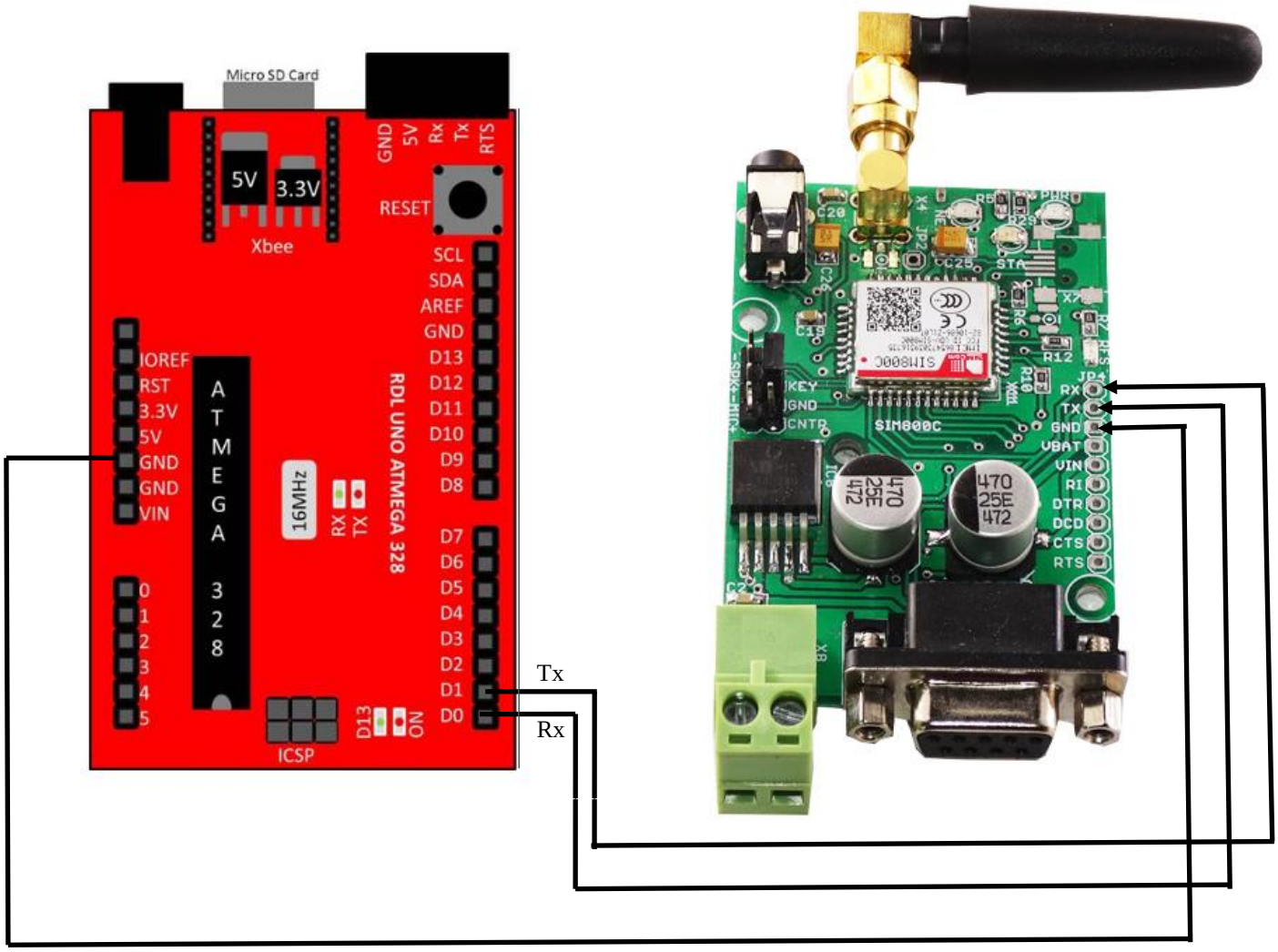



COM27 - Tera Term VT
File Edit Setup Control Window Help
AT
OK
AT+CMGF=1
OK
AT+CMGS="7795250138"
> Hello RDL
+CMGS: 115
OK
ATD7795250138;
OK

.

11. Block Diagrams

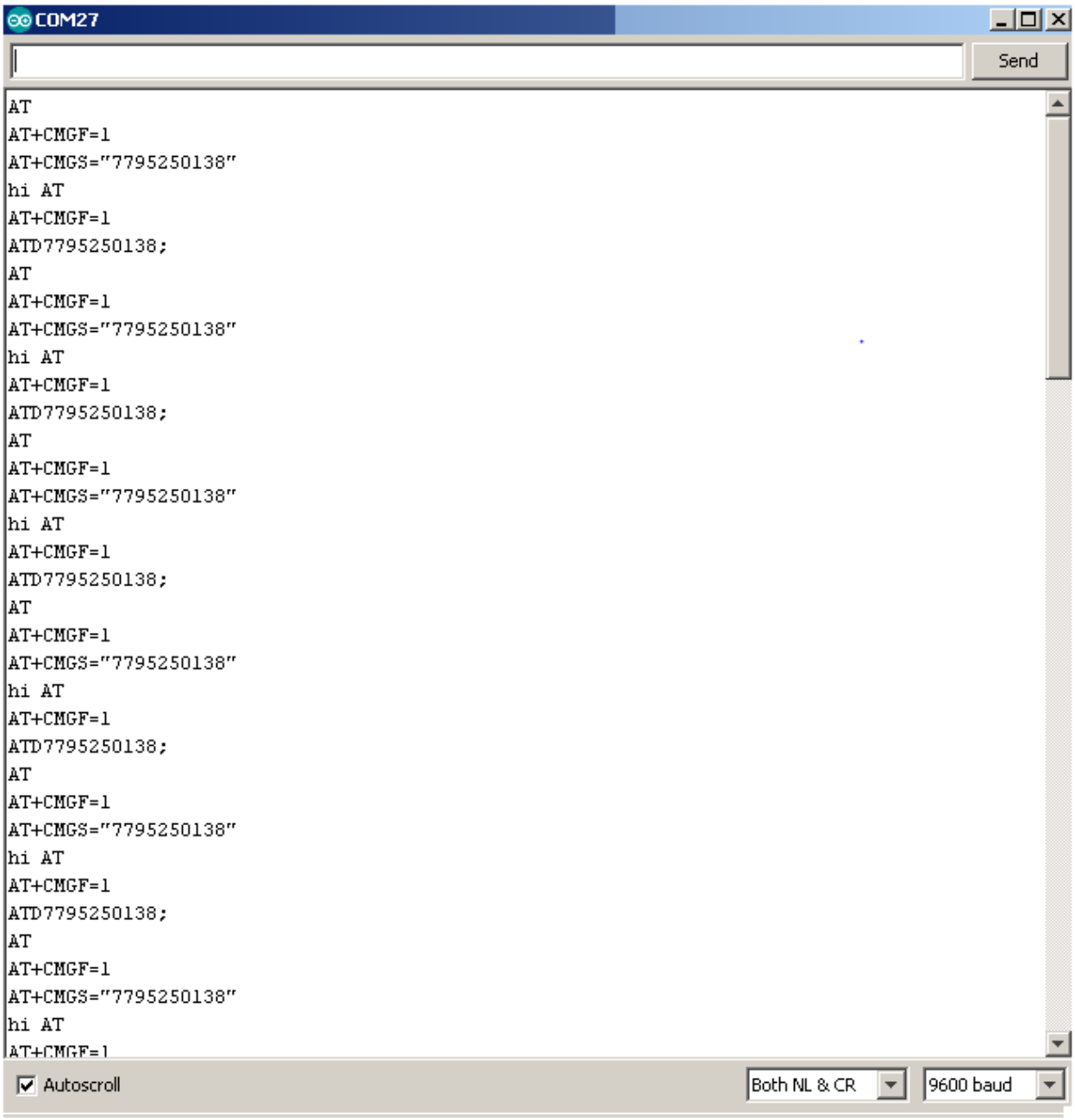
Interfacing Arduino UNO and GSM GPRS SIM800C Module



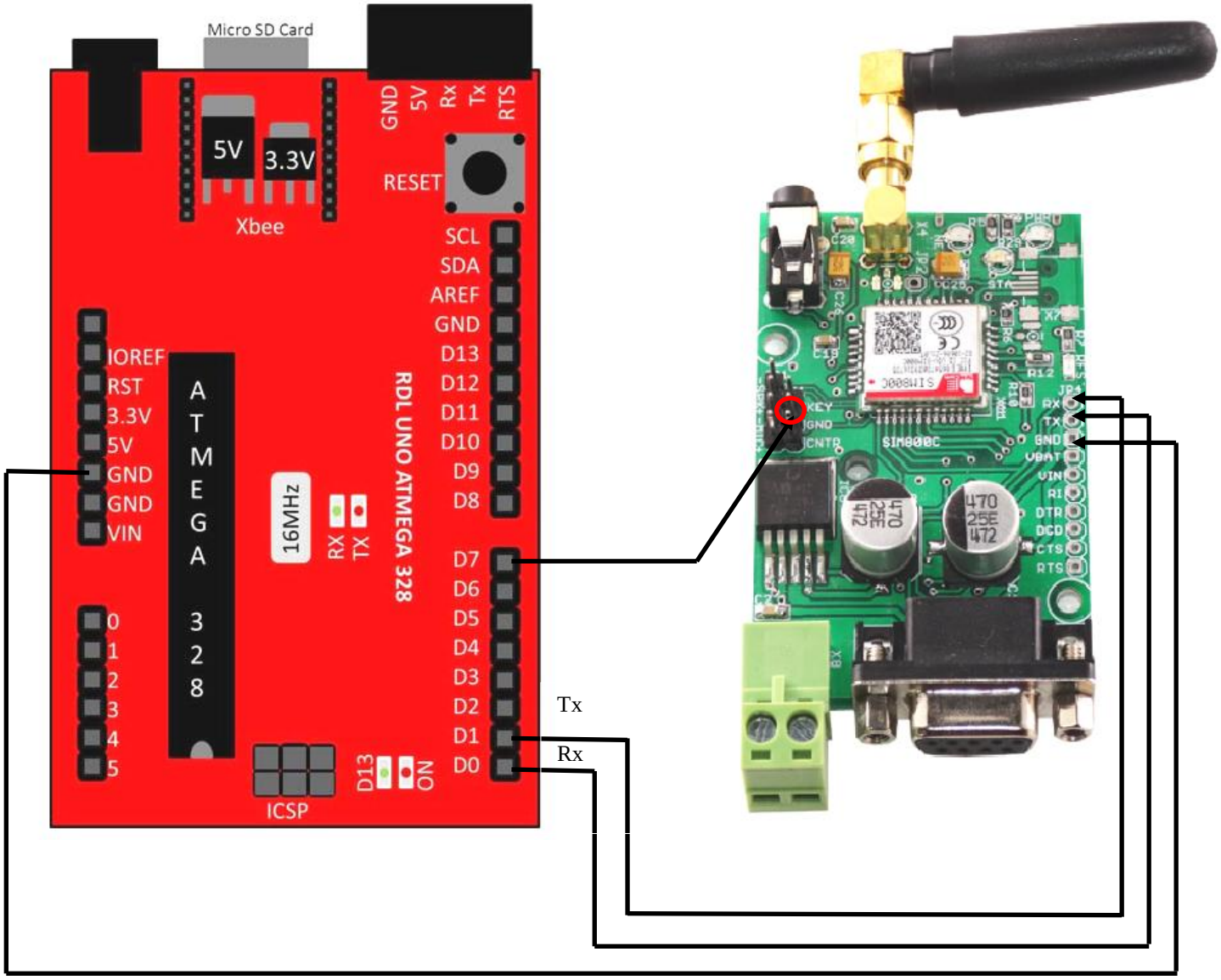
Arduino Code for Sending SMS and Making Voice Call

```
void setup()
{
Serial.begin(9600);
delay(5000);

}

void loop()
{
Serial.println("AT");//TO CHECK THE MODEM
delay(1000);
Serial.println("AT+CMGF=1"); //TO CHANGE THE SMS SENDING MODE
delay(1000);
Serial.println("AT+CMGS=\"7795250138\""); //CHANGE TO DESTINATION NUMBER
delay(1000);
Serial.print("hi"); // MESSAGE TO BE SENT
Serial.write(26);
delay(1000);
Serial.println("AT");
delay(1000);
Serial.println("AT+CMGF=1");
delay(1000);
Serial.println("ATD7795250138;"); //TO MAKE VOICE CALL
delay(1000);
}
```

Output


COM27
Send
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
ATD7795250138;
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
ATD7795250138;
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
ATD7795250138;
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
ATD7795250138;
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
ATD7795250138;
AT
AT+CMGF=1
AT+CMGS="7795250138"
hi AT
AT+CMGF=1
Autoscroll
Both NL & CR
9600 baud

Interfacing Arduino UNO and GSM GPRS SIM800C Module for GPRS Communication



Arduino Code for GPRS Connection

```
#include <SoftwareSerial.h>


SoftwareSerial mySerial(2, 3); // RX, TX


char aux_string[30];
String buffer1="/code2.php?V1=78&V2=98&V3=67&V4=65&V5=56"; //Values to be uploaded in the server
int8_t K=0;
int8_t answer;
int x = 0;

void setup ()
{
  pinMode(7, OUTPUT);
    digitalWrite(7, LOW);
  Serial.begin(9600);
  delay(100);
  mySerial.begin(9600);
  Serial.println("STARTING...");
  power_on();
  delay(3000);

  sendATcommand("AT+CMGF=1", "OK", 1000);//To Change SMS Sending Mode
  delay(1000);

  delay(1000);
  Serial.println("GSM IS READY...");
 }


void loop ()
 {

  UPLOAD();
 }


 void UPLOAD()
 {

   delay(300);
 answer = sendATcommand("AT+CIPSTART=\"TCP\",\"kanwalk.eu5.org\",\"80\"",
 "CONNECT OK", 15000);// To Start up TCP Connection,Change to your domain link
```

```
if (answer == 1)
 {
  K=10;
  delay(500);
  Serial.println("Connected");
  answer = sendATcommand("AT+CIPSEND", ">", 5000);  // To send Data through TCP
  if (answer == 1)
  {

  mySerial.print("GET ");
  mySerial.print(buffer1);
  mySerial.println(" HTTP/1.1");
  mySerial.print("Host: ");
  mySerial.println("kanwalk.eu5.org");
  mySerial.println("Connection: close");
  mySerial.println();
  mySerial.write(byte(0x1A));



 answer = sendATcommand("", "HTTP/1.1 200 OK", 15000);//To check the response of uploading
     if (answer == 1)
     {
     Serial.println("Uploaded");
     mySerial.flush();
     }
     }
    }
 }


 void power_on()
 {

    uint8_t answer=0;

     start:
     delay(2000);
       digitalWrite(7, HIGH);
     delay(2000);
       digitalWrite(7, LOW);
       answer=0;
       while(answer==0)
       {
```

```
  answer = sendATcommand("", "Call Ready", 20000);
  answer = sendATcommand("AT", "OK", 2000);//To Check the Modem
  answer = sendATcommand("AT+IPR=9600", "OK", 2000);//To Set A Particular Baud Rate
  if(answer==1)
     {
      answer = sendATcommand("AT", "OK", 2000);//To Check the Modem

        if(answer==1)
        {
          delay(300);
         sendATcommand("AT+CMGF=1","OK", 1000); //To Change SMS Sending Mode


        }
      }

   }


   sendATcommand("AT+CFUN=1", "OK", 1000);//To set the phone functionality
   answer = sendATcommand("AT+CSTT=\"Tata Docomo\",\"\",\"\"", "OK", 5000);//To Start Task and Set APN
   delay(1000);
   sendATcommand("AT+CIFSR", "OK", 15000);//To get Local IP Address

}


int8_t sendATcommand(char* ATcommand,char* expected_answer1,unsigned int timeout){

    uint8_t x=0,  answer=0;
    char response[200];
    unsigned long previous;

    memset(response, '\0', 200);// Initialize the string

    delay(100);

    while( mySerial.available() > 0) mySerial.read();// Clean the input buffer

    mySerial.println(ATcommand);// Send the AT command

      x = 0;
  previous = millis();

  // this loop waits for the answer
  do{
      if(mySerial.available() != 0){
          response[x] = mySerial.read();
```

```
        Serial.write(response[x]);
        x++;
        // check if the desired answer is in the response of the module
        if (strstr(response, expected_answer1) != NULL)
        {
            answer = 1;
        }
        if(x>180)
        x=0;
    }
    // Waits for the asnwer with time out
  }
  while((answer == 0) && ((millis() - previous) < timeout));

  return answer;
}
```

Arduino Code for Sending/Receiving SMS

```
int8_t answer;
int onModulePin= 2;
char aux_string[30];
char phone_number[]="*********";   // ********* is the number to call
char sms_text[]="Hello World";

void setup(){

    pinMode(onModulePin, OUTPUT);
    Serial.begin(9600);

    Serial.println("Starting...");
    power_on();

    delay(3000);
```

```
Serial.println("Connecting to the network...");

while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||
        sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );

Serial.print("Setting SMS mode...");
sendATcommand("AT+CMGF=1", "OK", 1000);    // sets the SMS mode to text
Serial.println("Sending SMS");
```

```
sprintf(aux_string,"AT+CMGS=\"%s\"", phone_number); //To send the sms
answer = sendATcommand(aux_string, ">", 2000); // send the SMS number
```

```
  if (answer == 1)
  {
      Serial.println(sms_text);
      Serial.write(0x1A);
      answer = sendATcommand("", "OK", 20000);
      if (answer == 1)
      {
          Serial.print("Sent ");
      }
      else
      {
          Serial.print("error ");
      }
  }
  else
  {
      Serial.print("error ");
      Serial.println(answer, DEC);
  }

}
 void loop(){

 }

 void power_on(){

     uint8_t answer=0;

     // checks if the module is started
     answer = sendATcommand("AT", "OK", 2000);
     if (answer == 0)
     {
         // power on pulse
         digitalWrite(onModulePin,HIGH);
         delay(3000);
         digitalWrite(onModulePin,LOW);

  // waits for an answer from the module
  while(answer == 0){     // Send AT every two seconds and wait for the answer
```

```
        answer = sendATcommand("AT", "OK", 2000);
    }
  }

}

int8_t sendATcommand(char* ATcommand, char* expected_answer, unsigned int timeout){

uint8_t x=0,  answer=0;
char response[100];
unsigned long previous;

memset(response, '\0', 100);    // Initialice the string

delay(100);

while( Serial.available() > 0) Serial.read();    // Clean the input buffer

Serial.println(ATcommand);    // Send the AT command

  x = 0;
  previous = millis();

  // this loop waits for the answer
  do{
      // if there are data in the UART input buffer, reads it and checks for the asnwer
      if(Serial.available() != 0){
          response[x] = Serial.read();
          x++;
          // check if the desired answer is in the response of the module
          if (strstr(response, expected_answer) != NULL)
          {
              answer = 1;
          }
      }
  // Waits for the asnwer with time out
  }while((answer == 0) && ((millis() - previous) < timeout));

  return answer;
  }
```

Arduino Power Saving Code

```
int8_t answer;
int onModulePin= 2;
char aux_string[30];
char phone_number[]="*********";   // ********* is the number to call
char sms_text[]="Hello World";

void setup(){

    pinMode(onModulePin, OUTPUT);
    Serial.begin(9600);

    Serial.println("Starting...");
    power_on();

    delay(3000);

Serial.println("Connecting to the network...");

while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||
        sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );//For Network Registration

Serial.print("Setting SMS mode...");
sendATcommand("AT+CMGF=1", "OK", 1000);// sets the SMS mode to text
Serial.println("Sending SMS");

sprintf(aux_string,"AT+CMGS=\"%s\"", phone_number); //To send the sms
answer = sendATcommand(aux_string, ">", 2000); // send the SMS number

 if (answer == 1)
 {
     Serial.println(sms_text);
     Serial.write(0x1A);
     answer = sendATcommand("", "OK", 20000);
     if (answer == 1)
     {
         Serial.print("Sent ");
     }
     else
```

```
        {
            Serial.print("error ");
        }
    }


    power_off();


  }


void loop(){

}

void power_on(){

    uint8_t answer=0;

    // checks if the module is started
    answer = sendATcommand("AT", "OK", 2000);
    if (answer == 0)
    {


      // power on pulse
      digitalWrite(onModulePin,HIGH);
      delay(3000);
      digitalWrite(onModulePin,LOW);

      // waits for an answer from the module
      while(answer == 0){     // Send AT every two seconds and wait for the answer
          answer = sendATcommand("AT", "OK", 2000);
      }
    }

  }
```

```
void power_off(){


    digitalWrite(onModulePin,HIGH);
    delay(3000);
    digitalWrite(onModulePin,LOW);



 }

 int8_t sendATcommand(char* ATcommand, char* expected_answer, unsigned int timeout){

 uint8_t x=0,  answer=0;
 char response[100];
 unsigned long previous;

 memset(response, '\0', 100);    // Initialice the string

 delay(100);


while( Serial.available() > 0) Serial.read();    // Clean the input buffer

Serial.println(ATcommand);    // Send the AT command

x = 0;
previous = millis();

// this loop waits for the answer
do{
    // if there are data in the UART input buffer, reads it and checks for the asnwer
    if(Serial.available() != 0){
        response[x] = Serial.read();
        x++;
        // check if the desired answer is in the response of the module
```

```
        if (strstr(response, expected_answer) != NULL)
        {
            answer = 1;
        }
    }
// Waits for the asnwer with time out
}while((answer == 0) && ((millis() - previous) < timeout));

return answer;
}
```

12. Documents