SIM5360E 3G Shield

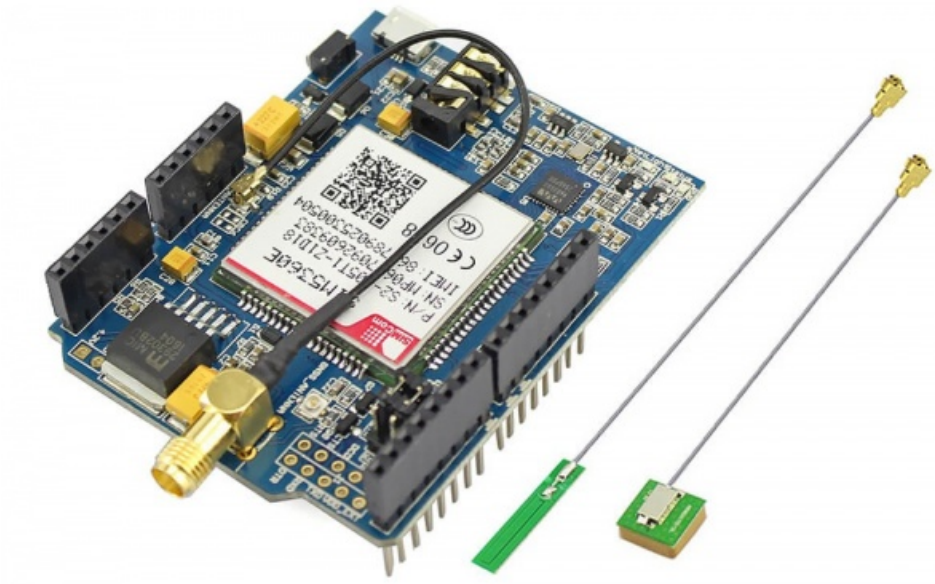
From Elecrow

Contents

- 1 Introduction
- 2 Features
- 3 Application Ideas
- 4 Cautions
- 5 Specifications
- 6 Interface Function
  - 6.1 Pins usage on Arduino
- 7 Usage
  - 7.1 Hardware installation
    - 7.1.1 1.Insert an Micro SIM card to SIM Card Holder
    - 7.1.2 2.Connect the Antenna
    - 7.1.3 3.Plug to Arduino/Crowduino
    - 7.1.4 Turn on the SIM5360E 3G shield
    - 7.1.5 Serial Port(UART) Communication
  - 7.2 Power Down the GPRS Shield
  - 7.3 Upload Sketch to Arduino
    - 7.3.1 Set Baud and Enable Charging Function
  - 7.4 Examples
    - 7.4.1 Sending SMS: using Software UART
    - 7.4.2 Making a call: using Software UART
  - 7.5 A Simple Source Code Example
  - 7.6 Using Sms to Control an LED Status
- 8 The usage of GPS Function
  - 8.1 AT Commands Examples
  - 8.2 Demo code of get the GPS information:
- 9 The usage of ECALL Function
  - 9.1 AT Comands Examples
  - 9.2 Demo code：using Software UART
- 10 FAQ
- 11 How to buy
- 12 Resources

Introduction

The SIM5360E series is Dual-Band HSPA+/WCDMA and Quad-Band GSM/GPRS/EDGE module solution in a SMT type which supports HSPA+ up to 14.4Mbps for downlink data transfer.supports 3G network and combines GPS technology for satellite navigation. Besides, it also supports A-GPS that available for indoor localization.and it also supports for eCALL.
The module is controlled by AT command via UART and supports 3.3V and 5V logical level.



Features

- Quad-band GSM/GPRS/EDGE 850/900/1800/1900MHz.
- HSPA+: Max. 14.4Mbps(DL), Max. 5.76Mbps(UL).
- WCDMA: Max. 384Kbps(DL), Max. 384Kbps(UL).
- GPRS multi-slot class 12, Max. 85.6Kbps(DL), Max. 85.6Kbps(UL).
- EDGE multi-slot class 12, Max. 236.8Kbps(DL), Max. 236.8Kbps(UL).
- GPRS mobile station class B.
- Controlled by AT Command (3GPP TS 27.007, 27.005 and SIMCOM enhanced AT Commands).
- Supports Real Time Clock.
- Supply voltage range 5V ~ 12V.
- Supports for eCALL.
- Integrated GPS/CNSS and supports A-GPS.
- Supports 3.0V to 5.0V logic level.
- Low power consumption, 1mA in sleep mode.
- Supports GPS NMEA protocol.
- Standard Micro SIM Card.

Application Ideas

- M2M (Machine 2 Machine) Applicatoions - To transfer control data using SMS or GPRS between two machines located at two different factories.
- Remote control of appliances - Send SMS while you are at your office to turn on or off your washing machine at home.
- Remote Weather station or a Wireless Sensor Network - Make it with [Crowduino v1.0|Crowduino v1.0] and create a sensor node capable of transferring sensor data (like from a weather station - temperature, humidity etc.) to a web server (like pachube.com (http://www.pachube.com)).
- Vehicle Tracking System - Install GPRS+GSM+GPS Shield in your car and publish your location live on the internet. Can be used as a automotive burglar alarm.

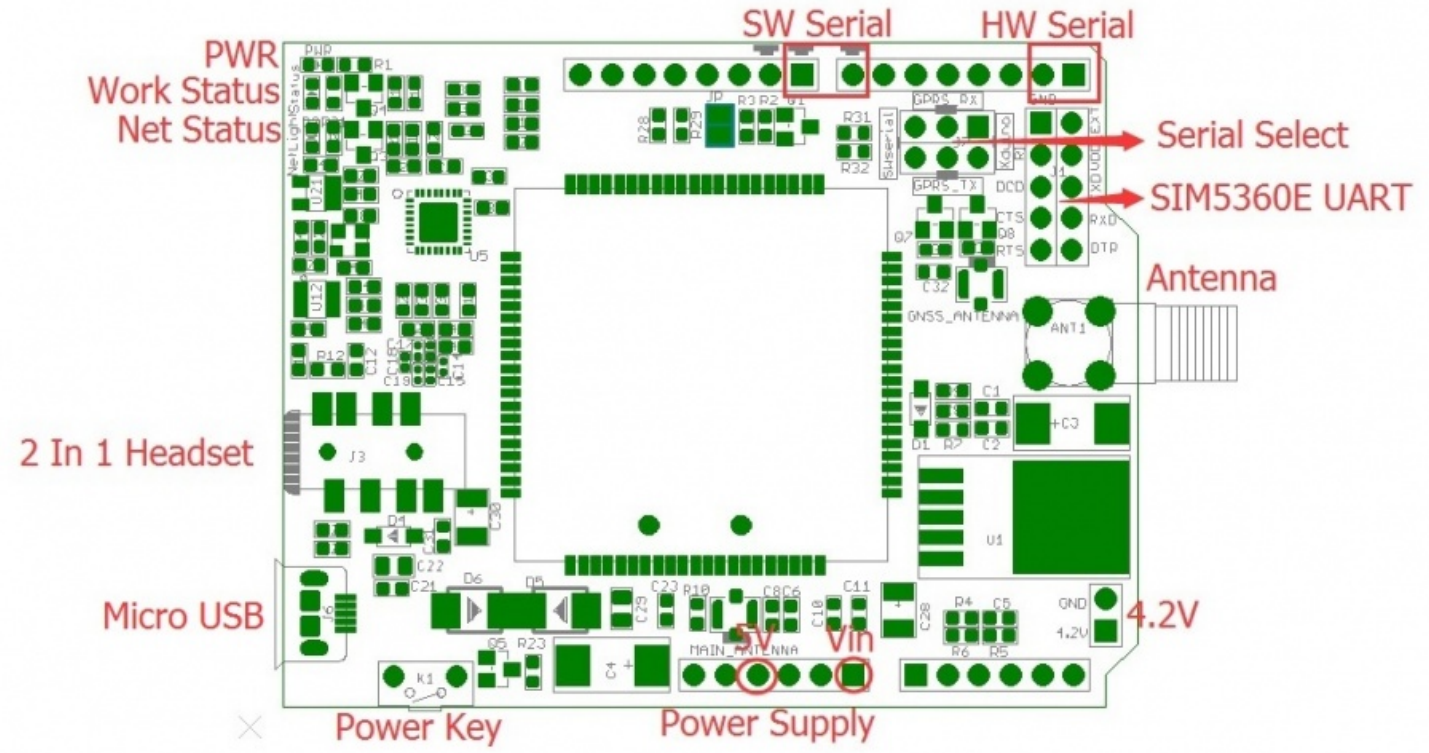
Cautions

- Make sure your SIM card is unlocked.
- The product is provided as is without an insulating enclosure. Please observe ESD precautions specially in dry (low humidity) weather.
- The factory default setting for the GPRS Shield UART is autobaoding. It supports baud rate from 1200 bps to 115200bps. (Can be changed using AT commands).

Specifications

Item	Min	Typical	Max	Unit
Voltage	4.8	5.0	12	VDC
Current	2	-	500	mA
Dimension(with antenna)	Arduino shield			mm
Net Weight	47±2			g

Interface Function



Power supply - Vin connected to external 5~9VDC power supply
Antenna interface - connected to external antenna
Serial port select - select either software serial port or hareware serial port to be connected to GPRS+GSM+GPS Shield Shield
Hardware Serial - D0/D1 of Arduino/Crowduino
Software serial - D7/D8 of Arduino/Crowduino
Status LED - tell whether the power of SIM5360E is on
Net light - tell the status about SIM5360E linking to the net
UART of SIM808 - UART pins breakout of SIM5360E
Microphone - to answer the phone call
Speaker - to answer the phone call
Micro USB - Upgrade the firmware
Power key - power up and down for SIM5360E

Pins usage on Arduino

D0 - Used if you select hardware serial port to communicate with GPRS+GSM+GPS Shield
D1 - Used if you select hardware serial port to communicate with GPRS+GSM+GPS Shield
D2 - Unused
D3 - Unused
D4 - Unused
D5 - Unused
D6 - Unused
D7 - Used if you select software serial port to communicate with GPRS+GSM+GPS Shield
D8 - Used if you select software serial port to communicate with GPRS+GSM+GPS Shield
D9 - Used for software control the power up or down of the SIM5360E
D10 - Unused
D11 - Unused
D12 - Unused
D13 - Unused
D14(A0) - Unused
D15(A1) - Unused
D16(A2) - Unused
D17(A3) - Unused
D18(A4) - Unused
D19(A5) - Unused

Usage

Hardware installation

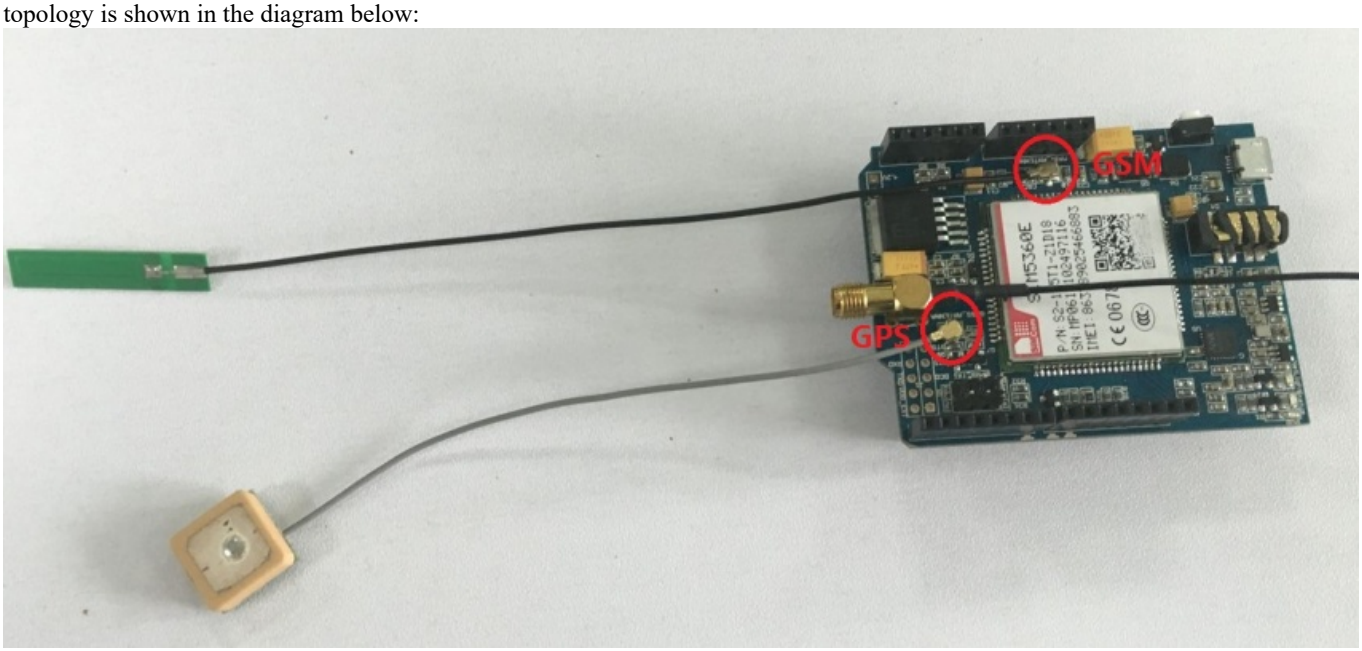
1.Insert an Micro SIM card to SIM Card Holder

6 Pin Holder for SIM Cards. Both 1.8 volts and 3.0 volts SIM Cards are supported by SIM5360E - the SIM card voltage type is automatically detected.

2.Connect the Antenna

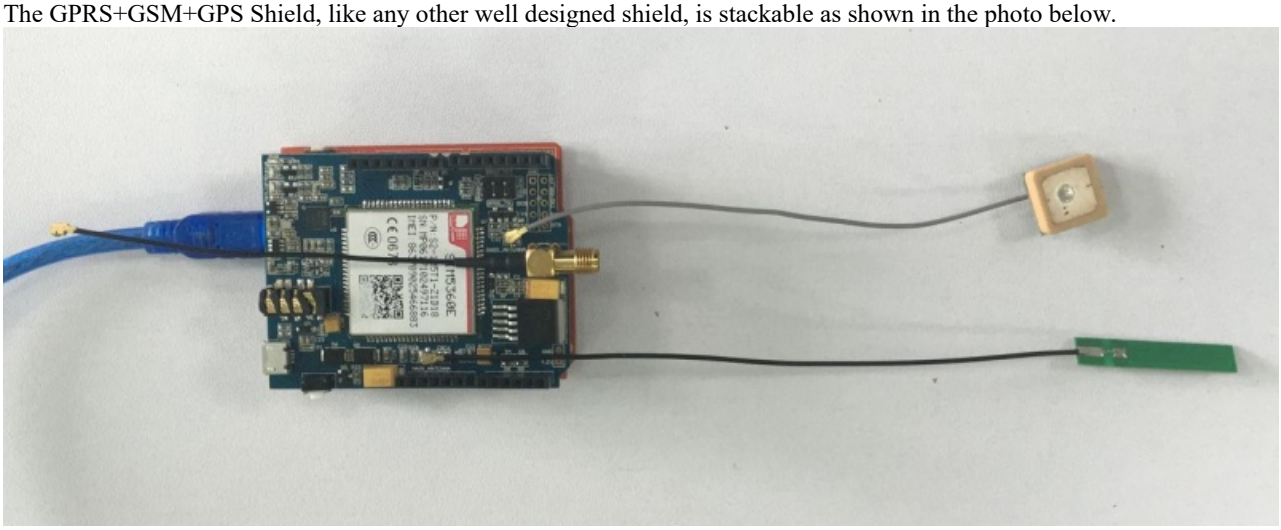
A miniature coaxial RF connector is present on the SIM5360E 3G Shield board to connect with a MAIN_Antenna. The connector present on the SIM5360E 3G Shield is called a U.FL connecto (http://en.wikipedia.org/wiki/Hirose_U.FL) .The GNSS_Antenna supplied with the GPRS Shield has an SMA connector (http://en.wikipedia.org/wiki/SMA_connector) (and not an RP-SMA connector) on it.The connection

topology is shown in the diagram below:



3.Plug to Arduino/Crowduino

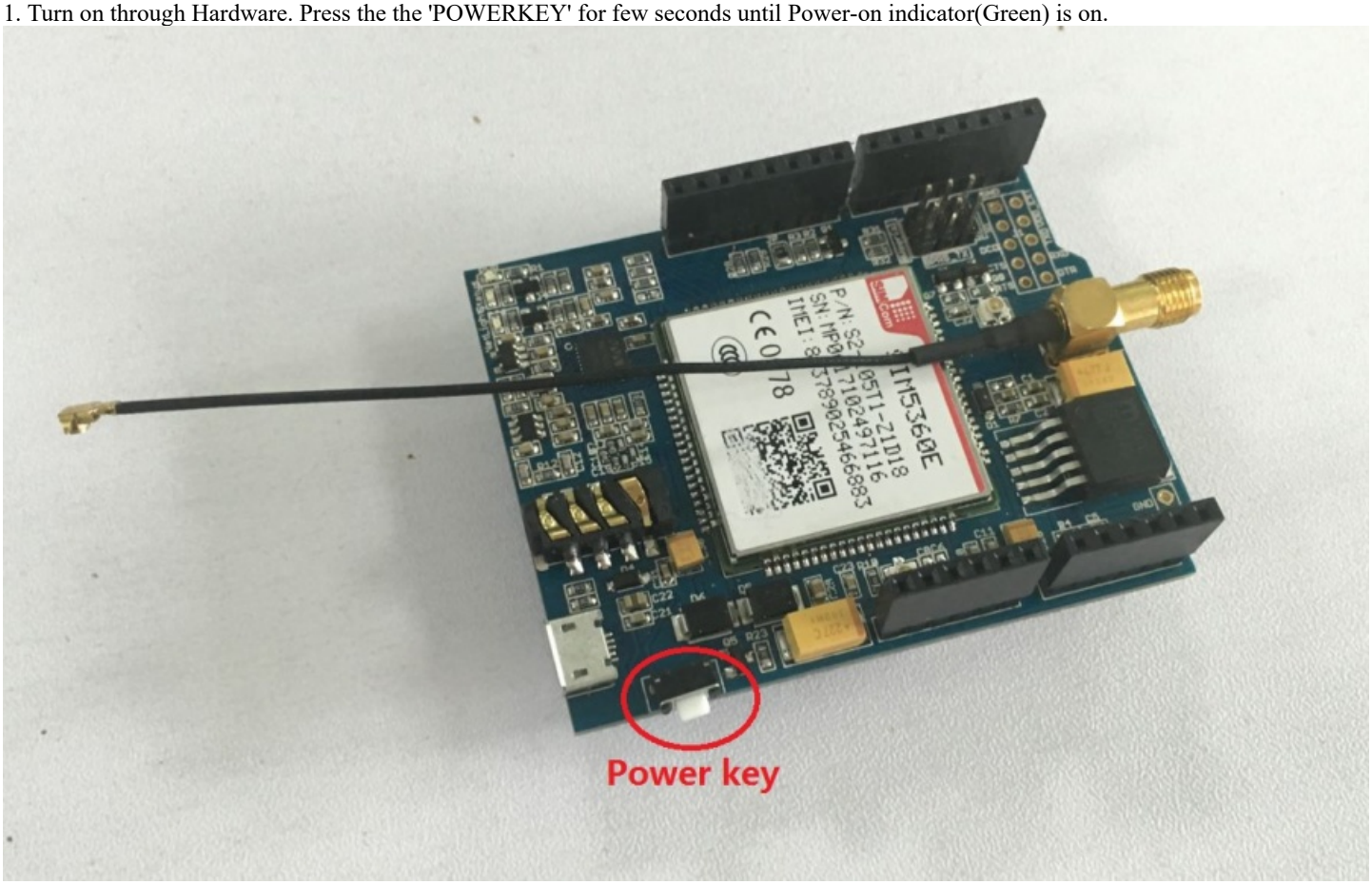
The GPRS+GSM+GPS Shield, like any other well designed shield, is stackable as shown in the photo below.



Turn on the SIM5360E 3G shield

There is two ways to turn on the SIM5360 3G Shield.

1. Turn on through Hardware. Press the the 'POWERKEY' for few seconds until Power-on indicator(Green) is on.

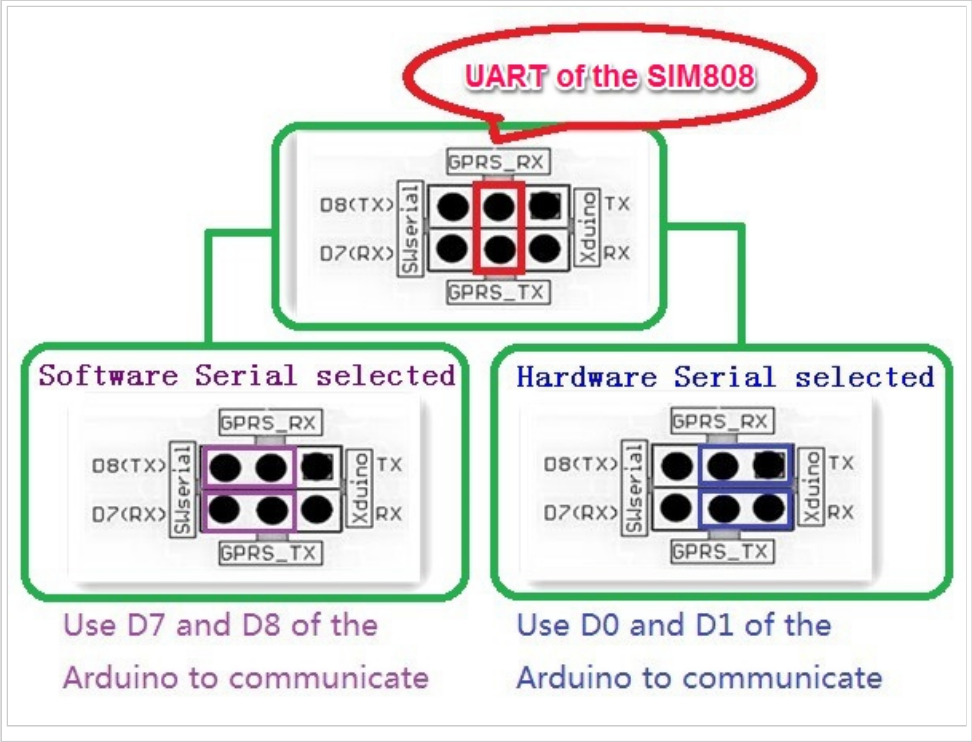


2. Turn on through Software. If the JP is soldered,run the following code, the SIM5360E will POWER on or POWER off.

```
int Powerkey = 9;
void setup() {
  pinMode(Powerkey, OUTPUT);   // initialize the digital pin as an output.
  power();                     //power on the sim5360 or power down the sim5360
}
void loop()
{

}

void power(void)
{
  digitalWrite(Powerkey, LOW);
  delay(1000);               // wait for 1 second
  digitalWrite(Powerkey, HIGH);
}
```

Serial Port(UART) Communication

The SIM5360 3G Shield is used UART protocol to communicate with an Arduino/Arduino clone; Users can use jumpers to connect (RX,TX) of the shield to either Software Serial(D8,D7) or Hardware Serial(D1,D0) of the Arduino.Detailed information is showed as the following picture:



Selectalbe 3G Shield Communication Port

Power Down the GPRS Shield

The GPRS Shield can be turned off by following ways:

- **1, Normal power down procedure**: Turn off the GPRS shield by using **Hardware Triger**; Press the **ON/OFF Button** about one seconds.

magic mesh (http://www.magicmeshoffer.com) The power down scenarios illustrates as following figure:

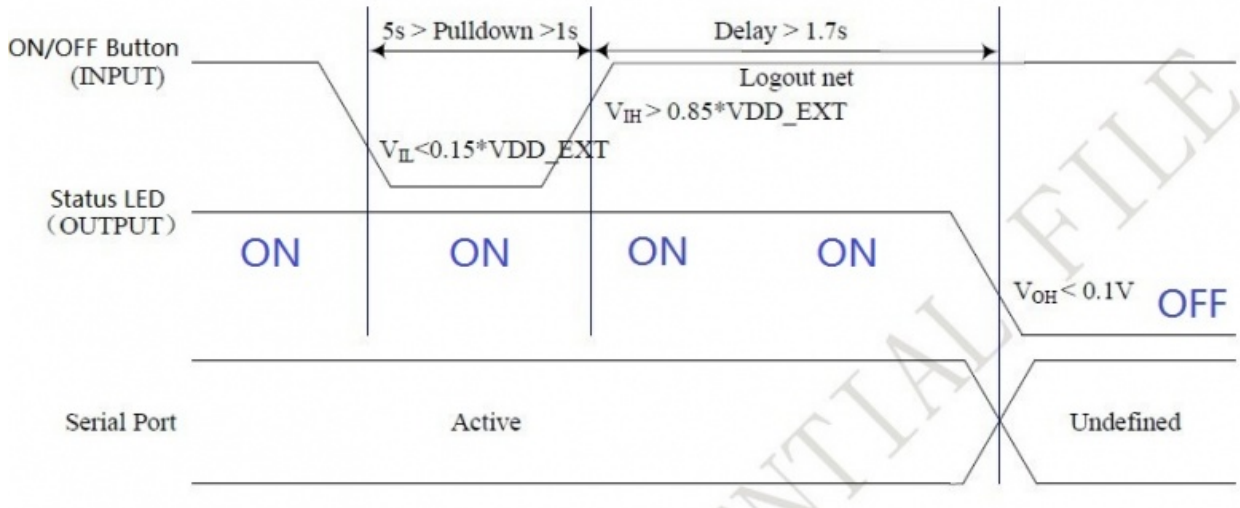


Figure of Timing of turning off GPRS Shield using Hardware Triger

The following code is power down subroutine for Arduino if using software triger:

```
int Powerkey = 9;
void setup() {
  pinMode(Powerkey, OUTPUT);   // initialize the digital pin as an output.
  power();                     //power on the sim5360 or power down the sim5360
}
void loop()
{

}

void power(void)
{
  digitalWrite(Powerkey, LOW);
  delay(1000);               // wait for 1 second
  digitalWrite(Powerkey, HIGH);
}
```

- **2, Normal power down procedure**: Turn off the GPRS shield by sending AT command **"AT+CPOF"** to SIM5360 module.

When GPRS Shield power dowm in **Normal power down procedure**, the procedure lets the SIM5360 log off from the network and allows the software to enter into a secure state and save data before completely disconnecting the power supply. Before the completion of the power down procedure the SIM5360 will send out result code:

NORMAL POWER DOWN

- **3, Over-voltage or Under-voltage Power Down**: The module software monitors VBAT voltage constantly.

①If the voltage ≤ 3.5V, the following URC will be reported:

UNDER-VOLTAGE WARNNING

②If the voltage ≥ 4.3V, the following URC will be reported:

OVER-VOLTAGE WARNNING

③If the voltage < 3.4V, the following URC will be reported,and the module will be automativally powered down.

UNDER-VOLTAGE POWER DOWN

④If the voltage > 4.4V, the following URC will be reported,and the module will be automativally powered down.

OVER-VOLTAGE POWER DOWN

- **4, Over-temperature or Under-temperature Power Down**: SIM5360 will constantly monitor the temperature of the module.

①If the temperature > +80°C, the following URC will be reported:

+CMTE:1

②If the temperature < -30°C, the following URC will be reported:

+CMTE:-1

③If the temperature > +85°C, the following URC will be reported,and the module will be automativally powered down.

+CMTE:2

④If the temperature < -40°C, the following URC will be reported,and the module will be automativally powered down.

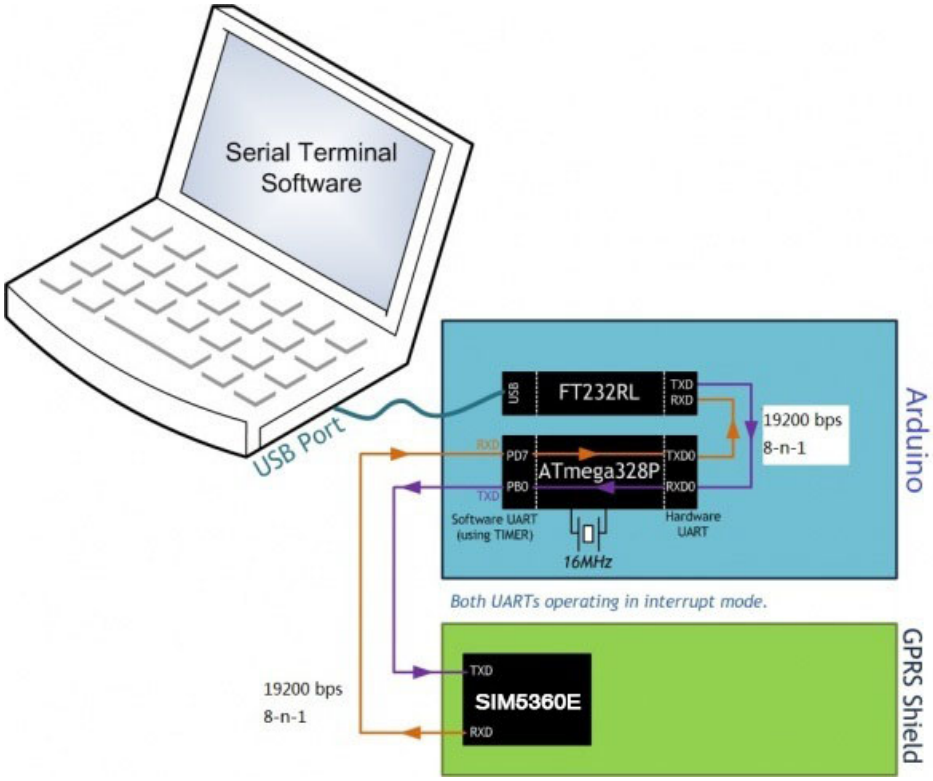
+CMTE:-2

When the GPRS Shield encounters POWER DOWN scenario, the AT commands can not be executed. The SIM5360 logs off from network and enters the POWER DOWN mode, only the RTC is still active. POWER DOWN can also be indicated by STATUS LED(Blue), which is off in this mode.

Note:

- To monitor the temperature, users can use the "**AT+CMTE**" command to read the temperature when GPRS Shield is powered on.

Upload Sketch to Arduino

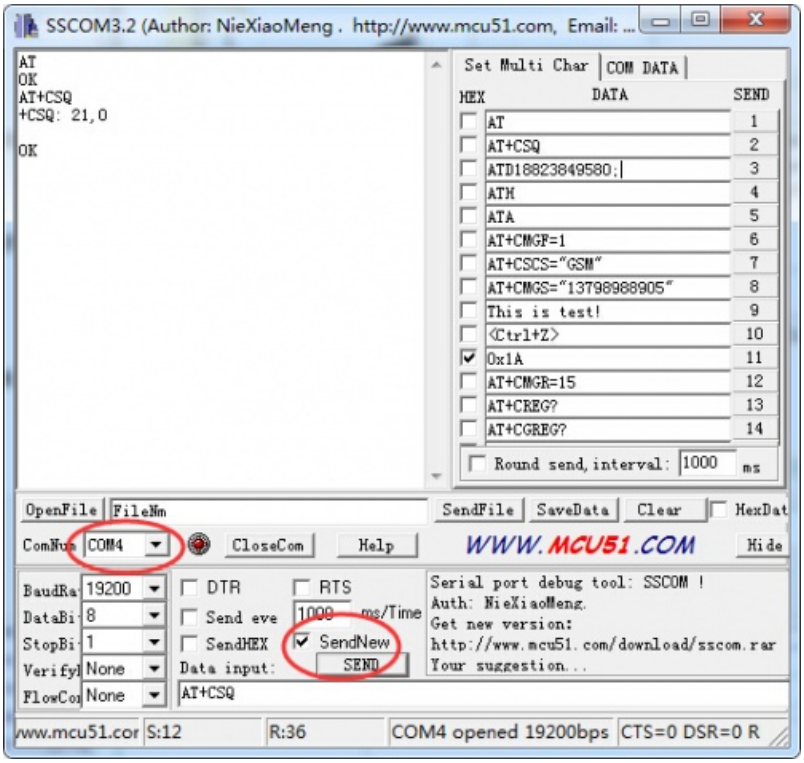


Data Stream among Computer, Arduino and GPRS Shield

The following sketch configures Arduino/Arduino clone as serial link between PC and the GPRS Shield(Jumpers on SWserial side). PC would need a serial terminal software to communicate with it - Window's built-in HyperTerminal, Arduino IDE's Serial Monitor, Serial Terminals(sscom32) (http://musicshield.googlecode.com/files/sscom32E.exe) or Bray++ Terminal (http://sites.google.com/site/terminalbpp/) .

After uploading the sketch to the Arduino board, press the ON/OFF button on the GPRS Shield to turn it on; Now you can see what you get on the serial terminal and the status of the three indicator LEDs, then communicate with your Shield.

```
//Serial Relay - Arduino will patch a
//serial link between the computer and the GPRS Shield
//at 19200 bps 8-N-1
//Computer is connected to Hardware UART
//GPRS Shield is connected to the Software UART

#include <SoftwareSerial.h>

SoftwareSerial GSMSerial(7, 8);

void setup()
{
GSMSerial.begin(19200);               // the GPRS/GSM baud rate
Serial.begin(19200);                 // the GPRS/GSM baud rate
}

void loop()
{
if(Serial.available())

GSMSerial.print((char)Serial.read());

else  if(GSMSerial.available())

Serial.print((char)GSMSerial.read());
}
```



Note:

- The "AT" or "at" prefix must be set at the beginning of each Command line. To terminate a Command line enter <CR>.

Set Baud and Enable Charging Function

It is recommended to execute this process when first time to use the module. In the Serial Monitor columns of following tables, input of AT commands are in back, module returns values are in orange.

Serial Monitor	Description
AT OK	Send command “AT” to synchronize baud rate. Serial port of module is by default set at auto-baud mode, and in this mode, it will not output any indications when the module is on.
AT+IPREX=19200 OK	Set baud rate at 19200bps, supports baud rate from 0bps to 4000000bps.
AT&W OK	Save the user setting to ME.
AT+CPOF OK +SIMCARD: NOT AVAILABLE	Power down the module.
START +CPIN: READY OPL UPDATING PNN UPDATING SMS DONE CALL READY PB DONE	Turn on the module again by the power button, it will response status about SMS and CALL.
AT+CBC +CBC: 0,75,3.810V OK	Inquire charging status and remaining battery capacity.
AT+CSQ +CSQ: 14,0 OK	Inquire GSM signal quality.

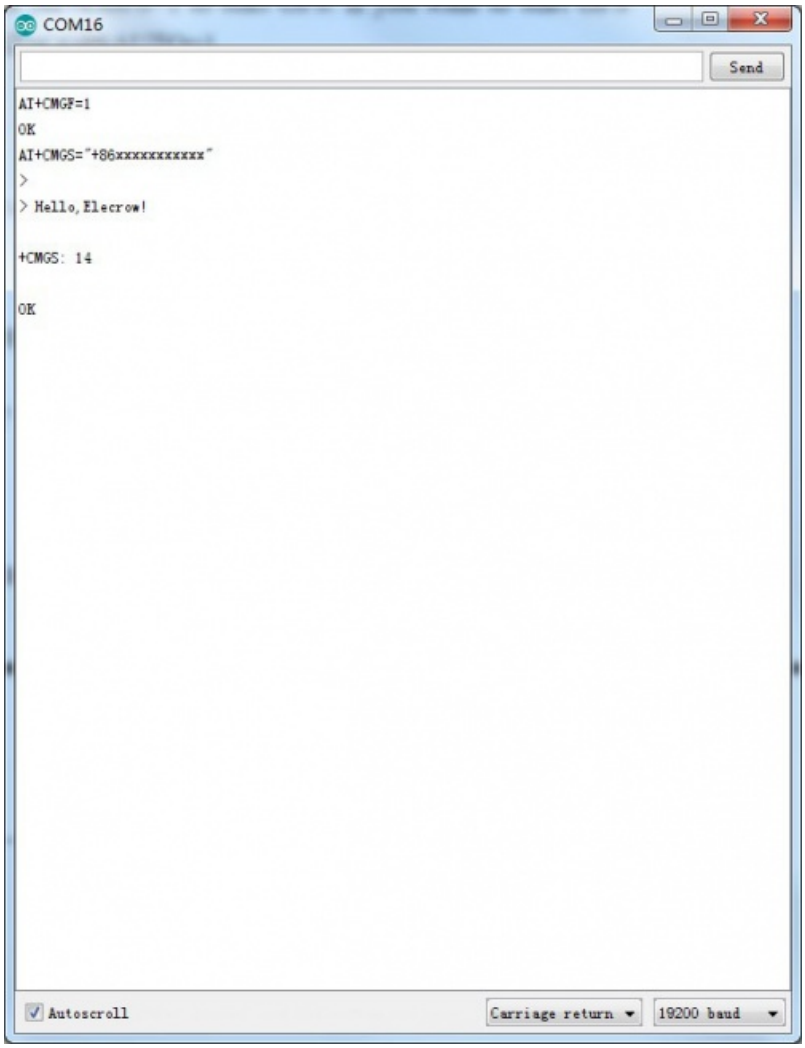
Examples

Sending SMS: using Software UART

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7,8); //
#define DEBUG true

void setup()
```

```
{
 mySerial.begin(19200);    // the GPRS baud rate
 Serial.begin(19200);    // the GPRS baud rate
 sendData("AT+CMGF=1",2000,DEBUG);//Because we want to send the SMS in text mode
 delay(100);
 sendData("AT+CMGS=\"+8613xxxxxxxx\"",2000,DEBUG);//send sms message, be careful need to add a country code before the cellphone number
 delay(100);
 sendData("Hello,Elecrow!",2000,DEBUG);//the content of the message
 delay(100);
 mySerial.println((char)26);//the ASCII code of the ctrl+z is 26
 delay(100);
}

void sendData(String command, const int timeout, boolean debug)
{
    String response = "";
    mySerial.println(command);
    long int time = millis();
    while( (time+timeout) > millis())
    {
      while(mySerial.available())
      {
        char c = mySerial.read();
        response+=c;
      }
    }
    if(debug)
    {
      Serial.print(response);
    }
    return response;
}

 void loop()
{
 if(mySerial.available())
      {
          char SerialInByte;
          SerialInByte = (unsigned char)mySerial.read();
          Serial.print( SerialInByte );
      }
}
```

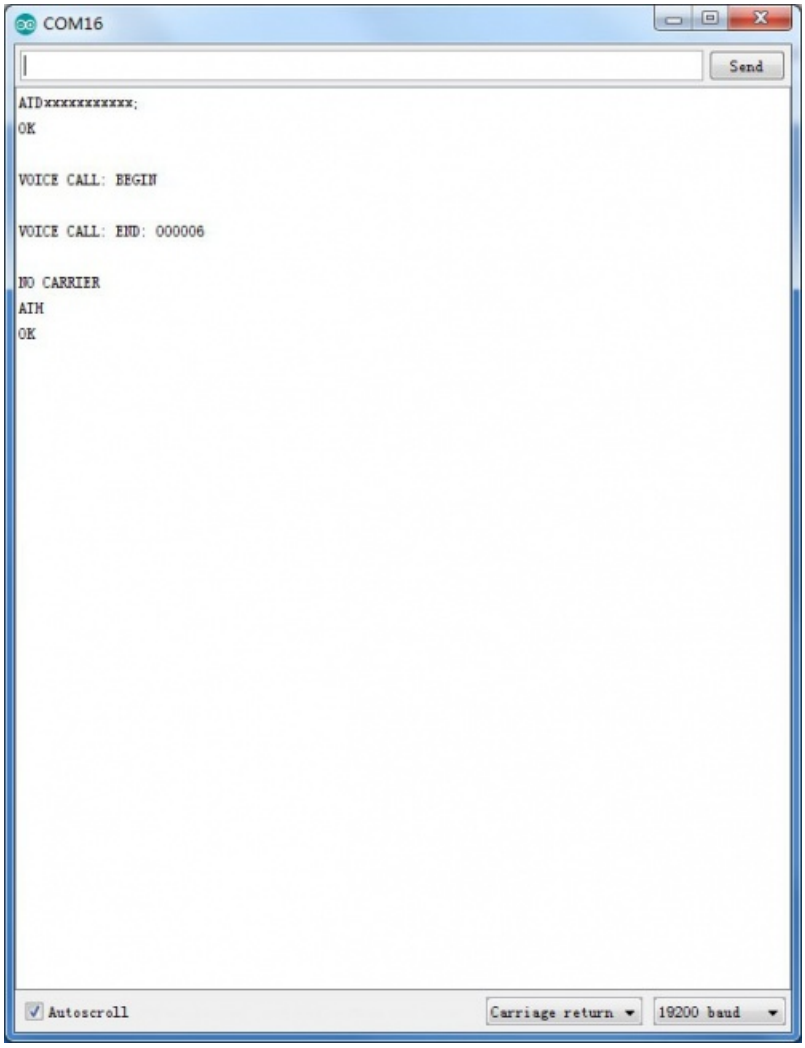
The result of sending SMS.



Making a call: using Software UART

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);
#define DEBUG true
void setup()
{
mySerial.begin(19200);               // the GPRS baud rate
Serial.begin(19200);               // the GPRS baud rate
sendData("ATDxxxxxxxxx;",2000,DEBUG); // xxxxxxxxx is the number you want to dial.


delay(10000);
delay(10000);

sendData("ATH",2000,DEBUG); //End the call.

}

void sendData(String command, const int timeout, boolean debug)
{
    String response = "";
    mySerial.println(command);
    long int time = millis();
    while( (time+timeout) > millis())
    {
      while(mySerial.available())
      {
        char c = mySerial.read();
        response+=c;
      }
    }
    if(debug)
    {
      Serial.print(response);
    }
    return response;
}

void loop()
{
if(mySerial.available())
      {
          char SerialInByte;
          SerialInByte = (unsigned char)mySerial.read();
          Serial.print( SerialInByte );
      }
}
```

The result of making a call.



A Simple Source Code Example

The demo code below is for the Xduino to send SMS message/show message/delete message/dial a voice call . It has been tested on Arduino Duemilanove but will work on any compatible variant, plesse note that this sketch uses the sorfware UART of ATmega328P. please follow the following steps for running this sketch.

1. With the 3G Shield removed, download this sketch into your Arduino.
2. Disconnect the Xduino from USB port to remove power source.
3. Set the Serial Port jumpers on the 3G Shield in SWserial position, to use the Soft Serial port of Arduino.
4. Connect the antenna to the 3G Shield and insert the SIM Card.
5. Mount the 3G Shield on Arduino.
6. Connect the Arduino to the computer by USB, and fire up your favorite serial terminal software on computer, choose the COM port for Arduino, set it to operate at 19200 8-N-1.
7. Type command in the terminal to execute different function, threr are 5 functions in the demo:
    1. If you input 'Q' in the terminal, the program will execute GetSignalQuality(),it will show the signal quality,
    2. If you input 'T', the demo will send a SMS message to another cellphone which you set(you need set the number in the code);
    3. If you input 'S', the demo will listing all SMS message ;
    4. If you input 'E', the demo will delete a SMS message ;
    5. If you input 'D', the program will dial a call to the other cellphone that you set(it is also need you set in the code);
8. If the program returns error in the terminal after you typed the command, don't worry, just try input the command again.

```
/*Note: this code is a demo for how to using 3G shield to send sms message, show message,delete message, dial a voice call.

The microcontrollers Digital Pin 7 and hence allow unhindered
communication with 3G Shield using SoftSerial Library.
IDE: Arduino 1.0 or later
Replace the following items in the code:
1.Phone number, don't forget add the country code
2.Replace the Access Point Name
3. Replace the Pachube API Key with your personal ones assigned
to your account at cosm.com
*/


#include <SoftwareSerial.h>

SoftwareSerial mySerial(7,8); //
#define DEBUG true
```

```
#define DEBUG true


//String SigQ[20];
char SigQ[20];
int value = 576;

void setup()
{

  mySerial.begin(19200);    // the GPRS baud rate
  Serial.begin(19200);    // the GPRS baud rate

}


void loop()
{
  if (Serial.available())     // This statement is never TRUE when receiving SMS
  switch(Serial.read())
  {
    case 'Q':
    GetSignalQuality();
    break;
    case 'T':
    SendTextMessage();
    break;
    case 'S':
    ShowSMS();
    break;
    case 'E':
    DeleteSMS();
    break;
    case 'D':
    DialVoiceCall();
    break;

  }
 if(mySerial.available())
     {
         char SerialInByte;
         SerialInByte = (unsigned char)mySerial.read();
         Serial.print( SerialInByte );
     }

}
///GetSignalQuality()
///get the signal quality of GSM model.

void GetSignalQuality(){
  String response = "";
  long int time = millis();
    Serial.println("Getting the sinal quality...");
    mySerial.println("AT+CSQ");
    delay(5);
    while( (time+1000) > millis()){
      while(mySerial.available()){
        response += char(mySerial.read());
      }
    }
     Serial.print(response);
     Serial.println("Tips:+CSQ: XX,QQ : It means the Signal Quality poorly when the XX is less than 16!");
}


///SendTextMessage()
///this function is to send a sms message
void SendTextMessage()
{
  //message 1
  sendData("AT+CMGF=1",2000,DEBUG);//Because we want to send the SMS in text mode
  delay(100);
  sendData("AT+CMGS=\"+8613117913785\"",2000,DEBUG);//send sms message, be careful need to add a country code before the cellphone number
  delay(100);
  sendData("HELLO!",2000,DEBUG);//the content of the message
  delay(100);
  mySerial.println((char)26);//the ASCII code of the ctrl+z is 26
  delay(5000);
}


///ShowSMS()
///this function is to show sms message
void ShowSMS(){
  sendData("AT+CMGF=1",2000,DEBUG);
  sendData("AT+CPMS=\"SM\",\"SM\",\"SM\"",1000,DEBUG);
  sendData("AT+CNMI=2,1",1000,DEBUG);
  Serial.println("*** Show all SMS message ***");
  sendData("AT+CMGL=\"ALL\"",5000,DEBUG);
  Serial.println("***         END          ***");
}

///DeleteSMS()
///this function is to delete sms message
void DeleteSMS(){
  sendData("AT+CMGF=1",2000,DEBUG);
  Serial.println("*** Delete SMS message ***");
  sendData("AT+CMGD=0",5000,DEBUG);
  Serial.println("***         END            ***");
}
```

```
///DialVoiceCall
///this function is to dial a voice call
void DialVoiceCall()
{
sendData("ATD13117913785;",2000,DEBUG); // xxxxxxxxx is the number you want to dial.
delay(10000);
delay(10000);
}

void sendData(String command, const int timeout, boolean debug)
{
    String response = "";
    mySerial.println(command);
    long int time = millis();
    while( (time+timeout) > millis())
    {
      while(mySerial.available())
      {
        char c = mySerial.read();
        response+=c;
      }
    }
    if(debug)
    {
      Serial.print(response);
    }
    return response;
}
```

Using Sms to Control an LED Status

This example is controbuted by MChobby, for more information please visit: http://mchobby.be/wiki/index.php?title=SmsCommand

Send a SMS message "on" or "off" from your cellphone to the 3G Shield to control the Digital Pin 13(LED) Status.

- The default Buffer of Rx in SoftwareSerial.h is 32/64, you may experience some data lose while the returns of SIM5360E are many(Receiving SMS/TCPIP), you can try to change the Buffer of Rx in SoftwareSerial.h into

#define _SS_MAX_RX_BUFF *128* // RX buffer size

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);

// EN: String buffer for the GPRS shield message

String msg = String("");
// EN: Set to 1 when the next GPRS shield message will contains the SMS message

int SmsContentFlag = 0;

// EN: Pin of the LED to turn ON and OFF depending on the received message

int ledPin = 13;

// EN: Code PIN of the SIM card (if applied)

//String SIM_PIN_CODE = String( "XXXX" );

void setup()
{
  mySerial.begin(19200);               // the GPRS baud rate
  Serial.begin(19200);                 // the GPRS baud rate

  // Initialize la PIN
  pinMode( ledPin, OUTPUT );
  digitalWrite( ledPin, LOW );
}

void loop()
{
    char SerialInByte;

    if(Serial.available())
    {
       mySerial.print((unsigned char)Serial.read());
     }
    else  if(mySerial.available())
    {
        char SerialInByte;
        SerialInByte = (unsigned char)mySerial.read();

        // EN: Relay to Arduino IDE Monitor

        Serial.print( SerialInByte );

        // ------------------------------------------------------------------
```

```
       // EN: Program also listen to the GPRS shield message.

       // --------------------------------------------------------------------

       // EN: If the message ends with <CR> then process the message

       if( SerialInByte == 13 ){
         // EN: Store the char into the message buffer

        ProcessGprsMsg();
        }
        if( SerialInByte == 10 ){
           // EN: Skip Line feed

        }
        else {
          // EN: store the current character in the message string buffer

          msg += String(SerialInByte);
        }
    }
}

// EN: Make action based on the content of the SMS.
//     Notice than SMS content is the result of the processing of several GPRS shield messages.

void ProcessSms( String sms ){
  Serial.print( "ProcessSms for [" );
  Serial.print( sms );
  Serial.println( "]" );


  if( sms.indexOf("on") >= 0 ){
    digitalWrite( ledPin, HIGH );
    Serial.println( "LED IS ON" );
    return;
  }
  if( sms.indexOf("off") >= 0 ){
    digitalWrite( ledPin, LOW );
    Serial.println( "LED IS OFF" );
    return;
  }
}

// EN: Send the SIM PIN Code to the GPRS shield

//void GprsSendPinCode(){
//  if( SIM_PIN_CODE.indexOf("XXXX")>=0 ){
//    Serial.println( "*** OUPS! you did not have provided a PIN CODE for your SIM CARD. ***" );
//    Serial.println( "*** Please, define the SIM_PIN_CODE variable . ***" );
//    return;
// }
//  mySerial.print("AT+CPIN=");
// mySerial.println( SIM_PIN_CODE );


// EN: Request Text Mode for SMS messaging

void GprsTextModeSMS(){
  mySerial.println( "AT+CMGF=1" );
  mySerial.println( "AT+CPMS=SM,SM,SM" );
  mySerial.println( "AT+CNMI=2,1" );
}

void GprsReadSmsStore( String SmsStorePos ){
  // Serial.print( "GprsReadSmsStore for storePos " );
  // Serial.println( SmsStorePos );
  mySerial.print( "AT+CMGR=" );
  mySerial.println( SmsStorePos );
}

// EN: Clear the GPRS shield message buffer

void ClearGprsMsg(){
  msg = "";
}

// EN: interpret the GPRS shield message and act appropiately

void ProcessGprsMsg() {
  Serial.println("");
  Serial.print( "GPRS Message: [" );
  Serial.print( msg );
  Serial.println( "]" );

//  if( msg.indexOf( "+CPIN: SIM PIN" ) >= 0 ){
//     Serial.println( "*** NEED FOR SIM PIN CODE ***" );
 //    Serial.println( "PIN CODE *** WILL BE SEND NOW" );
 //    GprsSendPinCode();
//  }

  if( msg.indexOf( "Call Ready" ) >= 0 ){
     Serial.println( "*** GPRS Shield registered on Mobile Network ***" );
     GprsTextModeSMS();
  }

  // EN: unsolicited message received when getting a SMS message
  // FR: Message non sollicité quand un SMS arrive
  if( msg.indexOf( "+CMTI" ) >= 0 ){
     Serial.println( "*** SMS Received ***" );
     // EN: Look for the coma in the full message (+CMTI: "SM",6)
```

```
    //     In the sample, the SMS is stored at position 6
    int iPos = msg.indexOf( "," );
    String SmsStorePos = msg.substring( iPos+1 );
    Serial.print( "SMS stored at " );
    Serial.println( SmsStorePos );

    // EN: Ask to read the SMS store
    GprsReadSmsStore( SmsStorePos );
  }

  // EN: SMS store readed through UART (result of GprsReadSmsStore request)
  if( msg.indexOf( "+CMGR:" ) >= 0 ){
    // EN: Next message will contains the BODY of SMS
    SmsContentFlag = 1;
    // EN: Following lines are essentiel to not clear the flag!
    ClearGprsMsg();
    return;
  }

  // EN: +CMGR message just before indicate that the following GRPS Shield message
  //     (this message) will contains the SMS body

  if( SmsContentFlag == 1 ){
    Serial.println( "*** SMS MESSAGE CONTENT ***" );
    Serial.println( msg );
    Serial.println( "*** END OF SMS MESSAGE ***" );
    ProcessSms( msg );
  }

  ClearGprsMsg();
  // EN: Always clear the flag

  SmsContentFlag = 0;
}
```

The usage of GPS Function

AT Commands Examples

Demonstration	Syntax	Expect Result
Start GPS session	AT+CGPS=1,1 (or AT+CGPS=1)	OK
Stop GPS session	AT+CGPS=0,1 (or AT+CGPS=0)	OK
Get GPS fixed position information	AT+CGPSINFO	+CGPSINFO:3113.343286,N,12121.234064,E,250311,072809.0,44.1,0.0,0 OK
Set URC reporting every 2(0-255)GPS fix	AT+CGPSINFO=2	OK
Set AGPS transport security	AT+CGPSSSL=0	OK
Configure NMEA sentence type	AT+CGPSNMEA＝511	OK
Delete the GPS information	AT+CGPSDEL	OK

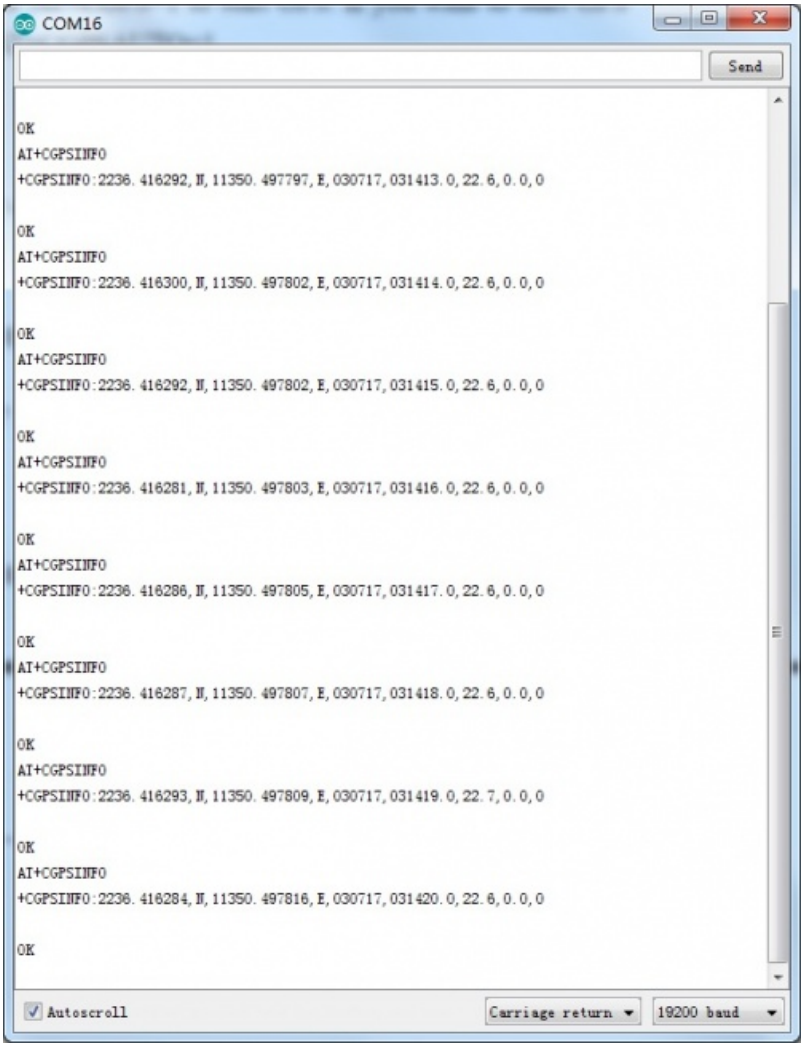
More information you can refer the SIM5360_GPS_Application_Note_V0.02 (https://www.elecrow.com/wiki/index.php?title=File:SIM5360_GPS_Application_Note_V0.02.pdf) .

Demo code of get the GPS information:

```
/*
 * Created by Island
 * Modified by keen
 * Date: 17/03/2017
 */
#include <SoftwareSerial.h>
#define DEBUG true
SoftwareSerial mySerial(7,8);

void setup(){
  Serial.begin(19200);
  mySerial.begin(19200);
  getgps();
}

void loop(){
   sendData( "AT+CGPSINFO",1000,DEBUG);
}
```

```
void getgps(void){
   sendData( "AT+CGPS=1,1",1000,DEBUG);
   sendData( "AT+CGPSINFO=0",1000,DEBUG);
}

void sendData(String command, const int timeout, boolean debug){
    String response = "";
    mySerial.println(command);
    delay(5);
    if(debug){
    long int time = millis();
    while( (time+timeout) > millis()){
      while(mySerial.available()){
        response += char(mySerial.read());
      }
    }
      Serial.print(response);
    }
}
```

The result of get the GPS information.



The usage of ECALL Function

AT Comands Examples

AT comands	Description
AT+CECALLFMT	set MSD packing format (this command must set to 1 for Europe ecall test)
AT+CECALLCFG	Configure vehicle information
AT+CECALLPOS	Set longitude and latitude
AT+CECALLTIME	Set time stamp
AT+ CMSDMESSAGEID	Set the message identifier of msd data
AT+CMSDOIDDATA	Set the optional additional data

AT+CMSDCONTROL	Set the control data in Minimum set of data (MSD)
AT+CMSD	input hex Minimum set of data(MSD)
AT+CECALLTOUT	Set T5, T6, T7 timeout value
AT+CECALLS	Make an ecall and send the MSD information once automatically
AT+CECALLE	Hang up ecall

Demo code：using Software UART

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(7, 8);
#define DEBUG true


void setup()
{
  mySerial.begin(19200);               // the GPRS baud rate
  Serial.begin(19200);               // the GPRS baud rate
  sendData("AT+CECALLFMT=1",2000,DEBUG);    //Set MSD packing format
  sendData("AT+CECALLCFG=5,18,8,\"WMJVDSVDSYA012345\",14,10,-10,20,-20",2000,DEBUG);     //Configure vehicle information
  sendData("AT+CECALLPOS=\"121.354138\",\"31.221938\"",2000,DEBUG);     //Set longitude and latitude
  sendData("AT+CECALLTIME=1,2011,10,20,15,30,30",2000,DEBUG);          //Set time stamp
  sendData("AT+CMSDCONTROL=1,1,1",2000,DEBUG);                        //Set the control data in Minimum set of data (MSD)
  sendData("AT+CMSDMESSAGEID=1",2000,DEBUG);                         //Set the initiatory message identifier of msd
  sendData("AT+CECALLTOUT=\"T5\",4000",2000,DEBUG);             //Set T5, T6, T7 timeout value
  sendData("AT+CMSDOIDDATA=\"1.2.125\",\"30304646\"",2000,DEBUG);  //Set the optional additional data
  sendData("AT+CECALLS=13xxxxxxxxx,0",2000,DEBUG);   //Make an eCall, xxxxxxxxx is the number you want to dial.
  delay(40000);
  delay(40000);
  sendData("AT+CECALLE",2000,DEBUG); //Hang up eCall.

}


void sendData(String command, const int timeout, boolean debug)
{
    String response = "";
    mySerial.println(command);
    long int time = millis();
    while( (time+timeout) > millis())
    {
      while(mySerial.available())
      {
        char c = mySerial.read();
        response+=c;
      }
    }
    if(debug)
    {
      Serial.print(response);
    }
    return response;
}


void loop()
{
if(mySerial.available())
      {
          char SerialInByte;
          SerialInByte = (unsigned char)mySerial.read();
          Serial.print( SerialInByte );
      }
}
```

FAQ

You can list you question here or contact with **techsupport@elecrow.com** for technology support.

How to buy

Resources

- SIM5360E Design files (https://www.elecrow.com/wiki/index.php?title=File:3G_Shield.zip)
- SIM5360_GPS_Application_Note_V0.02 (https://www.elecrow.com/wiki/index.php?title=File:SIM5360_GPS_Application_Note_V0.02.pdf)
- SIMCOM_SIM5360_ATC_EN_V0.16 (https://www.elecrow.com/download/SIMCOM_SIM5360_ATC_EN_V0.16%20.pdf)

Retrieved from "https://www.elecrow.com/wiki/index.php?title=SIM5360E_3G_Shield&oldid=17743"

- This page was last modified on 3 July 2017, at 10:26.

- This page has been accessed 8,028 times.