GPS/GPRS/GSM Module V3.0 (SKU:TEL0051)

From Robot Wiki

Contents

Introduction

This is a GPS/GPRS/GSM shield from DFRobot. This shield with a Quad-band GSM/GPRS engine works on frequencies EGSM 900MHz/DCS 1800MHz and GSM850 MHz/PCS 1900MHz. It also supports GPS technology for satellite navigation. It's possible for your robot and control system to send messages and use the GSM network.

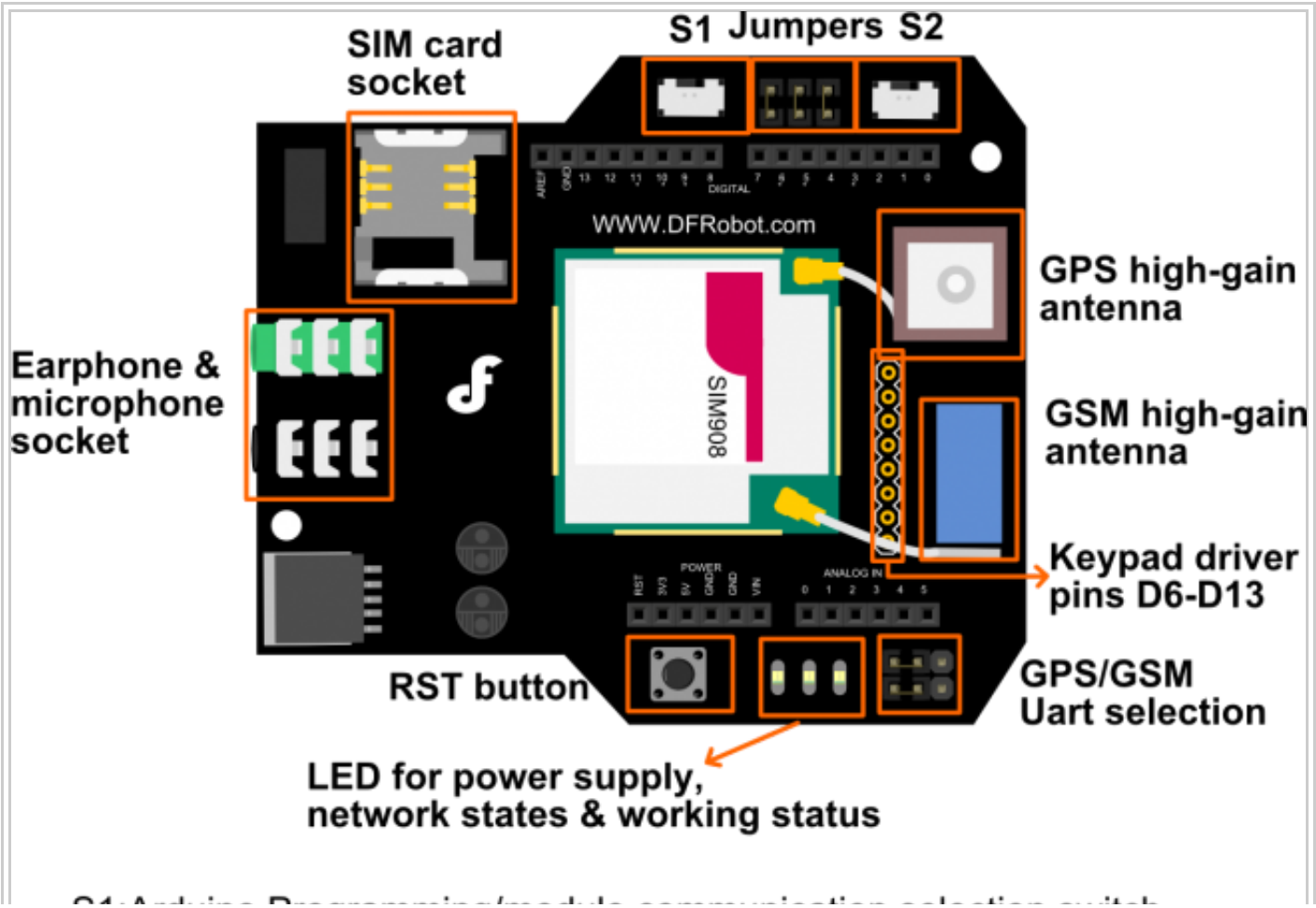
It is controlled via AT commands(GSM07.07 ,07.05 and SIMCOM enhanced AT Commands). And the design of this shield allows you to drive the GSM & GPS function directly with the computer and the Arduino Board. It includes a high-gain SMD antenna for GPS & GSM.

This GPS/GPRS/GSM shield uses an embedded SIM908 chip from SIMCom.Featuring an industry-standard interface and GPS function, the combination of both technologies allows goods, vehicles and people to be tracked seamlessly at any location and anytime with signal coverage.

Specification

- Power supply: 6-12v
- Low power consumption (100mA@7v - GSM mode)
- Quad-Band 850/900/1800/1900MHz
- GPRS multi-slot class 10
- Support GPS technology for satellite navigation
- Embeded high-gain SMD antennas for GPS & GSM
- Directly support 4*4 button pad
- USB/Arduino control switch
- Board Surface:Immersion Gold
- Size: 81x70mm

Pin Out

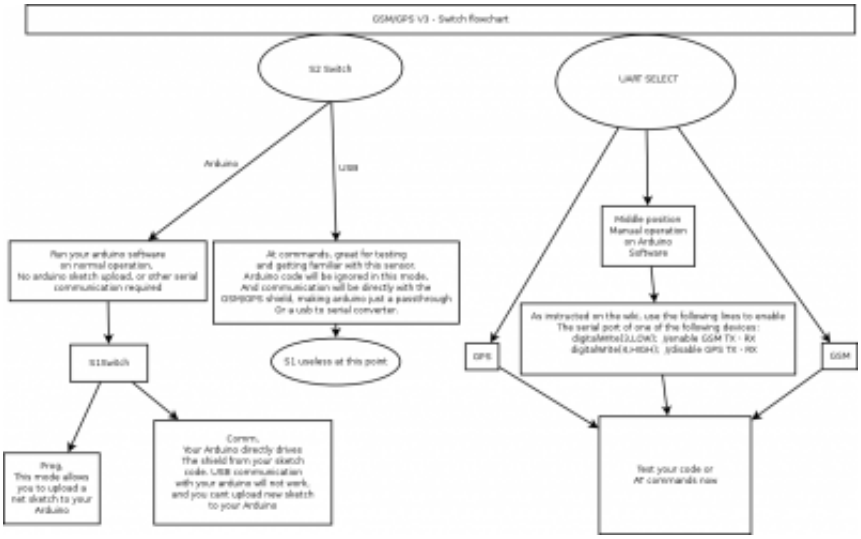


S1:Arduino Programming/module communication selection switch
S2:USB/Arduino serial communication selection switch
Jumpers:Connect the module driver pins to D3-D5 on the Arduino

```
NOTE: Two jumper caps of GPS/GSM UART SELECTION have been changed to a switch."Take off the jumper
 "slid the switch in the middle".
```

More details about switches:

- **Switch S1:** PC upload program to Arduino board/PC communicates with GPS/GPRS/GSM Module(Arduino programming/module communication).
- **Switch S2:** GPS/GPRS/GSM Module directly connects with PC through USB port or module communicates with Arduino board, which communicates with PC(USB/Arduino serial communication).



Tutorial

Hardwares prepare:

1. Arduino Uno
2. GPS/GPRS/GSM Module V3.0
3. SIM Card
4. Earphone & Microphone
5. External power supply via the power jack

It is recommended you to supply 7~12v power via the power jack. When using the GSM mode, the module requires an external power. But the power consumption is not high,just 200mA@7v, when calling.

Module driver pin jumpers: Applying the Module Pin Jumpers(J10-J12) will allocate Pin 3,4,5 for driving the module. Removing the jumpers will release the above Pins, and you could connect the driver pins to the other digital pins on your board to avoid the pin conflict.Read more (http://www.dfrobot.com/forum/index.php?topic=17186.msg21374#msg21374)

How to drive the GSM Mode via USB port

1. If your module works, the indicator Start LED will light up, this means that the module is running correctly. The LED marked "NET" is used to drive a network status indication LED.
2. Send the AT commands to the module by using Coolterm (http://freeware.the-meiers.org/CoolTermWin.zip) (or use the Arduino serial monitor).

```
Note: If you want to program the Arduino, please disconnect the coolterm to release the communicat
```

GSM mode & GPS mode Selection

Except using UART selection jumper caps, you could switch GSM and GPS function with the IO pins also. Please remove the jumper caps connected for hardware UART selection first!

Enable GPS mode & disable GSM mode:

- digitalWrite(4,LOW);//Enable GPS mode
- digitalWrite(3,HIGH);//Disable GSM mode

Enable GSM mode & disable GPS mode:

- digitalWrite(3,LOW);//Enable GSM mode
- digitalWrite(4,HIGH);//Disable GPS mode

Network indication

State	SIM908 behavior
Off	SIM908 is not running
64ms On/ 800ms Off	SIM908 not registered the network
64ms On/ 3,000ms Off	SIM908 registered to the network
64ms On/ 300ms Off	PPS GPRS communication is established

Following the steps included in the sketch below first!

```
?
  // Product name: GPS/GPRS/GSM Module V3.0
  // # Product SKU : TEL0051
1 // # Version     : 0.1
2
3 // # Description:
4 // # The sketch for driving the gsm mode via the USB interface
5
6 // # Steps:
7 // #        1. Turn the S1 switch to the Prog(right side)
8 // #        2. Turn the S2 switch to the USB side(left side)
9 // #        3. Take off the GSM/GPS jumper caps from the Uart select
10// #        4. Upload the sketch to the Arduino board(Make sure turn off other
11Serial monitor )
12// #        5. Turn the S1 switch to the comm(left side)
13// #        7. RST the board
14
15// #        wiki link-
16http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051)
17
18void setup()
19 {
20   //Init the driver pins for GSM function
21    pinMode(3,OUTPUT);
22    pinMode(4,OUTPUT);
23    pinMode(5,OUTPUT);
24   //Output GSM Timing
25    digitalWrite(5,HIGH);
26    delay(1500);
27    digitalWrite(5,LOW);
28 }
29 void loop()
30 {
31    // Use these commands instead of the hardware switch 'UART select' in order to
32enable each mode
33    // If you want to use both GMS and GPS. enable the required one in your code and
34disable the other one for each access.
35    digitalWrite(3,LOW);//enable GSM TX、RX
      digitalWrite(4,HIGH);//disable GPS TX、RX
   }
```

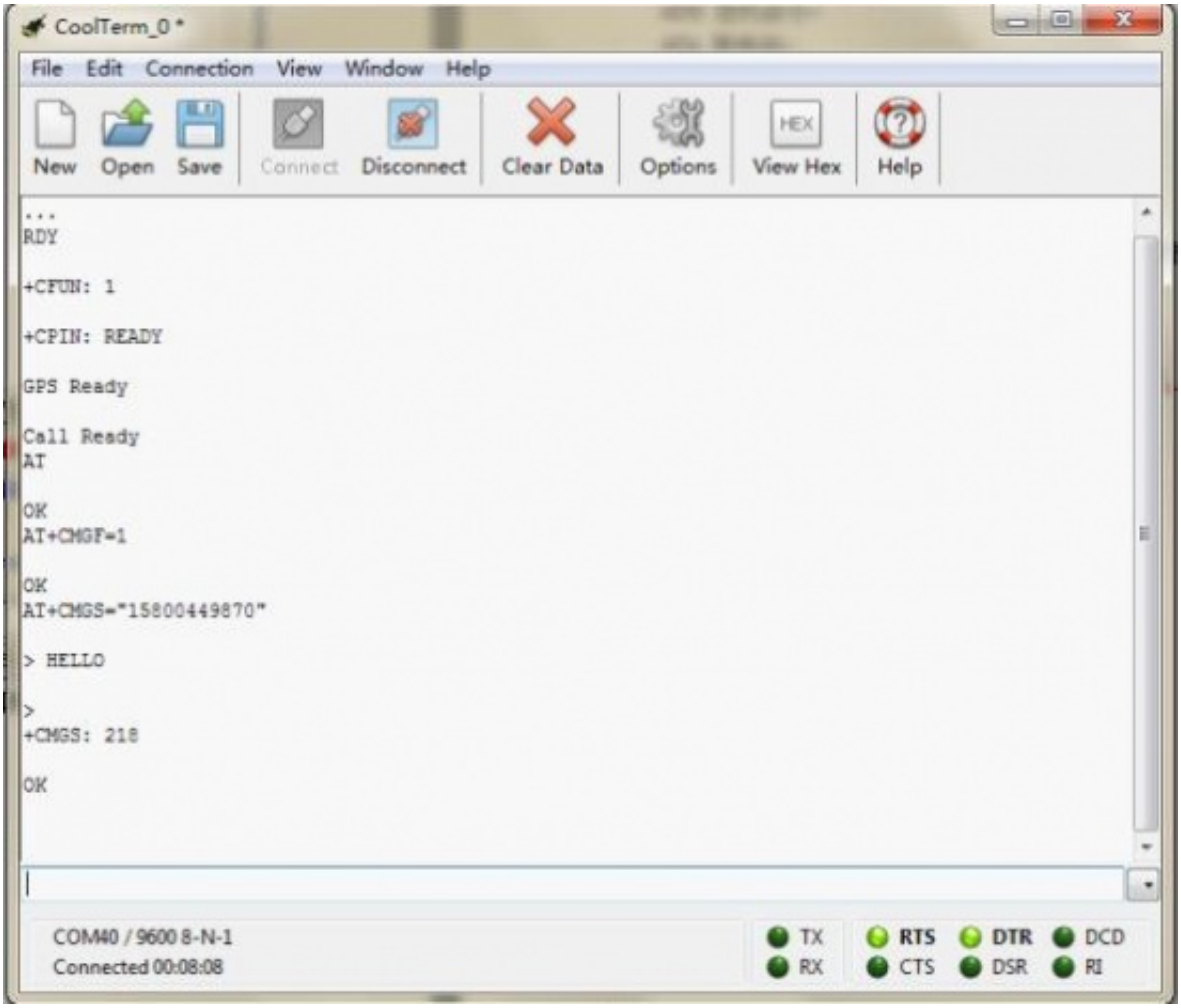
How to Send a message

Steps:

1. Send:AT
2. Send:AT+CMGF=1 (set the message to text format)
3. Send:AT+CMGS="XXXXX" （xxxx is the number of receiver ）
4. After you see '<' then send the message you want to say
5. press 'ctrl+z'(If you want to cancle,you can press Esc)

NOTE: Please do some settings as Recommended settings before connection.
If you have trouble sending Ctrl+Z, please refers to Ways to send Ctrl +Z in Coolterm

Then you will see



After several seconds,the receiver will get a message from this shield

Ways to send Ctrl +Z in Coolterm

1. After typing the message text, press enter key, it will show:

```
Call Ready
AT OK
AT+CMGF=1 OK
AT+CMGS="15827246249" > This is a message from DFRobot GSM/GPRS/GPS Module
>
```

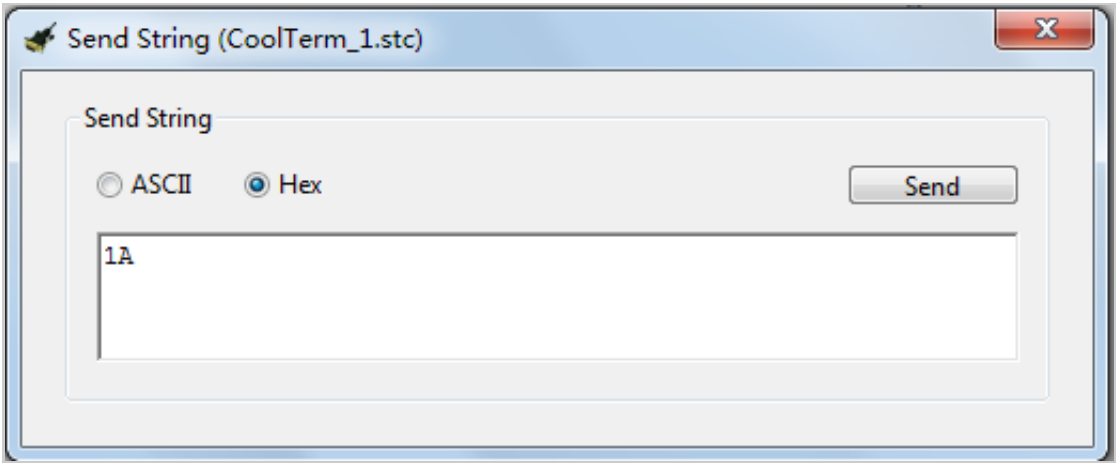
'Enter' after text finished

```
    Then, in the input area, pressing Ctrl+Z will send out the single ctrl character successfully
```

```
Call Ready
AT OK
AT+CMGF=1 OK
AT+CMGS="15827246249" > This is a message from DFRobot GSM/GPRS/GPS Module
>

+CMGS: 145

OK
```

After press Ctrl+Z

2. Ctrl characters can also be sent out in their hexadecimal format, in which form 1A is the value of Ctrl+Z:



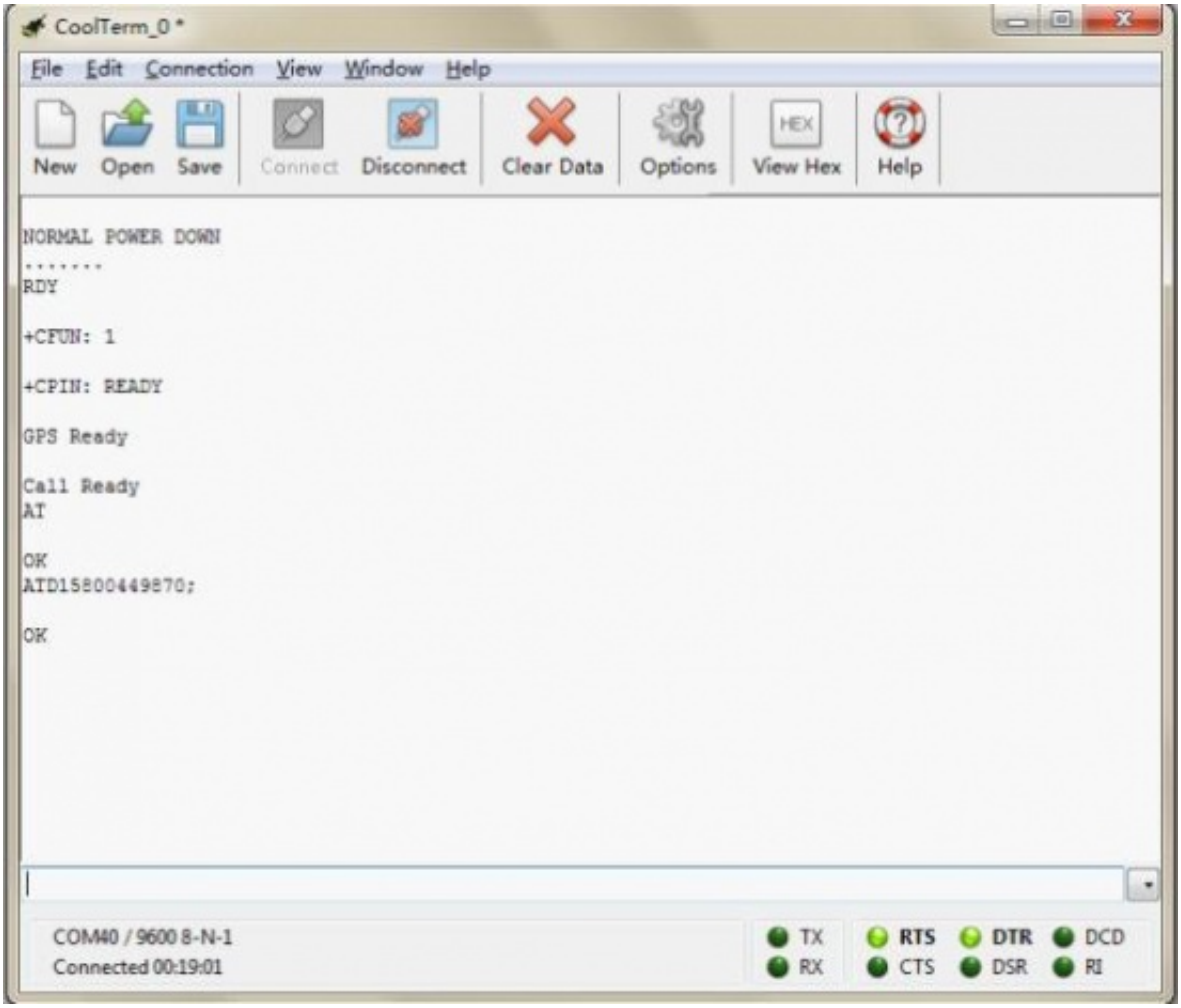
Sending Ctrl+Z with in hexadecimal format

How to Make a phone call

Steps:

1. Send:AT
2. Send:ATDXXXXXXX;（xxxxxxx is the number of receiver,don't forget the **;**）

Then you will see



After several seconds,the receiver will get a phone call from this shield

Some AT commands

- ATH : Hang up the phone
- ATA : Answer the phone

How to drive the GPS Mode via USB port

You should take the module outdoor, so that you can get the GPS datas

?

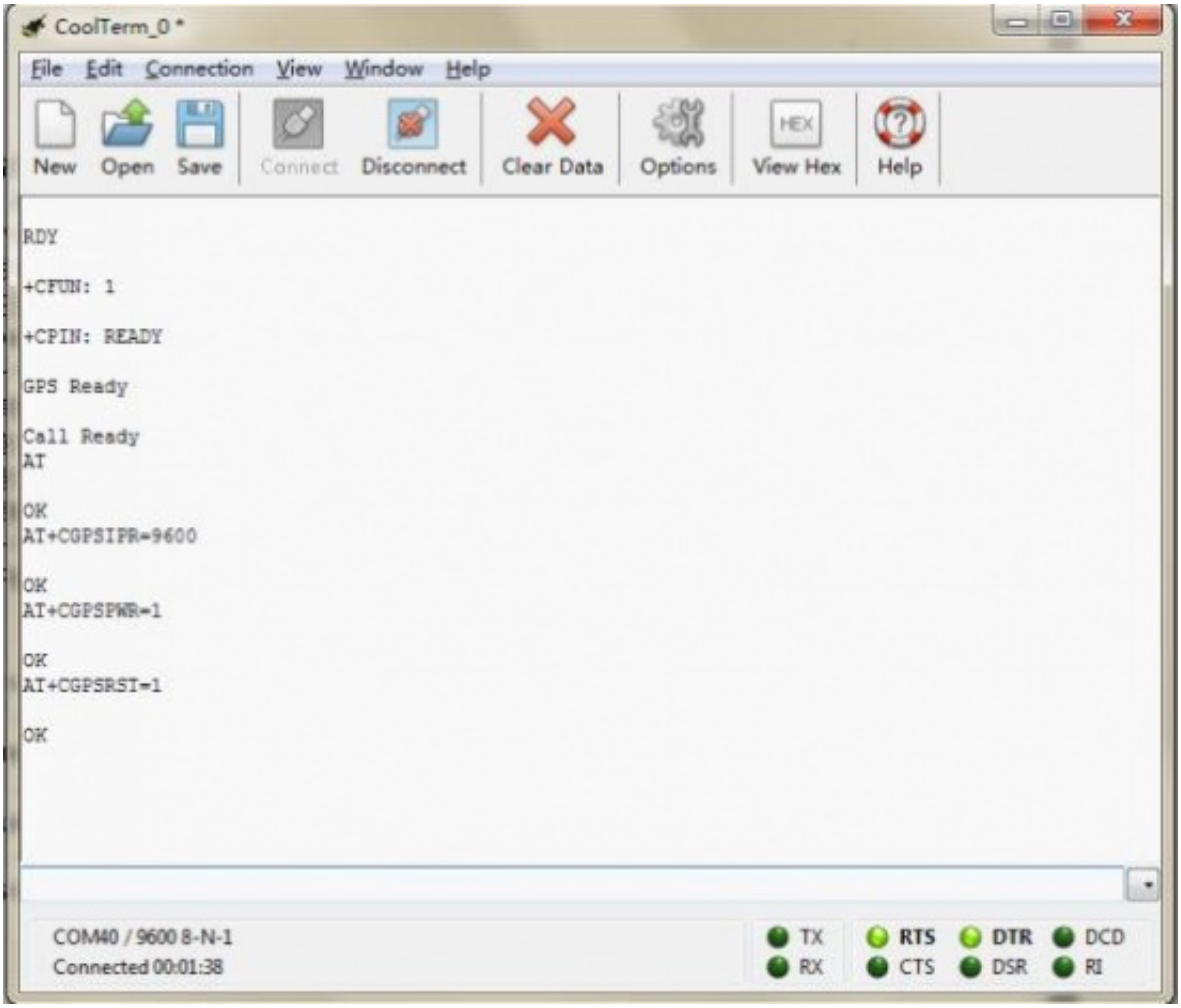
```
  // Product name: GPS/GPRS/GSM Module V3.0
1 // # Product SKU : TEL0051
2 // # Version     : 0.1
3
4 // # Description:
5 // # The sketch for driving the gps mode via the USB interface
6
7 // # Steps:
8 // #        1. Turn the S1 switch to the Prog(right side)
9 // #        2. Turn the S2 switch to the USB side(left side)
```

```
// #        3. Plug the GSM/GPS jumper caps to the GSM side
// #        4. Upload the sketch to the Arduino board(Make sure turn off other
Serial monitor )
// #        5. Turn the S1 switch to the comm(left side)
// #        7. RST the board until the START led is on

// #        wiki link-
http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051)

void setup()
 {
   //Init the driver pins for GSM function
    pinMode(3,OUTPUT);
    pinMode(4,OUTPUT);
    pinMode(5,OUTPUT);
  //Output GSM Timing
    digitalWrite(5,HIGH);
    delay(1500);
    digitalWrite(5,LOW);
 }
 void loop()
 {
    digitalWrite(3,HIGH);//disable GSM TX、RX
    digitalWrite(4,HIGH);//disable GPS TX、RX
  }
```

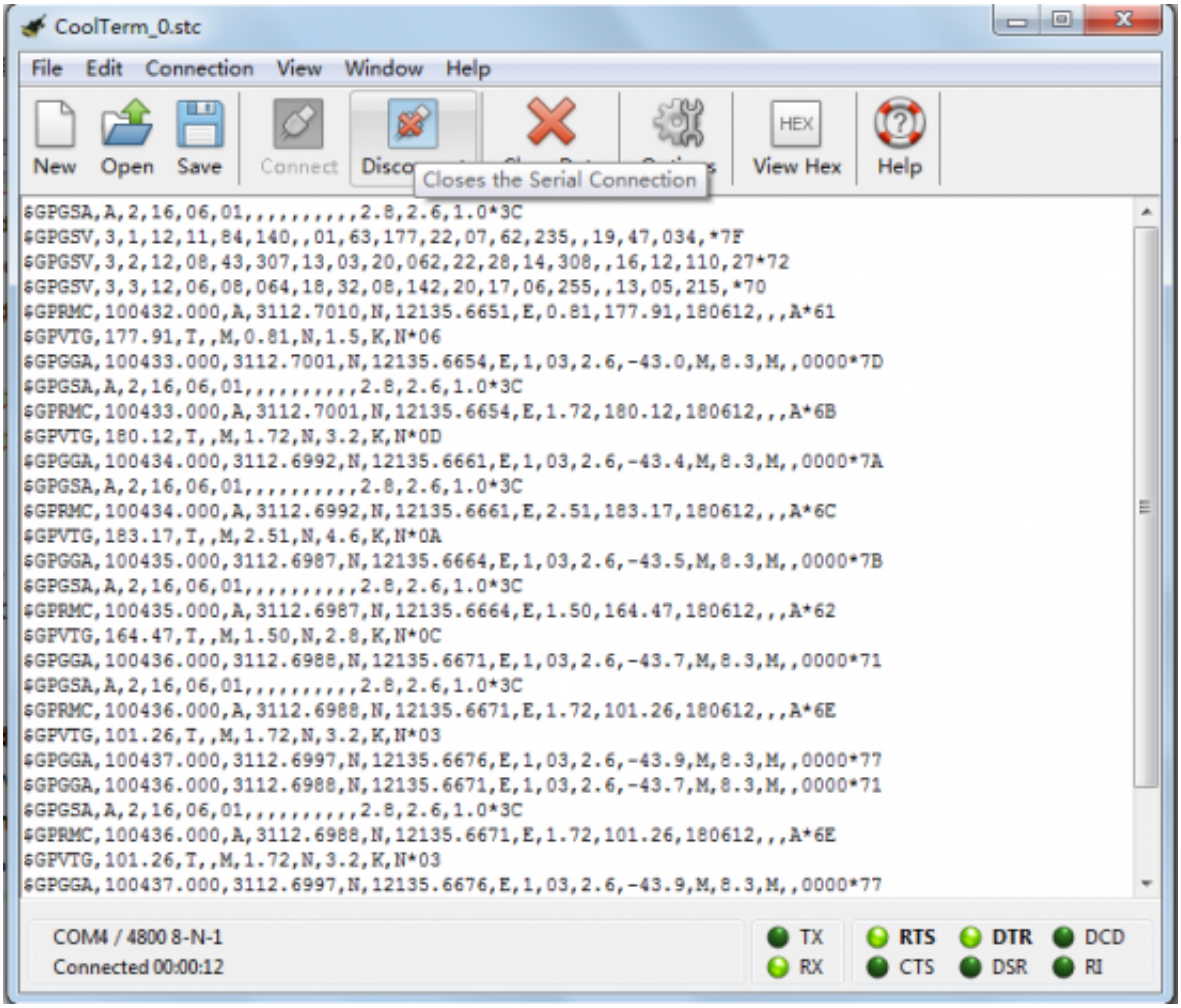
Steps:

1. Send:AT
2. Send:AT+CGPSIPR=9600 (set the baud rate)
3. Send:AT+CGPSPWR=1 （turn on GPS power supply）
4. Send:AT+CGPSRST=1 （reset GPS in autonomy mode）

Then you can see



Plug the jumper caps back to the GPS side.Then you will see as the following



For location of the data received, please refer to Location Mapping (GPRMC) (http://www.sanav.com/gps_tracking/webtrac-4/maps/MapLocationGPRMC.aspx)

How to drive the GSM Mode via Arduino board

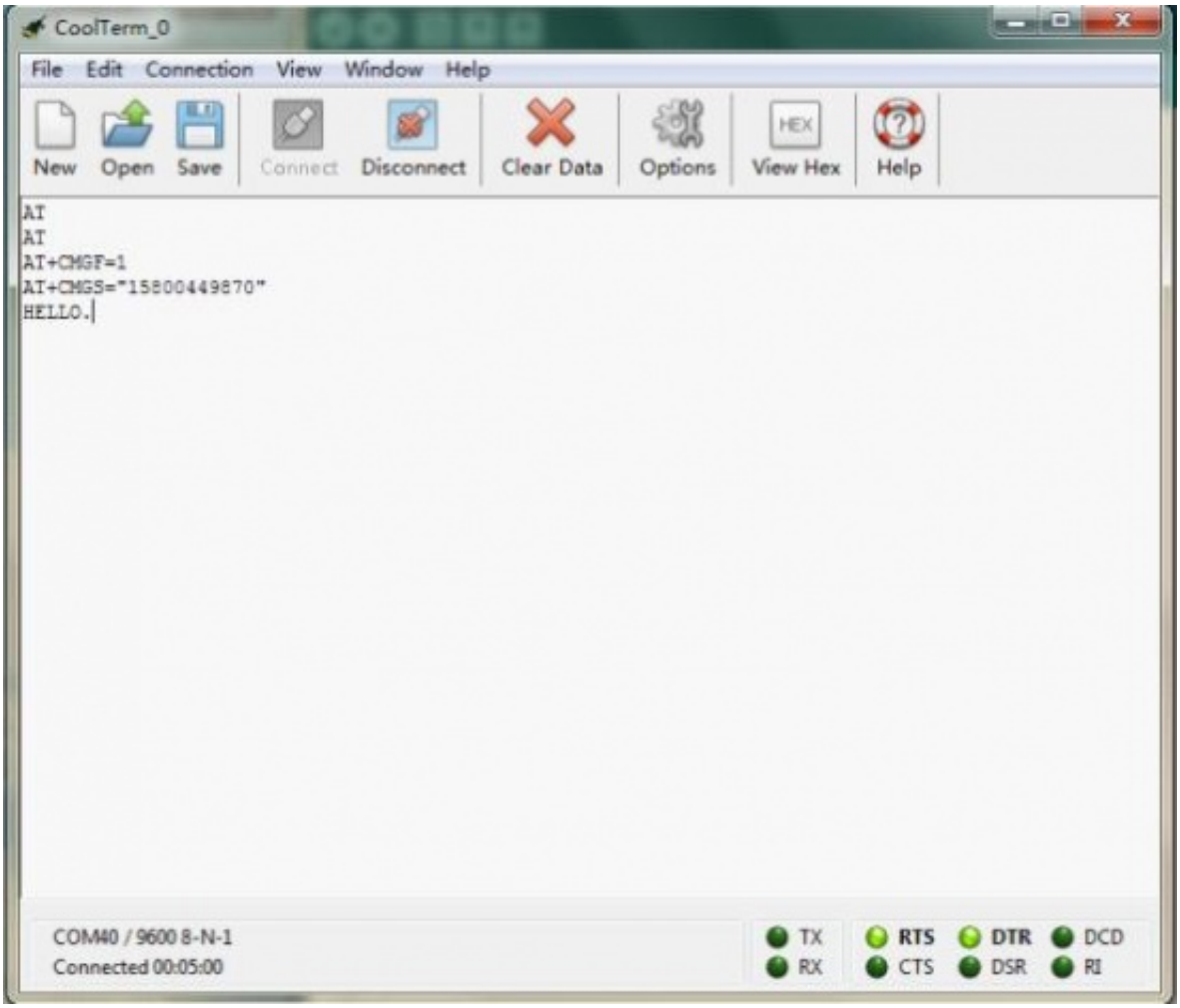
How to Send a message

?

```
// Product name: GPS/GPRS/GSM Module V3.0
// # Product SKU : TEL0051
// # Version     : 0.1

// # Description:
// # The sketch for driving the gsm mode via the Arduino board

// # Steps:
// #        1. Turn the S1 switch to the Prog(right side)
// #        2. Turn the S2 switch to the Arduino side(left side)
// #        3. Take off the GSM/GPS jumper caps from the Uart select
// #        4. Upload the sketch to the Arduino board
// #        5. Turn the S1 switch to the comm(left side)
```

```
// #        6. Plug the jumper caps back to GSM side
// #        7. RST the board

// #        wiki link-
http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051)

byte gsmDriverPin[3] = {
  3,4,5};//The default digital driver pins for the GSM and GPS mode
//If you want to change the digital driver pins
//or you have a conflict with D3~D5 on Arduino board,
//you can remove the J10~J12 jumpers to reconnect other driver pins for the module!
void setup()
{
  //Init the driver pins for GSM function
  for(int i = 0 ; i < 3; i++){
    pinMode(gsmDriverPin[i],OUTPUT);
  }
  digitalWrite(5,HIGH);//Output GSM Timing
  delay(1500);
  digitalWrite(5,LOW);
  digitalWrite(3,LOW);//Enable the GSM mode
  digitalWrite(4,HIGH);//Disable the GPS mode
  delay(2000);
  Serial.begin(9600); //set the baud rate
  delay(5000);//call ready
  delay(5000);
  delay(5000);
}

void loop()
{
  Serial.println("AT"); //Send AT command
  delay(2000);
  Serial.println("AT");
  delay(2000);
  //Send message
  Serial.println("AT+CMGF=1");
  delay(1000);
  Serial.println("AT+CMGS=\"15800449871\"");//Change the receiver phone number
  delay(1000);
  Serial.print("HELLO");//the message you want to send
  delay(1000);
  Serial.write(26);
  while(1);
}
```

You can see:



After several seconds,the receiver will get a message from this shield

How to Control your Arduino via SMS

Follow the forum discussion with more coding examples and options on this link Click Me ! (http://www.dfrobot.com/forum/index.php?topic=945.msg4514#msg4514)

?

```
// Product name: GPS/GPRS/GSM Module V3.0
// # Product SKU : TEL0051

// # Description:
// # The sketch for controling the GSM/GPRS/GPS module via SMS.
// # Steps:
// #        1. Turn the S1 switch to the Prog(right side)
// #        2. Turn the S2 switch to the USB side(left side)
// #        3. Plug the GSM/GPS jumper caps to the GSM side
// #        4. Upload the sketch to the Arduino board(Make sure turn off other
Serial monitor )
// #        5. Turn the S1 switch to the comm(left side)
```

```
// #        6. Turn the S2 switch to the Arduino(right side)
// #        7. RST the board until the START led is on(make sure you have >6V power
supply)
// #        8. Plug the long side of LED into pin 8 and short side into GND
// #        9. Start sending "LH" and "LL" to your board to turn LED on and off.

/*
 *  created:    2013-11-14
 *  by:     Grey
 *  Version:    0.3
 *  Attention: if you send the wrong SMS command to the module, just need to press
RST.
 *  This version can't watch the module status via the serial monitor, it only
display the Arduino command.
 *  If you want to watch the status,use the SoftwareSerial or the board with another
serial port plese.
 */


byte gsmDriverPin[3] = {
  3,4,5};//The default digital driver pins for the GSM and GPS mode
//If you want to change the digital driver pins
//or you have a conflict with D3~D5 on Arduino board,
//you can remove the J10~J12 jumpers to reconnect other driver pins for the module!
int ledpin = 8;
char inchar;
void setup()
{
  //Init the driver pins for GSM function
  for(int i = 0 ; i < 3; i++){
    pinMode(gsmDriverPin[i],OUTPUT);
  }
  pinMode(ledpin,OUTPUT);
  Serial.begin(9600);                                    //set the baud rate
  digitalWrite(5,HIGH);                                   //Output GSM Timing
  delay(1500);
  digitalWrite(5,LOW);
  digitalWrite(3,LOW);                                    //Enable the GSM mode
  digitalWrite(4,HIGH);                                   //Disable the GPS mode
  delay(2000);
  delay(5000);                                            //call ready
  delay(5000);
  Serial.println("AT+CMGD=1,4");                         //Delete all SMS in box
}
```

```
53
54void loop()
55{
56  if(Serial.available()>0)
57  {
58    inchar=Serial.read();
59    if(inchar=='T')
60    {
61      delay(10);
62      inchar=Serial.read();
63      if (inchar=='I')                                    //When the GSM module get
64the message, it will display the sign '+CMTI "SM", 1' in the serial port
65      {
66        delay(10);
67        Serial.println("AT+CMGR=1");                     //When Arduino read the
68sign, send the "read" AT command to the module
69        delay(10);
70      }
71    }
72    else if (inchar=='L')
73    {
74      delay(10);
75      inchar=Serial.read();
76      if (inchar=='H')                                   //Thw SMS("LH") was
77display in the Serial port, and Arduino has recognize it.
78      {
79        delay(10);
80        digitalWrite(ledpin,HIGH);                       //Turn on led
81        delay(50);
82        Serial.println("AT+CMGD=1,4");                  //Delete all message
83        delay(500);
84      }
85      if (inchar=='L')                                  //Thw SMS("LH") was display
86in the Serial port, and Arduino has recognize it.
87      {
88        delay(10);
89        digitalWrite(ledpin,LOW);                        //Turn off led
90        delay(50);
91        Serial.println("AT+CMGD=1,4");                  //Delete all message
92        delay(500);
        }
      }
    }
  }
```

When you send the SMS "LH" to the module, it WILL turn led on; when you send the SMS "LL", it will be turned off.

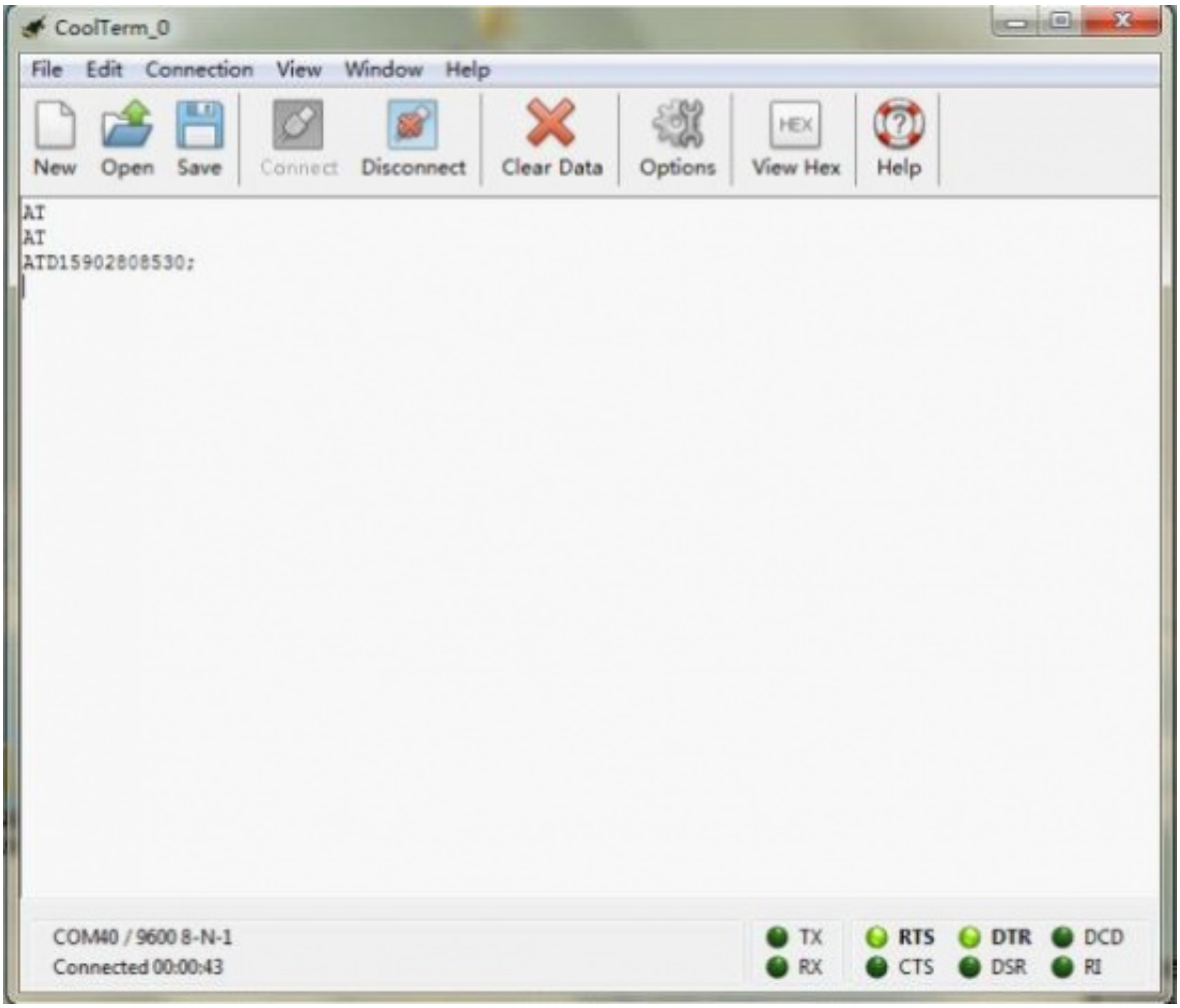
How to Make a phone call

?

```
// Product name: GPS/GPRS/GSM Module V3.0
// # Product SKU : TEL0051
// # Version     : 0.1

// # Description:
// # The sketch for driving the gsm mode via the Arduino board

// # Steps:
// #        1. Turn the S1 switch to the Prog(right side)
// #        2. Turn the S2 switch to the Arduino side(left side)
// #        3. Take off the GSM/GPS jumper caps from the Uart select
// #        4. Upload the sketch to the Arduino board
// #        5. Turn the S1 switch to the comm(left side)
// #        6. Plug the jumper caps back to GSM side
// #        7. RST the board

// #        wiki link-
http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051)

byte gsmDriverPin[3] = {
  3,4,5};//The default digital driver pins for the GSM and GPS mode
//If you want to change the digital driver pins
//or you have a conflict with D3~D5 on Arduino board,
//you can remove the J10~J12 jumpers to reconnect other driver pins for the module!

void setup()
{
 //Init the driver pins for GSM function
  for(int i = 0 ; i < 3; i++){
    pinMode(gsmDriverPin[i],OUTPUT);
  }
  digitalWrite(5,HIGH);//Output GSM Timing
  delay(1500);
  digitalWrite(5,LOW);
  digitalWrite(3,LOW);//Enable the GSM mode
  digitalWrite(4,HIGH);//Disable the GPS mode
  delay(2000);
  Serial.begin(9600); //set the baud rate
  delay(5000);//call ready
```

```
40   delay(5000);
41   delay(5000);
42}
43
44void loop()
45{
46    Serial.println("AT");//Send AT command
47    delay(2000);
48    Serial.println("AT");
49    delay(2000);
50  //Make a phone call
51    Serial.println("ATD15902808530;");//Change the receiver phone number
52    while(1);
  }
```

You can see:



After several seconds,the receiver will get a phone call from this shield

How to drive the GPS Mode via Arduino board

?

```
// Product name: GPS/GPRS/GSM Module V3.0
// # Product SKU : TEL0051
// # Version     : 0.1

// # Description:
// # The sketch for driving the gps mode via the Arduino board

// # Steps:
// #        1. Turn the S1 switch to the Prog(right side)
// #        2. Turn the S2 switch to the Arduino side(left side)
// #        3. Take off the GSM/GPS jumper caps from the Uart select
// #        4. Upload the sketch to the Arduino board
// #        5. Turn the S1 switch to the comm(left side)
// #        6. Remove the jumpers(old version) or set the UART select switch to
middle.
// #        7. RST the board

// #        If you get 'inf' values, go outdoors and wait until it is connected.
// #        wiki link-
http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051)

double Datatransfer(char *data_buf,char num)//convert the data to the float type
{                                           //*data_buf: the data
array
  double temp=0.0;                          //the number of the right of a decimal
point
  unsigned char i,j;

  if(data_buf[0]=='-')
  {
    i=1;
    //process the data array
    while(data_buf[i]!='.')
      temp=temp*10+(data_buf[i++]-0x30);
    for(j=0;j<num;j++)
      temp=temp*10+(data_buf[++i]-0x30);
    //convert the int type to the float type
    for(j=0;j<num;j++)
      temp=temp/10;
    //convert to the negative numbe
    temp=0-temp;
  }
  else//for the positive number
  {
    i=0;

```

```
    while(data_buf[i]!='.')
      temp=temp*10+(data_buf[i++]-0x30);
    for(j=0;j<num;j++)
      temp=temp*10+(data_buf[++i]-0x30);
    for(j=0;j<num;j++)
      temp=temp/10 ;
  }
  return temp;
}

char ID()//Match the ID commands
{
  char i=0;
  char value[6]={
    '$','G','P','G','G','A'  };//match the gps protocol
  char val[6]={
    '0','0','0','0','0','0'  };

  while(1)
  {
    if(Serial.available())
    {
      val[i] = Serial.read();//get the data from the serial interface
      if(val[i]==value[i]) //Match the protocol
      {
        i++;
        if(i==6)
        {
          i=0;
          return 1;//break out after get the command
        }
      }
      else
        i=0;
    }
  }
}

void comma(char num)//get ','
{
  char val;
  char count=0;//count the number of ','

  while(1)
  {
```

```
    if(Serial.available())
    {
      val = Serial.read();
      if(val==',')
        count++;
    }
    if(count==num)//if the command is right, run return
      return;
  }

}
void UTC()//get the UTC data -- the time
{
  char i;
  char time[9]={
    '0','0','0','0','0','0','0','0','0'
  };
  double t=0.0;

  if( ID())//check ID
  {
    comma(1);//remove 1 ','
    //get the datas after headers
    while(1)
    {
      if(Serial.available())
      {
        time[i] = Serial.read();
        i++;
      }
      if(i==9)
      {
        i=0;
        t=Datatransfer(time,2);//convert data
        t=t+80000.00;//convert to the chinese time GMT+8 Time zone
        Serial.println(t);//Print data
        return;
      }
    }
  }
}
void latitude()//get latitude
{
  char i;
  char lat[10]={
```

```
    '0','0','0','0','0','0','0','0','0','0'
  };


  if( ID())
  {
    comma(2);
    while(1)
    {
      if(Serial.available())
      {
        lat[i] = Serial.read();
        i++;
      }
      if(i==10)
      {
        i=0;
        Serial.println(Datatransfer(lat,5),5);//print latitude
        return;
      }
    }
  }
}
void lat_dir()//get dimensions
{
  char i=0,val;

  if( ID())
  {
    comma(3);
    while(1)
    {
      if(Serial.available())
      {
        val = Serial.read();
        Serial.write(val);
        Serial.println();
        i++;
      }
      if(i==1)
      {
        i=0;
        return;
      }
    }
```

```
}
}
void longitude()//get longitude
{
  char i;
  char lon[11]={
    '0','0','0','0','0','0','0','0','0','0','0'
  };

  if( ID())
  {
    comma(4);
    while(1)
    {
      if(Serial.available())
      {
        lon[i] = Serial.read();
        i++;
      }
      if(i==11)
      {
        i=0;
        Serial.println(Datatransfer(lon,5),5);
        return;
      }
    }
  }
}
void lon_dir()//get direction data
{
  char i=0,val;

  if( ID())
  {
    comma(5);
    while(1)
    {
      if(Serial.available())
      {
        val = Serial.read();
        Serial.write(val); //Serial.println(val,BYTE);
        Serial.println();
        i++;
      }
      if(i==1)
```

```
      {
        i=0;
        return;
      }
    }
  }
}
void altitude()//get altitude data
{
  char i,flag=0;
  char alt[8]={
    '0','0','0','0','0','0','0','0'
  };

  if( ID())
  {
    comma(9);
    while(1)
    {
      if(Serial.available())
      {
        alt[i] = Serial.read();
        if(alt[i]==',')
          flag=1;
        else
          i++;
      }
      if(flag)
      {
        i=0;
        Serial.println(Datatransfer(alt,1),1);//print altitude data
        return;
      }
    }
  }
}
void setup()
{
  pinMode(3,OUTPUT);//The default digital driver pins for the GSM and GPS mode
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  digitalWrite(5,HIGH);
  delay(1500);
  digitalWrite(5,LOW);

```

```
  digitalWrite(3,LOW);//Enable GSM mode
  digitalWrite(4,HIGH);//Disable GPS mode
  delay(2000);
  Serial.begin(9600);
  delay(5000);//GPS ready

  Serial.println("AT");
  delay(2000);
  //turn on GPS power supply
  Serial.println("AT+CGPSPWR=1");
  delay(1000);
  //reset GPS in autonomy mode
  Serial.println("AT+CGPSRST=1");
  delay(1000);

  digitalWrite(4,LOW);//Enable GPS mode
  digitalWrite(3,HIGH);//Disable GSM mode
  delay(2000);

  Serial.println("$GPGGA statement information: ");
}
void loop()
{
  while(1)
  {
    Serial.print("UTC:");
    UTC();
    Serial.print("Lat:");
    latitude();
    Serial.print("Dir:");
    lat_dir();
    Serial.print("Lon:");
    longitude();
    Serial.print("Dir:");
    lon_dir();
    Serial.print("Alt:");
    altitude();
    Serial.println(' ');
    Serial.println(' ');
  }
}
```

GPS Sample Code

This code introduces some testing features and useful code. It's mainly coded for Leonardo devices using the second serial port to handle GSM/GPS communications while USB serial port is left for debugging purposes. Data from GPS is parsed and stored on variables to be used freely on your code.

```
?
1  /********************* start of gps_gsm_sim908.h ***********************/
2
3  /*
4   *  by 2013-08-02
5   *  test on Leonardo
6   *  Serial1 to GPS
7   *
8   */
9
10 //debug
11 //#define DEBUG
12
13 #include <Arduino.h>
14
15
16 #define gps_enable()    digitalWrite (4, LOW)
17 #define gps_disable()   digitalWrite (4, HIGH)
18
19 #define gsm_enable()    digitalWrite (3, LOW)
20 #define gsm_disable()   digitalWrite (3, HIGH)
21
22 #define GPS_BUF_SIZE 500
23 #define GGA_NUM 15
24 #define RMC_NUM 14
25
26 //
27 char *gga_table[GGA_NUM] = {
28     "Message ID",           //0
29     "UTC Time",             //1
30     "Latitude",             //2
31     "N/S Indicator",        //3
32     "Longitude",            //4
33     "E/W Indicator",        //5
34     "Position Fix Indicator",   //6
35     "Satellites Used",      //7
36     "HDOP",             //8
37     "MSL Altitude",         //9
38     "Units(M)",             //10
39     "Geoid Separation",         //11
40     "Units",            //12
41     "Diff.Ref.Station ID",      //13
```

```
    "Checksum",         //14
};

//
char *gprmc_table[RMC_NUM] = {
    "Message ID",           //0
    "UTC Time",                     //1
    "Status",                       //2
    "Latitude",                     //3
    "N/S Indicator",                //4
    "Langitude",                    //5
    "E/W Indicator",                //6
    "Speed Over Ground",            //7
    "Course Over Ground",           //8
    "Date",                         //9
    "Magnetic Variation",           //10
    "East/West Indicator",          //11
    "Mode",                         //12
    "Checksum",                     //13
};


//save data from GPS
uint8_t gps_buf[GPS_BUF_SIZE];

//save pointer of gga block
uint8_t* gga_p[GGA_NUM];
uint8_t* gprmc_p[RMC_NUM];

// check sum using xor
uint8_t checksum_xor (uint8_t *array, uint8_t leng) {
    uint8_t sum = array[0];
    for (uint8_t i=1; i<leng; i++) {
        sum ^= array[i];
    }
    return sum;
}


//
void start_gps () {
    digitalWrite (5, HIGH);
    delay (1500);
    digitalWrite (5, LOW);
    delay (1500);
```

```
87
88      gsm_enable ();
89      gps_disable ();
90
91      delay (2000);
92      #ifdef DEBUG
93      Serial.println ("waiting for GPS! ");
94      #endif
95
96      Serial1.println ("AT");
97      #ifdef DEBUG
98      Serial.println ("Send AT");
99      #endif
100     delay (1000);
101     Serial1.println ("AT+CGPSPWR=1");
102     #ifdef DEBUG
103     Serial.println ("Send AT+CGPSPWR=1");
104     #endif
105     delay (1000);
106     Serial1.println ("AT+CGPSRST=1");
107     #ifdef DEBUG
108     Serial.println ("Send AT+CGPSRST=1");
109     #endif
110     delay (1000);
111
112     gsm_disable ();
113     gps_enable ();
114
115     delay (2000);
116     #ifdef DEBUG
117     Serial.println ("$GPGGA statement information: ");
118     #endif
119 }
120
121 // read data to gps_buf[] from GPS
122 static int gps_read () {
123     uint32_t start_time = millis ();
124     while (!Serial1.available ()) {
125         if (millis() - start_time > 1500) {
126             #ifdef DEBUG
127             Serial.println ("restart GPS......");
128             #endif
129             start_gps ();
130         }
131     }
```

```
132     for (int i=0; i<GPS_BUF_SIZE; i++) {
133         delay (7);
134         if (Serial1.available ()) {
135             gps_buf [i] = Serial1.read ();
136         } else {
137             #ifdef DEBUG
138             Serial.print ("read ");
139             Serial.print (i);
140             Serial.println (" character");
141             #endif
142             return 1;
143         }
144     }
145     #ifdef DEBUG
146     Serial.println ("error! data is so big!");
147     #endif
148     return 0;
149 }
150 
151 //test head of gps_buf[] if is "$GPGGA" or not
152 static int is_GPGGA () {
153     char gga_id[7] = "$GPGGA";
154     for (int i=0; i<6; i++)
155         if (gga_id[i] != gps_buf[i])
156             return 0;
157     return 1;
158 }
159 
160 //
161 static uint8_t get_gga_leng () {
162     uint8_t l;
163     for (l=0; l<GPS_BUF_SIZE && gps_buf[l] != 0x0d ; l++);
164     return l;
165 }
166 
167 // build gga_p[] by gps_buf
168 static void build_gga_p () {
169     int p,b;
170     for (p=b=0; p<GGA_NUM && b<GPS_BUF_SIZE; p++,b++) {
171         gga_p[p] = (gps_buf+b);//
172         if (gps_buf[b] == ',')
173             continue;
174         for (b++; b<GPS_BUF_SIZE && gps_buf[b]!=','; b++);
175     }
176 }
```

```
177
178//test if fix
179int gps_gga_is_fix (void) {
180    if (gga_p[6][0] == '1')
181        return 1;
182    else
183        return 0;
184}
185
186//get gga checksum
187static uint8_t gps_gga_checksum () {
188    uint8_t sum = 0;
189    if (gga_p[14][0] != '*')
190        return 0;
191    if (gga_p[14][2] >= '0' && gga_p[14][2] <= '9')
192        sum = gga_p[14][2] - '0';
193    else
194        sum = gga_p[14][2] - 'A' + 10;
195    if (gga_p[14][1] >= '0' && gga_p[14][1] <= '9')
196        sum += (gga_p[14][1] - '0') * 16;
197    else
198        sum += (gga_p[14][1] - 'A' + 10) * 16;
199    return sum;
200}
201
202//check sum of gga
203static int checksum_gga () {
204    uint8_t sum = checksum_xor (gps_buf+1, get_gga_leng ()-4);
205    return sum - gps_gga_checksum ();
206}
207
208// set gga, change ',' to '\0'
209static void gps_gga_set_str () {
210    int i;
211    for (i=0; gps_buf[i] != 0x0d && i<GPS_BUF_SIZE; i++)
212        if (gps_buf[i] == ',')
213            gps_buf[i] = '\0';
214    //gps_buf[i] = '\0';
215}
216
217//
218int gps_get_gga (void) {
219    int stat = 0;
220    if (gps_read ()) {
221        if (is_GPGGA ()) {
```

```
222             build_gga_p (); // build *gga_p[] by gps_buf
223             gps_gga_set_str ();
224             if (checksum_gga () == 0)
225                 stat = 0;
226             else
227                 stat = 1;
228         } else
229             stat = 2;
230     } else
231         stat = 3;
232 
233     return stat;
234 }
235 
236 
237 //get UTC second
238 uint8_t gps_gga_utc_ss () {
239     return (gga_p[1][4]-'0')*10+gga_p[1][5]-'0';
240 }
241 
242 //get UTC minute
243 uint8_t gps_gga_utc_mm () {
244     return (gga_p[1][2]-'0')*10+gga_p[1][3]-'0';
245 }
246 
247 //get UTC hour
248 uint8_t gps_gga_utc_hh () {
249     return (gga_p[1][0]-'0')*10+gga_p[1][1]-'0';
250 }
251 
252 //return UTC Time string, hhmmss
253 char* gps_gga_utc_s () {
254     return (char*)gga_p[1];
255 }
256 
257 //get latitude
258 double gps_gga_lat () {
259     return atof ((char*)gga_p[2]);
260 }
261 
262 //get latitude
263 char* gps_gga_lat_s () {
264     return (char*)gga_p[2];
265 }
266 
```

```
267//get longitude
268double gps_gga_long () {
269    return atof ((char*)gga_p[4]);
270}
271
272//get longitude
273char* gps_gga_long_s () {
274    return (char*)gga_p[4];
275}
276
277//get HDOP
278double gps_gga_HDOP () {
279    return atof ((char*)gga_p[8]);
280}
281
282//get HDOP
283char* gps_gga_HDOP_s () {
284    return (char*)gga_p[8];
285}
286
287//get N/S
288char* gps_gga_NS () {
289    return (char*)gga_p[3];
290    /*
291    if (gga_p[3][0] == '\0')
292        return '0';
293    else if (gga_p[3][0] == 'N' || gga_p[3][0] == 'S')
294        return gga_p[3][0];
295    else
296        return '?';
297        */
298}
299
300//get E/W
301char* gps_gga_EW () {
302    return (char*)gga_p[5];
303    /*
304    if (gga_p[5][0] == '\0')
305        return '0';
306    else if (gga_p[5][0] == 'E' || gga_p[5][0] == 'W')
307        return gga_p[5][0];
308    else
309        return '?';
310        */
311}
```

```
312
313//
314double gps_gga_MSL () {
315    return atof ((char*)gga_p[9]);
316}
317
318//
319char* gps_gga_MSL_s () {
320    return (char*)gga_p[9];
321}
322
323//get gpggpa Geoid Separation
324double gps_gga_geoid_sep () {
325    return atof ((char*)gga_p[11]);
326}
327
328//get gpggpa Geoid Separation
329char* gps_gga_geoid_sep_s () {
330    return (char*)gga_p[11];
331}
332
333#ifdef DEBUG
334//
335void gps_gga_print () {
336    for (int i=0; i<GPS_BUF_SIZE && gps_buf[i]!=0xd; i++)
337        Serial.print ((char)gps_buf[i]);
338    Serial.println ();
339}
340#endif
341
342/*
343void send_string (char* numble, char*string) {
344        char num_buf[25];
345        sprintf (num_buf, "AT+CMGS=\"%s\"", numble);
346        gsm_enable ();
347        gps_disable ();
348        delay (2000);
349        Serial1.println ("AT");
350        delay (200);
351        Serial1.println ("AT");
352        delay (200);
353        Serial1.println ("AT+CMGF=1");
354        delay (200);
355        Serial1.println (num_buf);
356        delay (200);
```

```
357         Serial1.println (string);
358         Serial1.write (26);
359 }
360 */
361
362 // set mobile numble
363 void gsm_set_numble (char *numble) {
364         char num_buf[25];
365         sprintf (num_buf, "AT+CMGS=\"%s\"", numble);
366         gsm_enable ();
367         gps_disable ();
368         delay (2000);
369         Serial1.println ("AT");
370         delay (200);
371         Serial1.println ("AT");
372         delay (200);
373         Serial1.println ("AT+CMGF=1");
374         delay (200);
375         Serial1.println (num_buf);
376         delay (200);
377 }
378
379 // send message to mobile
380 void gsm_send_message (char *message) {
381     Serial1.println (message);
382 }
383
384 //
385 void gsm_end_send () {
386     Serial1.write (26);
387     delay (200);
388     gsm_disable ();
389     gps_enable ();
390     delay (2000);
391 }
392
393
394 //
395 void gps_init () {
396     pinMode (3, OUTPUT);
397     pinMode (4, OUTPUT);
398     pinMode (5, OUTPUT);
399
400 }
401
```

```
/********************* end of gps_gsm_sim908.h ***********************/
```

sample code with gps_gsm_sim908.h

```
?
/********************* start of sample code ************************/

/*
 *  created:    2013-08-02
 *  by:     lisper (leyapin@gmail.com)
 *  Version:    0.1
 *  test gps gsm on Leonardo &XBEE R3
 *
 */

//  #ifndef DEBUG
//  #define DEBUG
//  #endif

#include "gps_gsm_sim908.h"

//
void setup () {
    gps_init ();    //init GPS pin

    #ifdef DEBUG
    Serial.begin (9600);    //serial0 connect computer
    while (!Serial);
    #endif

    Serial1.begin (9600);   //serial1 connect GPS
    while (!Serial1);

    #ifdef DEBUG
    Serial.println ("start GPS! ");
    #endif

    start_gps ();   //open GPS
}

//
void loop () {
    int stat = gps_get_gga ();  // read data from GPS, return 0 is ok
    #ifdef DEBUG
    Serial.println ("gps_get_gga () return stat:");
    Serial.println (stat);  //for test
```

```
42      #endif
43
44      if (stat == 0 || stat == 1) {
45          if (gps_gga_is_fix ()) {    //true if fix
46              //send_message ("18501683475", gps_gga_utc_s ());
47              gsm_set_numble ("18501683475"); //
48              gsm_send_message (gps_gga_utc_s ());
49              gsm_send_message (gps_gga_EW ());
50              gsm_send_message (gps_gga_NS ());
51              gsm_send_message (gps_gga_lat_s ());
52              gsm_send_message (gps_gga_long_s ());
53              gsm_end_send ();
54              while (1);
55          }
56      }
57
58      //
59      switch (stat) {
60      case 0:
61          #ifdef DEBUG
62          Serial.println ("data checksum is ok");
63          #endif
64          break;
65      case 1:
66          #ifdef DEBUG
67          Serial.println ("GPGGA ID is error!");
68          #endif
69          break;
70      case 2:
71          #ifdef DEBUG
72          Serial.println ("data is error!");
73          #endif
74          break;
75      }
76
77      #ifdef DEBUG
78      Serial.println ("$GPGGA data:");
79      gps_gga_print ();   //for test
80      #endif
81
82
83
84      /*
85      if (gps_gga_is_fix () == 0) //check if is fix
86          Serial.println ("can't fix! please go outside!");
```

```
87      else {
88          Serial.println ("ok! is fix!");
89
90          Serial.println ("gps_gga_utc_hh ()");
91          Serial.println (gps_gga_utc_hh ());
92          Serial.println ("gps_gga_utc_mm ()");
93          Serial.println (gps_gga_utc_mm ());
94          Serial.println ("gps_gga_utc_ss ()");
95          Serial.println (gps_gga_utc_ss ());
96
97          Serial.println ("gps_gga_NS ()");
98          Serial.println (gps_gga_NS (), 6);
99          Serial.println ("gps_gga_EW ()");
100         Serial.println (gps_gga_EW (), 6);
101
102         Serial.println ("gps_gga_lat ()");
103         Serial.println (gps_gga_lat (), 6);
104         Serial.println ("gps_gga_long ()");
105         Serial.println (gps_gga_long (), 6);
106         Serial.println ("gps_gga_HDOP ()");
107         Serial.println (gps_gga_HDOP (), 6);
108         Serial.println ("gps_gga_MSL ()");
109         Serial.println (gps_gga_MSL (), 6);
110         Serial.println ();
111     }
112     */
113 }
114
115
116 /********************* end of sample code ************************/
```

Trouble shooting

- Check external battery power levels, the shield needs extra power.
- Check signal range, best to be on an area with full coverage.
- Signal for GPS has best performance on a clear direct line of sight with sky, even better with less buildings around.
- GPS needs time to connect to at least 4 satellites to output data.

- AT command tester by one of our contributors. (http://www.dfrobot.com/wiki/index.php/AT_command_tester)

Version history

- GPS/GPRS/GSM Module V2.0 (SKU:TEL0051)

(http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V2.0_(SKU:TEL0051))

→ Go Shopping GPS/GPRS/GSM Shield V3.0 (Arduino Compatible)(SKU:TEL0051)
(http://www.dfrobot.com/index.php?route=product/product&filter_name=gps&product_id=673)

Categories: Product Manual | TEL Series | Shield
