

```
/*
SMS_SS v 0.921/20160712 - a-gsm 2.064 send/read/list SMS example utility
COPYRIGHT (c) 2014-2016 Dragos Iosub / R&D Software Solutions srl

*****IMPORTANT NOTICE*****
"agsm_basic_lbr.h", "agsm_SMS_lbr.ino" and "agsm_basic_lbr.h", "agsm_SMS_lbr.ino"
ARE REQUIERED IN ORDER TO RUN THIS EXAMPLE!!!!!!!!!!!!!!!!!!!!!!
Download the "a-gsm kickstart for Arduino" from the itbrainpower.net download section.
Uncompress the archive and copy the files mentined above in the folder
where is this utility, then you can compile this code.

You may want to modify "message" and "destinationNumber" variables found at lines 51/52
*****END of NOTICE*****
```

You are legally entitled to use this SOFTWARE ONLY IN CONJUNCTION WITH a-gsm DEVICES USAGE. Modifications, derivates and redistribution of this software must include unmodified this COPYRIGHT NOTICE. You can redistribute this SOFTWARE and/or modify it under the terms of this COPYRIGHT NOTICE. Any other usage may be permitted only after written notice of Dragos Iosub / R&D Software Solutions srl.

This SOFTWARE is distributed is provide "AS IS" in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Dragos Iosub, Bucharest 2016.  
<http://itbrainpower.net>

```
*/
/*
In order to make your Arduino serial communication reliable (especially for Arduino Uno)
with a-gsm shield, you must edit:
C:\Program Files\Arduino\libraries\SoftwareSerial\SoftwareSerial.h
```

```
comment the line 42
#define _SS_MAX_RX_BUFFER 64

this, will look after that like:
//#define _SS_MAX_RX_BUFFER 64

and add bellow the next line:
#define _SS_MAX_RX_BUFFER 128
```

You just increased the RX buffer size for UNO and other "snails".

```
Now you can compile and upload your code supporting highier buffered serial input data.
*/

//next 2 definition: leave them commented for standard conectivity over Software serial
//#define usejLader //un-comment this if you use micro and nano GSM 3G
adapter for ArduinoNano --Do not use it with a-gsm!!!!
//#define HARDWARESERIAL //remove comment to use Serial1 for communication on
AT MEGA 2560...DUE..

//#define atDebug //uncomment this line if you want to catch OK/ERROR/TIMEOUT modem response

//modify the next 2 line to be convenient for you
```

```

char message[]="Hi!\r\nThis message was sent from a-gsm v2.064 Arduino shield connected with
my Arduino board."; //no more than 160 chars -text SMS maximum lenght
char destinationNumber[]=" "; //usually phone number with International prefix eg. +40
for Romania - in some networks you must use domestic numbers

#define SMSLibDebug //use this definition to output the SMS send status
messages
//#define atDebug //un-comment this if you want to debug the AT exchange

/*do not change under this line! Instead, make one copy for playing with.*/
#define powerPIN 7//Arduino Digital pin used to power up / power down the modem
#define resetPIN 6//Arduino Digital pin used to reset the modem
#define statusPIN 5//Arduino Digital pin used to monitor if modem is powered

#if (ARDUINO >= 100)
#include "Arduino.h"
#if !defined(HARDWARESERIAL)
#include <SoftwareSerial.h>
#endif
#else
#include "WProgram.h"
#if !defined(HARDWARESERIAL)
#include <NewSoftSerial.h>
#endif
#endif

#if defined(HARDWARESERIAL)
#define BUFFDSIZE 1024
#else
#if defined(__AVR_ATmega1280__) /*AT MEGA ADK*/ || defined(__AVR_ATmega2560__) /*AT MEGA
2560*/ || defined(__AVR_ATmega32U4__) /*LEONARDO*/
SoftwareSerial agsmSerial(10,3); //RX==>10,TX soft==>3...read
#define BUFFDSIZE 1024
#else/*UNO*/
#define UNO_MODE //Arduino UNO
#define BUFFDSIZE 200 //240
#if defined usejLader
SoftwareSerial agsmSerial(3, 2); //RX==>3 ,TX soft==>2
#else
SoftwareSerial agsmSerial(2, 3); //RX==>2 ,TX soft==>3
#endif
#endif
#endif

//#include "agsm_basic_lbr.h"
#include "agsm_SMS_lbr.h"

#define printDebugLN(x){Serial.println(x);}

int state=0, i=0, powerState = 0;
char ch;
char buffd[256];
//char IMEI[18];

```

```

unsigned long offsetTime;
int totalChars = 0;
//int ready4SMS = 0;
//int ready4Voice = 0;

char readBuffer[200];

void setup(){
  agsmSerial.begin(9600);
  Serial.begin(57600);
  clearagsmSerial();
  clearSerial();
  delay(10);

  modemHWSSetup(); //configure Arduino IN and OUT to be used with
  modem

  Serial.flush();
  agsmSerial.flush();
  delay(1000);
  Serial.println(F("a-gsm SMS send/read/list SMS example"));
  Serial.flush();
  //You may want to modify "message" and "destinationNumber"
  if(strlen(destinationNumber)<1){
    Serial.print(F("destinationNumber not initialized. Edit SM_SS.ino and set the
    destinationNumber(line 48) with your phone number.\r\n\r\nNow the program will stop."
    ));
    delay(1000);
    exit(0);
  }
  Serial.println(F("sit back and relax until a-gsm is ready"));
  delay(100);

  powerOnModem();

  clearBUFFD();
  while(strlen(buffd)<1){
    getIMEI();
    delay(500);
  }

  ready4SMS = 0;
  ready4Voice = 0;

  Serial.println(F("a-gsm ready.. let's run the example"));
  Serial.print(F("a-gsm IMEI: ")); Serial.flush();
  Serial.println(buffd); Serial.flush();
  //setAUDIOchannel(20);
  delay(1000);
}

void loop(){
  int SMSindex = 1;
  char * pch0;
  char * pch1;
  int res;

```

```

int i;
int smslen=0;

switch(state){
  case 0://check modem status
    if(!getModemState()) restartMODEM();
    else
      state++;
    i=0;
    res= 0;
    while(res != 1){
      res = sendATcommand("","OK","ERROR",2);
      if (res != 1) {
        if(i++ >= 10) {
          printDebugLN(F("AT err...restarting"));
          restartMODEM();
        }
      }
      delay(500);
    }
    sendATcommand("+IPR=0;&w","OK","ERROR",2);
    delay(2000);
  break;

  case 1:
    clearBUFFD();
    //next some init strings...
    aGsmCMD("AT+QIMODE=0",200);
    aGsmCMD("AT+QINDI=0",200);
    aGsmCMD("AT+QIMUX=0",200);
    aGsmCMD("AT+QIDNSIP=0",200);
    offsetTime=0;
    clearBUFFD();
    state++;
  break;

  case 2:
    printDebugLN(F("try CPIN..."));
    if(!offsetTime) offsetTime = millis();
    if ((millis() - offsetTime) > 20000) restartMODEM();
    if(sendATcommand("+CPIN?", "READY")==1){
      offsetTime=0; state++;
      printDebugLN(F("READY"));
    }else{}
    clearagsmSerial(); delay(100);
    offsetTime = millis();
  break;

  case 3:
    if(!offsetTime) offsetTime = millis();
    if ((millis() - offsetTime) > 30000) restartMODEM();

    printDebugLN(F("Query GSM registration?"));
    res = registration(GSM);
    if(res==1){
      offsetTime=0; state++;
      printDebugLN(F("READY, HOME NETWORK"));
    }

```

```

}else if(res==5){
    offsetTime=0; state++;
    printDebugLN(F("READY, ROAMING"));
}else{
    Serial.print(F("."));
}
offsetTime = millis();
break;

case 4: //init SIM/MODEM
printDebugLN(F("Query State of Initialization"));
if(sendATcommand("+QINISTAT", "3")==1){
    offsetTime=0; state++;
    printDebugLN(F("READY"));
}else{Serial.print(F(".")); delay(100);}
clearagsmSerial(); delay(100);
offsetTime = millis();
break;

case 5://let's prepare modem for SMS usage
if(!offsetTime) offsetTime = millis();
if ((millis() - offsetTime) > 5000) restartMODEM();
clearBUFFD();
clearagsmSerial();

printDebugLN(F("Prepare the modem for SMS usage!"));
setupMODEMforSMSusage();
delay(10000);
offsetTime = millis();
state++;
break;

case 6://let's send one SMS to the destination receiptment
if(!offsetTime) offsetTime = millis();
if ((millis() - offsetTime) > 5000) restartMODEM();

printDebugLN(F("Let's send one SMS..."));

sendSMS(destinationNumber,message);

//if(sendSMS(destinationNumber,message,"147")==1) printDebugLN("test SMS has been send
with succes!");

//in real app you would like to use it like bellow
//if(sendSMS(destinationNumber,message) == 1) printDebugLN("SMS SUCCESS");
//else printDebugLN("SMS FAILURE");

clearBUFFD();
clearagsmSerial();

delay(10000);
offsetTime = millis();
state++;
break;

case 7://let's read SMSs from active memmory
printDebugLN(F("Let's find SMS storage used/free locations!")); delay(100);

```

```

clearBUFFD();
//listSMSstatus();
listSMS(); //this function do the job
Serial.print(F("noSMS: "));
printDebugLN(noSMS);
Serial.print(F("totSMS: "));
printDebugLN(totSMS);

delay(5000);

clearBUFFD();
clearagsmSerial();

state++;
break;

case 8://let's read one SMS
printDebugLN(F("Read the first SMS stored!")); delay(100);

//start READ SMS
SMSindex = 1; //h used as SMS index here we read only
first index... h >=1, h <= totSMS!!!!
clearBUFFD();
readSMS(SMSindex); //do the job the SMS content it is returned
in buffd

Serial.print("List the message >> ");
printDebugLN(buffd); //here is the message ....

delay(15000);

clearBUFFD();
clearagsmSerial();

state++;
break;

default:
//restartMODEM();
printDebugLN(F("That's all folks!"));
delay(10000);
//state=0;
break;
}
}

```