```
/*
SMS_SS v 0.921/20171130 - a-gsmII 2.105/b-gsmgnss 2.105 send/read/list SMS example utility
COPYRIGHT (c) 2014-2017 Dragos Iosub / R&D Software Solutions srl

*****************************IMPORTANT NOTICE*****************************************
"agsmII_basic_lbr.h", "agsmII_SMS_lbr.ino" and "agsmII_basic_lbr.h", "agsmII_SMS_lbr.ino"
or,
"bgsmgnss_basic_lbr.h", "bgsmgnss_SMS_lbr.ino" and "bgsmgnss_basic_lbr.h", "bgsmgnss_SMS_lbr.ino"
ARE REQUIERED IN ORDER TO RUN THIS EXAMPLE!!!!!!!!!!!!!!!!!!!

Download the "a-gsmII kickstart for Arduino"/"b-gsmgnss kickstart for Arduino" from here:
https://itbrainpower.net/downloads
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 55/56
*****************************END of NOTICE*******************************************

You are legaly entitled to use this SOFTWARE ONLY IN CONJUNCTION WITH
a-gsmII/b-gsmgnss 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 permited 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 2017.
http://itbrainpower.net
*/
/*
In order to make your Arduino serial communication reliable (especially for Arduino Uno) with a-gsmII/b-gsmgnss shield, you must edit:
C:\Program Files\Arduino\libraries\SoftwareSerial\SoftwareSerial.h

comment the line 42
#define _SS_MAX_RX_BUFF 64

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

and add bellow the next line:

#define _SS_MAX_RX_BUFF 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
```

```
49  //#define usejLader                     //un-comment this if you use micro and nano
    GSM 3G adapter for ArduinoNano --Do not use it with a-gsmII//b-gsmgnss!!!!
50  //#define HARDWARESERIAL                //remove comment to use Serial1 for
    communication on AT MEGA 2560...DUE..
51
52  //#define atDebug //uncomment this line if you want to catch OK/ERROR/TIMEOUT modem
    response
53
54  //modify the next 2 line to be convenient for you
55  char message[]="Hi!\r\nThis message was sent from a-gsmII/b-gsmgnss v2.105 Arduino
    shield connected with my Arduino board.";  //no more than 160 chars -text SMS
    maximum lenght
56  char destinationNumber[]="";      //usually phone number with International prefix
    eg. +40 for Romania - in some networks you must use domestic numbers
57
58  #define SMSLibDebug                   //use this definition to output the SMS send
    status messages
59  //#define atDebug                     //un-comment this if you want to debug the
    AT exchange
60
61
62  /*do not change under this line! Insteed, make one copy for playing with.*/
63  #define powerPIN        7//Arduino Digital pin used to power up / power down the modem
64  #define resetPIN        6//Arduino Digital pin used to reset the modem
65  #define statusPIN       5//Arduino Digital pin used to monitor if modem is powered
66
67  #if (ARDUINO >= 100)
68    #include "Arduino.h"
69      #if !defined(HARDWARESERIAL)
70          #include <SoftwareSerial.h>
71      #endif
72  #else
73    #include "WProgram.h"
74      #if !defined(HARDWARESERIAL)
75          #include <NewSoftSerial.h>
76      #endif
77  #endif
78
79  #if defined(HARDWARESERIAL)
80      #define BUFFDSIZE 1024
81  #else
82      #if defined(__AVR_ATmega1280__) /*AT MEGA ADK*/|| defined(__AVR_ATmega2560__)
        /*AT MEGA 2560*/|| defined(__AVR_ATmega32U4__) /*LEONARDO*/
83          SoftwareSerial agsmSerial(10,3);  //RX==>10,TX soft==>3...read
84          #define BUFFDSIZE 1024
85      #else/*UNO*/
86          #define UNO_MODE  //Arduino UNO
87          #define BUFFDSIZE 200 //240
88          #if defined usejLader
89              SoftwareSerial agsmSerial(3, 2);  //RX==>3 ,TX soft==>2
90          #else
91              SoftwareSerial agsmSerial(2, 3);  //RX==>2 ,TX soft==>3
92          #endif
93      #endif
94  #endif
95
96
97  #include "agsmII_SMS_lbr.h"
98
99  #define printDebugLN(x){Serial.println(x);}
100
101
```

```
102
103 int state=0, i=0, powerState = 0;
104 char ch;
105 char buffd[256];
106 //char IMEI[18];
107 unsigned long offsetTime;
108 int totalChars = 0;
109 //int ready4SMS = 0;
110 //int   ready4Voice = 0;
111
112 char readBuffer[200];
113
114 void setup(){
115     agsmSerial.begin(9600);
116     Serial.begin(57600);
117     clearagsmSerial();
118     clearSerial();
119     delay(10);
120
121     modemHWSetup();                          //configure Arduino IN and OUT to be
      used with modem
122
123     Serial.flush();
124     agsmSerial.flush();
125     delay(1000);
126     Serial.println(F("a-gsmII/b-gsmgnss SMS send/read/list SMS example"));
127     Serial.flush();
128     //You may want to modify "message" and "destinationNumber"
129     if(strlen(destinationNumber)<1){
130         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."));
131         delay(1000);
132         exit(0);
133     }
134     Serial.println(F("seat back and relax until your a-gsmII/b-gsmgnss is ready"));
135     delay(100);
136
137     powerOnModem();
138
139     clearBUFFD();
140     while(strlen(buffd)<1){
141         getIMEI();
142         delay(500);
143     }
144
145     ready4SMS = 0;
146     ready4Voice = 0;
147
148     Serial.println(F("a-gsmII/b-gsmgnss ready.. let's run the example"));
149     Serial.print(F("a-gsmII/b-gsmgnss IMEI: ")); Serial.flush();
150     Serial.println(buffd); Serial.flush();
151     //setAUDIOchannel(20);
152     delay(1000);
153 }
154
155
156 void loop(){
157   int SMSindex = 1;
158   char * pch0;
159   char * pch1;
160   int res;
```

```
161     int i;
162     int smslen=0;
163 
164     switch(state){
165       case 0://check modem status
166         if(!getModemState()) restartMODEM();
167         else
168           state++;
169         i=0;
170         res= 0;
171         while(res != 1){
172           res = sendATcommand("","OK","ERROR",2);
173           if (res != 1) {
174             if(i++ >= 10) {
175               printDebugLN(F("AT err...restarting"));
176               restartMODEM();
177             }
178           }
179           delay(500);
180         }
181         sendATcommand("+IPR=0;&w","OK","ERROR",2);
182         delay(2000);
183       break;
184 
185       case 1:
186         clearBUFFD();
187         //next some init strings...
188         aGsmCMD("AT+QIMODE=0",200);
189         aGsmCMD("AT+QINDI=0",200);
190         aGsmCMD("AT+QIMUX=0",200);
191         aGsmCMD("AT+QIDNSIP=0",200);
192         offsetTime=0;
193         clearBUFFD();
194         state++;
195         break;
196 
197       case 2:
198         printDebugLN(F("try CPIN..."));
199         if(!offsetTime) offsetTime = millis();
200         if ((millis() - offsetTime) > 20000) restartMODEM();
201         if(sendATcommand("+CPIN?","READY")==1){
202           offsetTime=0; state++;
203           printDebugLN(F("READY"));
204         }else{}
205         clearagsmSerial(); delay(100);
206         offsetTime = millis();
207         break;
208 
209       case 3:
210         if(!offsetTime) offsetTime = millis();
211         if ((millis() - offsetTime) > 30000) restartMODEM();
212 
213         printDebugLN(F("Query GSM registration?"));
214         res = registration(GSM);
215         if(res==1){
216           offsetTime=0; state++;
217           printDebugLN(F("READY, HOME NETWORK"));
218         }else if(res==5){
219           offsetTime=0; state++;
220           printDebugLN(F("READY, ROAMING"));
221         }else{
222           Serial.print(F("."));
```

```
223         }
224         offsetTime = millis();
225       break;
226
227       case 4: //init SIM/MODEM
228         printDebugLN(F("Query State of Initialization"));
229         if(sendATcommand("+QINISTAT","3")==1){
230           offsetTime=0; state++;
231           printDebugLN(F("READY"));
232         }else{Serial.print(F(".")); delay(100);}
233         clearagsmSerial(); delay(100);
234         offsetTime = millis();
235       break;
236
237       case 5://let's prepare modem for SMS usage
238         if(!offsetTime) offsetTime = millis();
239         if ((millis() - offsetTime) > 5000) restartMODEM();
240         clearBUFFD();
241         clearagsmSerial();
242
243         printDebugLN(F("Prepare the modem for SMS usage!"));
244         setupMODEMforSMSusage();
245         delay(10000);
246         offsetTime = millis();
247         state++;
248       break;
249
250       case 6://let's send one SMS to the destination receipment
251         if(!offsetTime) offsetTime = millis();
252         if ((millis() - offsetTime) > 5000) restartMODEM();
253
254         printDebugLN(F("Let's send one SMS..."));
255
256         sendSMS(destinationNumber,message);
257
258         //if(sendSMS(destinationNumber,message,"147")==1) printDebugLN("test SMS has
            been send with succees!");
259
260         //in real app you would like to use it like bellow
261         //if(sendSMS(destinationNumber,message) == 1) printDebugLN("SMS SUCCESS");
262         //else printDebugLN("SMS FAILURE");
263
264         clearBUFFD();
265         clearagsmSerial();
266
267         delay(10000);
268         offsetTime = millis();
269         state++;
270       break;
271
272       case 7://let's read SMSs from active memmory
273         printDebugLN(F("Let's find SMS storage used/free locations!")); delay(100);
274         clearBUFFD();
275         //listSMSstatus();
276         listSMS();                          //this function do the job
277         Serial.print(F("noSMS: "));
278         printDebugLN(noSMS);
279         Serial.print(F("totSMS: "));
280         printDebugLN(totSMS);
281
282         delay(5000);
283
```

```
284         clearBUFFD();
285         clearagsmSerial();
286 
287         state++;
288       break;
289 
290       case 8://let's read one SMS
291         printDebugLN(F("Read the first SMS stored!")); delay(100);
292 
293         //start READ SMS
294         SMSindex = 1;                               //h used as SMS index here we read
            only first index... h >=1, h <= totSMS!!!!
295         clearBUFFD();
296         readSMS(SMSindex);                          //do the job the SMS content it is
            returned in buffd
297 
298         Serial.print("List the message >> ");
299         printDebugLN(buffd);                        //here is the message ....
300 
301         delay(15000);
302 
303         clearBUFFD();
304         clearagsmSerial();
305 
306         state++;
307       break;
308 
309       default:
310         //restartMODEM();
311         printDebugLN(F("That's all folks!"));
312         delay(10000);
313         //state=0;
314       break;
315     }
316 
317   }
318 
```