```
/*
DTMF_RECEIVE v 0.921/20171130 - a-gsmII 2.105/b-gsmgnss 2.105 receive and decode DTMF example utility
COPYRIGHT (c) 2014-2017 Dragos Iosub / R&D Software Solutions srl

*****************************IMPORTANT NOTICE*****************************************
"agsmII_basic_lbr.h", "agsmII_DTMF_lbr.ino" and "agsmII_basic_lbr.h", "agsmII_DTMF_lbr.ino"
or,
"bgsmgnss_basic_lbr.h", "bgsmgnss_DTMF_lbr.ino" and "bgsmgnss_basic_lbr.h", "bgsmgnss_DTMF_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 the variables found at lines 40/41 (use the same values used in DTMF SEND)
*****************************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 the Arduino serial communication (especially for Arduino Uno) with a-gsmII/b-gsmgnss shield reliable you must
edit C:\Program Files\Arduino\libraries\SoftwareSerial\SoftwareSerial.h
comment at line 42
  #define _SS_MAX_RX_BUFF 64 ( will look like: //#define _SS_MAX_RX_BUFF 64 )
and add at next line
  #define _SS_MAX_RX_BUFF 128
You just increased increase the RX buffer size speed for UNO and other snails...
*/

//use the same values for the next two parameters, as the one used in DTMF_SEND example
int DTMFlenght=100;//DTMF lenght in miliseconds - 90-100ms best values for manual
int DTMFpause=100;//pause lenght between DTMF in miliseconds - 90-100ms best values for manual

//#define atDebug //uncomment this to debug serial communication with a-gsmII/b-gsmgnss

//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-gsmII/b-gsmgnss!!!!
//#define HARDWARESERIAL                //remove comment to use Serial1 for communication on AT MEGA 2560...DUE..

/*do not change under this line! Insteed, 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 "agsmII_DTMF_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);

```

```
106       modemHWSetup();                          //configure Arduino IN and OUT to be
          used with modem
107
108       Serial.flush();
109       agsmSerial.flush();
110       delay(1000);
111       Serial.println(F("a-gsmII/b-gsmgnss DTMF RECEIVE/DECODE example"));
112       Serial.flush();
113
114       Serial.println(F("seat back and relax until a-gsmII/b-gsmgnss is ready"));
115       delay(100);
116
117       powerOnModem();
118
119       clearBUFFD();
120       while(strlen(buffd)<1){
121           getIMEI();
122           delay(500);
123       }
124
125       ready4SMS = 0;
126       ready4Voice = 0;
127
128       Serial.println(F("a-gsmII/b-gsmgnss ready.. let's run the example"));
129       Serial.print(F("a-gsmII/b-gsmgnss IMEI: ")); Serial.flush();
130       Serial.println(buffd); Serial.flush();
131       //setAUDIOchannel(20);
132       delay(1000);
133   }
134
135
136
137   void loop(){
138     int callStatus;
139     int res=0;
140
141     switch(state){
142       case 0://check modem status
143         if(!getModemState()) restartMODEM();
144         else
145           state++;
146         i=0;
147         res= 0;
148         while(res != 1){
149           res = sendATcommand("","OK","ERROR",2);
150           if (res != 1) {
151             if(i++ >= 10) {
152               printDebugLN(F("AT err...restarting"));
153               restartMODEM();
154             }
155           }
156           delay(500);
157         }
158         sendATcommand("+IPR=0;&w","OK","ERROR",2);
159         delay(2000);
160       break;
161
162       case 1:
163         clearBUFFD();
164         //next some init strings...
165         aGsmCMD("AT+QIMODE=0",200);
166         aGsmCMD("AT+QINDI=0",200);
```

```
167         aGsmCMD("AT+QIMUX=0",200);
168         aGsmCMD("AT+QIDNSIP=0",200);
169         offsetTime=0;
170         clearBUFFD();
171         state++;
172         break;
173
174       case 2:
175         printDebugLN(F("try CPIN..."));
176         if(!offsetTime) offsetTime = millis();
177         if ((millis() - offsetTime) > 20000) restartMODEM();
178         if(sendATcommand("+CPIN?","READY")==1){
179           offsetTime=0; state++;
180           printDebugLN(F("READY"));
181         }else{}
182         clearagsmSerial(); delay(100);
183         offsetTime = millis();
184         break;
185
186       case 3:
187         if(!offsetTime) offsetTime = millis();
188         if ((millis() - offsetTime) > 30000) restartMODEM();
189
190         printDebugLN(F("Query GSM registration?"));
191         res = registration(GSM);
192         if(res==1){
193           offsetTime=0; state++;
194           printDebugLN(F("READY, HOME NETWORK"));
195         }else if(res==5){
196           offsetTime=0; state++;
197           printDebugLN(F("READY, ROAMING"));
198         }else{
199           Serial.print(F("."));
200         }
201         offsetTime = millis();
202       break;
203
204       case 4: //init SIM/MODEM
205         printDebugLN(F("Query State of Initialization"));
206         if(sendATcommand("+QINISTAT","3")==1){
207           offsetTime=0; state++;
208           printDebugLN(F("READY"));
209         }else{Serial.print(F(".")); delay(100);}
210         clearagsmSerial(); delay(100);
211         offsetTime = millis();
212       break;
213
214
215       case 5://Modem full initialised?
216         if(!offsetTime) offsetTime = millis();
217         if ((millis() - offsetTime) > 5000) restartMODEM();
218         clearBUFFD();
219         clearagsmSerial();
220
221         printDebugLN(F("It is Modem full initialised?")); delay(100);
222         setupMODEMforDTMFRusage();            //includes ATS=2 --> enable autoanswer
            at second ring
223         delay(5000);
224         offsetTime = millis();
225         state++;
226       break;
227
```

```
228
229       case 6://let's decode some DTMF
230         if(!offsetTime) offsetTime = millis();
231         if ((millis() - offsetTime) > 5000) restartMODEM();
232
233
234         printDebugLN(F("Let's decode some DTMF!"));
235
236         printDebugLN(F("Waiting for remote call! Please Start the DTMF SEND Arduino
            device!"));
237
238         callStatus =-2;//go to loop and force dial
239         while(callStatus!=0) {//remmember! Previous set ATS0=2 inside
            setupMODEMforDTMFRusage() -->auto answer at second ring detection
240             if(callStatus < 0) {//no connection, BUSY, ERROR
241               //delay(250);
242             }
243             delay(750);
244             callStatus = getcallStatus();
245         }
246         printDebugLN(F("Connected!"));
247         delay(2000);//wait a little bit
248         printDebugLN(F("Try to receive some data... Waiting for *** terminator or 55
            second!"));
249
250         memset(readBuffer,0x00,sizeof(readBuffer));// clear readBuffer for receive
            DTMF string
251
252         res = listen4DTMF(readBuffer,"***", 55);//listen for DTMF for 55 seconds,
            looking for "***" TERMINATOR
253         //printDebugLN(F("RAW Received data:"));
254         if(res<0){
255           printDebugLN(F("terminator *** NOT detected - 55 seconds passed"));
256         }else{
257           printDebugLN(F("terminator *** detected"));
258         }
259         printDebugLN(F("\r\nReceived data:"));
260         printDebugLN(readBuffer);
261
262         delay(5000);//wait a little bit
263
264         hangup();
265         printDebugLN(F("Hang up active call"));
266
267         disableAutoanswer();
268
269         disableDTMFdetection();
270         delay(5000);//wait a little bit
271         printDebugLN(F("That's all folks!"));
272
273         clearBUFFD();
274         clearagsmSerial();
275
276         delay(10000);
277         offsetTime = millis();
278         state++;
279       break;
280
281
282       default:
283         //restartMODEM();
284         delay(1000000);
```

```
285         state=0;
286       break;
287     }
288 
289   }
290 
```