

```

1  /*
2  DTMF_RECEIVE v 0.921/20171130 - a-gsmII 2.105/b-gsmgnss 2.105 receive and decode
   DTMF example utility
3  COPYRIGHT (c) 2014-2017 Dragos Iosub / R&D Software Solutions srl
4
5  *****IMPORTANT
   NOTICE*****
6  "agsmII_basic_lbr.h", "agsmII_DTMF_lbr.ino" and "agsmII_basic_lbr.h",
   "agsmII_DTMF_lbr.ino"
7  or,
8  "bgsmgnss_basic_lbr.h", "bgsmgnss_DTMF_lbr.ino" and "bgsmgnss_basic_lbr.h",
   "bgsmgnss_DTMF_lbr.ino"
9  ARE REQUIERED IN ORDER TO RUN THIS EXAMPLE!!!!!!!!!!!!!!!!!!!!!!
10
11  Download the "a-gsmII kickstart for Arduino"/"b-gsmgnss kickstart for Arduino" from
   here:
12  https://itbrainpower.net/downloads
13  Uncompress the archive and copy the files mentined above in the folder
14  where is this utility, then you can compile this code.
15
16  You may want to modify the variables found at lines 40/41 (use the same values used
   in DTMF SEND)
17  *****END of
   NOTICE*****
18
19  You are legaly entitled to use this SOFTWARE ONLY IN CONJUNCTION WITH
   a-gsmII/b-gsmgnss DEVICES USAGE. Modifications, derivates and redistribution
20  of this software must include unmodified this COPYRIGHT NOTICE. You can redistribute
   this SOFTWARE and/or modify it under the terms
21  of this COPYRIGHT NOTICE. Any other usage may be permitted only after written notice
   of Dragos Iosub / R&D Software Solutions srl.
22
23  This SOFTWARE is distributed is provide "AS IS" in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied
24  warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
25
26  Dragos Iosub, Bucharest 2017.
27  http://itbrainpower.net
28  */
29  /*
30  In order to make the Arduino serial communication (especially for Arduino Uno) with
   a-gsmII/b-gsmgnss shield reliable you must
31  edit C:\Program Files\Arduino\libraries\SoftwareSerial\SoftwareSerial.h
32  comment at line 42
33  #define _SS_MAX_RX_BUFF 64 ( will look like: //define _SS_MAX_RX_BUFF 64 )
34  and add at next line
35  #define _SS_MAX_RX_BUFF 128
36  You just increased increase the RX buffer size speed for UNO and other snails...
37  */
38
39  //use the same values for the next two parameters, as the one used in DTMF_SEND
   example
40  int DTMFlenght=100;//DTMF lenght in miliseconds - 90-100ms best values for manual
41  int DTMFpause=100;//pause lenght between DTMF in miliseconds - 90-100ms best values
   for manual
42
43  //define atDebug //uncomment this to debug serial communication with
   a-gsmII/b-gsmgnss
44
45  //next 2 definition: leave them commented for standard conectivity over Software
   serial
46  //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!!!!
47 //define HARDWARESERIAL //remove comment to use Serial1 for
communication on AT MEGA 2560...DUE..
48
49 /*do not change under this line! Instead, make one copy for playing with.*/
50 #define powerPIN 7//Arduino Digital pin used to power up / power down the modem
51 #define resetPIN 6//Arduino Digital pin used to reset the modem
52 #define statusPIN 5//Arduino Digital pin used to monitor if modem is powered
53
54 #if (ARDUINO >= 100)
55 #include "Arduino.h"
56 #if !defined(HARDWARESERIAL)
57 #include <SoftwareSerial.h>
58 #endif
59 #else
60 #include "WProgram.h"
61 #if !defined(HARDWARESERIAL)
62 #include <NewSoftSerial.h>
63 #endif
64 #endif
65
66 #if defined(HARDWARESERIAL)
67 #define BUFFDSIZE 1024
68 #else
69 #if defined(__AVR_ATmega1280__) /*AT MEGA ADK*/|| defined(__AVR_ATmega2560__)
/*AT MEGA 2560*/|| defined(__AVR_ATmega32U4__) /*LEONARDO*/
70 SoftwareSerial agsmSerial(10,3); //RX==>10,TX soft==>3...read
71 #define BUFFDSIZE 1024
72 #else/*UNO*/
73 #define UNO_MODE //Arduino UNO
74 #define BUFFDSIZE 200 //240
75 #if defined usejLader
76 SoftwareSerial agsmSerial(3, 2); //RX==>3 ,TX soft==>2
77 #else
78 SoftwareSerial agsmSerial(2, 3); //RX==>2 ,TX soft==>3
79 #endif
80 #endif
81 #endif
82
83
84 #include "agsmII_DTMF_lbr.h"
85
86 #define printDebugLN(x){Serial.println(x);}
87
88 int state=0, i=0, powerState = 0;
89 char ch;
90 char buffd[256];
91 //char IMEI[18];
92 unsigned long offsetTime;
93 int totalChars = 0;
94 int ready4SMS = 0;
95 int ready4Voice = 0;
96 char readBuffer[200];
97
98
99 void setup(){
100 agsmSerial.begin(9600);
101 Serial.begin(57600);
102 clearagsmSerial();
103 clearSerial();
104 delay(10);
105

```

```

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     setupMODEMforDTMFUsage(); //includes AT+Q=2 --> enable autoanswer
223     at second ring
224     delay(5000);
225     offsetTime = millis();
226     state++;
227     break;

```

```

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) { //remember! Previous set ATSO=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
```