```
/*
DTMF_RECEIVE v 0.921/20160707 - a-gsm 2.064 receive and decode DTMF example utility
COPYRIGHT (c) 2014-2016 Dragos Iosub / R&D Software Solutions srl

*****************************IMPORTANT NOTICE*****************************************
"agsm_basic_lbr.h", "agsm_DTMF_lbr.ino" and "agsm_basic_lbr.h", "agsm_DTMF_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 the variables found at lines 36/37 (use the same as in DTMF SEND)
*****************************END of NOTICE*****************************************

You are legaly 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 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 2016.
http://itbrainpower.net
*/
/*
In order to make the Arduino serial communication (especially for Arduino Uno) with a-gsm
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 next two parameters as 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-gsm

//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..

/*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 "agsm_basic_lbr.h"
#include "agsm_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);

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

    Serial.flush();
```

```
    agsmSerial.flush();
    delay(1000);
    Serial.println(F("a-gsm DTMF RECEIVE/DECODE example"));
    Serial.flush();

    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 callStatus;
  int res=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://Modem full initialised?
    if(!offsetTime) offsetTime = millis();
    if ((millis() - offsetTime) > 5000) restartMODEM();
    clearBUFFD();
    clearagsmSerial();

    printDebugLN(F("It is Modem full initialised?")); delay(100);
    setupMODEMforDTMFRusage();            //includes ATS=2 --> enable autoanswer at second
```

```
    ring
    delay(5000);
    offsetTime = millis();
    state++;
  break;


  case 6://let's decode some DTMF
    if(!offsetTime) offsetTime = millis();
    if ((millis() - offsetTime) > 5000) restartMODEM();


    printDebugLN(F("Let's decode some DTMF!"));

    printDebugLN(F("Waiting for remote call! Please Start the DTMF SEND Arduino device!"));

    callStatus =-2;//go to loop and force dial
    while(callStatus!=0) {//remmember! Previous set ATS0=2 inside
    setupMODEMforDTMFRusage() -->auto answer at second ring detection
        if(callStatus < 0) {//no connection, BUSY, ERROR
          //delay(250);
        }
        delay(750);
        callStatus = getcallStatus();
    }
    printDebugLN(F("Connected!"));
    delay(2000);//wait a little bit
    printDebugLN(F("Try to receive some data... Waiting for *** terminator or 55 second!"));

    memset(readBuffer,0x00,sizeof(readBuffer));// clear readBuffer for receive DTMF string

    res = listen4DTMF(readBuffer,"***", 55);//listen for DTMF for 55 seconds, looking for
    "***" TERMINATOR
    //printDebugLN(F("RAW Received data:"));
    if(res<0){
      printDebugLN(F("terminator *** NOT detected - 55 seconds passed"));
    }else{
      printDebugLN(F("terminator *** detected"));
    }
    printDebugLN(F("\r\nReceived data:"));
    printDebugLN(readBuffer);

    delay(5000);//wait a little bit

    hangup();
    printDebugLN(F("Hang up active call"));

    disableAutoanswer();

    disableDTMFdetection();
    delay(5000);//wait a little bit
    printDebugLN(F("That's all folks!"));

    clearBUFFD();
    clearagsmSerial();

    delay(10000);
```

```
      offsetTime = millis();
      state++;
    break;


    default:
      //restartMODEM();
      delay(1000000);
      state=0;
    break;
  }

}
```