```
/*
SD_SS.ino  v 0.921/20160609 - a-gsm 2.064  microSD read/write/delete example utility
COPYRIGHT (c) 2014-2016 Dragos Iosub / R&D Software Solutions srl

******************************IMPORTANT NOTICE*******************************************
"agsm_basic_lbr.h", "agsm_FS_lbr.ino" and "agsm_basic_lbr.h", "agsm_FS_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 message variable found at line 45
*******************************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 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_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
//#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                       //un-comment this if you want to debug the AT exchange

/*next one - change general read buffer(buffd) size 1024 uncommented, 256 commented */
```

```
char message[]="Hello world!";                //test.tst content - no more than 1024(128 in
UNO_MODE) chars! see buffd size, bellow


/*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_FS_lbr.h"

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

#if defined(UNO_MODE)
  char buffd[256];
#else
  char buffd[512];
#endif

int state=0, i=0, powerState = 0;
char ch;
//char IMEI[18];
unsigned long offsetTime;
int totalChars = 0;
int ready4SMS = 0;
int ready4Voice = 0;
//char buffd[BUFFDSIZE];
//int noSMS=0, totSMS=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 SD read/write/delete file 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 res;

  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://list files on SD
    //case 7:
      if(!offsetTime) offsetTime = millis();
      if ((millis() - offsetTime) > 5000) restartMODEM();

      printDebugLN(F("list all text files")); delay(100);
      listModemFile("*.txt");
      printDebugLN(buffd);
      delay(1000);

      printDebugLN(F("The file system it is NOT case SENSITIVE!")); delay(100);
      delay(10000);

      state++;
    break;

    case 6:
      printDebugLN(F("Try to read inexistent test.TXT!")); delay(100);
      clearBUFFD();
      memset(readBuffer,0x00, sizeof(readBuffer));
      if(readModemFile("test.TXT", readBuffer)>0) {
            printDebugLN(readBuffer);
      }else printDebugLN(F("Error reading file"));
      memset(readBuffer,0x00, sizeof(readBuffer));
      clearBUFFD();

      delay(10000);

      printDebugLN(F("Try to write test.txt with defined message")); delay(100);
      if(writeModemFile("test.txt", message)>0) {
        printDebugLN(F("Success writing file!"));
      }else printDebugLN(F("Error writing file!"));

      delay(10000);

      clearBUFFD();
      printDebugLN("now list text files"); delay(100);
      listModemFile("*.txt");
      printDebugLN(buffd);

      delay(10000);

      clearBUFFD();
      printDebugLN(F("Now read test.txt!")); delay(100);
      memset(readBuffer,0x00, sizeof(readBuffer));
      readModemFile("test.TXT", readBuffer);
      printDebugLN(readBuffer); delay(10000);

      clearBUFFD();
      printDebugLN(("Now delete test.txt!")); delay(100);
      deleteModemFile("test.txt");
      delay(5000);

      printDebugLN(F("now list text files")); delay(100);
      listModemFile("*.txt");
      printDebugLN(buffd); delay(10000);

      delay(2000);

      printDebugLN(F("That's all folks!"));
      state++;
    break;

    case 7:
    default:
      //restartMODEM();
      delay(1000000);
      state=0;
    break;
  }

}
```