

```

#include "CArdCom.h"

//-----
CArdCom::CArdCom()
{
}
//-----
CArdCom::~CArdCom()
{
    Close();          //close port
}
//-----
bool CArdCom::Init(string comport,int baud,int bits,int par,int stop)
{
    //settings for communication and others
    Port=comport;
    Baud=baud;
    Bits=bits;
    Par=par;
    StopBit=stop;

    int requiredSize = mbstowcs( NULL, Port.c_str(), 0);
    wchar_t      *port=new wchar_t[requiredSize+1];

    #ifdef UNICODE
        mbstowcs(port,Port.c_str(),1+strlen(Port.c_str()));
    #else
        strcpy(port,Port.c_str());
    #endif

    Open(port);
    delete []port;
    Setup(EBaudrate(Baud),EDataBits(Bits),EParity(Par),EStopBits(StopBit)
);

    Sleep(2000); //time for arduino to wake up

    if(IsOpen())
    {
        return true;
    }
    else
    {
        return false;
    }
}
//-----
//send message to arduino with all symbol around and check sum (ToDo)
bool CArdCom::SendMessageArd(string message)
{
    if(!IsOpen())
    {
        return false;
    }
    if(message.length()<3)
    {
        return false;
    }
    Buffer1=START+message+END+"0"+STOP;
    //purge port to be ready for incomming data

```

```

Purge();
//write data to port
if(Write(Buffer1.c_str(),Buffer1.length())!=ERROR_SUCCESS)
{
    return false;
}

ReadMessage(Buffer1,7); //only characters for ACK
string ideof, val;
if(!ProcMessages(Buffer1, ideof, val)) return false;
if(ideof==string(ACK))
{
    return true; //ACK OK
}
else
{
    return false; //NACK or no response
}

}
//-----
bool CCardCom::ReadMessage(string &message, unsigned int count)
{
    if(count>20) count=20;
    if(!IsOpen())
    {
        return false;
    }
    SetupReadTimeouts(EReadTimeout(0));
    char buffer[20];
    memset(buffer, 0x00, 20*sizeof(char));
    Sleep(50); //timeout for controller to send response
    Read(buffer, count);
    message=buffer;
    return true;
}
//-----
bool CCardCom::ProcMessages(string in, string &ideof, string &data)
{
    int start=0, end=0;
    bool stop=false;
    char buffer[5];
    string pom;
    for(int i=0; i<in.length(); i++)
    {
        sprintf(buffer, "%c", in.c_str()[i]);
        pom=buffer;
        if(pom==string(START))
        {
            start=i;
        }
        if(pom==string(END))
        {
            end=i;
        }
        if(pom==string(STOP))
        {
            stop=true;
            break; //waiting only for one message
        }
    }
}

```

```

    }
    ndef=in.substr(start+1,start+3);
    data=in.substr(start+4,end-(start+4));

    return stop;
}
//-----
bool CArdCom::SetPin(unsigned int port)
{
    char temp[255];
    _itoa(port,temp,10);
    Buffer2=string(SET)+temp;
    return SendMessageArd(Buffer2);
}
//-----
bool CArdCom::ResetPin(unsigned int port)
{
    char temp[255];
    _itoa(port,temp,10);
    Buffer2=string(RES)+temp;
    return SendMessageArd(Buffer2);
}

```