Programme dans Arduino

```
/*ARDUINO - comunication
 *Start - 25.6.2009
*/

#define START 0x01
#define END 0x02
#define STOP 0x03

#define ADR //adress of the first LED pin

class Comunication
{
    private:
      int numberBytes;        //number of all received bytes
      int received;           //flag for receiving message
      int processed;          //flag for processing message
      char message[200];      // message
      char idef[4];            //identificator
      char body[20];
      void ReceiveMessage();  //read message from serial port
      void ProcessMessage();  //process message
      void GetIdef();        //get IDEF of message
      int GetBody();        //get body of message
      void Set(int);        //set target pin to 1
      void Reset(int);      //reset target pin to 0
      void DoLight();          //
      void SetSwitch();

    public:
      Comunication();       //costructor
      void Init();          //inicialization
      void Send(char*);      //send a char to serial port
      void Send(int);    //send int
      void Send(float);    //send float
      void HandlMessage();   //handl message from PC
      ~Comunication();       //destructor


};


int StrToInt(char *str, int count)
{
  int i = 0;
  int number = 0;


  for(i=count; i>=0; i--)
  {
    number += (str[i] - 48) * round(pow(10, count - i));
    //strlen(str)
  }
```

```
  return number;
}
Comunication::~Comunication()
{

}
/*-------------------------------------------------------------*/
Comunication::Comunication()
{
    numberBytes = 0;
    received = 0;
    processed = 0;
 }

/*-------------------------------------------------------------*/
void Comunication::Init()
{

}
/*-------------------------------------------------------------*/
void Comunication::Set(int pin)
  {
    digitalWrite(pin, 1);
  }

  /*-------------------------------------------------------------*/
void Comunication::Reset(int pin)
  {
    digitalWrite(pin, 0);
  }

 /*-------------------------------------------------------------*/

int Comunication:: GetBody()
{
  int i = 0;

  while((message[i+3] != END)&&(i<20))
  {
    body[i] = message[i+3];
    i++;
  }

  //body[i] = 0x00;

  return StrToInt(body, i-1);
}

/*-------------------------------------------------------------*/

void Comunication::Send(char *info)
/*form of message:
 *  START|IDEF|message|END |SUM|*****STOP
 *  0x01 |XXX |body..|0x02|X |*****0x03
 */
  {
    Serial.print(START, BYTE);     //Start
```

```
    Serial.print(info);       //ACK, NCK, message
    Serial.print(END, BYTE);       //End
    Serial.print(3, DEC);       //sum
    Serial.print(STOP, BYTE);       //STOP
  }

/*-------------------------------------------------------------*/
void Comunication::Send(int info)
 {
    Serial.print(START, BYTE);     //Start
    Serial.print("GET");
    Serial.print(info, DEC);       //value
    Serial.print(END, BYTE);       //End
    Serial.print(3, DEC);       //sum
    Serial.print(STOP, BYTE);       //STOP
  }

/*-------------------------------------------------------------*/
  void Comunication::Send(float info)
 {
    Serial.print(START, BYTE);     //Start
    Serial.print("MES");
    Serial.print(info, DEC);       //value
    Serial.print(END, BYTE);       //End
    Serial.print(3, DEC);       //sum
    Serial.print(STOP, BYTE);       //STOP
  }

/*-------------------------------------------------------------*/
void Comunication::GetIdef()
{
  int i = 0;
  for(i=0; i<3; i++)idef[i] = message[i];
  idef[3] = 0x00;
}

/*-------------------------------------------------------------*/
void Comunication:: DoLight()
{
  int i = 0;
  char LED[100];

  while((message[i+3] != END))
  {
    LED[i] = message[i+3];
    i++;
  }
  //to do add code for writting value to PWM pin, the values are stored
in LED array
  //values 0x00, 0x01, 0x02, 0x03 are reservad for the comunication and
cannot by used

}

/*-------------------------------------------------------------*/
void Comunication:: SetSwitch()
{
```

```
  int Byte = 0, Bit = 0, i = 0, pom2;
  char LED[100];

  for (Byte=0; Byte<7; Byte++)       //get state of digital I/O from
message
  {
    for (Bit=0; Bit<7; Bit++)
    {
      if(((message[Byte + 3] - 48) & (1 << Bit)) != 0)
digitalWrite(Byte * 8 + Bit, HIGH);
      else digitalWrite(Byte * 8 + Bit, LOW);
      //here you can do whatever you wont to do, each bit of 7 bytes
can represent a pin
    }
  }
//this is just example of use

}

/*-------------------------------------------------------------*/
void Comunication::ReceiveMessage()
{
  char Cpom;

  do
  {
      Cpom = Serial.read();
      switch (Cpom)
      {
        case START:{
                    received = 0;
                    numberBytes = 0;
                    }break;
        case END:{
                   message[numberBytes] = END;
                   numberBytes++;
                  }break;
        case STOP:{
                   message[numberBytes] = STOP;
                   received = 1;
                  }break;
        default:{
                  message[numberBytes] = Cpom;
                  numberBytes++;
                }
      }//switch
  }while ((Serial.available() > 0));
}

/*-------------------------------------------------------------*/
void Comunication::HandlMessage()
{
    if(Serial.available() > 0)  ReceiveMessage();
    if(received == 1) ProcessMessage();
}

/*-------------------------------------------------------------*/
```

```
void Comunication::ProcessMessage()
{
  received = 0;                                  //clear flag

  GetIdef();
   if(strcmp("CON",idef) == 0)
  {
        processed = 1;
        Send("ACK");
          //set flag
   }
  if(strcmp("LIG",idef) == 0)
  {
        DoLight();
        processed = 1;
        Send("ACK");
           //set flag
   }
   if(strcmp("SWI",idef) == 0)
  {
        SetSwitch();
        processed = 1;
        Send("ACK");
           //set flag
   }
   if(strcmp("SET",idef) == 0)
  {
        Set(GetBody());
        processed = 1;
        Send("ACK");
           //set flag
   }
  if(strcmp("RES",idef) == 0)
  {
        Reset(GetBody());
        processed = 1;
        Send("ACK");
            //set flag
  }
  if(!processed)
    Send("NCK");                        //send error

  processed = 0;                        //clear flag
}
/*-------------------------------------------------------------*/
/*-------------------------------------------------------------*/
/*-------------------------------------------------------------*/

Comunication Com;

/*-------------------------------------------------------------*/
void setup()
{
    Com.Init();
}

/*-------------------------------------------------------------*/
```

```
void loop()
{
  Com.HandlMessage();
}
```