```
/*
Riadenie skenera
verzia 27.1.2022 (vratane kurenia, vylepsene nastavovanie DAC)
Frantisek Kundracik

Krokove motory

CONNECTIONS - karusel 0:
driver + -> +5V  (D zbernica)
driver - -> GND (D zbernica)
driver IN1 -> PIN 8
driver IN2 -> PIN 9
driver IN3 -> PIN 10
driver IN4 -> PIN 11

CONNECTIONS - karusel 1:
driver + -> +5V (D zbernica)
driver - -> GND (D zbernica)
driver IN1 -> PIN 4
driver IN2 -> PIN 5
driver IN3 -> PIN 6
driver IN4 -> PIN 7

Senzor filtrov
napajanie 5V -> PIN 12
senzor GND -> GND  (D zbernica)
brana 0 (žltý) -> A0  (D zbernica)
brana 1 (oranžový) -> A1  (D zbernica)

Displej
VCC -> +5V (D zbernica)
GND -> GND (D zbernica)
SDA -> A4/SDA (D zbernica)
SCL -> A5/SCL (D zbernica)
LED -> PIN 3 (Arduino)

12-bit D/A prevodnik MCP4725
GND -> GND (A zbernica)
VCC -> +5V (A zbernica)
SDA -> A4/SDA (D zbernica)
SCL -> A5/SDA (D zbernica)
OUT -> CtrlVoltageIN (A zbernica)

16-bit A/D prevodnik ADS1115
VCC -> +5V (A zbernica)
GND -> GND (A zbernica)
SDA -> A4/SDA (D zbernica)
SCL -> A5/SCL (D zbernica)
ADDR -> NC (Adresa 0x48)
A0 -> PMT (koax. kabel signal)
A1 -> PMT (koax. kabel GND)
A2 -> ctrlVoltageOUT (A zbernica)

Teplomer Dallas
VCC -> +5V (digitalna zbernica)
GND -> GND (digitalna zbernica)
DATA -> PIN 2

kurenie signal -> PIN 13
*/

#include <Stepper.h>
```

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_ADS1015.h>
#include <Adafruit_MCP4725.h>
#include <DallasTemperature.h>

const int IN1_0 =  11;
const int IN2_0 =  10;
const int IN3_0 =  9;
const int IN4_0 =  8;

const int IN1_1 =  7;
const int IN2_1 =  6;
const int IN3_1 =  5;
const int IN4_1 =  4;

const int stepsPerRevolution = 4096;  // 4096 = 64 krokov/otacku,  1:64 prevodovka
- reálne je ale 2048!!!!

//POZOR! čísla nie sú v poradí, 10 a 9 sú prehodené, hoci piny sú zapojené v
poradí!
Stepper myStepper[2]={
                      Stepper(stepsPerRevolution, IN4_0, IN2_0, IN3_0, IN1_0),
                      Stepper(stepsPerRevolution, IN4_1, IN2_1, IN3_1, IN1_1)
                      };

//----- premenne pre ovladanie filtrov -----------------
const int brana =  12;  //PIN12 - zapinanie/vypinanie svetla
const int senzor[2] = {A0,A1};  //Piny na zaznamenanie signalu
int filter[2]={0,0};  //0-11 - cislo aktualneho filtra
bool filterOK[2]={false,false};  //ci je filter synchronizovany
const int senzor_level=300; //prah pre otvorenu branu
int s=-1; //smer otacania karuselu
int fastspeed=6;  //rychle otacanie: motor stiha maximalne po 8
int slowspeed=3; //pomale otacanie: 3 - optimum, 2-uz prilis pomaly (pohyb karuselu
je uz trhany a hlucny)
//cca 170 krokov na posun o jeden filter, siroke okno = 47-49 krokov
int jump1 = 35; //posun zhruba do stredu dlheho okna(stred dlheho okna je cca 24
krokov)
int jump2 =60; //posun cez rozhranie (dost velky na prekonanie vole v prevode)
int jump3 = 150-jump1-jump2; //vacsi prvy posun (dost velky na prekonanie vole),
dalsi filter je na 170
bool je_posunuty=false; //ze karusel je pootoceny po hladani pozicie 0
//-------------------------------------------------------

//----- premenne pre displej ----------------------------
LiquidCrystal_I2C lcd(0x27,16,2);  // nastavenie adresy (0x3F) u novsich modelov
alebo (0x27) u starsich
int display_brightness_pin=3;
int brightness_level=32;
//-------------------------------------------------------

//------- premenne pre kurenie --------------------------
const int kurenie=13;
float minteplota=5.0; //minimalna akceptovana teplota
char deg=' '; //symbol pre stupen (zavisi od teploty)
//-------------------------------------------------------

//----------- premenne pre 16-bit A/D prevodnik ----------
Adafruit_ADS1115 ads(0x48); //default adresa prevodnika ADS1115
float signalVoltage; //aktualne namerana hodnota
int repetitions=100;  //pocet spriemerovanych hodnot
int CTRL=0; //meranie control voltage
int SIGNAL=1; //meranie vystupu fotonasobica
//--------------------------------------------------------

//----------- premenne pre 12-bit D/A prevodnik ----------
Adafruit_MCP4725 dac;   //adresa I2C=A0
float dac_scale=4096/1.1861; //zavisi od delica, na ktory je vystup pripojeny
(1.1861 pre prototyp s 3k3 a 1k0), sluzi iba na prvotny odhad cisla pre D/A, nie je
kriticke
                             //je to vlastne hodnota napatia, ktore sa nastavi pri
poziadavke nastavit viac, nez je mozne (napr. 1.2V) = hodnota pre level=4095
float ctrlVoltage; //aktualne namerana hodnota
float ctrlRequiredVoltage=0.4; //zelana hodnota. 0.4 - velmi male zosilnenie ako
default, pod 0.5 uz prudko klesa k nule
uint32_t level; //ciselna uroven D/A prevodnika v bitoch
float ofset=0.0003; //pridanie 0.0003V zabranuje preblikavaniu typu 0.800 a 0.799
na displeji
//--------------------------------------------------------

//----------- premenne pre teplomer ---------------------
OneWire oneWire(2);  //pripojeny na PIN 2
DallasTemperature teplomer(&oneWire);
float teplota=0; //aktualne namerana teplota
//-------------------------------------------------------

char command[9]=""; //prikaz

//------------------------ setup() ---------------------------------------


void setup() {
  pinMode(brana, OUTPUT);
  digitalWrite(brana,LOW); //zhasnut senzor

  // initialize the serial port:
  Serial.begin(115200);

  // set the speed:
  myStepper[0].setSpeed(fastspeed);  //motor stiha maximalne asi po  8
  myStepper[1].setSpeed(fastspeed);  //motor stiha maximalne asi po  8

  //nastavíme adresu a typ displeje
  lcd.init();                      // initializace lcd
  setBrightness(brightness_level);
  lcd.backlight();

  pinMode(kurenie,OUTPUT);   //pin kurenia ako output
  digitalWrite(kurenie,LOW); //vypnut kurenie

  ads.begin();
  ads.setGain(GAIN_ONE); //GAIN_TWOTHIRDS, GAIN_ONE, GAIN_TWO, GAIN_FOUR,
GAIN_EIGHT, GAIN_SIXTEEN
                           // GAIN ONE single ended = 0 ... 4.096V s rozlisenim
0.125 mV,
                           // differential -4.096V ... + 4.096V, rozlisenie 0.125
mV
  dac.begin(0x60);

  char buf[18]="A/D-D/A check...";
  displayText(buf);
  unsigned long millis1=millis();

  level = (ctrlRequiredVoltage+ofset)*dac_scale; //odhad pozadovanej urovne D/A pre
default hodnotu napätia
```

```
  setCtrlVoltage(ctrlRequiredVoltage);
  ctrlVoltage=getVoltage(CTRL);


  //test ci I2C funguje (ci je zapnuty spinac napatia k prevodnikom)
  unsigned long millis2=millis();
  if(abs(millis2-millis1)>1000)
  {
    strcpy(buf,"Turn switch on!");
    displayText(buf);
    for(int i=0;i<10;i++)  //zabliikat pri plnom jase
    {
      analogWrite(display_brightness_pin, 255);
      delay(500);
      analogWrite(display_brightness_pin, 0);
      delay(500);
    }
  }
  analogWrite(display_brightness_pin, brightness_level);
  teplomer.begin();
}

//----------------- loop() --------------------------------------------------------
--------------
void loop() {
  char cmd[16]="";
  char param[16]="";

  setCtrlVoltage(ctrlRequiredVoltage);
  ctrlVoltage=getVoltage(CTRL); //refresh control voltage
  int repetitions_bak=repetitions;
  repetitions=10; //fast measuring in the loop
  signalVoltage=getVoltage(SIGNAL);
  repetitions=repetitions_bak; //original averaging
  teplomer.requestTemperatures();
  teplota = teplomer.getTempCByIndex(0);
  if(teplota<=minteplota)
   {
      digitalWrite(kurenie,HIGH);
      deg=42; //hviezdicka
   }
   else
   {
      digitalWrite(kurenie,LOW);
      deg=223; //stupen
   }
  refreshDisplay();

  if(Serial.available()>=8)
  {
    Serial.readBytes(command, 8);
    displayText(command);
    while(Serial.available()>0) Serial.read(); //ignore endl characters
    strcpy(cmd,command);
    strcpy(param,command+3); //extrahovat parametre
    cmd[3]=0; //extrahovat povel

    if(strcmp(cmd,"IDN")==0) //identifikacia IDNXXXXX, XXXXX-dummy
    {
      Serial.println("SKY-SCAN");
      return;
    }
    if(strcmp(cmd,"SFL")==0) //set filter command SFLabbXX:
a=karusel(0,1),bb=filter(00-11), XX=dummy
    {
      param[6]=0; //remove dummy
      int number=atoi(param+1); //extract filter position
      if(param[0]=='0')
{setFilter(0,number);sprintf(command,"FLT0%02dXX",filter[0]);Serial.println(command
);}
      if(param[0]=='1')
{setFilter(1,number);sprintf(command,"FLT1%02dXX",filter[1]);Serial.println(command
);}
      return;
    }
    if(strcmp(cmd,"GFL")==0) //get filter command GFLaXXXX: a=karusel(0,1),
XXXX=dummy
    {
      if(param[0]=='0')
{sprintf(command,"FLT0%02dXX",filter[0]);Serial.println(command);}
      if(param[0]=='1')
{sprintf(command,"FLT1%02dXX",filter[1]);Serial.println(command);}
      return;
    }

    if(strcmp(cmd,"RFL")==0) //restart filter command RFLaXXXX: a=karusel(0,1),
XXXX=dummy
    {
      if(param[0]=='0') {
        filterOK[0]=resetFilter(0);
        if(filterOK[0]) {sprintf(command,"FLT0ISOK");Serial.println(command);}
        else {sprintf(command,"FLT0LOST");Serial.println(command);}
        }
      if(param[0]=='1') {
        filterOK[1]=resetFilter(1);
        if(filterOK[1]) {sprintf(command,"FLT1ISOK");Serial.println(command);}
        else {sprintf(command,"FLT1LOST");Serial.println(command);}
       }
       return;
    }
    if(strcmp(cmd,"SCV")==0) //set control voltage command SCVabbbbb: SCV08500 =
0.8500V
    {
      param[8]=0; //end of text
      int number=atoi(param+1); //extract filter position
      if(param[0]=='0') {ctrlRequiredVoltage=(float)number/10000;}
      if(param[0]=='1') {ctrlRequiredVoltage=1.0+(float)number/10000;}
      level = (ctrlRequiredVoltage+ofset)*dac_scale; //odhad požadovanej úrovne D/A
      if(level<0) level=0;
      if(level>4095) level=4095;
      setCtrlVoltage(ctrlRequiredVoltage);
      ctrlVoltage=getVoltage(CTRL);
      float val=ctrlVoltage;
      int val_int = (int) val;   // compute the integer part of the float
      int val_fra = (int) ((val - (float)val_int) * 10000);   // compute 4 decimal
places (and convert it to int)
      snprintf (command, 16, "CVT%d%04d", val_int, val_fra); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"GCV")==0) //get control voltage command GCVXXXXX: XXXXX=dummy
    {
      ctrlVoltage=getVoltage(CTRL);
      float val=ctrlVoltage;
      int val_int = (int) val;   // compute the integer part of the float
      int val_fra = (int) ((val - (float)val_int) * 10000);   // compute 4 decimal
places (and convert it to int)
      snprintf (command, 16, "CVT%d%04d", val_int, val_fra); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"GSV")==0) //get signal voltage command GSVXXXXX: XXXXX=dummy
    {
      setCtrlVoltage(ctrlRequiredVoltage);
      int stavkurenia=digitalRead(kurenie); //zistit stav kurenia
      digitalWrite(kurenie,LOW);      //vypnut kurenie (svetlo z LED, rusenie)
      lcd.noBacklight();    //vypnut display (svetlo)
      signalVoltage=getVoltage(SIGNAL);
      lcd.backlight(); //zapnut displej
      digitalWrite(kurenie,stavkurenia); //obnovit stav kurenia
      float val=signalVoltage;
      int val_int = (int) val;   // compute the integer part of the float
      int val_fra = (int) ((val - (float)val_int) * 10000);   // compute 4 decimal
places (and convert it to int)
      snprintf (command, 16, "SVT%d%04d", val_int, val_fra); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"SNM")==0) //set number of measurements to be averaged command
SNMnnnnn: nnnnn=integer: 00100 = 100 samples
    {
      repetitions=atoi(param);
      snprintf (command, 16, "NMA%05d", repetitions); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"GNM")==0) //get number of measurements to be averaged command
SNMXXXXX: XXXXX´dummy
    {
      snprintf (command, 16, "NMA%05d", repetitions); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"SDB")==0) //set display brightness command SDB00nnn: nnn = 000
... 255
    {
      brightness_level=atoi(param);
      if(brightness_level>255) brightness_level=255;
      if(brightness_level<0) brightness_level=0;
      analogWrite(display_brightness_pin, brightness_level);
      snprintf (command, 16, "DBR%05d", brightness_level); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"STP")==0) //set minimum temperature command STPabbbb: STP+0125
(v desatinach stupna) = +12.5oC
    {
      param[8]=0; //end of text
      int number=atoi(param);
      minteplota=(float)number/10;
      char znamienko;
      if(minteplota<0) znamienko='-';
      else znamienko='+';
      float val=minteplota;
      if(znamienko=='-') val=-val; //convert to positive value
      int val_int = (int) val;   // compute the integer part of the float
      int val_fra = (int) ((val - (float)val_int) * 10);   // compute 1 decimal
place (and convert it to int)
```

```
      snprintf (command, 16, "TPV%c%03d%01d", znamienko,val_int, val_fra); //
      Serial.println(command);
      return;
    }
    if(strcmp(cmd,"GTP")==0) //get current temperature command GTPXXXXX:
XXXXX=dummy
    {
      char znamienko;
      float val=teplota;
      if(teplota<0) znamienko='-';
      else znamienko='+';
      if(znamienko=='-') val=-val; //convert to positive value
      int val_int = (int) val;   // compute the integer part of the float
      int val_fra = (int) ((val - (float)val_int) * 10);   // compute 1 decimal
place (and convert it to int)
      snprintf (command, 16, "TPV%c%03d%01d", znamienko,val_int, val_fra); //
      Serial.println(command);
      return;
    }
  snprintf(command,16,"UNKNOWN!");
  Serial.println(command);
  }
}

void setFilter(int index, int n)
{

  while(filter[index]!=n)
  {
    int delta=n-filter[index]; //vzdialenost od zelaneho filtra
    if(delta<0) delta +=12;    //korekcia na prechod cez 11
    if(delta>1) myStepper[index].setSpeed(fastspeed);
    else myStepper[index].setSpeed(slowspeed);
    nextFilter(index);
  }

  stop_motors();
  refreshDisplay();
}

void nextFilter(int index)
{
  int level;
  digitalWrite(brana, HIGH);  //rozsvietit senzor
  int jump=jump3;
  if(je_posunuty) jump-=jump1;
  for(int i=0;i<abs(jump/s);i++)  //posunut sa cez hranu o jump1
  {
    myStepper[index].step(s);
  }
  while ((level=analogRead(senzor[index]))>senzor_level) //cakat kym sa senzor
nezakryje (pre istotu ak sme sa neposunuli dost)
  {
    myStepper[index].step(s);
  }
  for(int i=0;i<abs(jump2/s);i++) //trochu pohnut cez rozhranie
  {
    myStepper[index].step(s);
  }
  while ((level=analogRead(senzor[index]))<senzor_level) //cakat kym sa senzor
neodkryje - filter je na pozicii
  {
    myStepper[index].step(s);
  }

  je_posunuty=false;
  filter[index]++;
  if(filter[index]>11) filter[index]=0;

  digitalWrite(brana, LOW);  //zhasnut senzor

}

boolean resetFilter(int index)
{
  char buf[17]="";
  char tmp[17]="";
  strcpy(buf,"Rst filter ");
  strcpy(tmp, itoa(index,tmp,2));
  strcat(buf,tmp);
  strcat(buf,"...");
  displayText(buf);
  je_posunuty=false;
  int found=0;
  boolean is_lost=false;
  myStepper[index].setSpeed(slowspeed);//pomaly beh, aby nasiel spolahlivo poziciu
0

  while(found == 0)
  {
    nextFilter(index);
    digitalWrite(brana, HIGH);  //rozsvietit senzor

    for(int i=0;i<abs(jump1/s);i++) //posunut karusel (ci je dlhy otvor)
    {
      myStepper[index].step(s);
    }
    je_posunuty=true;
    if(analogRead(senzor[index])>senzor_level) //je to dlhy otvor - pozicia 0
    {
      found=1;
      if(filter[index]!=0) //pozicia filtrov bola "stratena"
        {
          is_lost=true;
        }
    }
  }
  myStepper[index].setSpeed(fastspeed);//rychly beh

  for(int i=0;i<11;i++)
  {
    nextFilter(index);
  }
  myStepper[index].setSpeed(slowspeed); //uz pomaly
  nextFilter(index);
  filter[index]=0;
  stop_motors();
  return !is_lost;
}

void stop_motors()
{
  digitalWrite(IN1_0, LOW);
  digitalWrite(IN2_0, LOW);
  digitalWrite(IN3_0, LOW);
  digitalWrite(IN4_0, LOW);
  digitalWrite(IN1_1, LOW);
  digitalWrite(IN2_1, LOW);
  digitalWrite(IN3_1, LOW);
  digitalWrite(IN4_1, LOW);
}

void refreshDisplay()
{
  char buf[19]="";
  char tmp[19]="";
  //show filter status
  strcpy(buf,"F: ");
  strcat(buf,itoa(filter[1],tmp,10));
  strcat(buf,"/");
  strcat(buf,itoa(filter[0],tmp,10));
  strcat(buf,"   ");
  lcd.setCursor ( 0, 0 );
  lcd.print(buf);
  //show control voltage
  float val=ctrlVoltage;
  bool isnegative=false;
  if(val<0) isnegative=true;
  val=fabs(val);
  int val_int = (int) val;   // compute the integer part of the float
  int val_fra = (int) ((val - (float)val_int) * 1000);   // compute 3 decimal
places (and convert it to int)
  snprintf (buf, 16, "%d.%03d V  ", val_int, val_fra); //
  lcd.setCursor ( 9, 0 );
  lcd.print(buf);
  //show signal voltage
  val=signalVoltage;
  isnegative=false;
  if(val<0) isnegative=true;
  val=fabs(val);
  val_int = (int) val;   // compute the integer part of the float
  val_fra = (int) ((val - (float)val_int) * 10000);   // compute 4 decimal places
(and convert it to int)
  snprintf (buf, 16, "%d.%04d V  ", val_int, val_fra); //
  lcd.setCursor ( 0, 1 );
  lcd.print(buf);
  lcd.setCursor ( 0, 1 );
  if(isnegative)
  {
   lcd.print("-");
  }
  //show temperature
  val=teplota;
  isnegative=false;
  if(val<0) isnegative=true;
  val=fabs(val);
  val_int = (int) val;   // compute the integer part of the float
  val_fra = (int) ((val - (float)val_int) * 10);   // compute 1 decimal places (and
convert it to int)
  snprintf (buf, 16, "%d.%01d%cC  ", val_int, val_fra,deg); //
  lcd.setCursor ( 10, 1 );
  lcd.print(buf);
  lcd.setCursor ( 9, 1 );
  if(isnegative)
  {
   lcd.print("-");
  }
}
```

```
void displayText(char *text) //displays any text in the first row
{
  lcd.setCursor ( 0, 0 );
  lcd.print(text);
  lcd.print("                ");
}

float getVoltage(int mode) //reads the signal from PMT or Contrl voltage
{
  int16_t adc0;  // we read from the ADC, we have a sixteen bit integer as a result
  if(mode==CTRL)
  {
    adc0 = ads.readADC_SingleEnded(2);  //vstupy = 0,1,2,3
  }
  if(mode==SIGNAL)
  {
    long sum=0;
    for(int i=0;i<repetitions;i++)
    {
      sum += ads.readADC_Differential_0_1();
    }
    adc0=sum/repetitions;
  }
  return(adc0 * 0.125/1.0)/1000.0;    //   0.125/GAIN   0.1875 at default gain of
2/3 ; /1000.0 - results in volts instead of millivolts
                                      // GAIN_ONE - rozsah +/- 4.096V (differential
vyuzije 16 bit) resp 0-4.096V single ended (vyuzije sa len 15 bit);
                                      // GAIN_TWO - rozsah +/- 2.048V atd.
                                      // GAIN_TWOTHIDRS -  rozsah +/- 6.144V (!!!
maximum je VDD + 0.3V - nevyuzije sa rozsah)
}

void setCtrlVoltage(float expected_voltage)
{
  while((getVoltage(CTRL)>(expected_voltage+ofset))&&(level>0))
  {
    level--;
    dac.setVoltage(level,false);
  }
  while((getVoltage(CTRL)<(expected_voltage+ofset))&&(level<4095))
  {
    level++;
    dac.setVoltage(level,false);
  }
}

void setBrightness(int level)
{
  analogWrite(display_brightness_pin, level);
}
```