Patryk Wrzesorek
249465
PN TP 16:10

BADANIE REGULATORA PROPORCJONALNEGO

1. Zadanie do wykonania

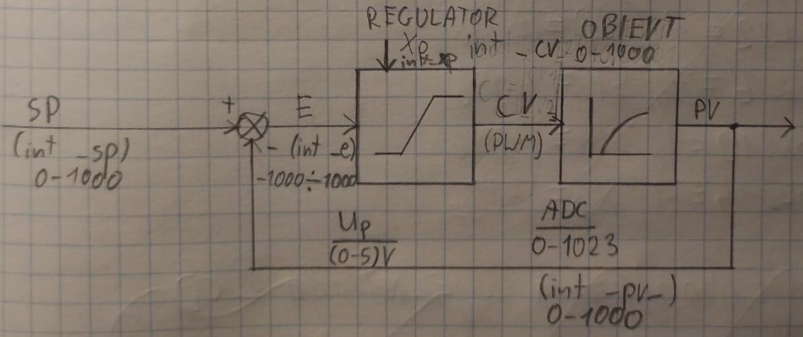
Opracować układ pomiarowy, zmontować układ do badania regulatora, opracować algorytm sterowania w układzie regulacji proporcjonalnej i przetestować regulator w warunkach laboratoryjnych.

Cz. 1. Badanie toru wykonawczego PWM

Cz. 2. Badanie regulatora

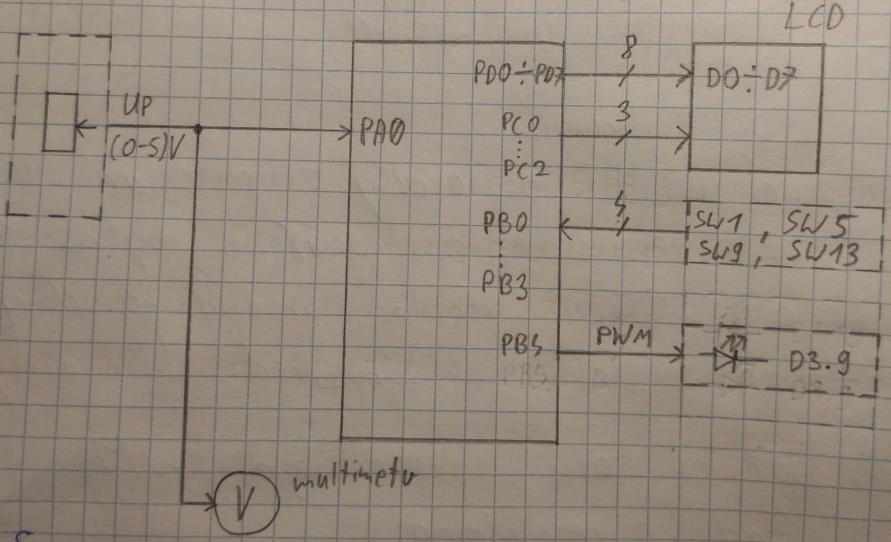
2. Założenia projektowe

2.1 Schemat blokowy typowego układu regulacji



2.2. Schemat blokowy podłączenia sygnałów w układzie do badania regulatora

Patryk Wieczorek



2.3. Zadawanie parametrów regulacji

Patryk Wieczorek

Zakres pomiarowy (0-400)°C / (0-5)V

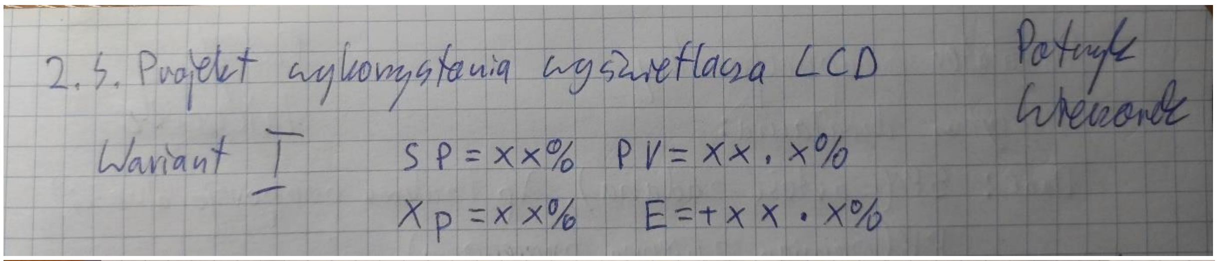
a) Po RESET SP=60%, X_p = 20%

b) Gdy SW1 = 1, SP = 50%

c) Gdy SW5 = 1, SP = 40%

d) Gdy SW9 = 1, X_p = 30%

e) Gdy SW13 = 1, X_p = 40%



2.5. Projekt wykorzystania wyświetlacza LCD

Patryk [illegible]

Wariant I

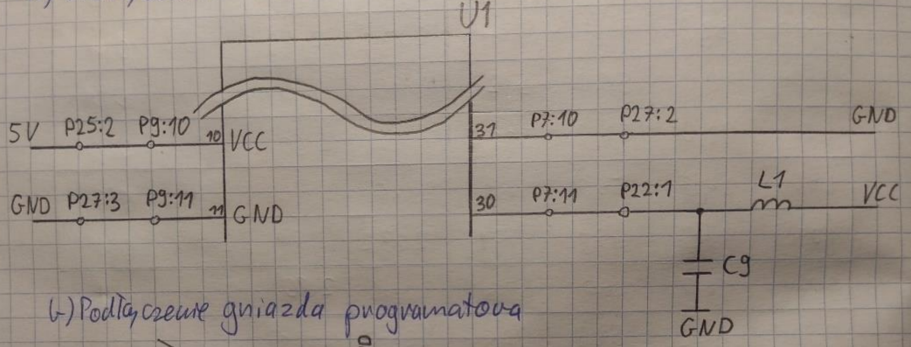
SP = xx% PV = xx.x%

Xp = xx% E = ±xx.x%

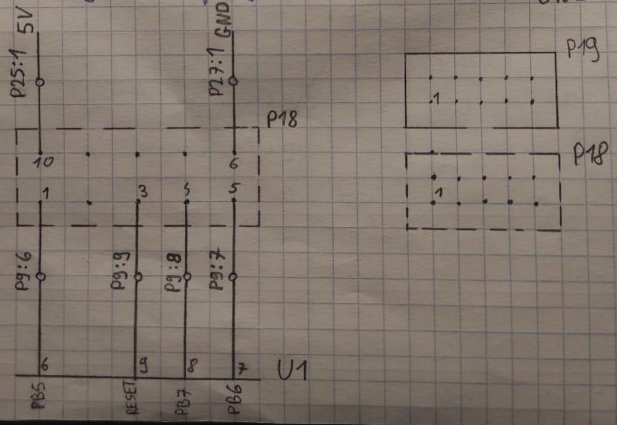
2.5 Schematy ideowe połączeń elektrycznych

Patryk [illegible]

a) Podłączenie zasilania mikrokontrolera

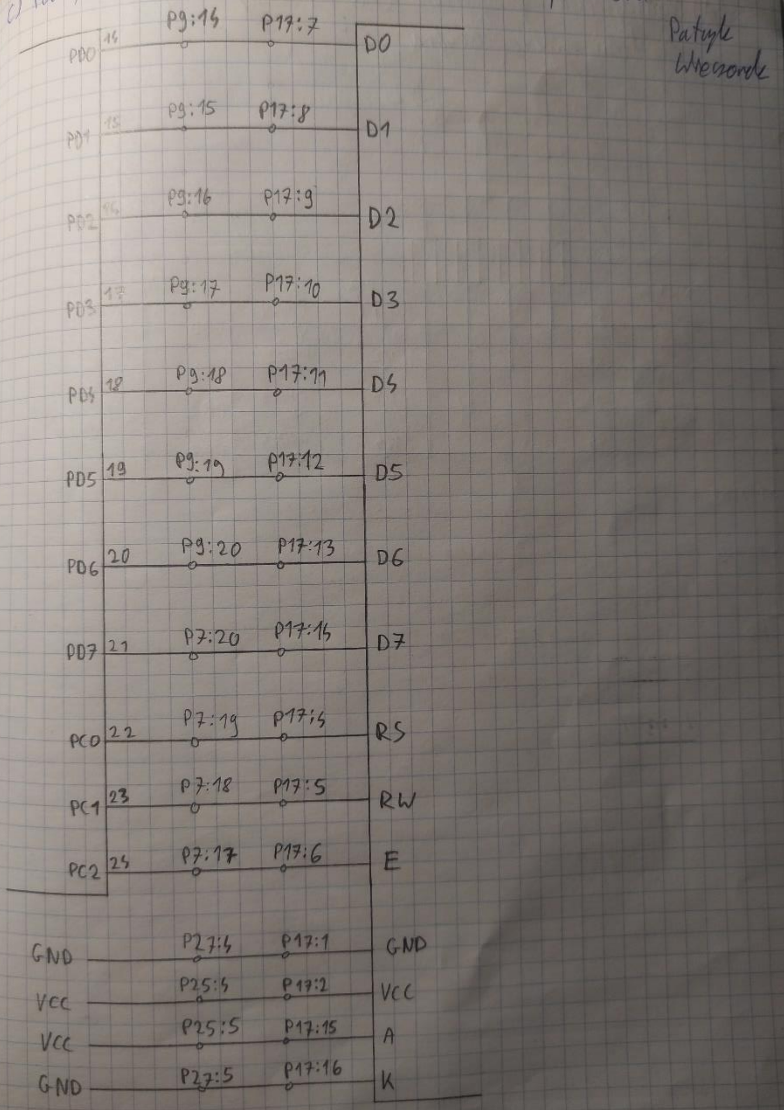


b) Podłączenie gniazda programatora



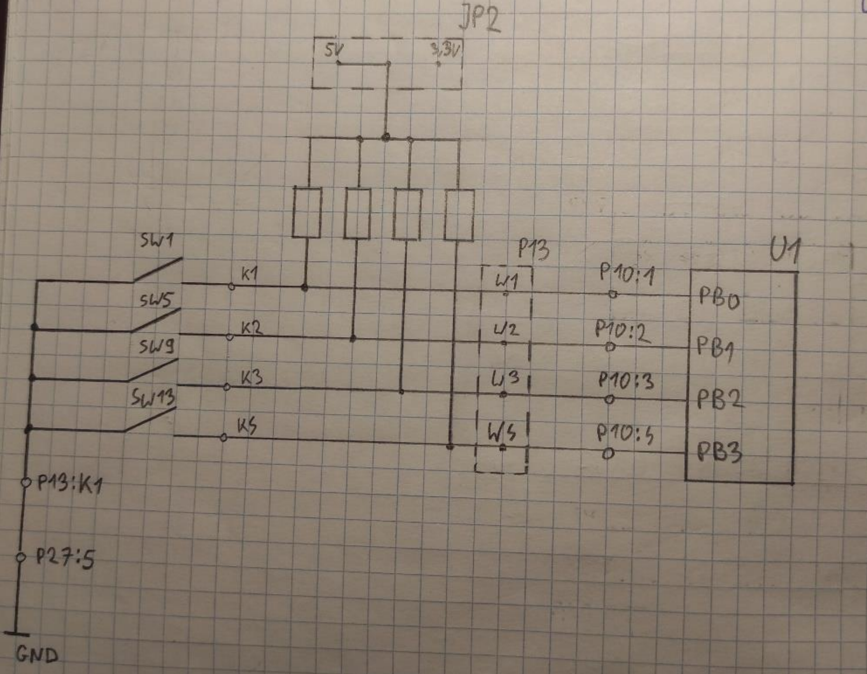
c) Podłączenie wyświetlacza LCD do mikroprocesora

Patryk Wieczorek



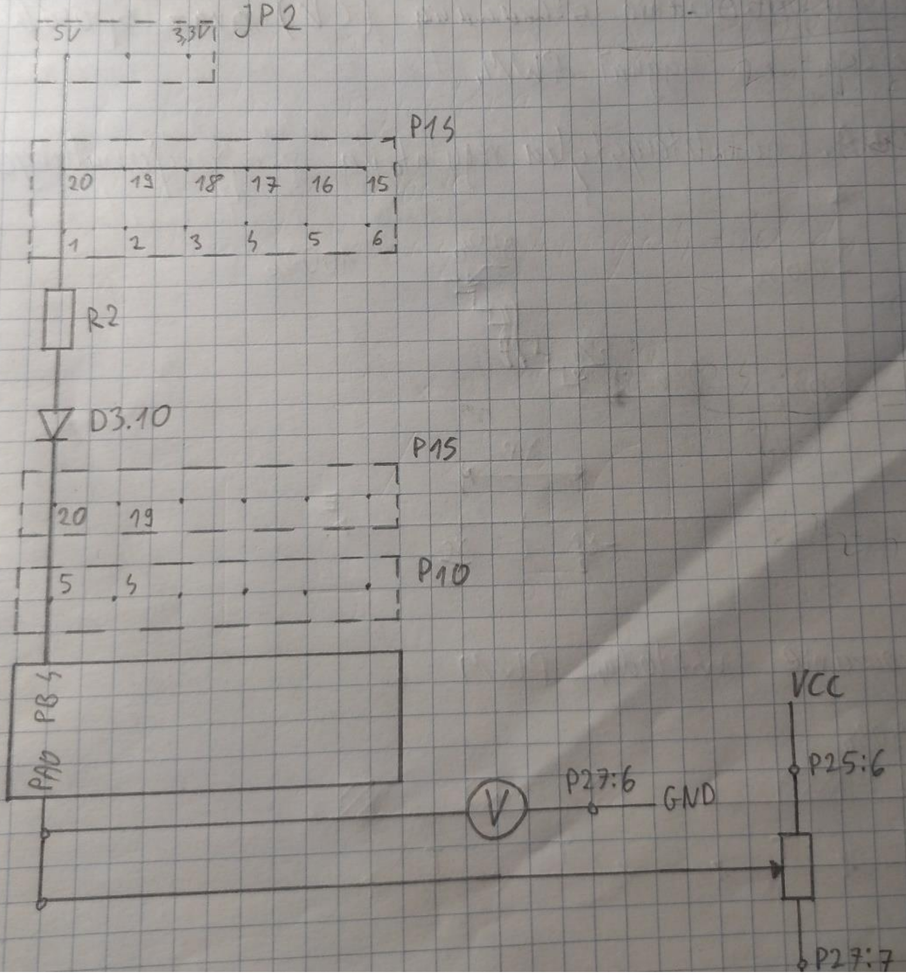
d) Podłączenie przycisków (SW1, SW5, SW9, SW13) do linii mikroprocesora

Patryk Wieczorek



Patryk Wieczorek

e) Podłączenie potencjometru do zadawania napięcia Up (symulacja pomiaru zmiennej procesowej), podłączenie woltomierza do pomiaru napięcia oraz podłączenie wyjścia regulatora do D3.10



Patryk [illegible]

3. Regulator proporcjonalny

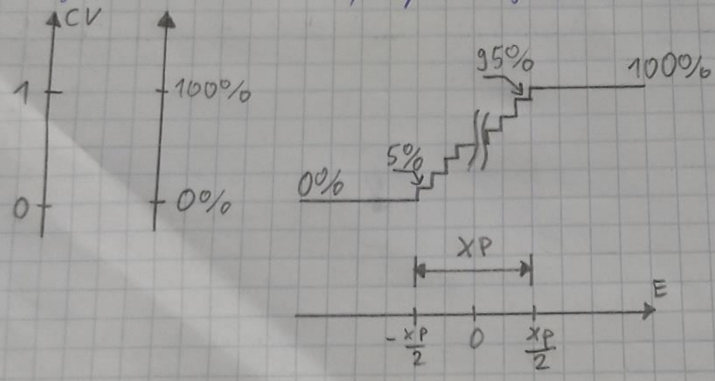
Algorytm działania:

Dane: SP (wartość zadana), X_p (zakres proporcjonalności),
PV (pomiar, zmienna procesowa)

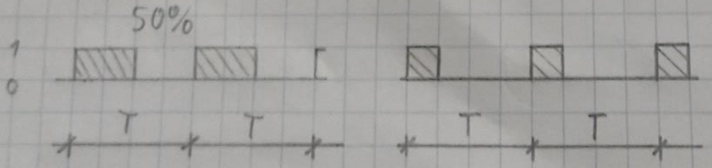
Założenia dodatkowe:

okres dla sygnału PWM T0=200[ms] (dla testów T0=20[s]), rozdzielczość sygnału sterującego 5% (CV=0%, 5%, 10%, ... 95%, 100%), do odmierzania czasu użyć funkcji dely_ms (ms=100 albo dla testów 1000). Opisać szczegółowo sposób obliczania sterowania (CV) oraz sposób przejścia na sygnał PWM.

Rys. 3A. Charakterystyka regulatora proporcjonalnego



Sterowanie metodą PWM



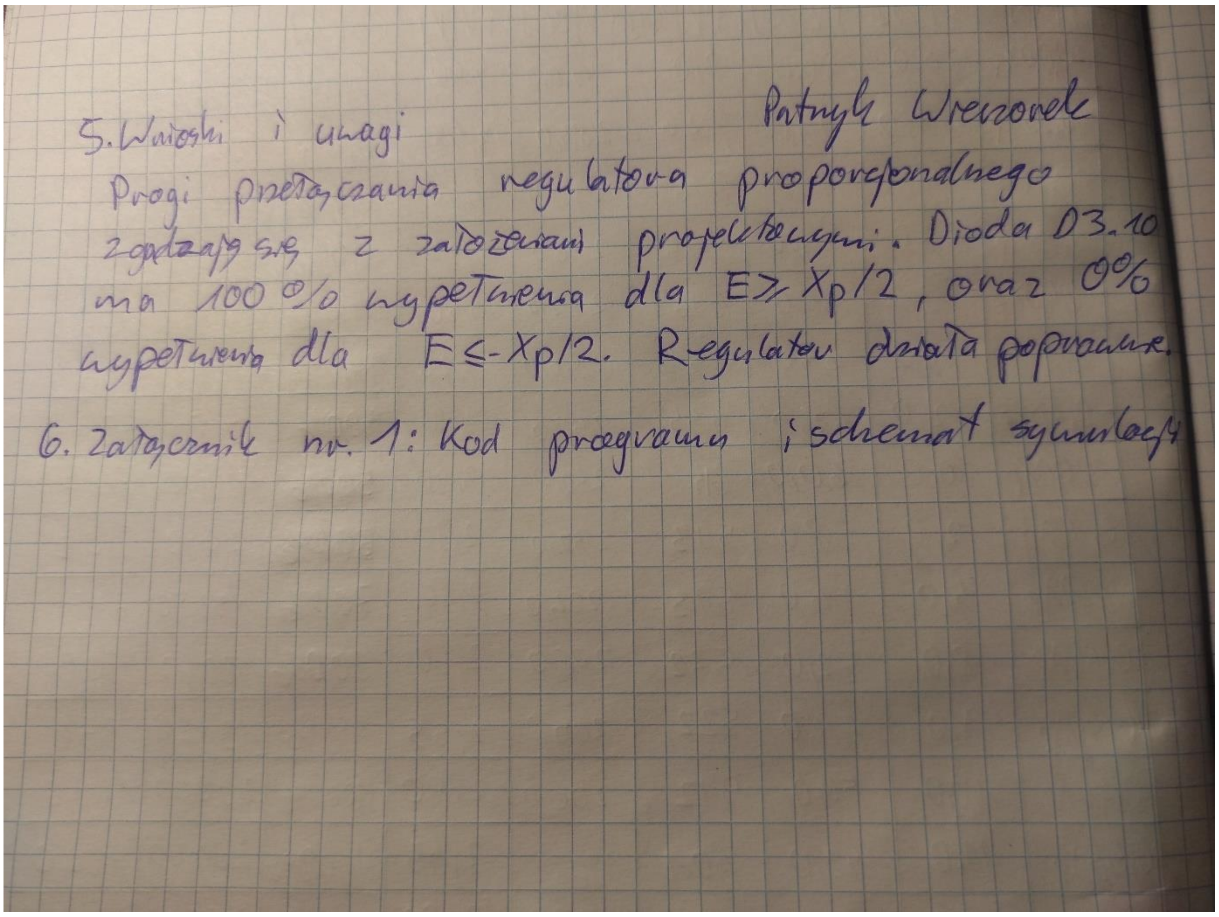
Patryk Wierzowiek

4. Tabela pomiarowa
Każda grupa oblicza dane do tabeli dla „własnych” danych.

Badanie regulatora dla SP=60%, X_p=20%, okres sygnału TO=200[s]
Autor: Zakres pomiarowy: (0-400)°C/(0-5)V

E[Xp]	E[%]	PV[%]	PV[ADC]	PV[°C]	PV[V]	CV[%]	+H [ms] pomiar	+H[s]/200[s] ×100%
-1,00Xp	-20,0	80,0	818	320,0	4,00	0,0%	0,00	0,0
-0,55Xp	-11,0	71,0	726	284,0	3,55	0,00%	0,00	0,0
-0,50Xp	-10,0	70,0	716	280,0	3,50	0,0%	0,00	0,0
-0,45Xp	-9,0	69,0	706	276,0	3,45	5,0%	10,80	5,4
-0,40Xp	-8,0	68,0	696	272,0	3,40	10,0%	20,40	10,2
-0,20Xp	-4,0	64,0	655	256,0	3,20	30,0%	61,20	30,6
-0,10Xp	-2,0	62,0	634	248,0	3,10	40,0%	80,80	40,4
0,00Xp	0,0	60,0	614	240,0	3,00	50,0%	100,60	50,3
0,10Xp	2,0	58,0	593	232,0	2,90	60,0%	120,40	60,2
0,20Xp	4,0	56,0	573	224,0	2,80	70,0%	139,40	69,7
0,40Xp	8,0	52,0	532	208,0	2,60	90,0%	180,60	90,3
0,45Xp	9,0	51,0	522	204,0	2,55	95,0%	191,20	95,6
0,50Xp	10,0	50,0	512	200,0	2,50	100,0%	200,00	100,0
0,55Xp	11,0	49,0	501	196,0	2,45	100,0%	200,00	100,0
1,00Xp	20,0	40,0	409	160,0	2,00	100,0%	200,00	100,0

0,4%

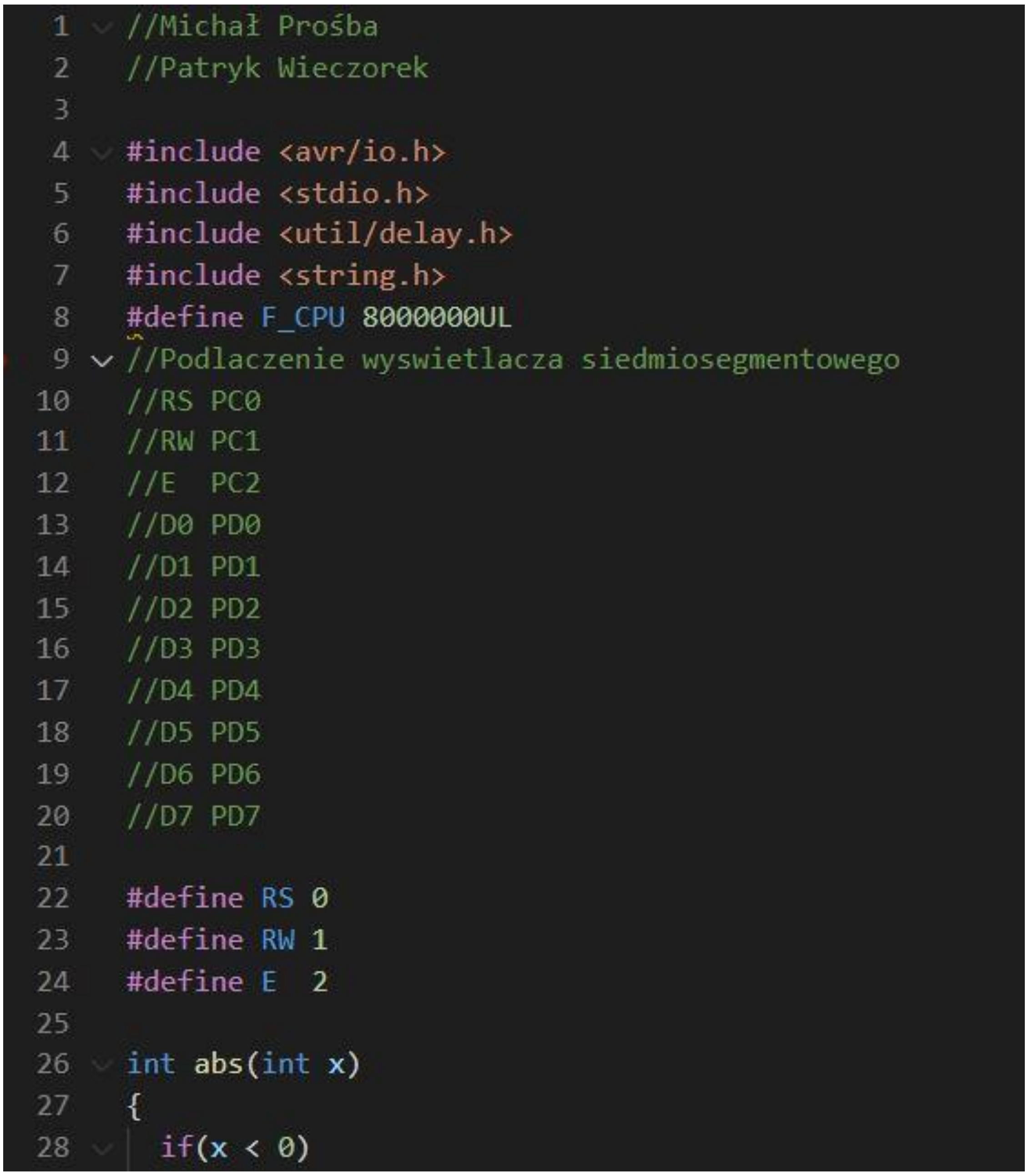


Patryk Wieczorek

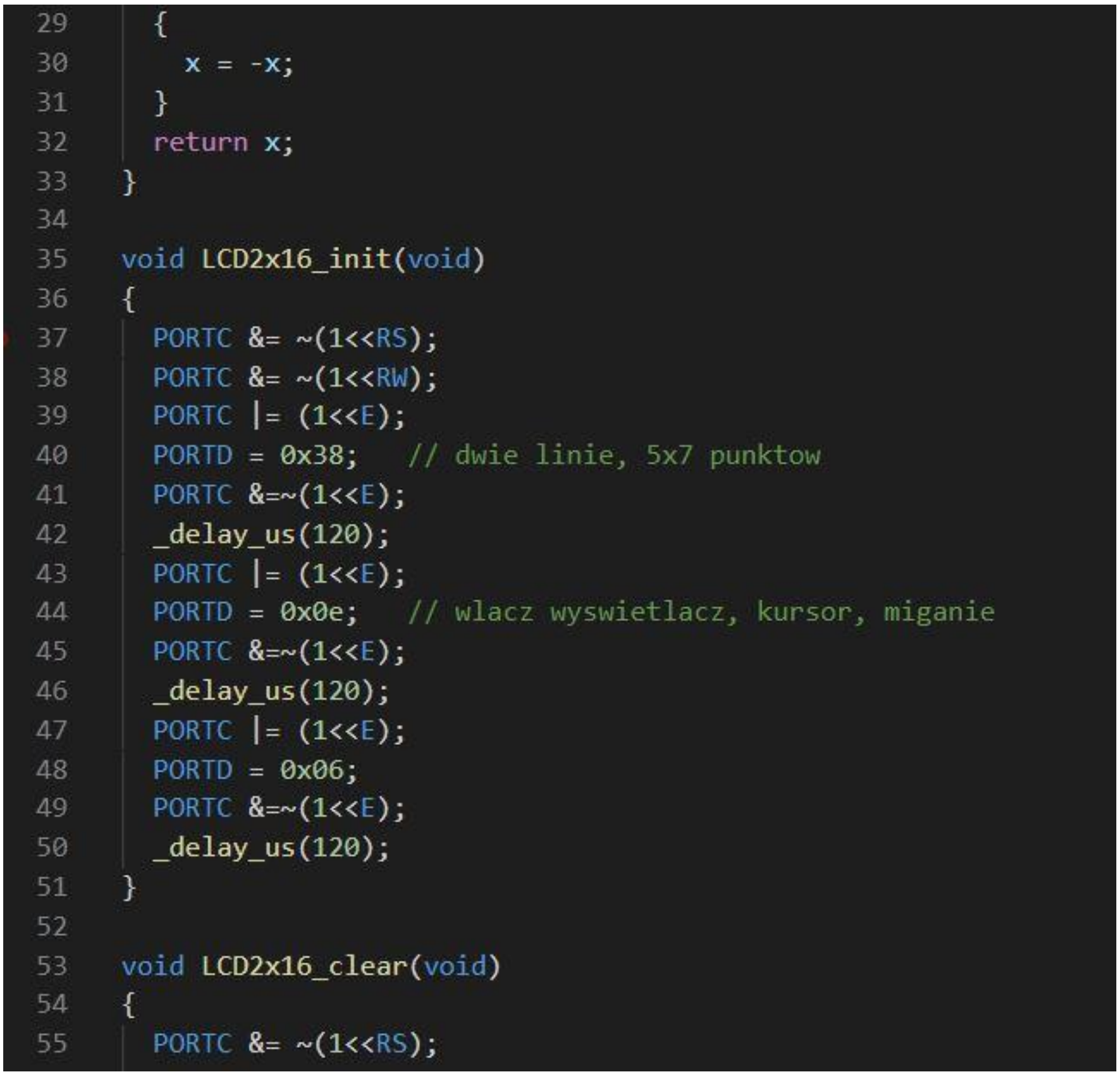
5. Wnioski i uwagi

Progi przełączania regulatora proporcjonalnego zgadzają się z założeniami projektowymi. Dioda D3.10 ma 100 % wypełnienia dla $E \geq X_p/2$, oraz 0% wypełnienia dla $E \leq -X_p/2$. Regulator działa poprawnie.

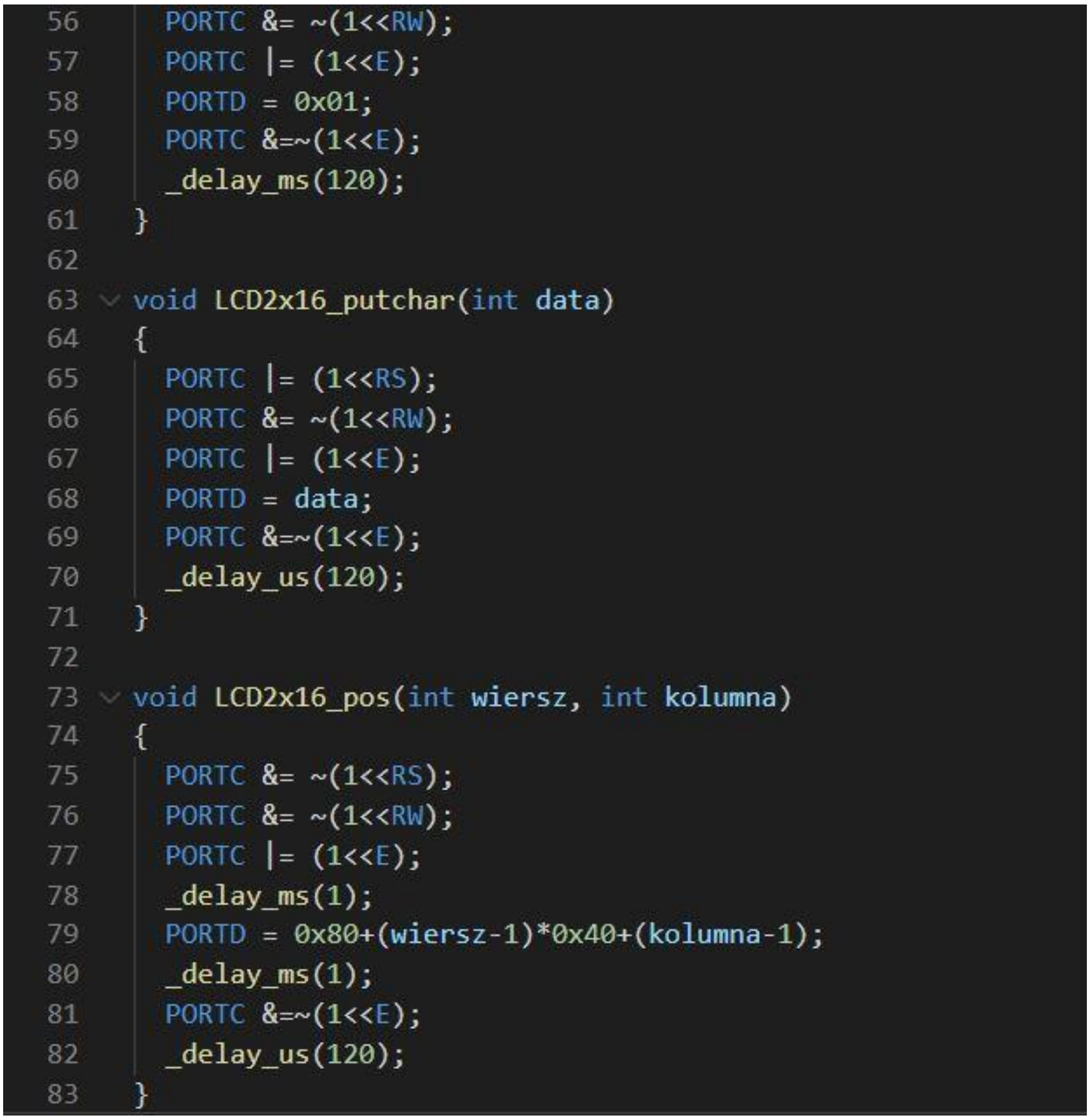
6. Załącznik nr. 1: Kod programu i schemat symulacji

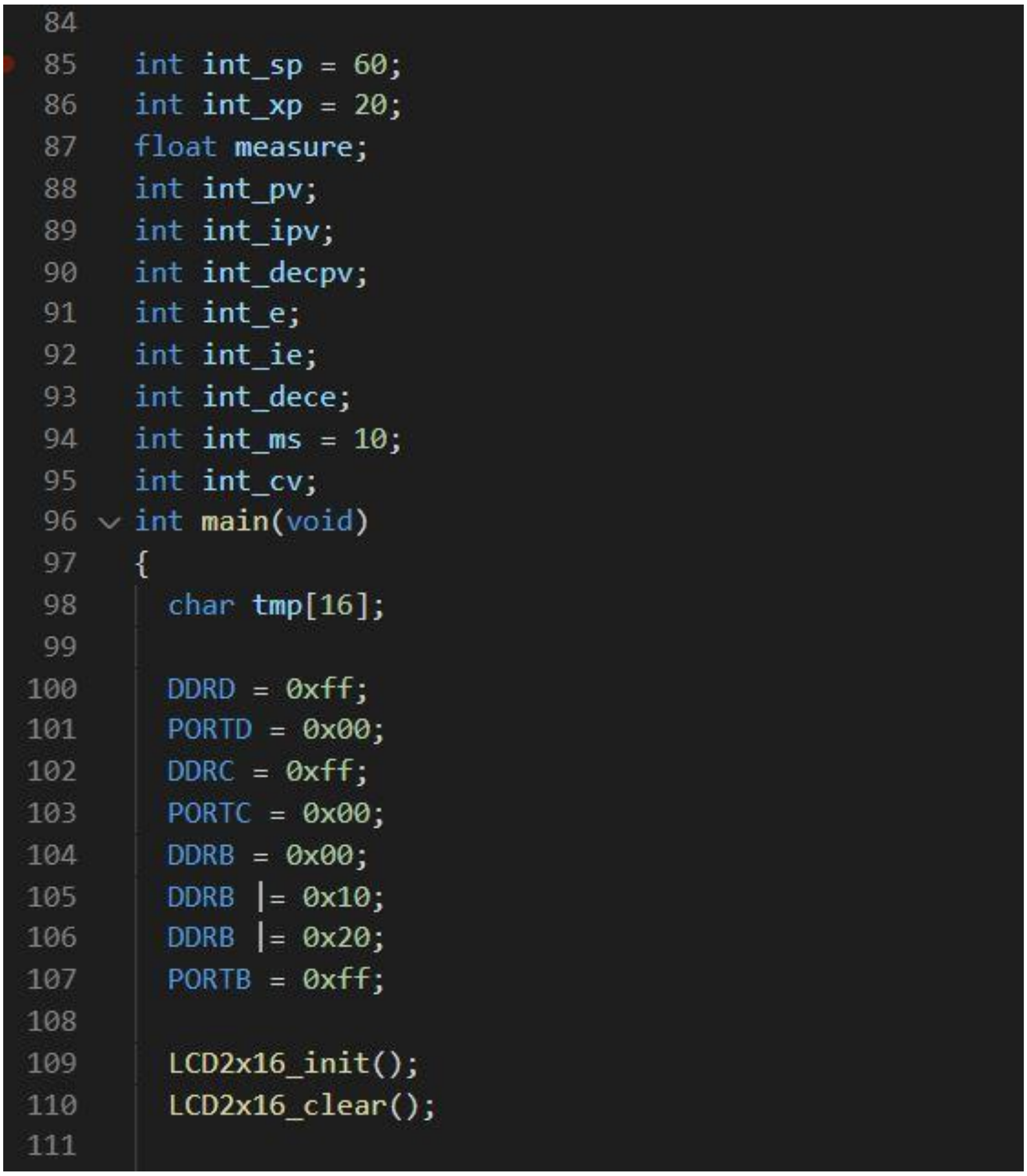


```c
//Michał Prośba
//Patryk Wieczorek

#include <avr/io.h>
#include <stdio.h>
#include <util/delay.h>
#include <string.h>
#define F_CPU 8000000UL
//Podlaczenie wyswietlacza siedmiosegmentowego
//RS PC0
//RW PC1
//E  PC2
//D0 PD0
//D1 PD1
//D2 PD2
//D3 PD3
//D4 PD4
//D5 PD5
//D6 PD6
//D7 PD7

#define RS 0
#define RW 1
#define E  2

int abs(int x)
{
  if(x < 0)
```

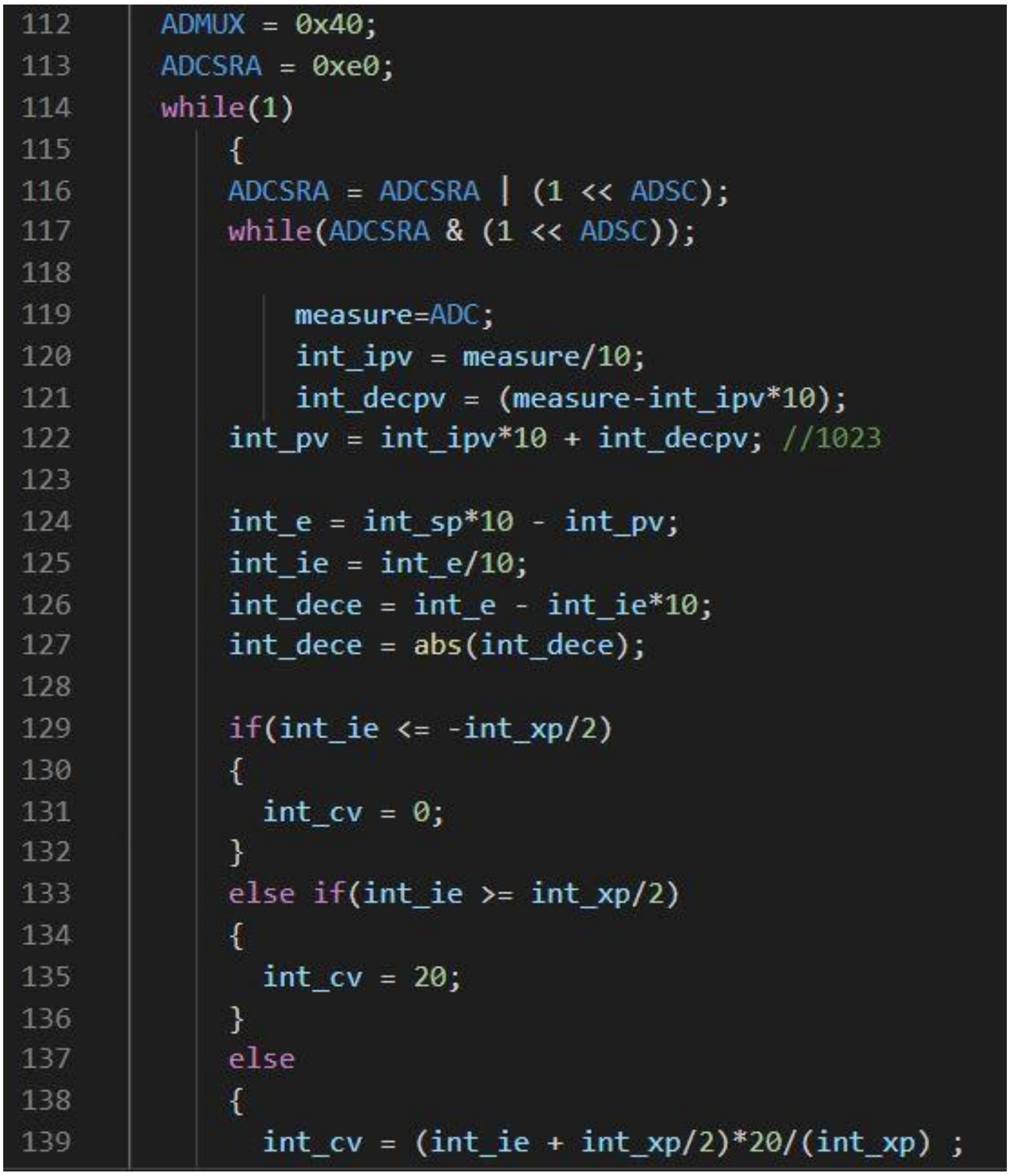


```c
  {
    x = -x;
  }
  return x;
}

void LCD2x16_init(void)
{
  PORTC &= ~(1<<RS);
  PORTC &= ~(1<<RW);
  PORTC |= (1<<E);
  PORTD = 0x38;   // dwie linie, 5x7 punktow
  PORTC &=~(1<<E);
  _delay_us(120);
  PORTC |= (1<<E);
  PORTD = 0x0e;   // wlacz wyswietlacz, kursor, miganie
  PORTC &=~(1<<E);
  _delay_us(120);
  PORTC |= (1<<E);
  PORTD = 0x06;
  PORTC &=~(1<<E);
  _delay_us(120);
}

void LCD2x16_clear(void)
{
  PORTC &= ~(1<<RS);
```

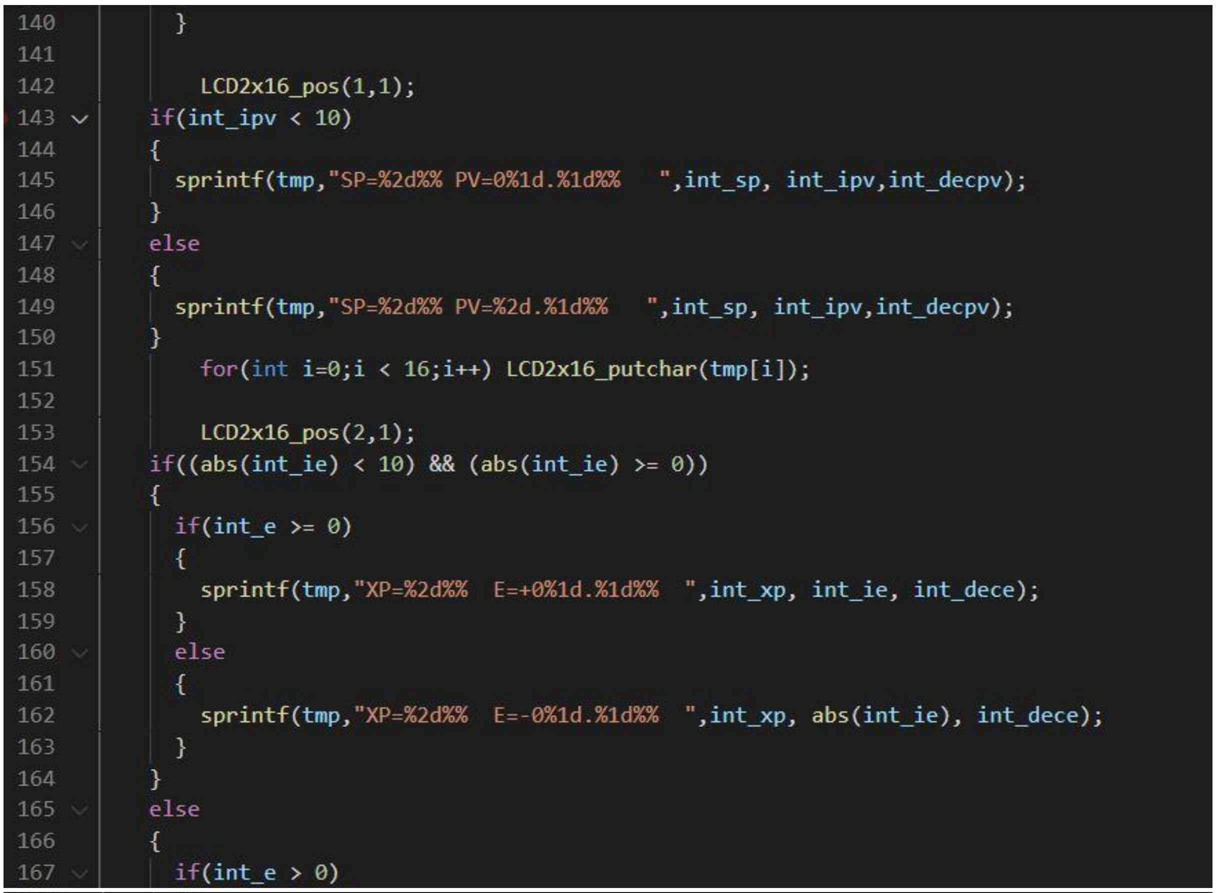

```
    PORTC &= ~(1<<RW);
    PORTC |= (1<<E);
    PORTD = 0x01;
    PORTC &=~(1<<E);
    _delay_ms(120);
}

void LCD2x16_putchar(int data)
{
    PORTC |= (1<<RS);
    PORTC &= ~(1<<RW);
    PORTC |= (1<<E);
    PORTD = data;
    PORTC &=~(1<<E);
    _delay_us(120);
}

void LCD2x16_pos(int wiersz, int kolumna)
{
    PORTC &= ~(1<<RS);
    PORTC &= ~(1<<RW);
    PORTC |= (1<<E);
    _delay_ms(1);
    PORTD = 0x80+(wiersz-1)*0x40+(kolumna-1);
    _delay_ms(1);
    PORTC &=~(1<<E);
    _delay_us(120);
}
```



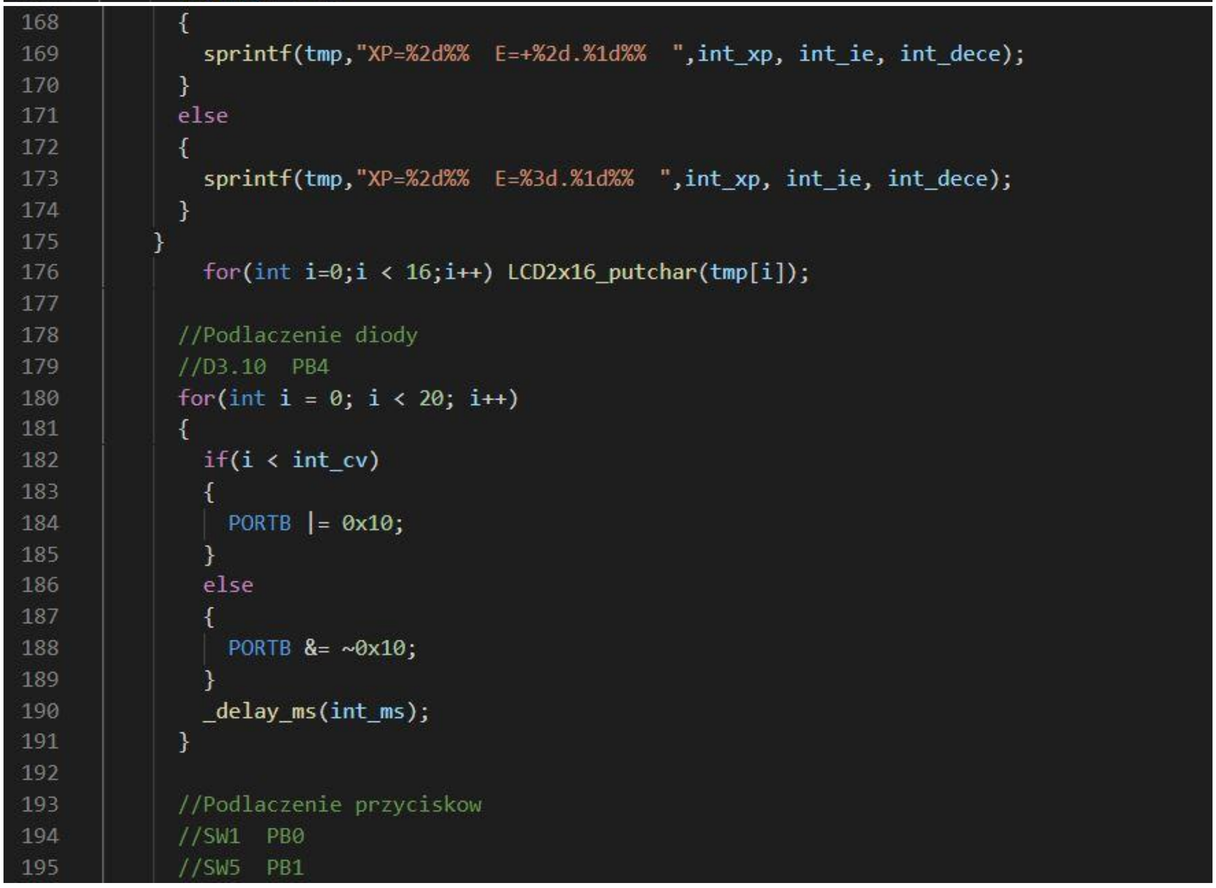
```c

int int_sp = 60;
int int_xp = 20;
float measure;
int int_pv;
int int_ipv;
int int_decpv;
int int_e;
int int_ie;
int int_dece;
int int_ms = 10;
int int_cv;
int main(void)
{
    char tmp[16];

    DDRD = 0xff;
    PORTD = 0x00;
    DDRC = 0xff;
    PORTC = 0x00;
    DDRB = 0x00;
    DDRB |= 0x10;
    DDRB |= 0x20;
    PORTB = 0xff;

    LCD2x16_init();
    LCD2x16_clear();

```



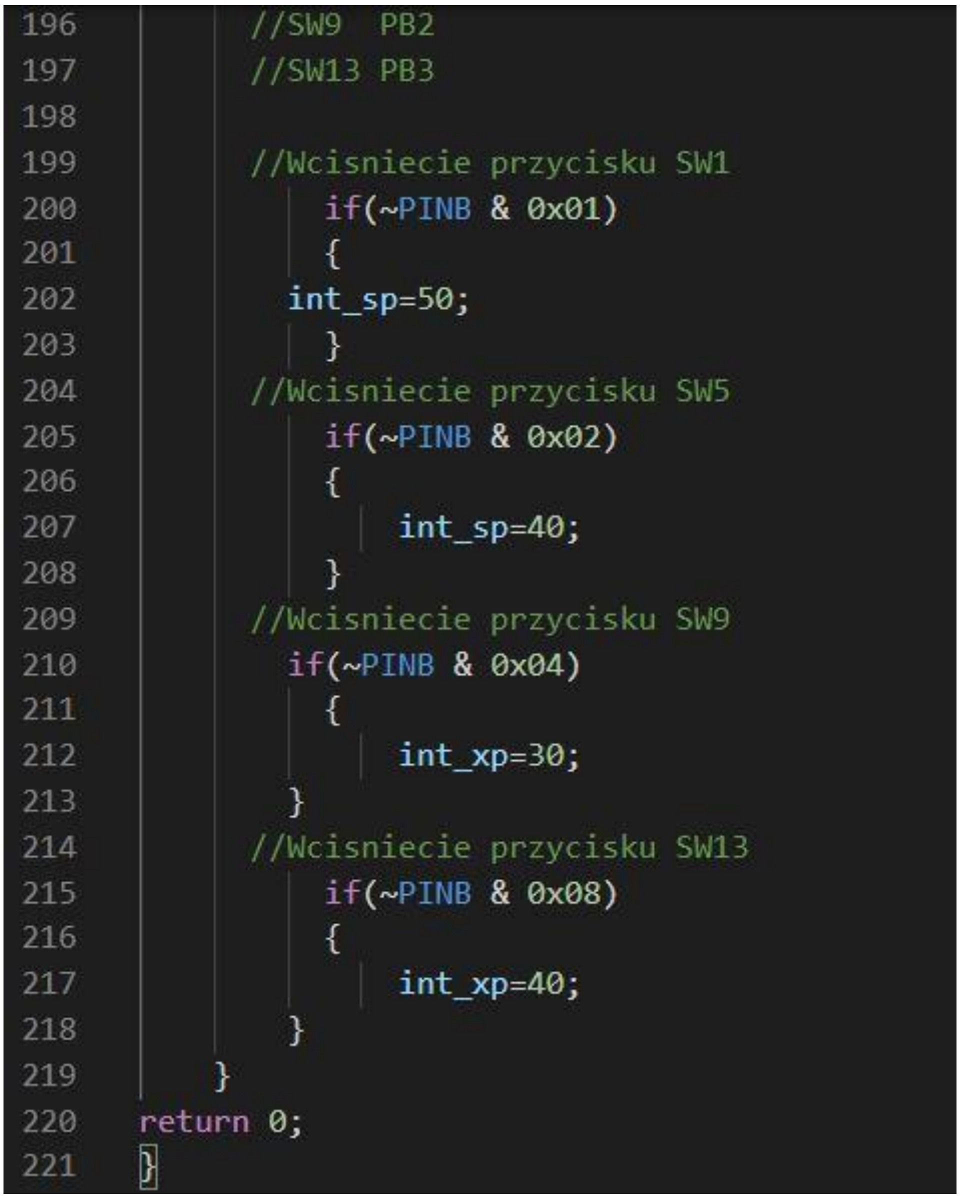
```
  ADMUX = 0x40;
  ADCSRA = 0xe0;
  while(1)
      {
      ADCSRA = ADCSRA | (1 << ADSC);
      while(ADCSRA & (1 << ADSC));

          measure=ADC;
          int_ipv = measure/10;
          int_decpv = (measure-int_ipv*10);
      int_pv = int_ipv*10 + int_decpv; //1023

      int_e = int_sp*10 - int_pv;
      int_ie = int_e/10;
      int_dece = int_e - int_ie*10;
      int_dece = abs(int_dece);

      if(int_ie <= -int_xp/2)
      {
        int_cv = 0;
      }
      else if(int_ie >= int_xp/2)
      {
        int_cv = 20;
      }
      else
      {
        int_cv = (int_ie + int_xp/2)*20/(int_xp) ;
```



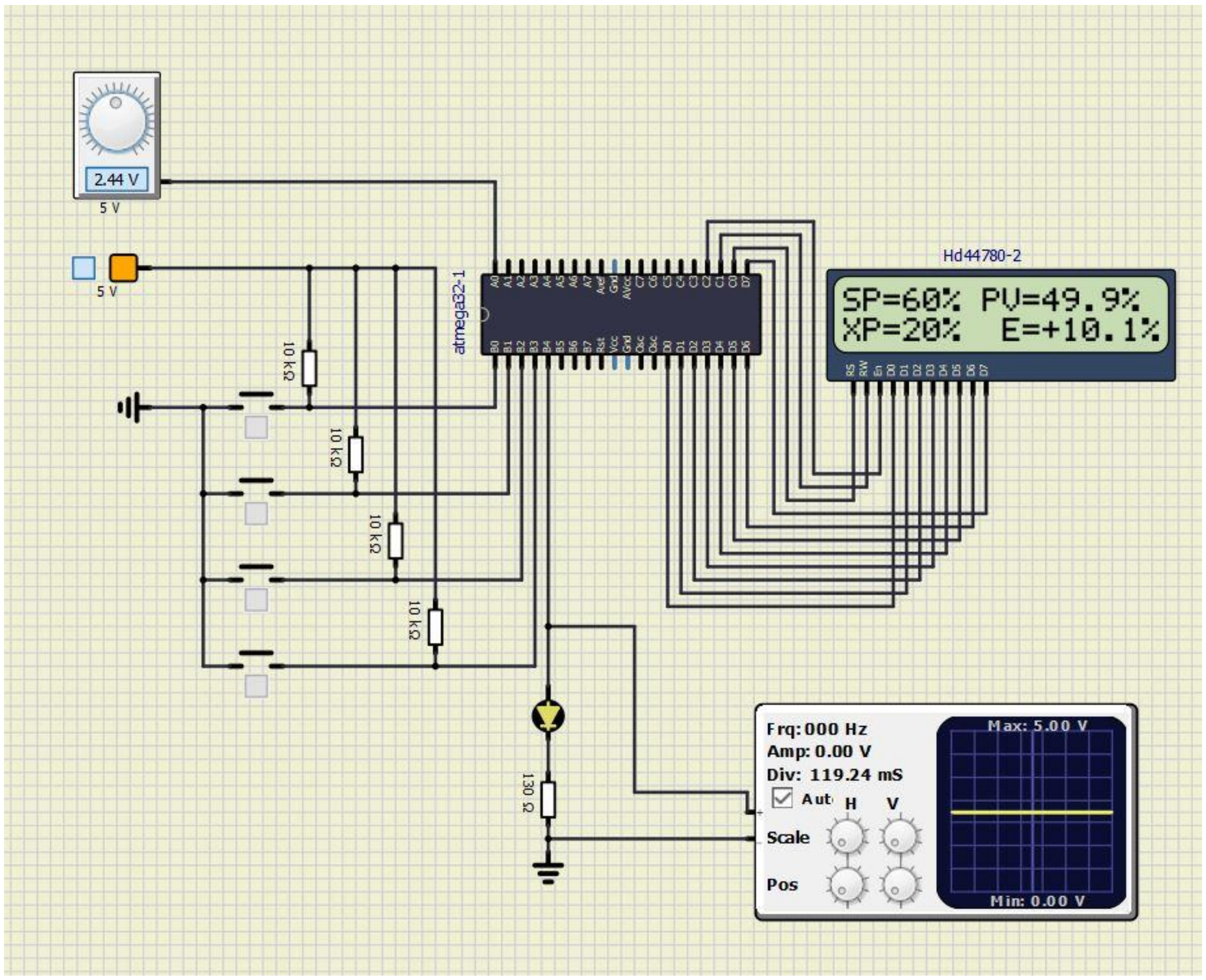
```c
  }

    LCD2x16_pos(1,1);
if(int_ipv < 10)
{
  sprintf(tmp,"SP=%2d%% PV=0%1d.%1d%%   ",int_sp, int_ipv,int_decpv);
}
else
{
  sprintf(tmp,"SP=%2d%% PV=%2d.%1d%%   ",int_sp, int_ipv,int_decpv);
}
    for(int i=0;i < 16;i++) LCD2x16_putchar(tmp[i]);

    LCD2x16_pos(2,1);
if((abs(int_ie) < 10) && (abs(int_ie) >= 0))
{
  if(int_e >= 0)
  {
    sprintf(tmp,"XP=%2d%%  E=+0%1d.%1d%%  ",int_xp, int_ie, int_dece);
  }
  else
  {
    sprintf(tmp,"XP=%2d%%  E=-0%1d.%1d%%  ",int_xp, abs(int_ie), int_dece);
  }
}
else
{
  if(int_e > 0)
```



```c
  {
    sprintf(tmp,"XP=%2d%%  E=+%2d.%1d%%  ",int_xp, int_ie, int_dece);
  }
  else
  {
    sprintf(tmp,"XP=%2d%%  E=%3d.%1d%%  ",int_xp, int_ie, int_dece);
  }
}
    for(int i=0;i < 16;i++) LCD2x16_putchar(tmp[i]);

  //Podlaczenie diody
  //D3.10  PB4
  for(int i = 0; i < 20; i++)
  {
    if(i < int_cv)
    {
      PORTB |= 0x10;
    }
    else
    {
      PORTB &= ~0x10;
    }
    _delay_ms(int_ms);
  }

  //Podlaczenie przyciskow
  //SW1  PB0
  //SW5  PB1
```

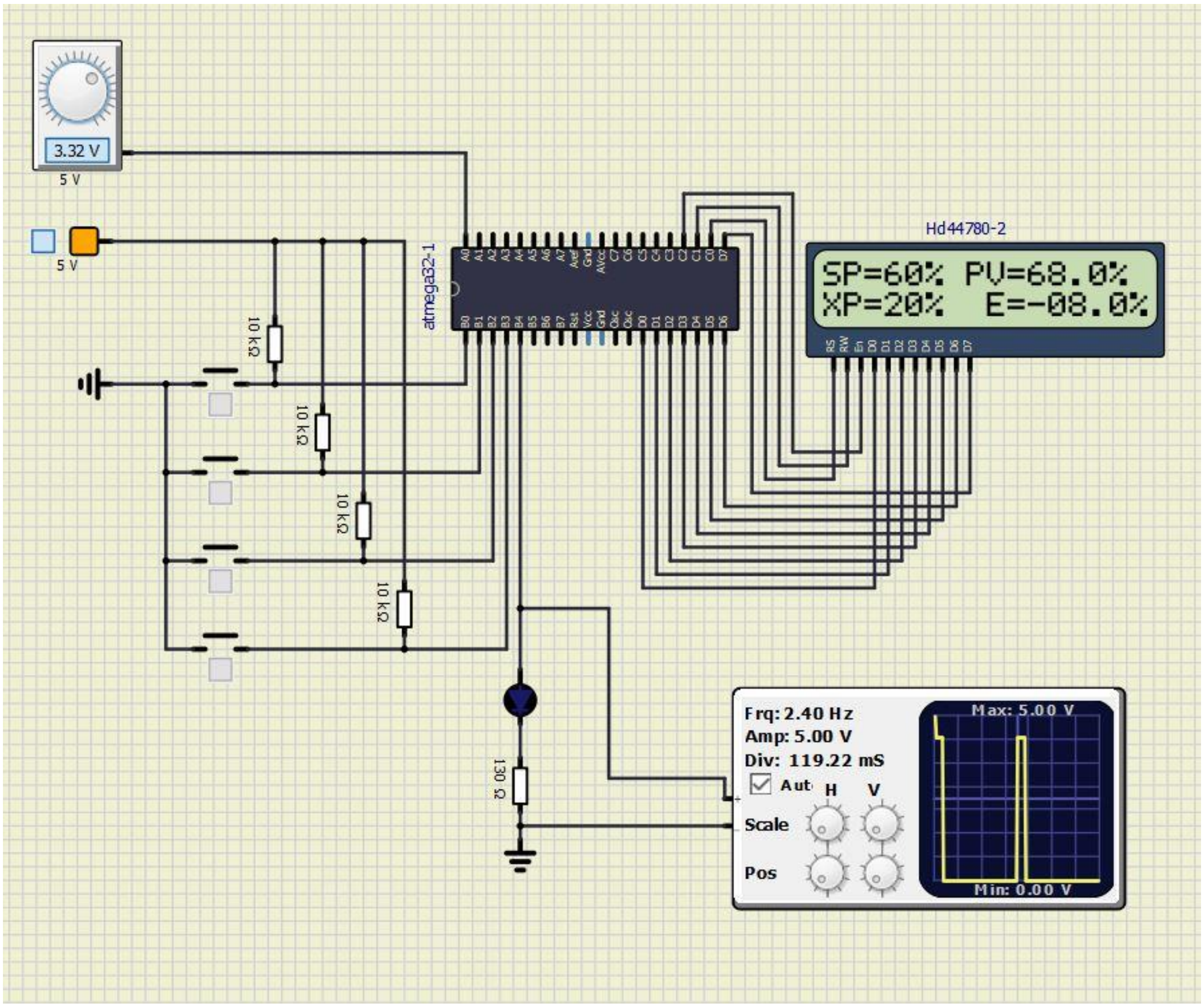


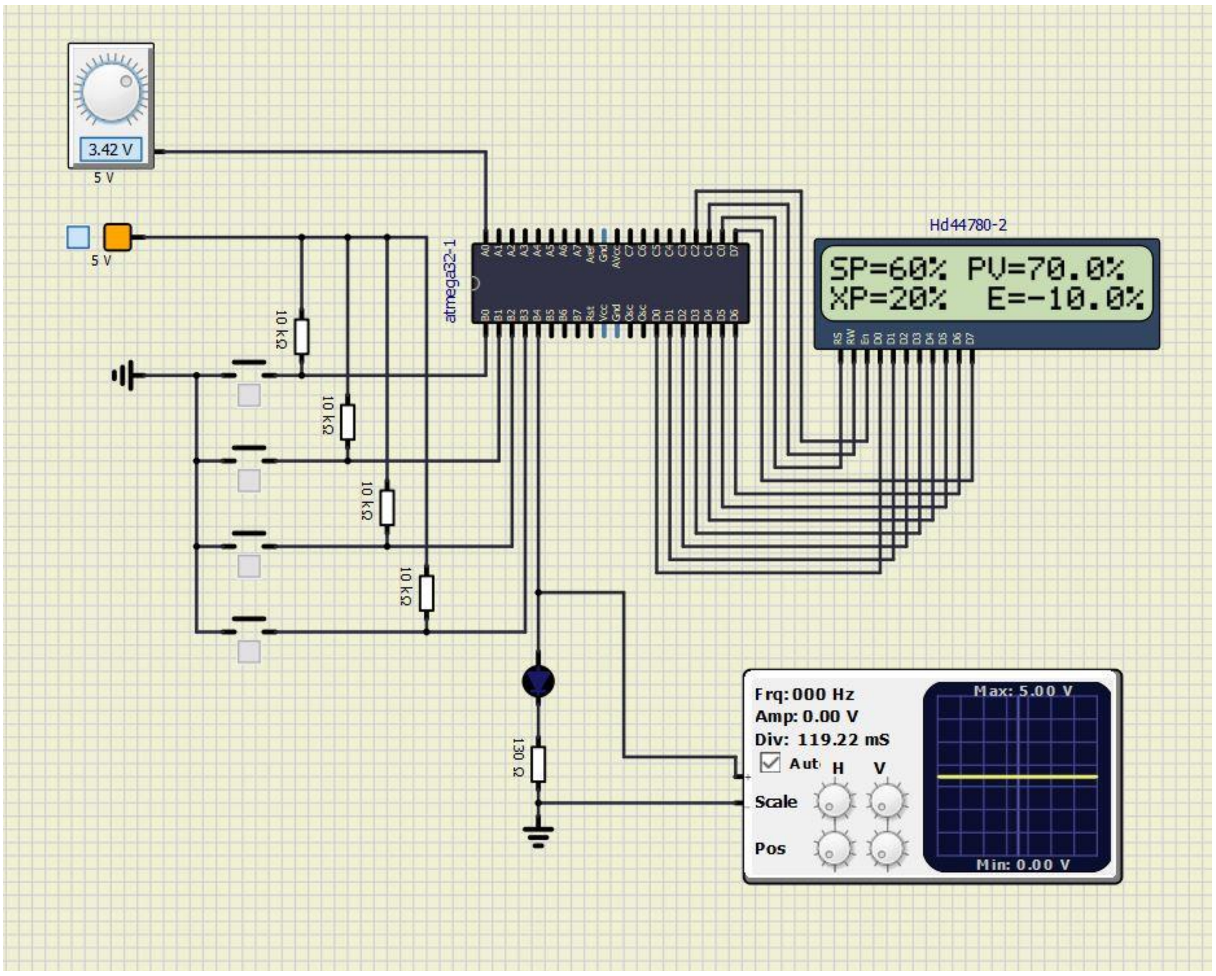
```
          //SW9  PB2
          //SW13 PB3

          //Wcisniecie przycisku SW1
              if(~PINB & 0x01)
              {
            int_sp=50;
              }
          //Wcisniecie przycisku SW5
              if(~PINB & 0x02)
              {
                  int_sp=40;
              }
          //Wcisniecie przycisku SW9
            if(~PINB & 0x04)
              {
                  int_xp=30;
            }
          //Wcisniecie przycisku SW13
              if(~PINB & 0x08)
              {
                  int_xp=40;
            }
        }
    return 0;
    }
```



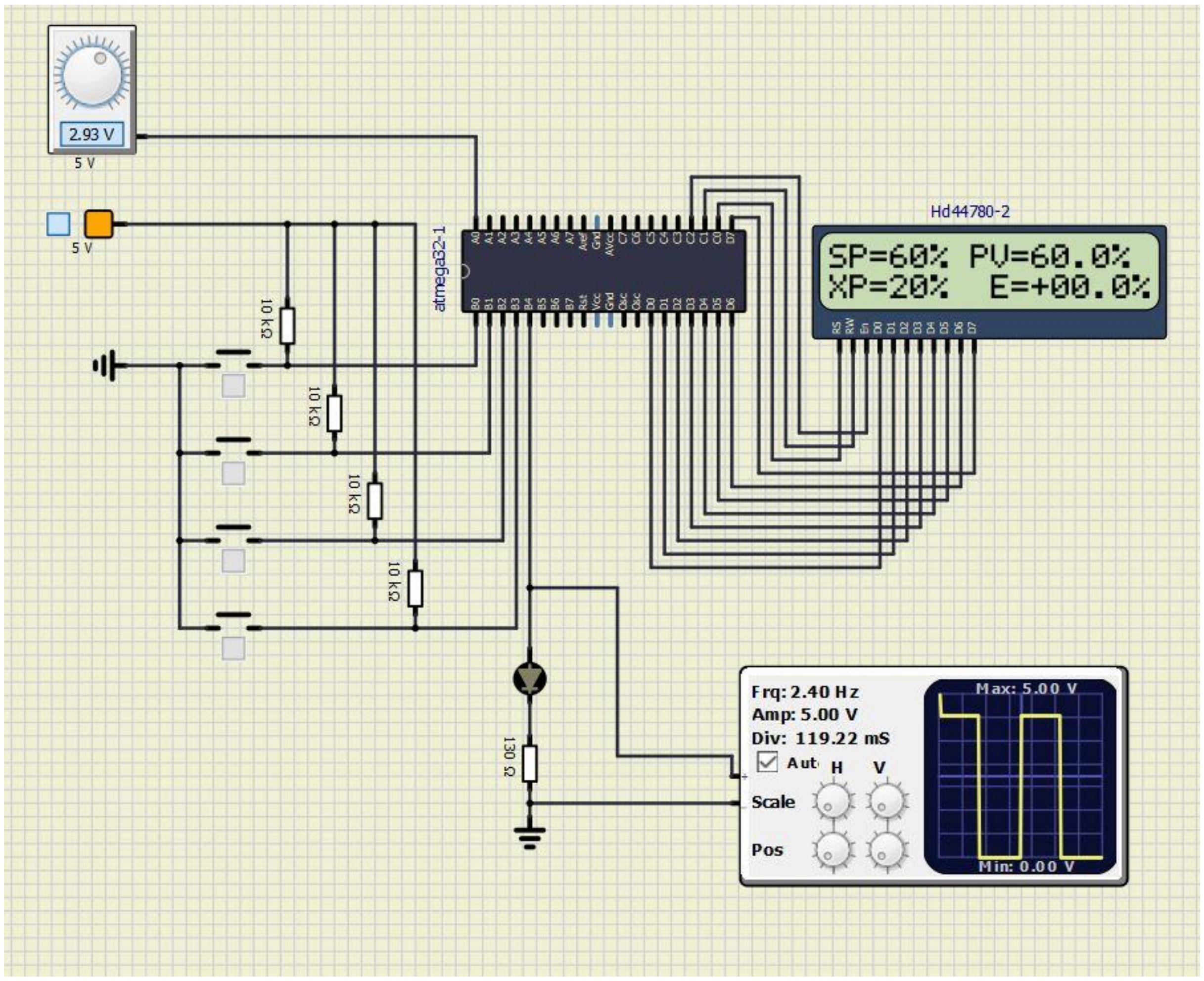
```
SP=60% PV=49.9%
XP=20%   E=+10.1%
```

```
Frq: 000 Hz
Amp: 0.00 V
Div: 119.24 mS
```

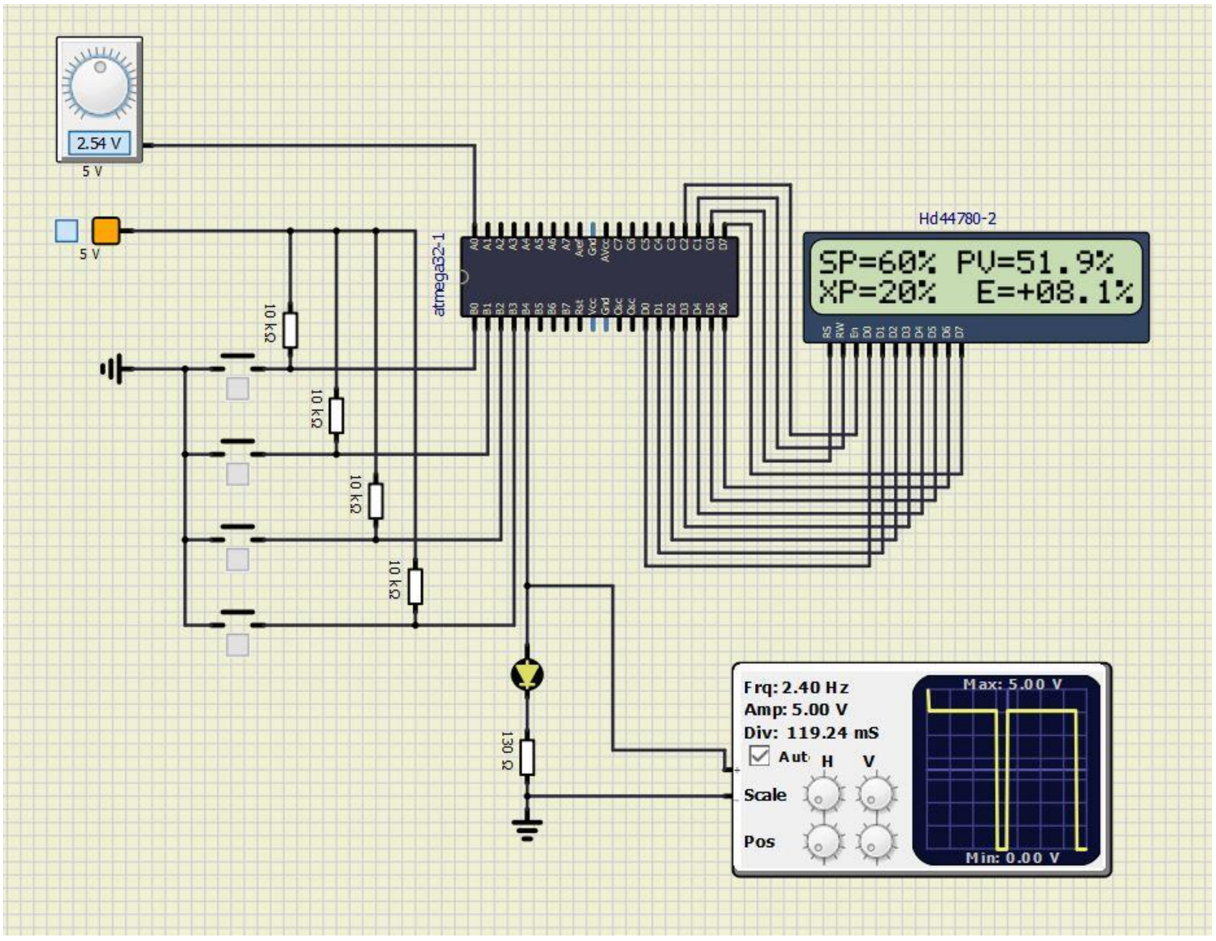

3.32 V
5 V
5 V
atmega32-1
Hd44780-2
SP=60% PV=68.0%
XP=20% E=-08.0%
10 kΩ
10 kΩ
10 kΩ
10 kΩ
130 Ω
Frq: 2.40 Hz
Amp: 5.00 V
Div: 119.22 mS
Auto
H
V
Scale
Pos
Max: 5.00 V
Min: 0.00 V



```
SP=60% PV=70.0%
XP=20%   E=-10.0%
```



```
SP=60% PV=60.0%
XP=20%  E=+00.0%
```


2.54 V
5 V
5 V
atmega32-1
Hd44780-2
SP=60% PV=51.9%
XP=20%   E=+08.1%
10 kΩ
10 kΩ
10 kΩ
10 kΩ
130 Ω
Frq: 2.40 Hz
Amp: 5.00 V
Div: 119.24 mS
Auto
H
V
Scale
Pos
Max: 5.00 V
Min: 0.00 V