// **3 p digitalni regulator** //

```
// Program v mikro C deluje na PIC16F88 kot tripoložajni regulator //
// Za programiranje rabimo mikro-C razvojno okolje //
// Program naložimo in testiramo na razvojni plošči Mikroelektronike//
// Regulator vzdržuje nastavljeno vrednost procesne veličine tropoložajno //
// Primeri možnih procesnih veličin: temperatura, pretok, nivo. //
// 3p aktuatorji regulirne veličine so lahko: grelec -hladilnik, ventil //

// DELOVANJE IN POVEZAVE za primer REGULACIJE PRETOKA //

// Programski algoritem v zanki ciklično skanira regulirano veličino in //
// jo primerja z programsko nastavljeno zgornjo ter spodnjo mejo. //
// V primeru odstopanja se izvrši vklop regulirne veličine. //
// Regulirna veličina preko aktuatorja korigira regulirano veličino.//
// V primeru izkrmiljenja senzorji javijo skrajni položaj aktuatorja//

// A4 je analogni vhod  za zajem meritve regulirane veličine //
// B6 in B7 sta digitalna vhoda, ostali pini so digitalni izhodi   //
// na B6 damo signal max odprtega aktuatorja - KS_OL //
// na B7 damo signal max zaprtega aktuatorja - KS_ZL //

// digitalni izhod B1 odpira aktuator, to signalizira LED2 //
// digitalni izhod B3 zapira aktuator,  to signalizira LED3 //
// LED so vezane na digitalne izhode B0, B1, B3, B4 //
// LED1 na B0 signalizira vklop KS_ZL //
// LED4 na B4 signalizira vklop KS_OL //
// Preko USB povezave PIC pošilja informacije na PC za nadzor pretoka //
// Preko USB povezave na PC inkrementiramo nastavitev pretoka gor/dol 1% //

//////////////////////////////////////////////////////////////////

// ZAČETEK PROGRAMA //
void main()
{

// DEKLARACIJE SPREMENLJIVK //
//unsigned pretok = 100; // shrani meritev  //

int pretok_i = 100;    // shrani 10 bitno meritev MAX 1024  //
unsigned pretok = 100; // shrani 8 bitno meritev  MAX 255 //
unsigned nastavitev = 100;  // shrani nastavitev reference  //
unsigned ref_sp = nastavitev - 15; // spodnja toleranca regulacije //
unsigned ref_zg = nastavitev + 15; // zgornja toleranca regulacije //
unsigned i = 0;  // zančna spremenljivka //
char podatek[4] = "100";  // shrani meritev kot znakovni niz //
char beseda[13] = "PRETOK m3: "; // shrani znakovni niz komentarja //
char alarm[4] = " KS"; // shrani znakovni niz za alarm //
char znak = 'R'; // shrani znak za delovanje regulatorja //
char znak_1 = '?'; // shrani promptni znak za vpis inkrementa //
```

```
char inkrement = '+';  // shrani znak za povečanje nastavitve pretoka  //

// INICIALIZACIJE IN RESET  //
Usart_Init(9600); // inicializacija Usart  //
PORTB = 0; // reset vseh pinov porta B  //

// NASTAVITEV MERITEV z uporabo ADC in analognih vhodov //
ANSEL = 0b00010000; // A4 definiramo za analogni vhod //
ADCON1 = 0x80;  // definiramo uro in Uref ? //
TRISA = 0b00010000;  // definiram A4 za vhod //

// NASTAVITEV DIGITALNIH VHODOV IN IZHODOV  //
TRISB = 192;  // B6 in B7  sta vhoda, ostali pini so izhodi  //


// GLAVNA NEPOGOJNA ZANKA //
while(1)
{


//  1. ZAJEM MERITVE IN SHRANITEV V SPREMENLJIVKO //

//   10 bitni zajem da maksimalno vrednost 1024 zato je intiger format  //
pretok_i = Adc_Read(4); // čita meritev na A4 //
//   pretok_i / 4 da maksimalno 255 in gre v 8 bitni unsigned format   //
pretok = pretok_i / 4;  // 10 bitno meritev pretvorimo v 8 bitno //


//  2. FUNKCIJA REGULACIJE  //

PORTB.F0 = 0; // reset signalizacije KS_ZL //
PORTB.F2 = 0; // reset Usart pina  //
PORTB.F4 = 0; // reset signalizacije KS_OL //
PORTB.F5 = 0; // reset Usart pina  //
// test nizkega pretoka in končnega stikala KS_OL //
if ((pretok < ref_sp)&&(PORTB.F6 == 0))
{
// korekcija nizkega pretoka - delovanje regulatorja //
PORTB.F1 = 1; // B1 = 1 za odpiranje lopute, LED2 gori  //
Usart_Write(znak); // znak 'R' za delovanje regulacije, izpis na PC //
}
// reset korekcije nizkega pretoka - mirovanje regulatorja//
else
PORTB.F1 = 0; // B1 = 0 za reset korekcije, LED2 ugasne  //
// test visokega pretoka in končnega stikala KS_ZL //
if ((pretok > ref_zg)&&(PORTB.F7 == 0))
{
// korekcija visokega pretoka - delovanje regulatorja//
PORTB.F3 = 1; // B3 = 1 za zapiranje lopute, LED3 gori  //
Usart_Write(znak); // znak 'R' za delovanje regulacije, izpis na PC //
```

```
}
// reset korekcije nizkega pretoka - mirovanje regulatorja//
else
PORTB.F3 = 0; // B3 = 0 za reset korekcije, LED4 ugasne  //
// test končnega stikala KS_OL //
if (PORTB.F6 == 1)   // tipka na B6  //
// signalizacija končnega stikala KS_OL z LED4 //
PORTB.F4 = 1; // B4 = 1 za signalizacijo KS_OL //
// test končnega stikala KS_ZL //
if (PORTB.F7 == 1)   //  tipka na B7 //
// signalizacija končnega stikala KS_ZL z LED1 //
PORTB.F0 = 1; // B0 = 1 za signalizacijo KS_ZL //


//  3. FUNKCIJA DALJINSKEGA NADZORA  //

// prikaz napisa za trenutni pretok //
for (i=0; i<14; i++)    // izpis "PRETOK m3: " na PC  //
{
Usart_Write(beseda[i]); // beseda = " PRETOK m3: " //
}
Delay_ms(500);
// prikaz številčne vrednosti trenutnega pretoka //
ByteToStr(pretok, podatek);  // številka se pretvori v znakovni niz //
                        // za "char" prikaz števil na PC //

for (i=0; i<5; i++)   // izpis desetiške vrednosti pretoka na PC  //
{
Usart_Write(podatek[i]); // 3 cifrni podatkovni niz  //
}
Delay_ms(500);
// prikaz alarma za vklop končnega stikala (izkrmiljena regulacija) //
if ((PORTB.F6==1)||(PORTB.F7==1)) // KS_OL, ali KS_ZL //
{
for (i=0; i<5; i++)   // izpis znakovnega niza " KS"   //
{
Usart_Write(alarm[i]); // 3 mestni znakovni niz alarm  //
}
}
Delay_ms(1000);


//  4. FUNKCIJA ON-LINE NASTAVITVE PRETOKA //

// vklop funkcije ponastavljanja kontrole pretoka  //
if ((PORTB.F6 == 1)&&(PORTB.F7 == 1)) // hkrati vklop obeh tipk //
{
// čakalna zanka za sprejem in čitanje nastavitve regulacije //
while (!Usart_Data_Ready()) // zanka za čitanje znaka + ali - //
{  // do vnosa znaka kateregakoli znaka regulacija začasno ne deluje //
```

```
Usart_Write(znak_1);   // prompt '?' za vpis + / - inkrementa na PC //
Delay_ms(500);
}
// test prisotnosti sprejetih podatkov po USB iz PC  //
if (Usart_Data_Ready())
{
// branje in shranitev sprejetega podatka za spremembo nastavitve pretoka //
inkrement = Usart_Read();
// test ustreznosti prejetega znaka //
if ((inkrement == '+')||(inkrement == '-'))
{
// test zahteve povišanja in povišanje nastavitve za 1 če je prejeto '+' //
if (inkrement == '+') nastavitev = nastavitev + 1;
// test zahteve znižanja in znižanje nastavitve za 1 če je prejeto '-' //
if (inkrement == '-') nastavitev = nastavitev - 1;
// nova kalkulacija spodnje in zgornje tolerance regulacije //
ref_sp = nastavitev - 15;
ref_zg = nastavitev + 15;
// prikaz številčne vrednosti nove nastavitve pretoka //
ByteToStr(nastavitev, podatek); // številka se pretvori v znakovni niz //
for (i=0; i<5; i++)   // prikaz številčne vrednosti nastavitve na PC //
{
Usart_Write(podatek[i]); // 3 cifrni podatkovni niz  //
}
Delay_ms(500);
}
// v primeru vnosa napačnega znaka program skoči sem //
}
}
// v primeru da ni vnosa znaka program skoči sem //


}
// GLAVNA NEPOGOJNA ZANKA  //

}
// KONEC PROGRAMA //

//////////////////////////////////////////////////////////////
```