**PROGRAMERSKI PRIROČNIK (PETER ZOROJA)**

[JAVA 2](#_Toc26431213)

[OSNOVE IN KONZOLNO PROGRAMIRANJE 2](#_Toc26431214)

[VIZUALNO PROGRAMIRANJE-»DOGODKOVNO« PROGRAMIRANJE 4](#_Toc26431215)

[OSNOVE 4](#_Toc26431216)

[DELO Z MIKROKONTROLERJI IN KNJIŽNICO »PanamaHitek\_Arduino« 5](#_Toc26431217)

[TABELA ASCII KODE 5](#_Toc26431218)

[C++ (ARDUINO) 6](#_Toc26431219)

[OSNOVE 6](#_Toc26431220)

[ČASOVNA PREKINITEV 7](#_Toc26431221)

[ČASOVNIKI (TIMERS) V ARDUINO MODULIH 8](#_Toc26431222)

[Timer0 8](#_Toc26431223)

[//časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 1), interval fp=62500 Hz do 16 MHz 8](#_Toc26431224)

[//časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 8), interval fp=7813 Hz do 2 MHz 8](#_Toc26431225)

[//časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 64), interval fp=977 Hz do 250000 Hz 9](#_Toc26431226)

[//časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 256), interval fp=244 Hz do 62500 Hz 9](#_Toc26431227)

[//časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 1024), interval fp=61 Hz do 15625 Hz 9](#_Toc26431228)

[//prekinitveni program, metoda za timer 0 10](#_Toc26431229)

[//Nastavitev časovne prekinitve timer 1 10](#_Toc26431230)

[//Prekinitvena funkcija timer 1 10](#_Toc26431231)

[//Nastavitev časovne prekinitve timer 2 10](#_Toc26431232)

[//Prekinitvena funkcija timer 2 11](#_Toc26431233)

[//Zakasnitvene metode pri uporabi časovne prekinitve za Arduino uno (empirične meritve) 11](#_Toc26431234)

[SPREMENLJIVKE PRI ARDUINO MODULIH 11](#_Toc26431235)

[ARDUINO UNO TEHNIČNE SPECIFIKACIJE 12](#_Toc26431236)

[ARDUINO NANO TEHNIČNE SPECIFIKACIJE 15](#_Toc26431237)

[FLOWCODE (PROGRAMIRANJE Z DIAGRAMI POTEKA-FLOWCHART) 15](#_Toc26431238)

[PREDSTAVITEV UČNIH MODELOV, NAPRAV 16](#_Toc26431239)

[Didaktični model s PIC mikrokrmilnikom PIC 16F877a ali PIC 16f877 16](#_Toc26431240)

[ROBO MIŠ mobilni robot 18](#_Toc26431241)

[Primeri mobilnih robotov 20](#_Toc26431242)

[ROBO MIŠ metode zakasnitve 22](#_Toc26431243)

# JAVA

## OSNOVE IN KONZOLNO PROGRAMIRANJE

**OSNOVE:**

---------------------------------------------------------------

public static void main(String[] args){

}

---------------------------------------------------------------

//branje tipkovnice

import java.util.\*;

import java.util.Scanner;

Scanner vhod=new Scanner (System.in);

String ime=vhod.next();

String ime=vhod.nextLine();

int a=vhod.nextInt();

Double a=vhod.nextDouble();

//branje znaka preko niza

String znakovni\_niz=vhod.next();

char znak=znakovni\_niz.charAt(0);

//metoda za branje znaka

public static void metoda\_za\_branje\_znaka() {

int dolzina\_znakovnega\_niza;

String znak\_String;

do {

znak\_String = vhod.nextLine();

dolzina\_znakovnega\_niza = znak\_String.length();

if (dolzina\_znakovnega\_niza != 1) {

System.out.println("NAPAKA! VNESI ZNAK!");

}

} while (dolzina\_znakovnega\_niza != 1);

char znak\_char = znak\_String.charAt(0);

}

---------------------------------------------------------------

//izpis

System.out.println();

System.out.print ();

System.out.printf ("%d",10); //izpis celih števil

System.out.printf ("\n"); //skok v novo vrstico

System.out.printf ("%f",20.123456789);//izpis dec. števil s šestimi decimalkami

System.out.printf ("%c",'A');//izpis znaka

System.out.printf ("%C",'a');//izpis samo velikih črk-znak

System.out.printf ("%s","peter");//izpis znakovnega niza

System.out.printf ("%S","peter");//izpis samo velikih črk-znakovni niz

System.out.printf ("%b",4<3);//boolova logika, izpis true/false

System.out.printf ("%B",4>3);//boolova logika, izpis TRUE/FALSE

System.out.printf ("%e",0.003);//eksponentni izpis številke

System.out.printf ("%10s%20s","d "+1,"a");//izpis z zamikom:-10 polj d 1 20 polj

Uporaba metode format za izpis (glej spodnji URL)

<https://docs.oracle.com/javase/tutorial/java/data/numberformat.html>

//izhod iz konzolne aplikacije ob vprašanju

System.out.println("IZHOD IZ PROGRAMA-PRITISNI TIPKO ENTER");

Scanner konec=new Scanner(System.in);

konec.nextLine();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

------------------------------------------------------------

//if struktura

if (ime.equals("LJUBLJANA")){

}else{

}

//if struktura (razširjena struktura)

if (ime.equals("LJUBLJANA")){

}else if(pogoj1){

} else if(pogoj2)

}else{

}

----------------------------------------------------------------------------------

//pogojno logični operator

stevilo=(A>=B) ? "A je večje ali enako B":"B je večje od A";

//switch struktura

switch (spremenljivka) {

case 1:

break;

case 2:

break;

default:

}

--------------------------------------------------------------

//while struktura

while (spremenljivka!=0){

}

--------------------------------------------------------------

//do while struktura

do {

}while(spremenljivka!=0);

--------------------------------------------------------------

//for struktura

for (int stevec=1; stevec<=stevilo; stevec++) {

}

--------------------------------------------------------------

//try..catch struktura, struktura za zaznavanje napake

try {

}

catch(Exception e) {

// del kode, ki se izvede ob napaki

}

--------------------------------------------------------------

//dolžina niza

int dolzina\_niza=znakovni\_niz.length();

//metoda, brez argumentov in ne vrača vrednosti

public static void metoda(){

}

metoda();// klic metode

//metoda, brez argumentov, vrača vrednost

public static int vraca\_vrednost metoda(){

return lokalna\_spremenljivka;

}

metoda();// klic metode

//metoda z argumenti, ne vrača vrednosti

public static void metoda(int lokalna\_spremenljivka){

}

int a=5;

metoda(a);// klic metode

//metoda z argumenti, vrača vrednost

public static int vraca\_vrednost metoda(int lokalna\_spremenljivka\_a){

return lokalna\_spremenljivka\_b;

}

int a=5;

metoda(a);// klic metode

Matematične metode

Math.sqrt(double spremenljivka); //kvadratni koren

Math.pow(double osnova, double eksponent); //metoda za potencirati število

Odpiranje konzolne eksekutivne jar datoteke v CMD-ju (kopiraš direktno iz IDE NetBeans ob generaciji .jar datoteke)

java –jar C:\Users\peterz\Desktop\Konzolne\_aplikacije\dist\Konzolna\_apk.jar

// zakasnitev

Thread.sleep(1000);//zakasnitev 1000ms

TimeUnit.SECONDS.sleep(3);

//Aritmetični operatorji: (rezultat je število)

|  |  |
| --- | --- |
| Operator | Namen |
| + | vsota števil, konkatenacija nizov (String-ov), pretvorba celoštevilskega števila v int |
| += | seštej in priredi vsoto, stakni in priredi niz (String) |
| - | razlika števil , aritmetična negacija števila |
| -= | odštej in priredi razliko |
| \* | zmnožek števil |
| \*= | zmnoži in priredi |
| / | količnik števil |
| /= | deli in priredi (divide and assign) |
| % | določi ostanek pri celoštevilskem deljenju (take remainder) |
| %= | določi ostanek pri celoštevilskem deljenju in priredi (take remainder and assign) |
| ++ | povečaj za 1 (increment by one) |
| -- | zmanjšaj za 1 (decrement by one) |

//Relacijski operatorji (rezultat je logična vrednost)

|  |  |
| --- | --- |
| Operator | Namen |
| > | večje kot (greater than) |
| >= | večje ali enako kot (greater than or equal to) |
| < | manjše kot (less than) |
| <= | manjše ali enako kot(less than or equal to) |
| == | test enakosti |
| != | test različnosti |

//Logični operatorji

|  |  |
| --- | --- |
| Operator | Namen |
| ! | logična negacija (boolean NOT) |
| != | logična neenakost (not equal to) |
| && | logični IN (boolean AND) |
| || | logični ALI (boolean OR) |
| == | logična enakost (boolean equals) |

//Bitni operatorji

|  |  |
| --- | --- |
| Operator | Namen |
| ~ | bitna negacija oz. eniški komplement (bitwise NOT) |
| | | bitni ALI (bitwise OR) |
| |= | bitni ALI in prirejanje (bitwise OR and assign) |
| ^ | bitni ekskluzivni ALI (bitwise XOR) |
| ^= | bitni ekskluzivni ALI in prirejanje (bitwise XOR and assign) |
| & | bitni IN (bitwise AND) |
| &= | bitni IN in prirejanje (bitwise AND and assign) |
| >> | pomik bitov v desno z upoštevanjem predznaka (shift bits right with sign extension) |
| >>= | pomik bitov v desno z upoštevanjem predznaka in prirejanje (shift bits right with sign extension and assign) |
| << | pomik bitov levo (shift bits left) |
| <<= | pomik bitov levo in prirejanje (shift bits left and assign) |
| >>> | pomik bitov desno brez upoštevanja predznaka (unsigned bit shift right) |
| >>>= | pomik bitov desno brez upoštevanja predznaka in prireditev (unsigned bit shift right and assign) |

//Nekategorizirani operatorji

|  |  |
| --- | --- |
| = | prirejanje (assignment) |
| ?: | pogojni operator (conditional) |
| . | operator dosega (selekcijski operator, selection) |
| :: | operator dosega (selekcijski operator, selection) |
| ( ) | vsiljena pretvorba tipa (cast), združevanje v aritmetičnih izrazih |
| [] | deklaracija tabele, kreiranje tabele, indeksiranje el. tabele |
| new | kreira nov objekt ali tabelo elementov |
| instanceof | preverjanje tipov (type checking) |

//tabele

String[] del\_dn\_v\_tednu={"pon","tor","sre","čet","pet"};

//ugotavljanje koliko elementov ima tabela

int velikost\_tabele= del\_dn\_v\_tednu.length;

//Deklaracija tabele:  
int[] tab;  
String[] tab1;  
double[] tab2;  
char[] tab3;

//Rezervacija prostora za tabelo v pomnilniku:  
tab=new int[10];  
tab1=new String[10];  
tab2=new double[10];  
tab3=new char[10];

//Deklarcaija z rezervacijo:  
int[] tab=new int[10];  
String[] tab1=new String[10];  
double[] tab2=new double[10];  
char[] tab3=new char[10];

//Naključno generirano število na intervalu [a,b]  
int s=(int)(Math.random()\*(b-a+1)+a);

// iz ascii v karakter oz. znak

c=(char)tab[i];

// 2D tabele

int a=0;

int[][] tabela = new int[2][2];

for (int vrstica = 0; vrstica < stevilo\_elementov\_vrstice; ++ vrstica) {

for (int stolpec = 0; vrstica < stevilo\_elementov\_stolpca; ++ stolpec) {

}

}

while (st\_tab!=tabela.length){

System.out.println("element v diagonali: "+ tabela[st\_tab][st\_tab]);

st\_tab++;

}

//iz znaka v niz

niz=String.valueOf(znak);

//ascii koda znaka

int ascii = (char) znak;

//metoda endsWith

String ime="PETER";

boolean odgovor;

odgovor=ime.endsWith("a");//true ali false

//iz znakovnega niza v tabelo znakov

char[] tabela = znakovni niz.toCharArray();

//metoda Character.isDigit

boolean odgovor = Character.isDigit(znak);// odgovor je true, če je znak število

**//izpis v dokument PrintStream**(<https://www.tutorialspoint.com/java/io/java_io_printstream.htm>, <https://docs.oracle.com/javase/7/docs/api/java/io/PrintStream.html>)

PrintStream dokument = new PrintStream(new File(ime\_dokumenta + ".txt"));//ustvarjanje dokumenta

System.setOut(dokument); //začenja se izpis v dokument

PrintStream konzola = new PrintStream((OutputStream)(System.out));//objekt konzolnega okna

System.setOut(konzola); //začenja se izpis v konzolo

**//izpis v dokument File, FileWriter, PrintWriter**

File datoteka = new File("a.txt");//poimenuje datoteko

FileWriter fw = new FileWriter(datoteka, true);//ustvari datoteko

PrintWriter pw = new PrintWriter(fw);//objekt za pisanje v datoteko

pw.println("podatek");//izpis

pw.close();//aktivira izpis

**// brisanje\_konzole\_in\_povrnitev\_na\_vrh**

new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();

**//pretvorbe binarni, desetiški zapis**

byte **desetiski** = (byte) 0b00000111;

**//iz desetiškega v binarno**

String **binarni**=Integer.toBinaryString((**desetiski** & 0xFF) + 0x100).substring(1);

**// iz binarnega v desetiško**

byte **iz\_bin\_v\_des**=Byte.parseByte(**binarni**, 2);

**//odpiranje slik, datotek iz projektne mape-relativno naslavljanje ./ Opomba: ob generaciji .exe datoteke morajo biti slike oz. drugi dokumenti kopirane v isto mapo kjer se nahaja .exe datoteka**

File file = new File("**./**PZ2018.jpg");

Desktop povezava = Desktop.getDesktop();

povezava.open(file);

## VIZUALNO PROGRAMIRANJE-»DOGODKOVNO« PROGRAMIRANJE

### OSNOVE

**//ČASOVNA PREKINITEV**

//knjižnice

import java.util.Date;

import java.util.TimerTask;

//objekt

java.util.Timer casovnik = new java.util.Timer();

//metoda

private void casovna\_prekinitev(long zakasnitev) {

// long zakasnitev perioda v ms

casovnik.scheduleAtFixedRate(new TimerTask() {

public void run() {

//vsebina v prekinitvi

}

}, new Date(), zakasnitev);

}

//V. P. BRANJE VNOSNEGA POLJA jTextField

a=Double.parseDouble(jTextField.getText());

b=Integer.parseInt(jTextField.getText());// pretvorba znakovnega niza v celoštevilčno spremenljivko

ime\_priimek=jTextField.getText(); //BRANJE ZNAKOVNEGA NIZA

tab[i]=jTextField.getText();

//V. P. IZPIS V LABELO

jLabel.setText(""+c);

jLabel.setText(tab[0]+" "+tab[1]+" "+tab[2]);

//V. P. IZPIS V jTextArea

jTextArea.setText(""+c);

jTextArea.setText(tab[0]+" "+tab[1]+" "+tab[2]);

jTextArea.append("abc"+"\n"); //izpis se nadaljuje v novi vrstici, "\n"-skok v novo vrstico

//PRETVORBA SPREMENLJIVKE char V String

jTextArea1.append(String.valueOf('a'));

//V. P. NASTAVITEV jTextArea ELEMENTA ZA PRIKAZ BESEDILA BREZ MOŽNOSTI UREJANJA

jTextArea1.setEditable(false);

jTextArea1.setCursor(null);

//V. P. SPREMINJANJE IKONE OKNA

import java.awt.Toolkit;//vnos knjižnice razredov

setIcon();//KLIC METODE setIcon

private void setIcon() {

setIconImage(Toolkit.getDefaultToolkit().getImage(getClass().getResource("cube.jpg")));

}

//V. P ODPIRANJE NOVEGA OKNA

Okno2 a = new Okno2();

a.setVisible(true);

//V. P METODA ZA ZAPIRANJE OKNA

import java.awt.Toolkit;

import java.awt.event.WindowEvent;

private void close() {

WindowEvent oknoExit=new WindowEvent(this,WindowEvent.WINDOW\_CLOSING);

Toolkit.getDefaultToolkit().getSystemEventQueue().postEvent(oknoExit);

}

//PRIKAZ SLIKE V LABELI

jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource(tabela\_slik[i])));

//PRIKAZ LABELE

jLabel.setVisible(true);

//SPREMINJANJE BARVE ELEMENTA label

import java.awt.Color;

jLabel1.setBackground(Color.red);

jLabel1.setOpaque(true);

//zagon brskalnika in odpiranje spletne strani v brskalniku ALI ODPIRANJE MAPE NA pc-JU

Desktop povezava = Desktop.getDesktop();

povezava.browse(new URI("http://www.trenitalia.com/"));

//ZAGON PROGRAMA NA PC-JU

Process p = Runtime.getRuntime().exec("C:/Program Files/NetBeans 8.2/bin/netbeans.exe");

//ZAGON PROGRAMA CMD NA PC-JU

Process p = Runtime.getRuntime().exec("cmd.exe /c start cmd");

//DIMENZIJA OKNA ČEZ CEL EKRAN

Toolkit dimenzija=Toolkit.getDefaultToolkit();

int x=(int) dimenzija.getScreenSize().getWidth();//prebere dimenzijo ekrana v x osi

public Okno() {

initComponents();

this.setSize(x,300);//aktivira dimenzijo

}

//BRANJE PODATKA TIPKE NA TIPKOVNICI

znak\_tipke = evt.getKeyChar();

//PRESLIKAVA INFORMACIJE IZ BYTE SPREMENLJIVKE V BITNE, BOOLEAN SPREMENLJIVKE

byte[] tabela = {(byte) 0b10000000, (byte) 0b01000000, (byte) 0b00100000, (byte) 0b00010000, (byte) 0b00001000, (byte) 0b00000100, (byte) 0b00000010, (byte) 0b00000001};

byte stevec = 0;

boolean[] biti = new boolean[8];//DEFINICIJA 8-MIH BITOV-TABELA

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//PRESLIKAVA BYTE V BITE

biti[7] = ((tabela[stevec] & (byte) 0b10000000) != 0);

biti[6] = ((tabela[stevec] & (byte) 0b01000000) != 0);

biti[5] = ((tabela[stevec] & (byte) 0b00100000) != 0);

biti[4] = ((tabela[stevec] & (byte) 0b00010000) != 0);

biti[3] = ((tabela[stevec] & (byte) 0b00001000) != 0);

biti[2] = ((tabela[stevec] & (byte) 0b00000100) != 0);

biti[1] = ((tabela[stevec] & (byte) 0b00000010) != 0);

biti[0] = ((tabela[stevec] & (byte) 0b00000001) != 0);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//PRIKAZ TRUE/FALSE V LABELI

b7.setText(String.valueOf(biti[7]));

b6.setText(String.valueOf(biti[6]));

b5.setText(String.valueOf(biti[5]));

b4.setText(String.valueOf(biti[4]));

b3.setText(String.valueOf(biti[3]));

b2.setText(String.valueOf(biti[2]));

b1.setText(String.valueOf(biti[1]));

b0.setText(String.valueOf(biti[0]));

### DELO Z MIKROKONTROLERJI IN KNJIŽNICO »PanamaHitek\_Arduino«

PanamaHitek\_Arduino objekt = new PanamaHitek\_Arduino(); //objekt razreda PanamaHitek\_Arduino

objekt.sendData("1"); // pošiljanje znakovnega niza

objekt.sendByte(0b00100100); //pošiljanje byte spremenljivke

objekt.arduinoTX("COM3", 9600); //vzpostavitev serijske komunikacije z mikrokrmilnikom in PC-jem za pošiljanje

//poslušanje informacij na serijskem COM vhodu oz. sprejemanje podatkov, objekt poslušalec definiraš kot globalni objekt oz **takoj pod razredom!**

SerialPortEventListener **poslusalec** = new SerialPortEventListener() {

@Override

public void serialEvent(SerialPortEvent serialPortEvent) {

try {

//če je podatek prisoten

if (objekt.isMessageAvailable()) {

podatek = objekt.printMessage(); //podatek se prebere, tipa znakovni niz

podatek\_byte = Byte.parseByte(podatek); //primer pretvorbe v byte

//in tukaj naprej pišeš kaj naj program naredi glede na dospeli podatek

}

} catch (SerialPortException | ArduinoException ex) {

Logger.getLogger(**Ime\_razreda**.class.getName()).log(Level.SEVERE, null, ex);

}

}

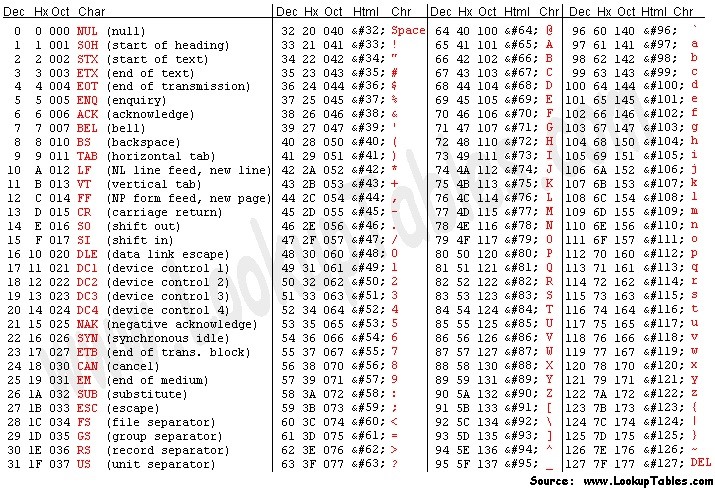
};

//konec tega »eventa«

//aktiviranje branja, pošiljanja podatka oz. zgornjega »eventa«

objekt.arduinoRXTX("COM3", 9600, **poslusalec**);//vzpostavitev komunikacije za pošiljanje in branje

### TABELA ASCII KODE



# C++ (ARDUINO)

## OSNOVE

//struktura skicirke

void setup() {

}

void loop() {

}

//if struktura

if (pogoj) {

}else{

}

//if..else if struktura

if (pogoj1){

}else if(pogoj 2){

} else if(pogoj 3){

else{

}

//do while struktura

do {

}while(spremenljivka!=0);

//while struktura

while (spremenljivka!=0){

}

//switch struktura

switch (spremenljivka) {

case 1:

break;

default:

}

//for struktura

for (int stevec=1; stevec<=stevilo; stevec++) {

}

// spremenljivke

int a = 0; // lahko definiramo vrednost spremenljivki (int-celoštevilčna)

int a, b, c; // brez definiranih vrednosti spremenljivk

int led2\_n = 6;

byte a=0; //lahko definiramo vrednost spremenljivki (byte-celoštevilčna)

boolean a; //enobitna spremenljivka (true/false oz. 1/0)

double a;//spremenljivka za uporabo decimalnih števil

String a="tekst"; // znakovna spremenljivka

byte tab\_cifre[]={0b01001000,0b11111001};//tabelarična spremenljivka

// definicija vhodov, izhodov

DDRD=DDRD|0b11110000; // kot izhodi Port D7-D4, kot vhodi Port D3-D0

pinMode(tipka\_n,INPUT); //vhod

pinMode(tipka\_n,OUTPUT); //izhod

//branje vhodov, postavitev izhodov

tipka=digitalRead(tipka\_n); //digitalno branje vhoda

tipke=PINC //branje celotnega porta C

digitalWrite(ledPin, HIGH); //digitalna postavitev stanja na izhod

//večbitni podatek na port-ih

PORTD=0b10101100;//postavitev izhodov na binarno stanje

//zakasnitev

delay(1000); //zakasnitev v ms

Aritmetični operatorji

= (enačaj)

+ (plus)

- (minus)

\* (krat)

/ (deljeno)

% (mod funkcija)

//Primerjalni operatorji

== (je enako)

!= (ni enako)

< (manjše)

> (večje)

<= (manjše ali enako)

>= (večje ali enako)

Boolean operatorji

&& (in)

|| (ali)

! (negator)

Bitni operatorji

& (bitni in)

| (bitni ali)

^ (bitni xor)

~ (bitni negator)

<< (bitni pomik v levo)

>> (bitni pomik v desno)

//Delo z LCD knjižnico, razredom

#include <LiquidCrystal.h>//uvoz razreda

//LiquidCrystal(rs, enable, d4, d5, d6, d7) , def. prključkov

LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

lcd.begin(16, 2);//dimenzije LCD prikazovalnika

lcd.setCursor(5, 0);//def. Pozicije izpisa

lcd.print("VREDNOST: ");//izpis znakovnega niza

*lcd*.print(spremenljivka); //izpis spremenljivke

*lcd*.print(data,BASE);//izpis spre, BASE JE LAHKO 2,10,8 ali 16

lcd.display();//prikaže izpis

lcd.noDisplay();//skrije izpis

//analogno branje vhodov

an\_vrednost=analogRead(nozica);//analogno branje

## ČASOVNA PREKINITEV

//spremenljivke v prekinitvenem delu programa

Spremenljivke, ki nastopajo v prekinitvenem delu programa imajo pri definiciji tipa spremenljivke dodano rezervirano besedo volatile

(Primer: volatile int spremenljivka=3;)

ČASOVNE PREKINITVE-priporočilo: NASTAVLJAJ FREKVENCE PRI **SPODNJIH MEJAH INTERVALA (empirična, praktična izkušnja)**!!!

Kaj moramo paziti kadar uporabljamo prekinitve: [link](https://www.robotshop.com/community/forum/t/arduino-101-timers-and-interrupts/13072)

### ČASOVNIKI (TIMERS) V ARDUINO MODULIH

Timer0  
Timer0 is a 8bit timer.  
In the Arduino world timer0 is been used for the timer functions, like [delay()](http://arduino.cc/en/Reference/Delay?utm_source=rb-community&utm_medium=forum&utm_campaign=arduino-101-timers-and-interrupts" \t "_blank), [millis()](http://arduino.cc/en/Reference/Millis?utm_source=rb-community&utm_medium=forum&utm_campaign=arduino-101-timers-and-interrupts) and[micros()](http://arduino.cc/en/Reference/Micros?utm_source=rb-community&utm_medium=forum&utm_campaign=arduino-101-timers-and-interrupts" \t "_blank). If you change timer0 registers, this may influence the Arduino timer function. So you should know what you are doing.  
  
Timer1  
Timer1 is a 16bit timer.  
In the Arduino world the [Servo library](http://arduino.cc/en/Reference/Servo?utm_source=rb-community&utm_medium=forum&utm_campaign=arduino-101-timers-and-interrupts" \t "_blank) uses timer1 on Arduino Uno (timer5 on Arduino Mega).  
  
Timer2  
Timer2 is a 8bit timer like timer0.  
In the Arduino work the [tone()](http://arduino.cc/en/Reference/Tone?utm_source=rb-community&utm_medium=forum&utm_campaign=arduino-101-timers-and-interrupts) function uses timer2.  
  
Timer3, Timer4, Timer5  
Timer 3,4,5 are only available on Arduino Mega boards. These timers are all 16bit timers.

### //časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 1), interval fp=62500 Hz do 16 MHz

//**maksimalna vrednost unsigned int= 65535**

void nastavitev\_prekinitve\_timer0\_prescaler\_1(unsigned int fp) {

cli();//ustavitev vseh prekinitev

TCCR0A = 0;// nastavitev začetne vrednosti registra TCCR0A

TCCR0B = 0;// nastavitev začetne vrednosti registra TCCR0B

TCNT0 = 0;//inicializacija števca TCNT0

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR0A+1))

OCR0A = ((16\*10^6)/(fp))-1; // OCR0A= ((16\*10^6) / (fp)) - 1 (vrednost mora biti manjša od <256)…………. fp=977 Hz

//CTC način delovanja

TCCR0A |= (1 << WGM01);

//CS02=0, CS01=0, CS00=1 (prescaler=1)

//CS02=0, CS01=1, CS00=0 (prescaler=8)

//CS02=0, CS01=1, CS00=1 (prescaler=64)

//CS02=1, CS01=0, CS00=0 (prescaler=256)

//CS02=1, CS01=0, CS00=1 (prescaler=1024)

TCCR0B |= (0 << CS02);

TCCR0B |= (0 << CS01);

TCCR0B |= (1 << CS00);

// omogoči prekinitev na timer0

TIMSK0 |= (1 << OCIE0A);

sei();//aktivacija vseh prekinitev

}

### //časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 8), interval fp=7813 Hz do 2 MHz

//**maksimalna vrednost unsigned int= 65535**

void nastavitev\_prekinitve\_timer0\_prescaler\_8(unsigned int fp) {

cli();//ustavitev vseh prekinitev

TCCR0A = 0;// nastavitev začetne vrednosti registra TCCR0A

TCCR0B = 0;// nastavitev začetne vrednosti registra TCCR0B

TCNT0 = 0;//inicializacija števca TCNT0

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR0A+1))

OCR0A = ((16\*10^6)/(fp\*8))-1; // OCR0A= ((16\*10^6) / (fp\*8)) - 1 (vrednost mora biti manjša od <256)…………. fp=977 Hz

//CTC način delovanja

TCCR0A |= (1 << WGM01);

//CS02=0, CS01=0, CS00=1 (prescaler=1)

//CS02=0, CS01=1, CS00=0 (prescaler=8)

//CS02=0, CS01=1, CS00=1 (prescaler=64)

//CS02=1, CS01=0, CS00=0 (prescaler=256)

//CS02=1, CS01=0, CS00=1 (prescaler=1024)

TCCR0B |= (0 << CS02);

TCCR0B |= (1 << CS01);

TCCR0B |= (0 << CS00);

// omogoči prekinitev na timer0

TIMSK0 |= (1 << OCIE0A);

sei();//aktivacija vseh prekinitev

}

### //časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 64), interval fp=977 Hz do 250000 Hz

//**maksimalna vrednost unsigned int= 65535**

void nastavitev\_prekinitve\_timer0\_prescaler\_64(unsigned int fp) {

cli();//ustavitev vseh prekinitev

TCCR0A = 0;// nastavitev začetne vrednosti registra TCCR0A

TCCR0B = 0;// nastavitev začetne vrednosti registra TCCR0B

TCNT0 = 0;//inicializacija števca TCNT0

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR0A+1))

OCR0A = ((16\*10^6)/(fp\*64))-1;// OCR0A= ((16\*10^6) / (fp\*64)) - 1 (vrednost mora biti manjša od <256)…………. fp=977 Hz

//CTC način delovanja

TCCR0A |= (1 << WGM01);

//CS02=0, CS01=0, CS00=1 (prescaler=1)

//CS02=0, CS01=1, CS00=0 (prescaler=8)

//CS02=0, CS01=1, CS00=1 (prescaler=64)

//CS02=1, CS01=0, CS00=0 (prescaler=256)

//CS02=1, CS01=0, CS00=1 (prescaler=1024)

TCCR0B |= (0 << CS02);

TCCR0B |= (1 << CS01);

TCCR0B |= (1 << CS00);

// omogoči prekinitev na timer0

TIMSK0 |= (1 << OCIE0A);

sei();//aktivacija vseh prekinitev

}

### //časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 256), interval fp=244 Hz do 62500 Hz

//**maksimalna vrednost unsigned int= 65535**

void nastavitev\_prekinitve\_timer0\_prescaler\_256(unsigned int fp) {

cli();//ustavitev vseh prekinitev

TCCR0A = 0;// nastavitev začetne vrednosti registra TCCR0A

TCCR0B = 0;// nastavitev začetne vrednosti registra TCCR0B

TCNT0 = 0;//inicializacija števca TCNT0

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR0A+1)

OCR0A = ((16\*10^6)/(fp\*256))-1;

//CTC način delovanja

TCCR0A |= (1 << WGM01);

//CS02=0, CS01=0, CS00=1 (prescaler=1)

//CS02=0, CS01=1, CS00=0 (prescaler=8)

//CS02=0, CS01=1, CS00=1 (prescaler=64)

//CS02=1, CS01=0, CS00=0 (prescaler=256)

//CS02=1, CS01=0, CS00=1 (prescaler=1024)

TCCR0B |= (1 << CS02);

TCCR0B |= (0 << CS01);

TCCR0B |= (0 << CS00);

// omogoči prekinitev na timer0

TIMSK0 |= (1 << OCIE0A);

sei();//aktivacija vseh prekinitev

}

### //časovna prekinitev s timer 0 registrom-metoda nastavitve prekinitve z argumentom (preskaler 1024), interval fp=61 Hz do 15625 Hz

//**maksimalna vrednost int= 32767**

void nastavitev\_prekinitve\_timer0\_prescaler\_1024(int fp) {

cli();//ustavitev vseh prekinitev

TCCR0A = 0;// nastavitev začetne vrednosti registra TCCR0A

TCCR0B = 0;// nastavitev začetne vrednosti registra TCCR0B

TCNT0 = 0;//inicializacija števca TCNT0

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR0A+1)

OCR0A = ((16\*10^6)/(fp\*1024))-1;

//CTC način delovanja

TCCR0A |= (1 << WGM01);

//CS02=0, CS01=0, CS00=1 (prescaler=1)

//CS02=0, CS01=1, CS00=0 (prescaler=8)

//CS02=0, CS01=1, CS00=1 (prescaler=64)

//CS02=1, CS01=0, CS00=0 (prescaler=256)

//CS02=1, CS01=0, CS00=1 (prescaler=1024)

TCCR0B |= (1 << CS02);

TCCR0B |= (0 << CS01);

TCCR0B |= (1 << CS00);

// omogoči prekinitev na timer0

TIMSK0 |= (1 << OCIE0A);

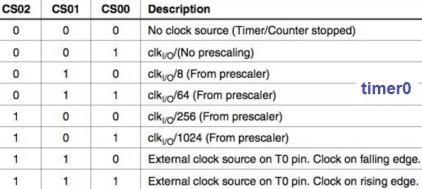
sei();//aktivacija vseh prekinitev

}

### //prekinitveni program, metoda za timer 0

ISR(TIMER0\_COMPA\_vect) {

}



Slika 1 Nastavitve preskalerja timer0

### //Nastavitev časovne prekinitve timer 1

void nastavitev\_prekinitve\_timer1(){

noInterrupts();// onemogoči vse prekinitve

TCCR1A = 0;//Timer1/Counter1 kontrolni register A začetna vrednost

TCCR1B = 0;//Timer1/Counter1 kontrolni register B začetna vrednost

TCNT1 = 0;// inicializacija timer-ja

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR1A+1))

OCR1A = 15624;// OCR1A=(16\*10^6/(fp\*prescaler))-1(za timer0 in timer2 mora vrednos biti <256 za timer1 pa <65536)

//CTC način delovanja

TCCR1B |= (1 << WGM12);

//CS12=0, CS11=0, CS10=1 (prescaler=1)

//CS12=0, CS11=1, CS10=0 (prescaler=8)

//CS12=0, CS11=1, CS10=1 (prescaler=64)

//CS12=1, CS11=0, CS10=0 (prescaler=256)

//CS12=1, CS11=0, CS10=1 (prescaler=1024)

TCCR1B |= (1 << CS10);

TCCR1B |= (0 << CS11);

TCCR1B |= (1 << CS12);

TIMSK1 |= (1 << OCIE1A);

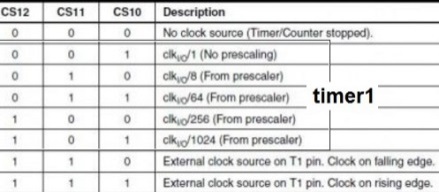
interrupts();// omogoči vse prekinitve

}

### //Prekinitvena funkcija timer 1

ISR(TIMER1\_COMPA\_vect){

}



Slika 2 Nastavitev preskalerja timer1

### //Nastavitev časovne prekinitve timer 2

void nastavitev\_prekinitve\_timer2() {

cli();//ustavitev vseh prekinitev

TCCR2A = 0;//nastavitev registra TCCR2A na začetna vrednost

TCCR2B = 0; //nastavitev registra TCCR2B na začetna vrednost

TCNT2 = 0; // inicializacija timer-ja

// nastavitev compare match registra za željeno frekvenco prekinitve fp=16\*10^6/(prescaler\*(OCR2A+1))

OCR2A = 249;// = ((16\*10^6) / (fp\*prescaler)) - 1 (OCR2A <256)

//CTC način delovanja

TCCR2A |= (1 << WGM21);

//CS22=0, CS21=0, CS20=1 (prescaler=1)

//CS22=0, CS21=1, CS20=0 (prescaler=8)

//CS22=0, CS21=1, CS20=1 (prescaler=32)

//CS22=1, CS21=0, CS20=0 (prescaler=64)

TCCR2B |= (0 << CS22);

TCCR2B |= (1 << CS21);

TCCR2B |= (0 << CS20);

// omogoči prekinitev na timer2

TIMSK2 |= (1 << OCIE2A);

sei();//aktivacija vseh prekinitev

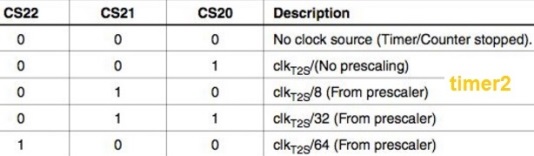
}

### //Prekinitvena funkcija timer 2

ISR(TIMER2\_COMPA\_vect){

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



Slika 3 Nastavitev preskalerja timer2

### //Zakasnitvene metode pri uporabi časovne prekinitve za Arduino uno (empirične meritve)

Spremenljivke, ki nastopajo v FOR ZANKI imajo pri definiciji tipa spremenljivke dodano rezervirano besedo volatile! To moramo dodati zaradi »pametnega prevajalnika« iz c++ kode v .hex. Prevajalnik namreč zanke ne upošteva, če ta nima instrukcij, ukazov v svoji strukturi. Če spremenljivko označimo kot volatile pa preslepimo prevajalnik.

void ena\_mili\_sek\_UNO() {

for (volatile int i = 0; i < 832; i++);

}

//\*\*\*\*\*\*\*\*\*\*\*

void mili\_sek\_UNO(volatile unsigned int ms){

for (volatile unsigned int i = 0; i < ms; i++){

for (volatile int i = 0; i < 832; i++);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*

void sek\_UNO(volatile unsigned int s){

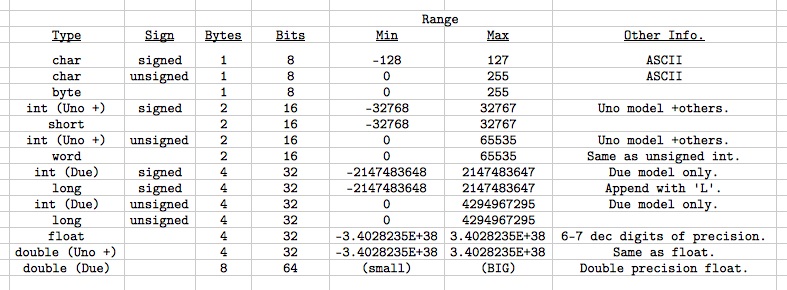
for (volatile unsigned int i = 0; i < s; i++){

mili\_sek\_UNO(1000);

}

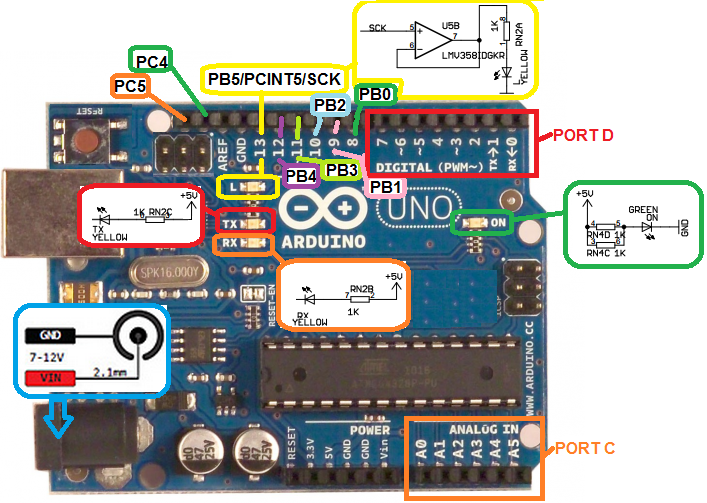
}

## SPREMENLJIVKE PRI ARDUINO MODULIH

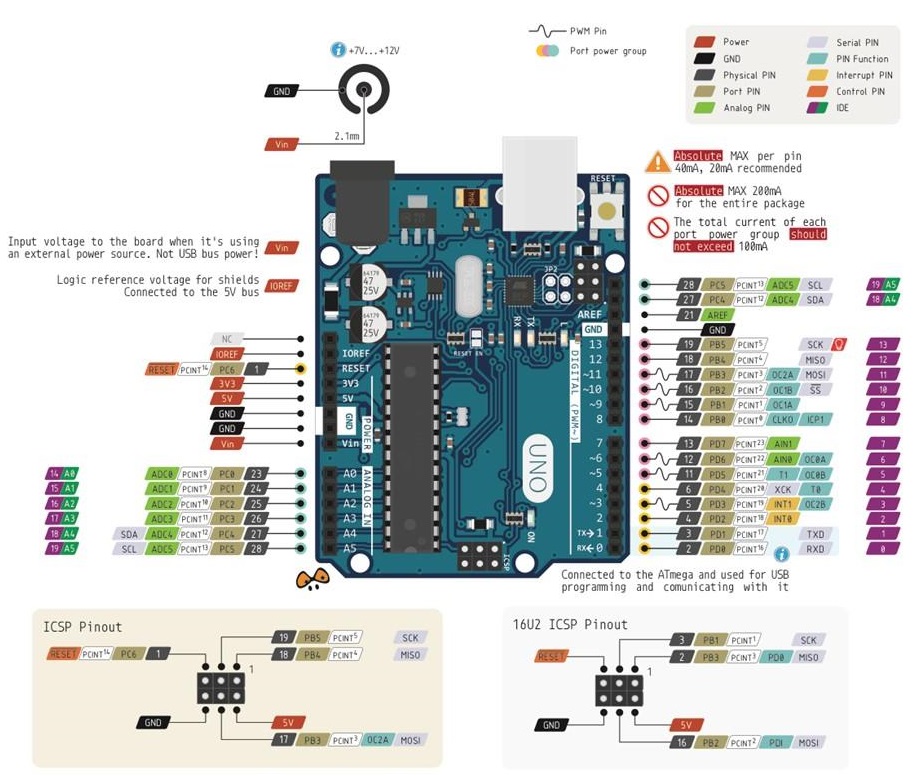


Slika Tipi spremenljivk, Arduino

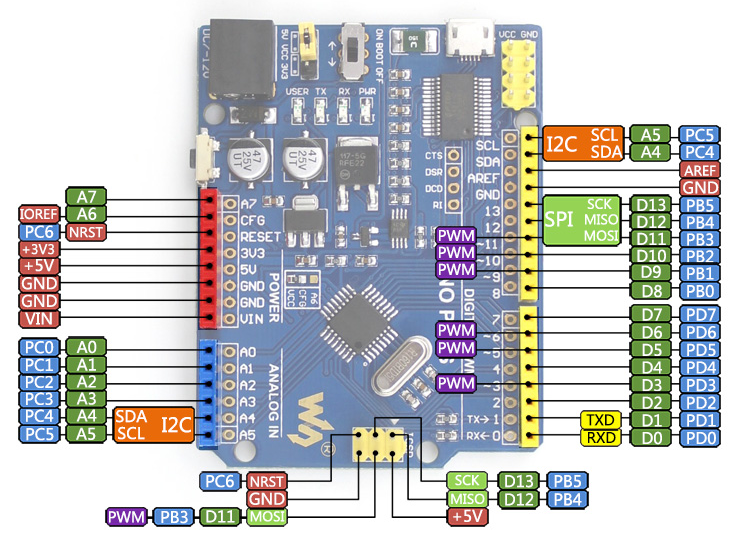
## ARDUINO UNO TEHNIČNE SPECIFIKACIJE



Slika Arduino Uno (primer slike 1)



Slika Arduino Uno (primer slike 2)

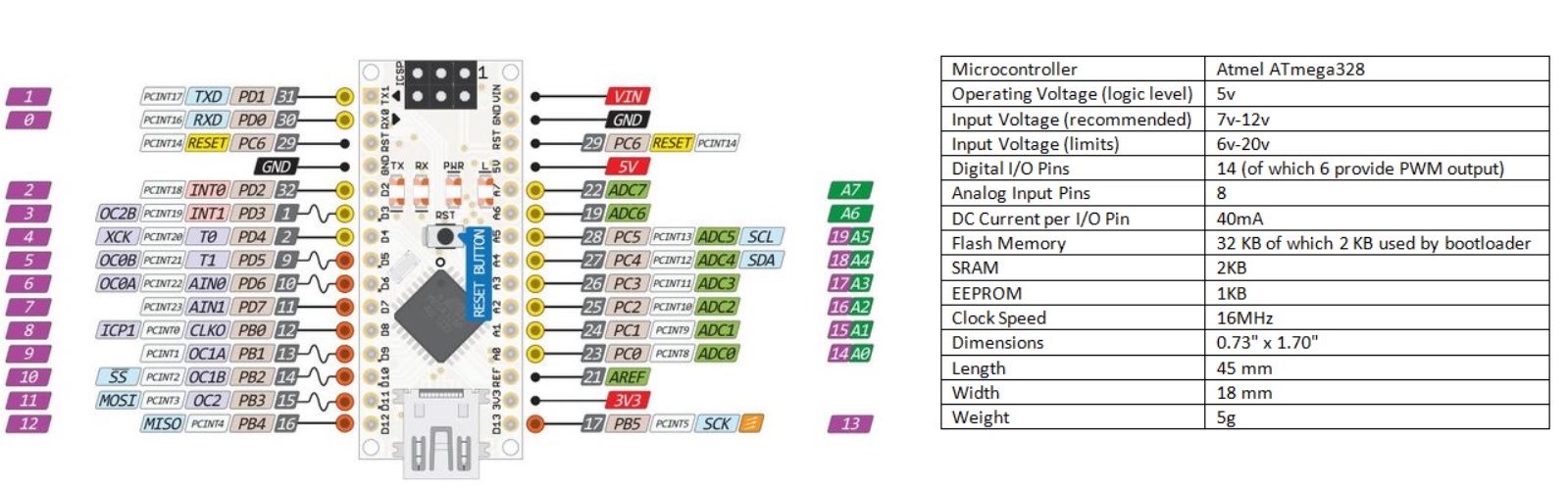


Slika Arduino Uno (primer slike 3)



Slika Karakteristike Arduino uno

## ARDUINO NANO TEHNIČNE SPECIFIKACIJE



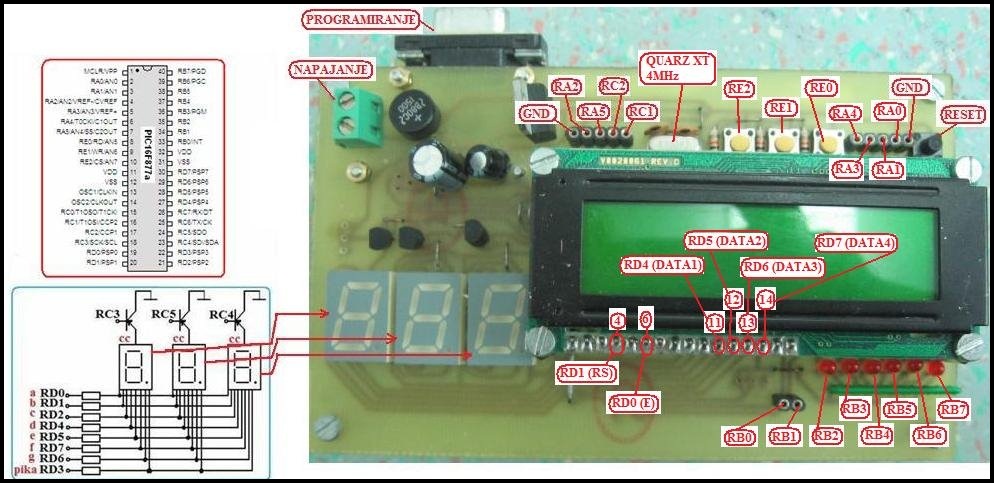
Slika Arduino nano

# FLOWCODE (PROGRAMIRANJE Z DIAGRAMI POTEKA-FLOWCHART)

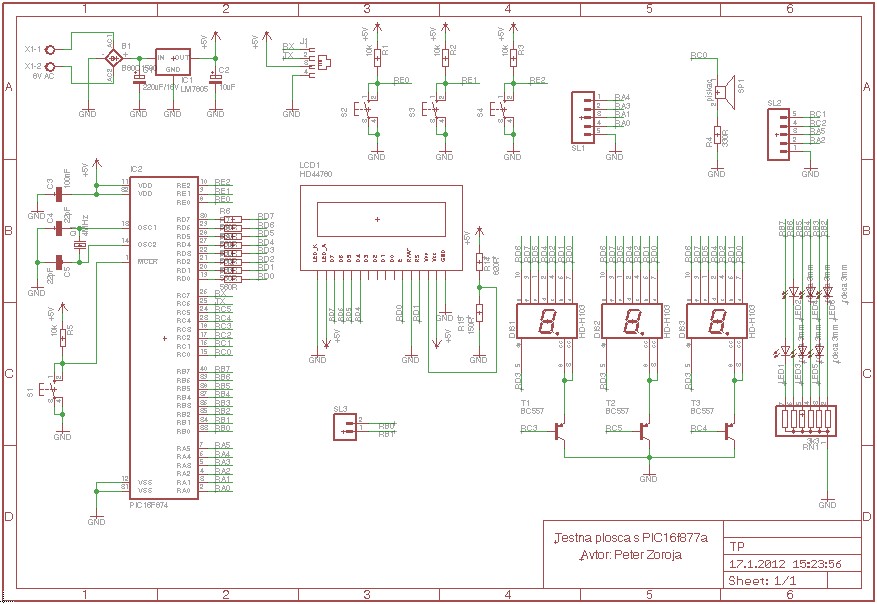
FLOWCODE je integrirano razvojno okolje (IDE-integrated development environment), ki omogoča vizualno programiranje mikrokrmilnikov in nekaterih drugih programljivih naprav blokovno z diagrami poteka oz. dopušča tudi programiranje v »tekstovni« obliki« z različnimi programskimi jeziki (link ponudnika: [**https://www.matrixtsl.com/flowcode/**](https://www.matrixtsl.com/flowcode/)).

## PREDSTAVITEV UČNIH MODELOV, NAPRAV

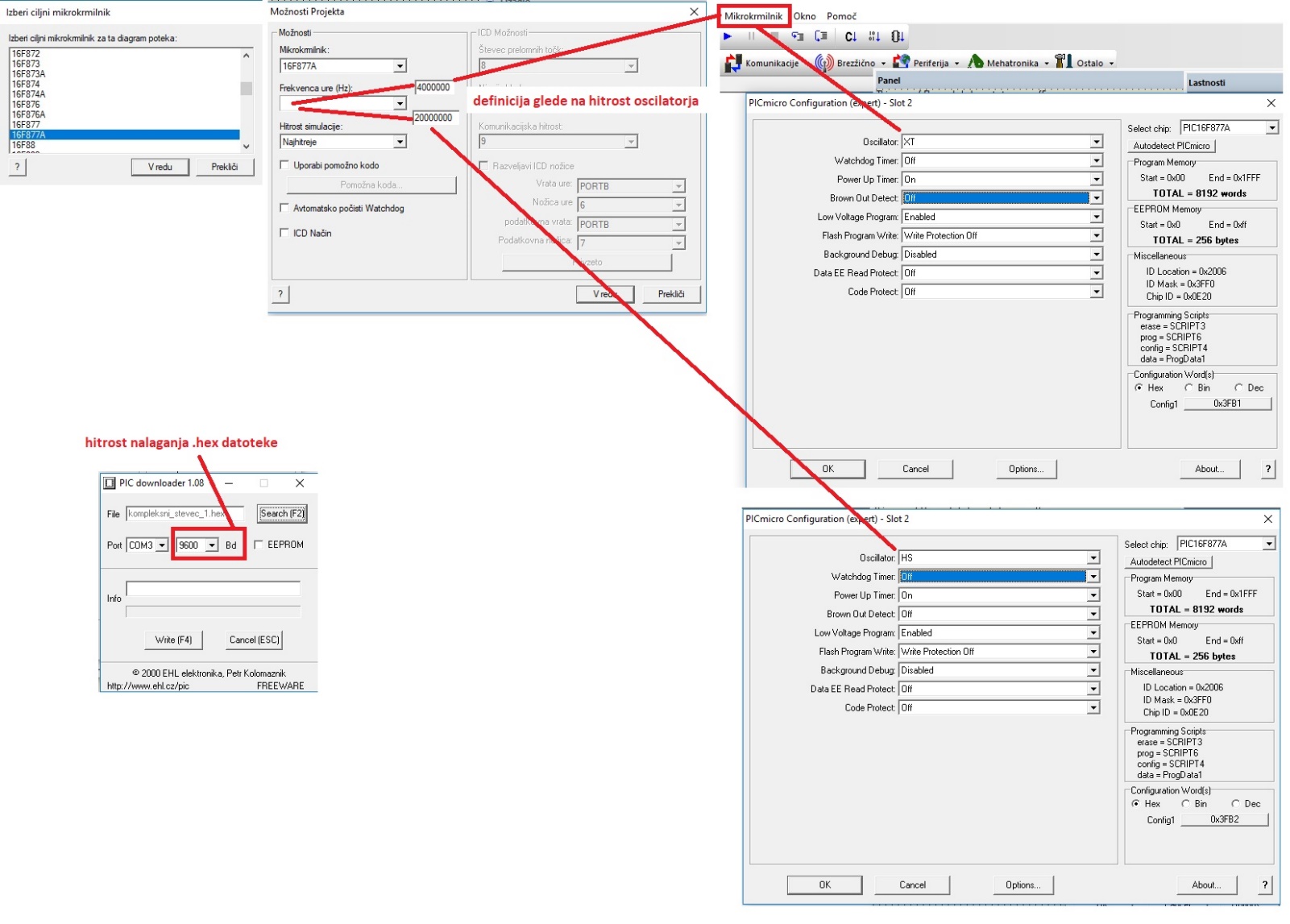
### Didaktični model s PIC mikrokrmilnikom PIC 16F877a ali PIC 16f877



Slika Testno vezja z mikrokrmilnikom PIC16f877a oz. s PIC16f877 (v vezju je lahko dan KVARC 4MHz ali KVARZ 20MHz)



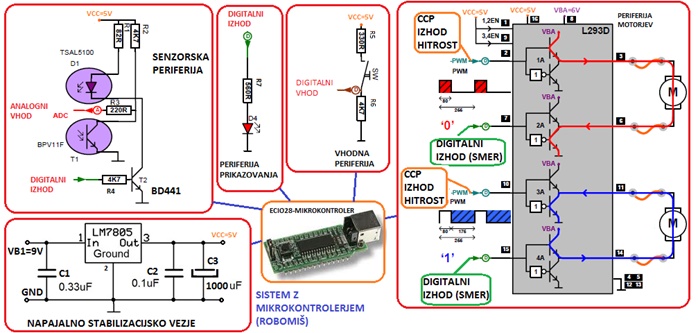
Slika Vezalna shema didaktičnega modela



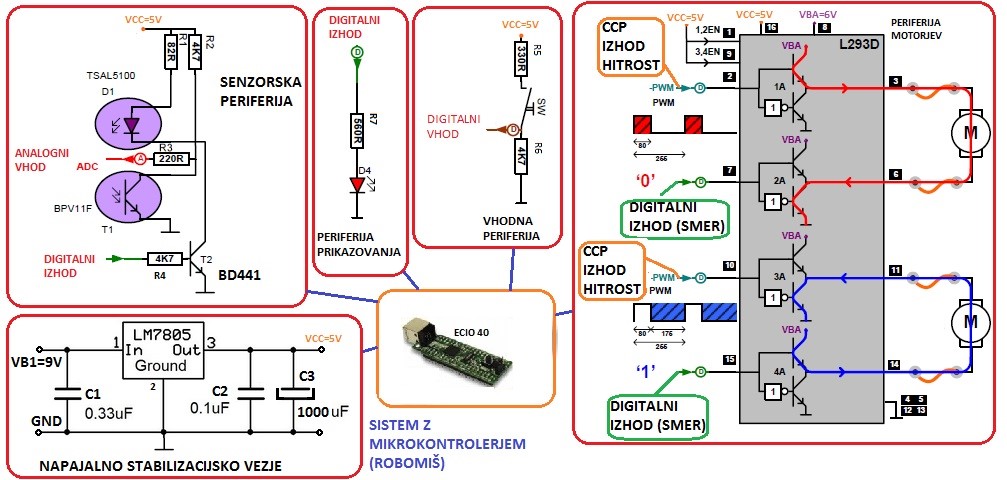
Slika Nastavitev konfiguracijske besede sistema

### ROBO MIŠ mobilni robot

Spletna stran-link: <https://sites.google.com/a/tscng.org/mobilna-robotika/#TOC-TEKMOVALNE-PREDLOGE-PROGRAMOV->



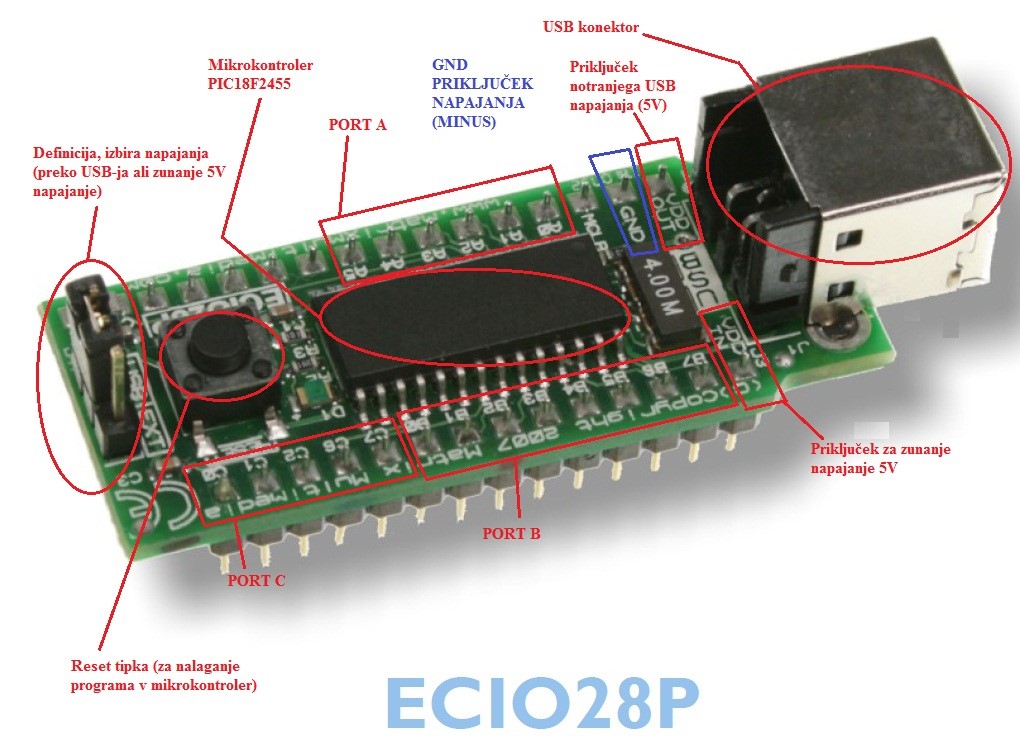
Slika Električna konfiguracija robota z ECIO28 mikrokrmilniškim modulom



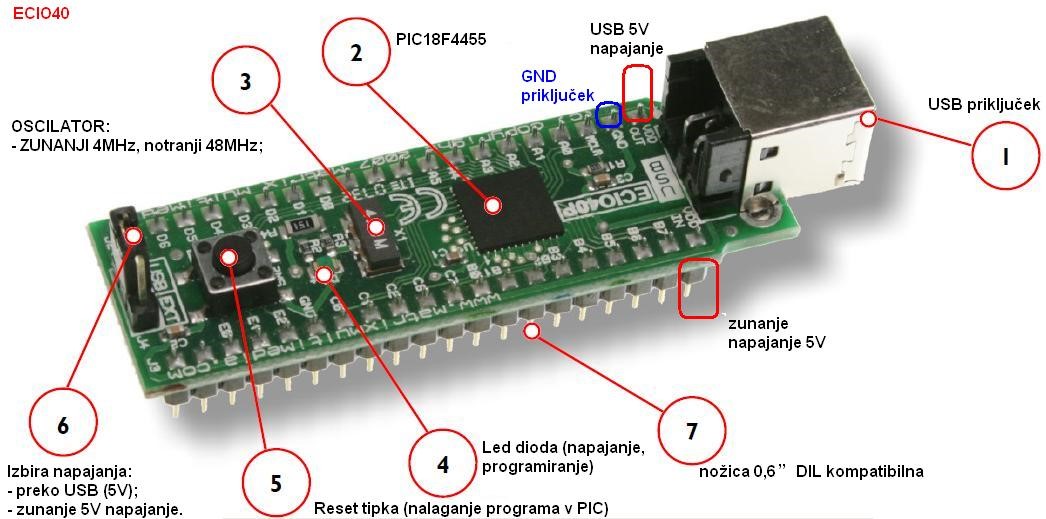
Slika Električna konfiguracija robota z ECIO40 mikrokrmilniškim modulom



Slika Modifikacija, poenostavljeno vezje napajanja z enim sklopom baterij (Ecio moduli delujejo v napajalnem intervalu od 2V do 5,5V)

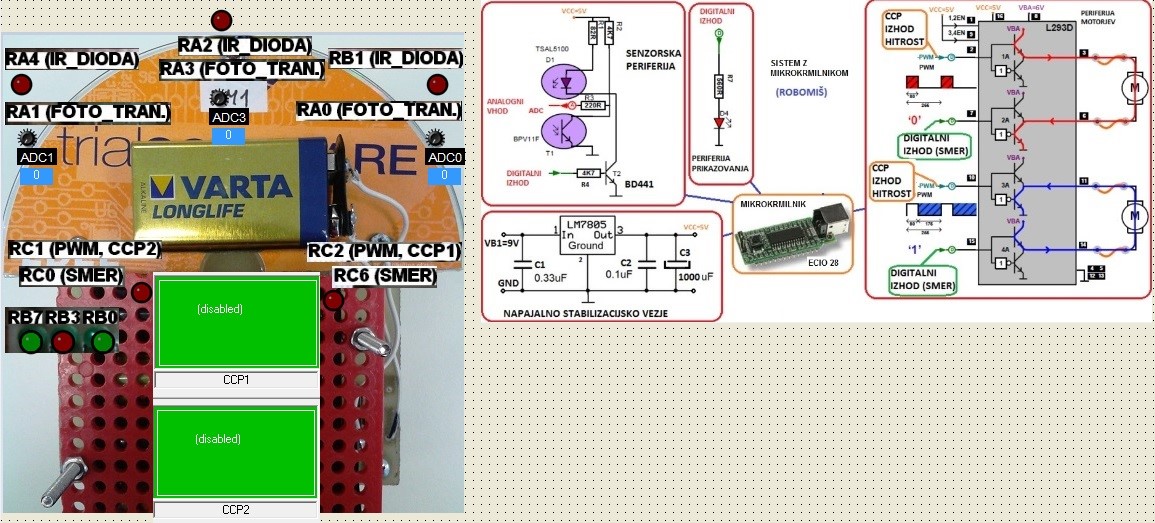


Slika ECIO 28

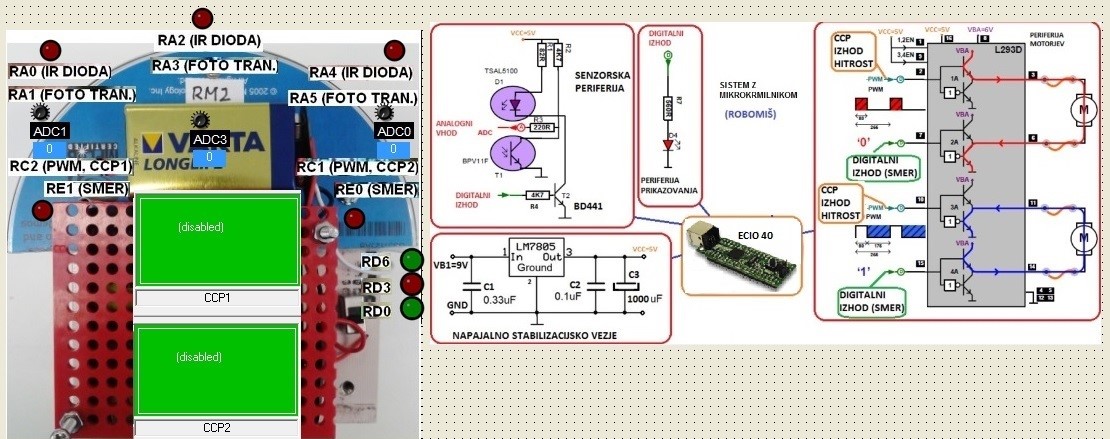


Slika ECIO 40

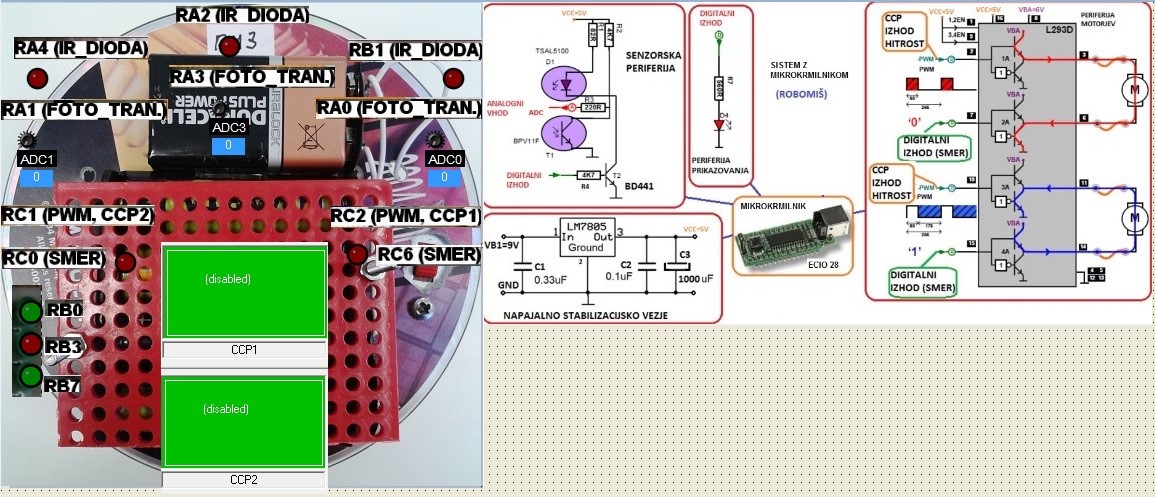
#### Primeri mobilnih robotov



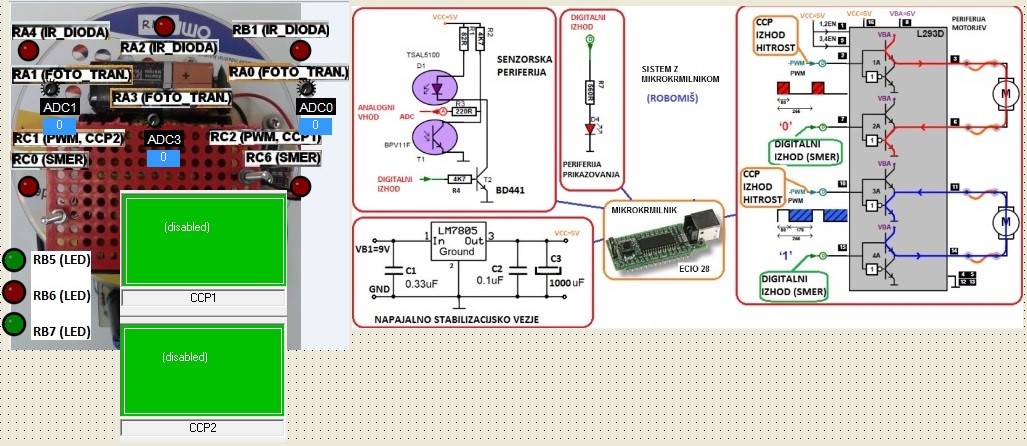
Slika RM1



Slika RM2



Slika RM3



Slika RM4

#### ROBO MIŠ metode zakasnitve

//deklaracije metod

void zakasnitev\_m\_sek( int ms);

void zakasnitev\_sek(int s);

//definicije metod

void zakasnitev\_m\_sek( int ms){

for(int i=1; i<=ms;i++){

for(int j=1; j<=1200;j++){

//zakasnitev\_1ms

}

}

}

void zakasnitev\_sek(int s){

for(int k=1; k<=s;k++){

for(int l=1; l<=1000;l++){

for(int m=1; m<=1200;m++){

//zakasnitev\_1ms

}

}

}

}