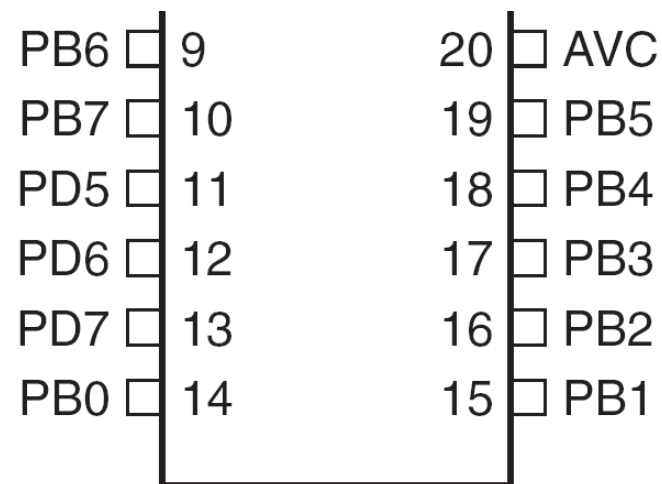


MIKROPROCESORI U INDUSTRIJI

Izlazni portovi

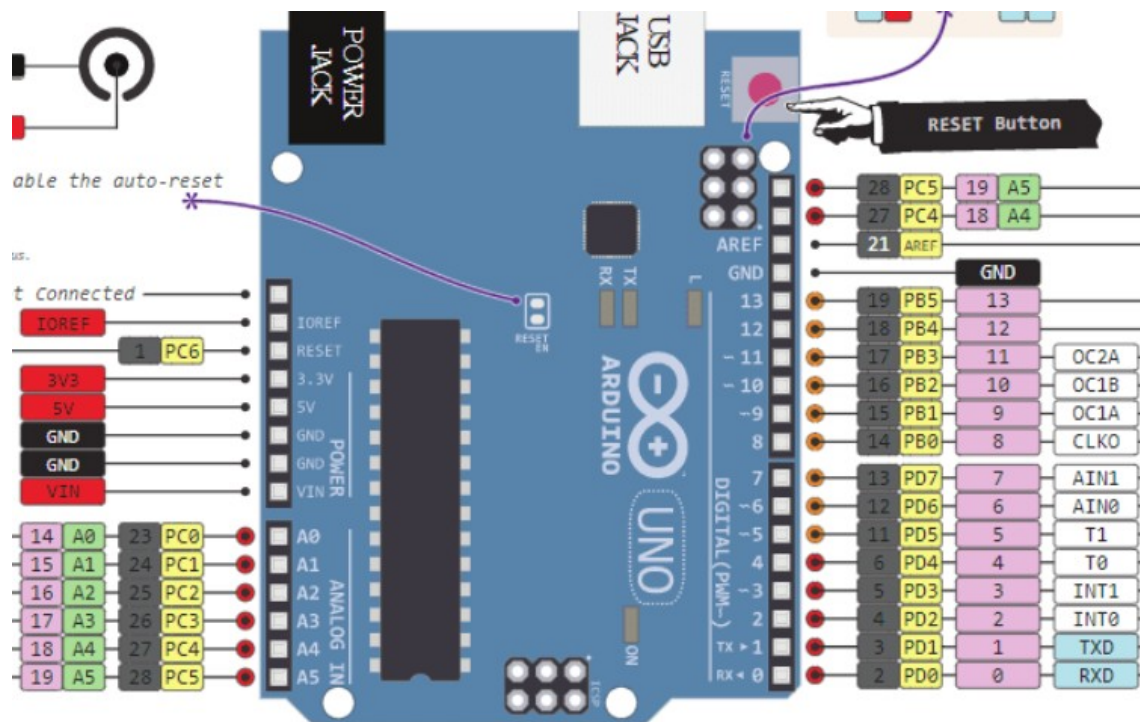
Microcontrollerski portovi i pinovi

- Priključci kroz koje mikrokontroler opšti sa spoljašnjom sredinom
 - Pr. PORTB
 - Pinovi PB0 – PB7
 - Ne moraju biti susjedni
 - Često bi-direcioni



Microkontrolerski portovi i pinovi

(PCINT14/RESET) PC6	1	28	PC5 (ADC5/SCL/PCINT13)
(PCINT16/RXD) PD0	2	27	PC4 (ADC4/SDA/PCINT12)
(PCINT17/TXD) PD1	3	26	PC3 (ADC3/PCINT11)
(PCINT18/INT0) PD2	4	25	PC2 (ADC2/PCINT10)
(PCINT19/OC2B/INT1) PD3	5	24	PC1 (ADC1/PCINT9)
(PCINT20/XCK/T0) PD4	6	23	PC0 (ADC0/PCINT8)
VCC	7	22	GND
GND	8	21	AREF
(PCINT6/XTAL1/TOSC1) PB6	9	20	AVCC
(PCINT7/XTAL2/TOSC2) PB7	10	19	PB5 (SCK/PCINT5)
(PCINT21/OC0B/T1) PD5	11	18	PB4 (MISO/PCINT4)
(PCINT22/OC0A/AIN0) PD6	12	17	PB3 (MOSI/OC2A/PCINT3)
(PCINT23/AIN1) PD7	13	16	PB2 ($\overline{SS}$ /OC1B/PCINT2)
(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)

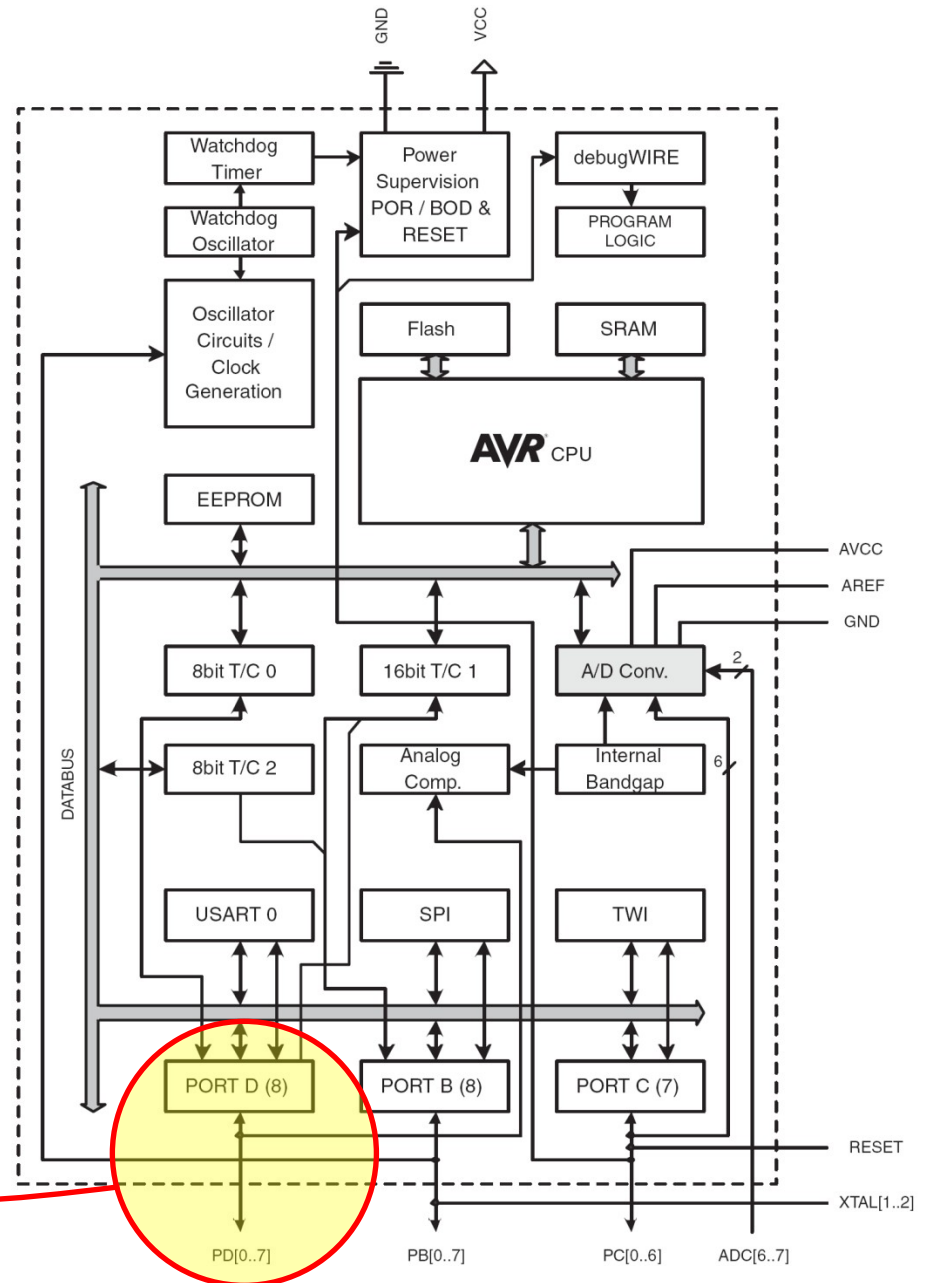
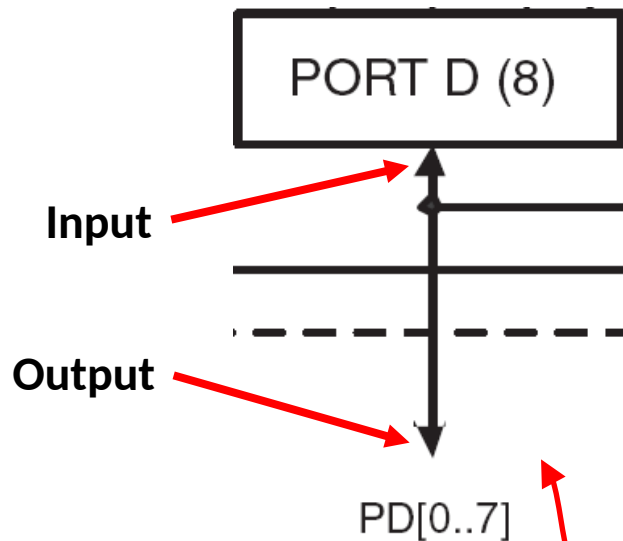


Port Pin – Usmjerenje podataka

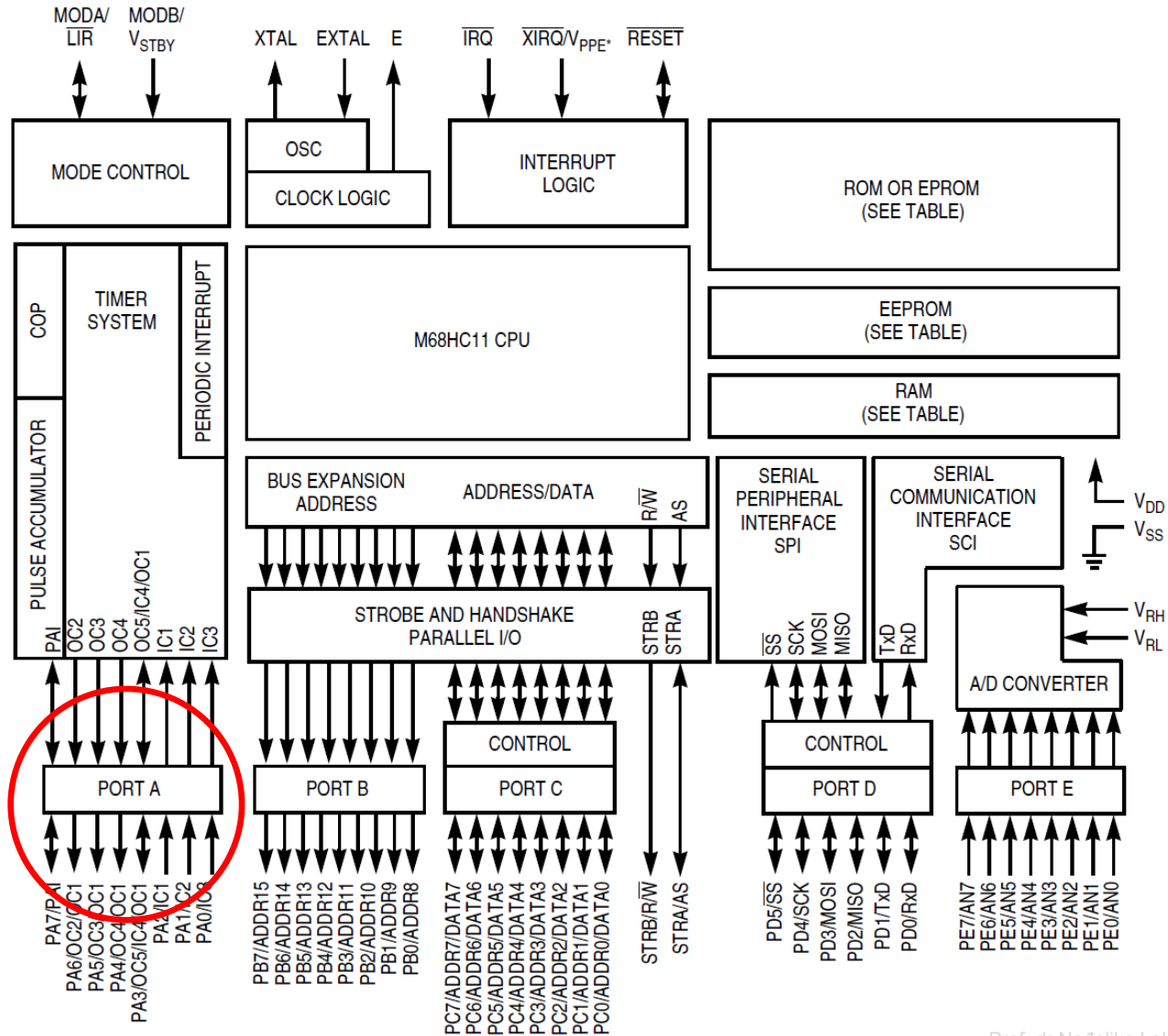
- Ulaz
 - Kada se želi uzeti informacija iz spoljašnjeg svijeta (senzori) u MCU
- Output
 - Kada se želi izmijeniti stanje nečega **izvan** MCU (uključiti ili isključiti motor, itd.) (aktuatori)
- Po uključenju napajanja svi pinovi su ulazni.
- Program može mijenjati usmjerenja podataka za svaki pin u svakom trenutku.

ATmega328

Blok diagram



M68HC11 mikrokontroler



Postavljanje smjera toka podatka za pin

- Arduino

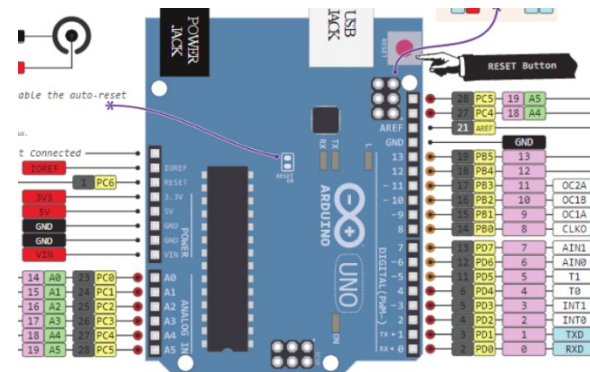
- `pinMode(pin_no., dir)`

- Pr. postaviti Arduino pin 3 (PD3) kao izlazni

- `pinMode(3, OUTPUT);`

- Napomena: jedan pin u jednom trenutku

- Predpostavimo da se želi postaviti pinove 3, 5, i 7 (PD3, PD5, i PD7) kao izlazne?
- Postoji li način da se oni postave istovremeno?
 - Da! Kako, slijedi kasnije...

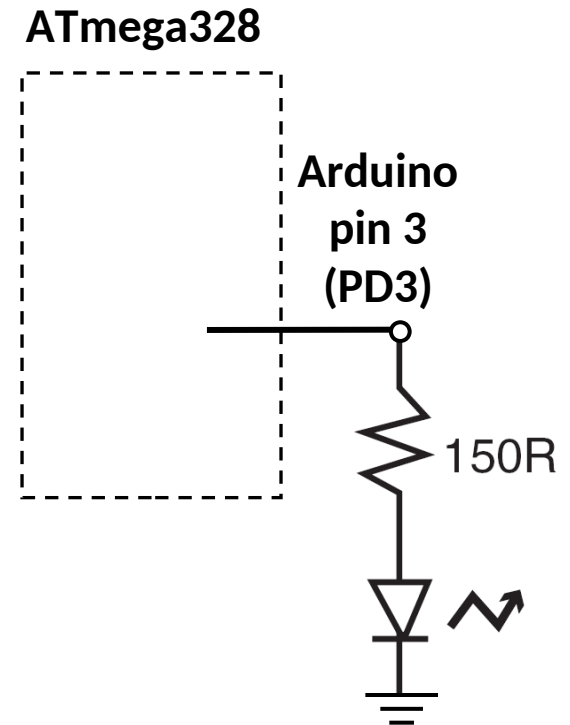


Napon na pinu

- Mikrokontroleri su u osnovi ***digitalni*** uređaji.
Za digitalne ulazno/izlazne (IO) pinove:
 - Informacija je ‘kodirana’ u dva diskretna stanja:
 - HIGH or LOW (logic: 1 or 0)
 - Naponi
 - TTL
 - » 5 V (za HIGH)
 - » 0 V (za LOW)
 - 3.3 V CMOS
 - » 3.3 V (za HIGH)
 - » 0 V (za LOW)

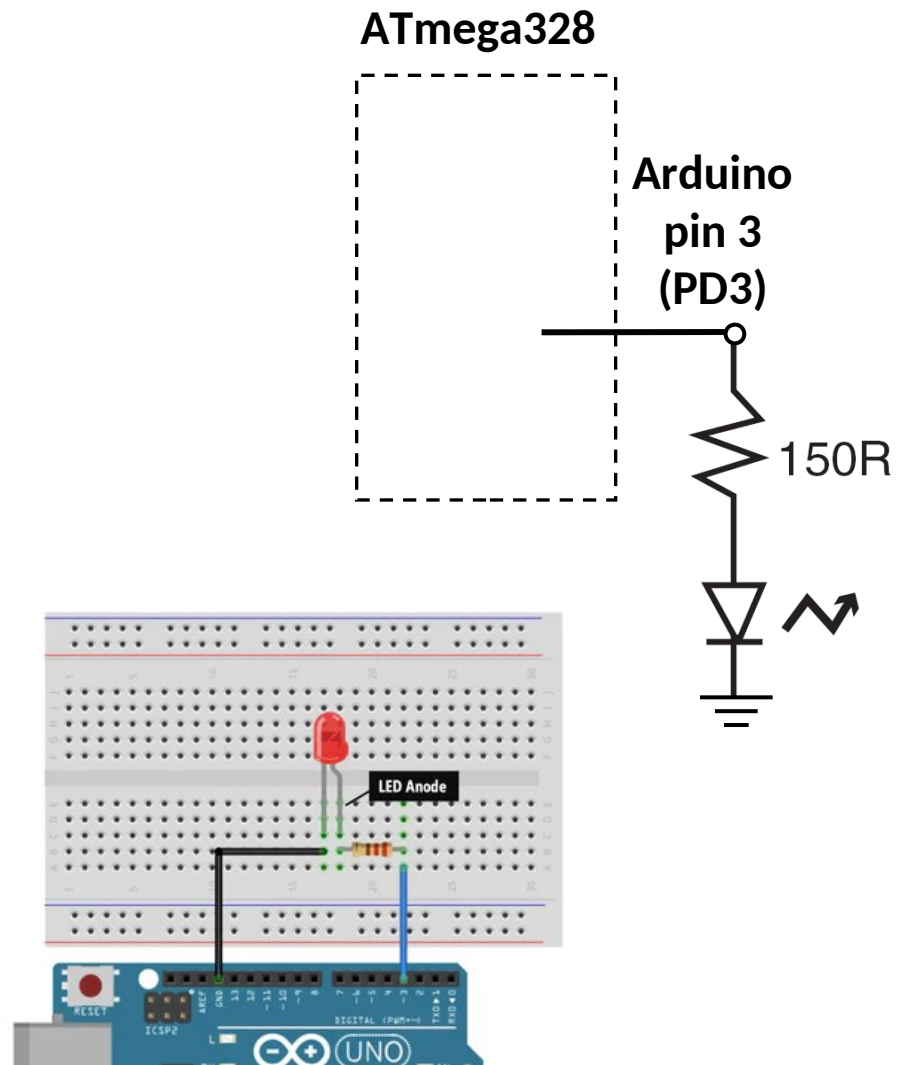
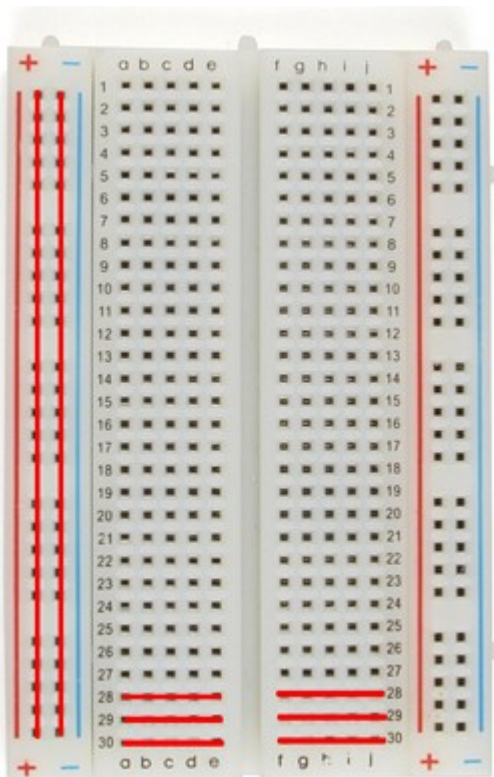
Pin upotrijebljen kao izlazni

- Uključiti LED, koja je povezana na Arduino pin 3 (PD3) (otpornik!)
 - Koji tok podataka treba biti za pin 3 (PD3)?
 - **pinMode(? , ?);**
 - Uključenje LED
 - **digitalWrite(3,HIGH);**
 - Isključenje LED
 - **digitalWrite(3,LOW);**



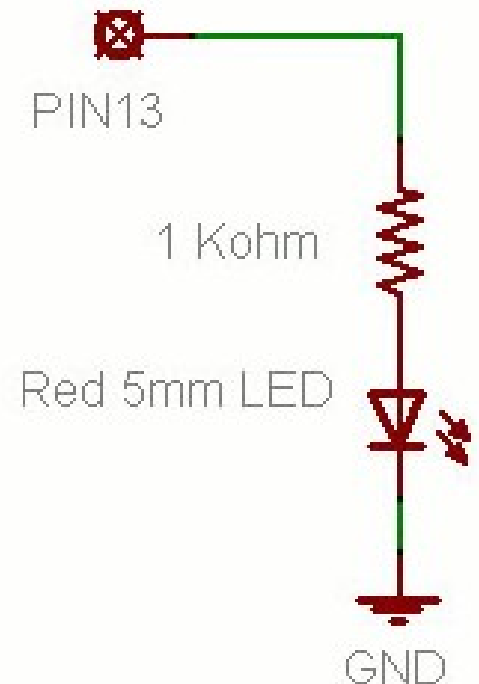
Pin upotrijebljen kao izlazni

- Uključenje LED
 - `digitalWrite(3,HIGH);`
- Isključenje LED
 - `digitalWrite(3,LOW);`



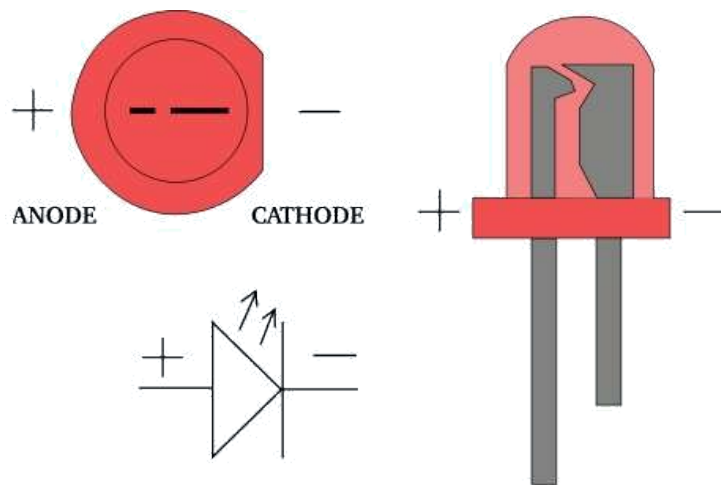
Osnovno LED kolo

- Povežite pin 13 mikrokontrolera na jedan kraj otpornika.
- Drugu nožicu otpornika spojite na dužu nožicu LED.
 - Veća otpornost znači slabije svjetlo.
 - Manja otpornost znači jače svjetlo.
 - Bez otpornosti znači pregorijevanje LED ili preopterećenje porta.
- Kraću nožicu LED spojite na negativni priključak napajanja (masu).

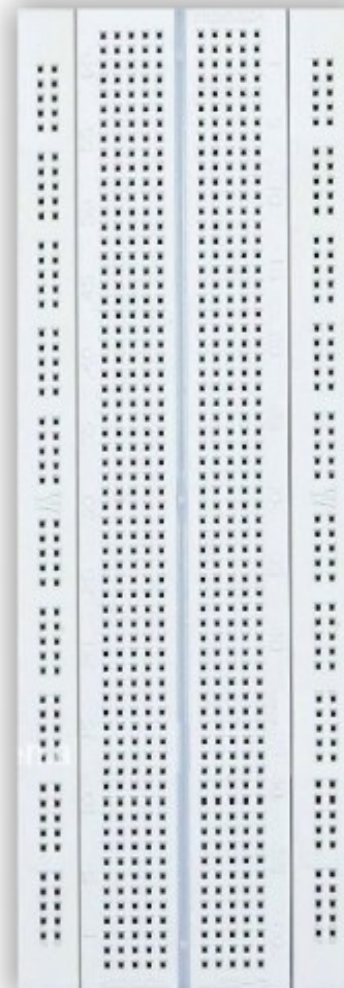
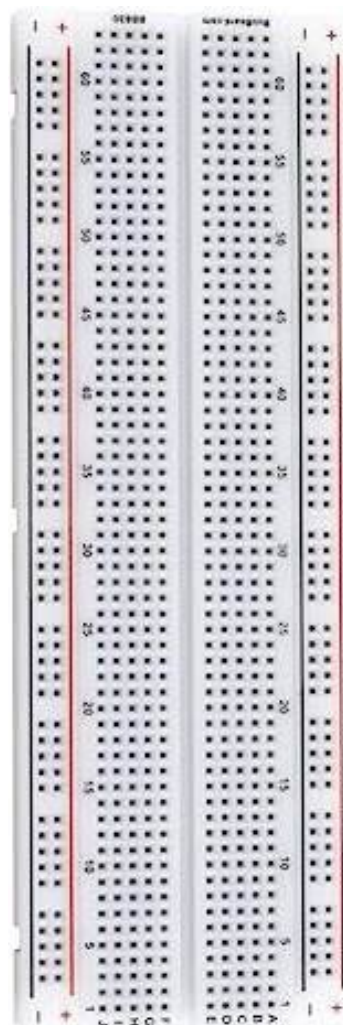
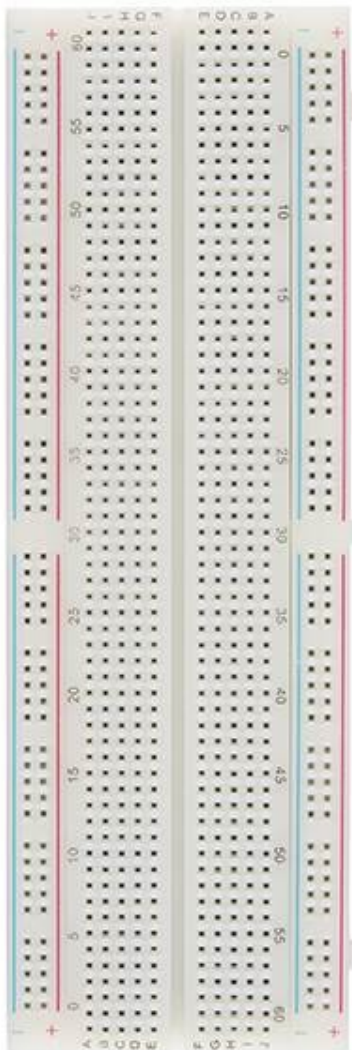


Blink Skeč (Treperenje)

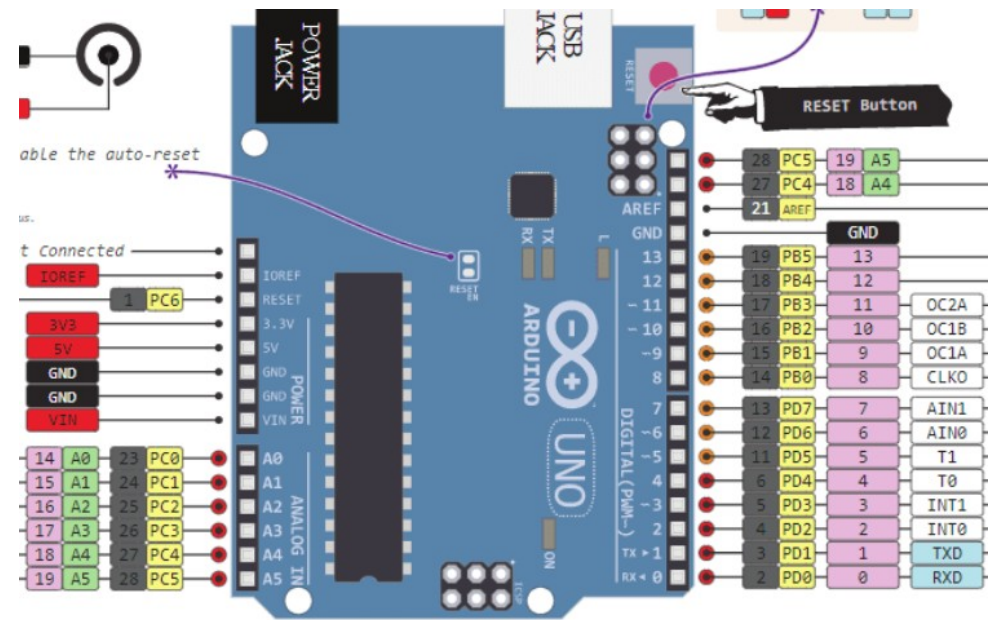
- **File > Examples > Digital > Blink**
- LED ima polaritet
 - Negativni je indikovani zasječenim obodom tijela diode i kraćom nožicom.



Varijante experimentalnih ploča



Pitanje od prije?



- Pitanje od prije:
 - Postoji li način da se tok podataka postavi za više pinova istovremeno?
- Sav rad na MCU dešava se kroz *registre* (posebne memorijske lokacije)
 - Registri u Atmega328 su dužine 8-bita.
- Data direction register (DDRx) upravlja tokom podataka za pinove u PORTx

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Izvor: http://www.atmel.com/dyn/products/product_card.asp?PN=ATmega328P p. 93

Data Direction Register

- Ako je bit nula -> pin će biti ulazni
 - Postavljenje bita na nulu == 'čišćenje bita' ('clearing the bit')
- Ako je bit jedan -> pin će biti izlazni
 - Postavljenje bita na jedinicu == 'postavljanje bita' ('setting the bit')
- Za istovremenu promjenu toka podataka za više pinova koji pripadaju portu PORTx:
 1. Određivanje koje bitove treba postaviti a koje očistiti u registru DDRx.
 2. Upisati binarni (hex) broj u DDRx.

ATmega328 registri za rad sa portovima

- Vidijeti ATmega328 data sheet, pp. 76-94
- Za digitalne IO, važni registri su:
 - DDRx
 - Data Direction bit u DDRx registru (read/write)
 - PORTx
 - PORTx data registar (read/write)
 - PINx
 - PINx registar (read only)

PORT Pin i registar - detalji

Figure 13-1. I/O Pin Equivalent Schematic

ATmega328 datasheet, pp. 76-94

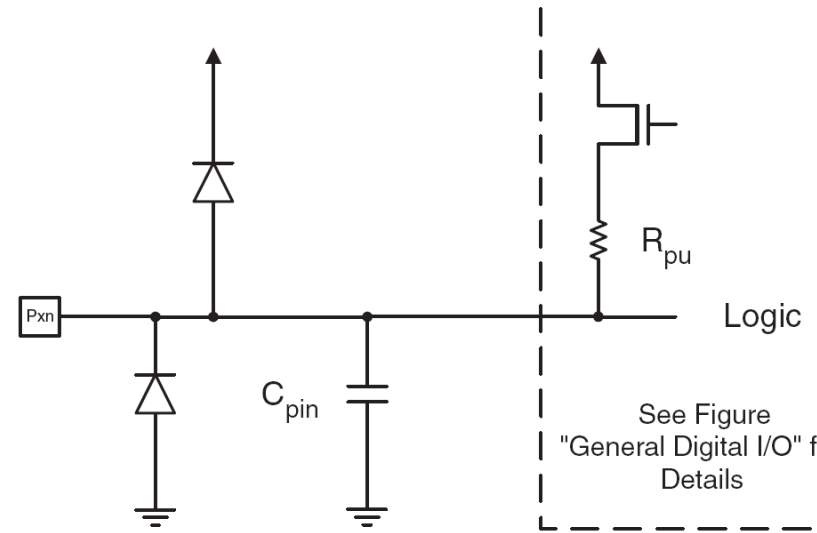
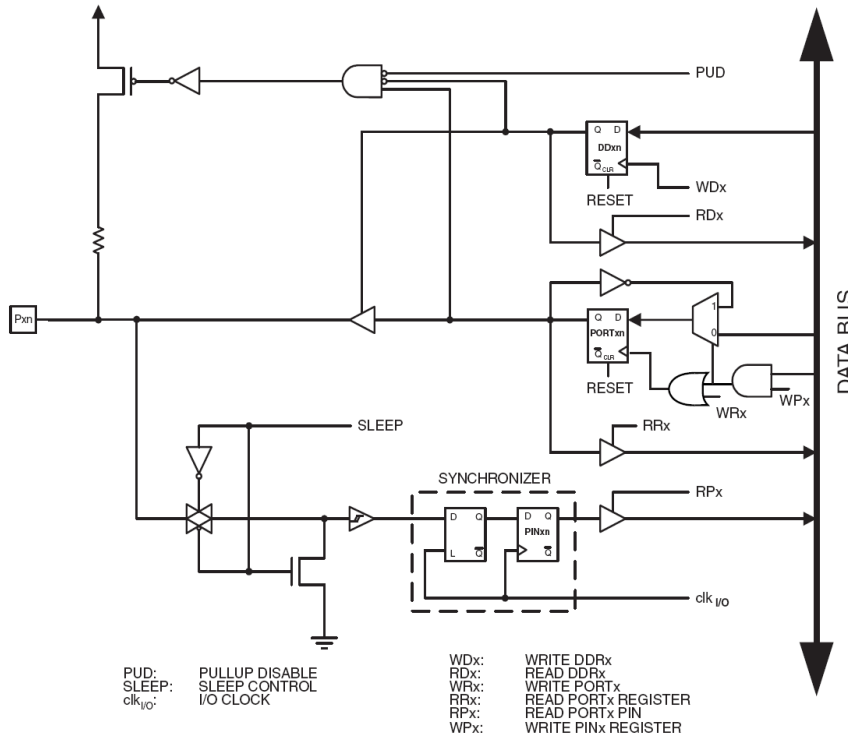


Figure 13-2. General Digital I/O⁽¹⁾



PORTD – The Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DDRD – The Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PIND – The Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x09 (0x29)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Primjer 1

- Postaviti Arduino pinove 3, 5, i 7 (PD3, PD5, i PD7) kao izlazne

- Arduino pristup

```
pinMode(3, OUTPUT);  
pinMode(5, OUTPUT);  
pinMode(7, OUTPUT);
```

Ili ako je upotrijebljena me106.h:

```
pinMode(PIN_D3, OUTPUT);  
pinMode(PIN_D5, OUTPUT);  
pinMode(PIN_D7, OUTPUT);
```

- Alternativni pristup

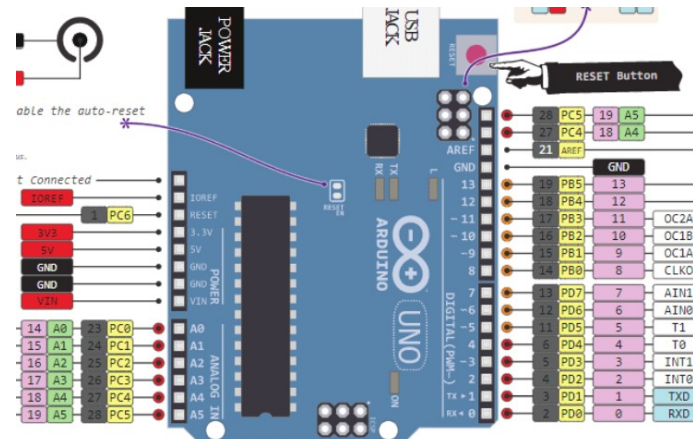
$DDRD = 0b10101000;$

ili

$DDRD = 0xA8;$

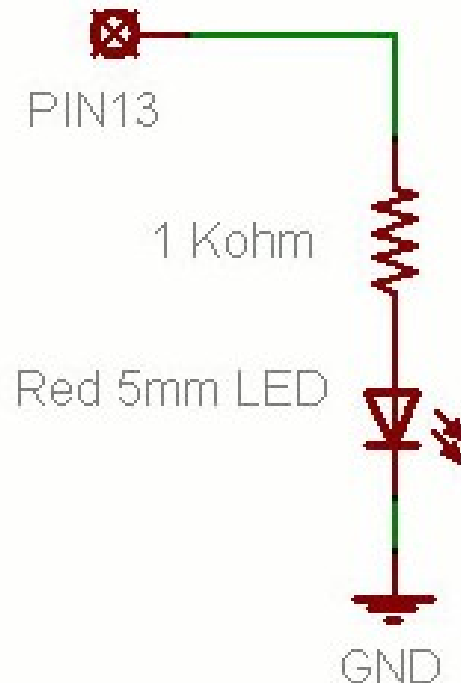
ili

$DDRD |= 1 \ll PD7 | 1 \ll PD5 | 1 \ll PD3;$



Prosto elektronsko kolo

- Najjednostavniji sklop.
- Uključi/isključi svjetlo.
- Struja teče iz pina (izvora napajanja), kroz potrošač (LED).



Struktura Arduino programa

- Arduino program == 'sketch'
 - Mora imati:
 - `setup()`
 - `loop()`
 - `setup()`
 - Konfiguriše pinove i registre
 - `loop()`
 - Pokreće glavno tijelo programa neprestano
 - Kao `while(1) {...}`
 - Gdje je `main()` ?
 - Arduino uprošćava stvari
 - Odrađuje za Vas

```
/* Blink - turns on an LED for DELAY_ON msec,
then off for DELAY_OFF msec, and repeats
BJ Furman rev. 1.1 Last rev: 22JAN2011
*/
#define LED_PIN 13 // LED on digital pin 13
#define DELAY_ON 1000
#define DELAY_OFF 1000

void setup()
{
    // initialize the digital pin as an output:
    pinMode(LED_PIN, OUTPUT);
}

// loop() method runs forever,
// as long as the Arduino has power

void loop()
{
    digitalWrite(LED_PIN, HIGH); // set the LED on
    delay(DELAY_ON); // wait for DELAY_ON msec
    digitalWrite(LED_PIN, LOW); // set the LED off
    delay(DELAY_OFF); // wait for DELAY_OFF msec
}
```

Funkcije: millis (micros)

Izbjegavanje upotrebe dužeg čekanja u skeču!

Vraća broj milisekundi (mikrosekundi) koji je prošao od kada je Arduino ploča počela da izvršava određeni program.

```
unsigned long myTime;
```

```
myTime = millis();
```

Funkcije: millis (micros)

Primjer upotrebe funkcije millis():

```
unsigned long startMillis; //globalne promjenljive
unsigned long currentMillis;
const unsigned long period = 1000; //vrijednost je u milisekundama
const byte ledPin = 13; //korištenje ugrađene diode LED

void setup()
{
    pinMode(ledPin, OUTPUT);
    startMillis = millis(); //inicijalno početno vrijeme
}

void loop()
{
    currentMillis = millis(); //dodjela trenutnog „vremena“ promjenljivoj
    if (currentMillis - startMillis >= period) //ispitivanje da li je period istekao
    {
        digitalWrite(ledPin, !digitalRead(ledPin)); //ako jeste, promjena stanja LED.
        startMillis = currentMillis; //VAŽNO je ubilj. startno vrijeme zadatog stanja LED.
    }
}
```

Funkcije: millis (micros)

Izbjegavanje problema preliivanja (overflow) kada se koristi millis() i micros()

Funkcije millis() i micros() prelijevaju nakon otprilike 50 dana i 70 minuti, respektivno. Potencijalni problem može se lako izbjeći malim prilagođenjem koda.

Umjesto ovako:

```
int period = 1000;
unsigned long time_now = 0;

void setup() {
    Serial.begin(115200);
}

void loop() {
    if(millis() > time_now + period){
        time_now = millis();
        Serial.println("Hello");
    }

    //Run other code
}
```

Napisati ovako:

```
int period = 1000;
unsigned long time_now = 0;

void setup() {
    Serial.begin(115200);
}

void loop() {
    if(millis() - time_now > period){
        time_now = millis();
        Serial.println("Hello");
    }

    //Run other code
}
```

Na ovo se može gledati kao na poređenje perioda, umesto da se radi sa vremenskim oznakama.

Funkcije: millis (micros)

Može izgledati da predhodno nema puno smisla. Može izgledati da se problem samo pomjera, a da se ne rješava.

Gledajući matematički, nema puno smisla, jer će rezultat

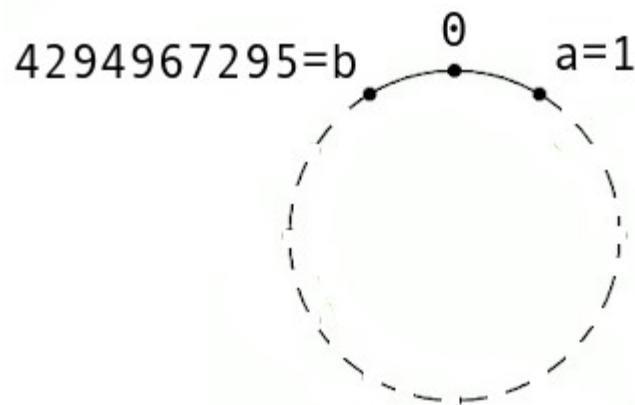
`millis()` - `time_now`

postati negativan, kada millis() „prelije“.

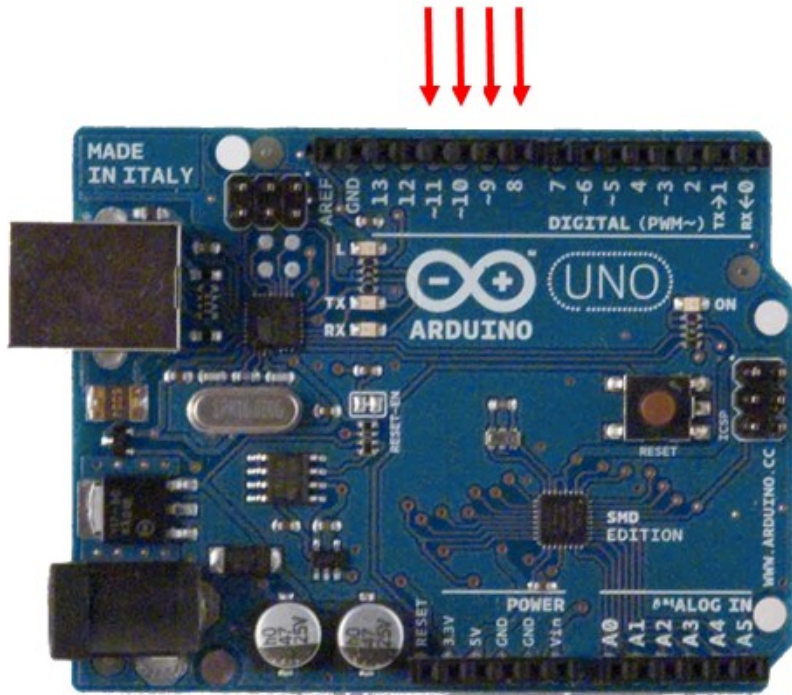
Rezultat oduzimanja: od veoma malog cijelog broja veoma velikog cijelog broja.

Dokaz da ipak neće:

```
void setup() {  
  Serial.begin(115200);  
  
  unsigned long a = 1;  
  
  //unsigned long maximum value  
  unsigned long b = 4294967295;  
  
  Serial.println(a-b);  
}  
  
void loop() {  
}
```



TRČEĆE SVJETLO - 4 LED



```
#define PERIOD 700
```

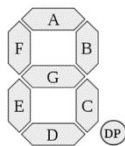
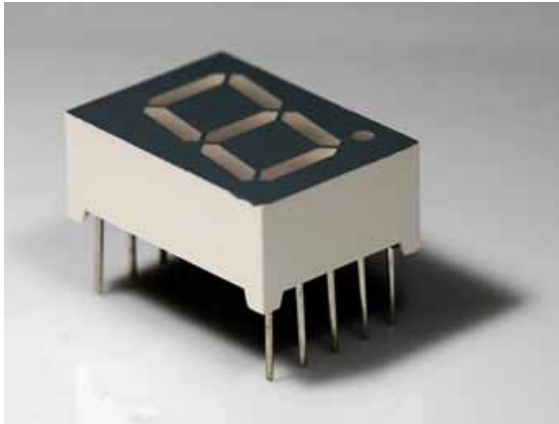
```
int i = 0;  
int LED[] = { 1, 2, 4, 8 };  
unsigned long start;
```

```
void setup() {  
  DDRB |= 0b00001111;  
  start = millis();  
}
```

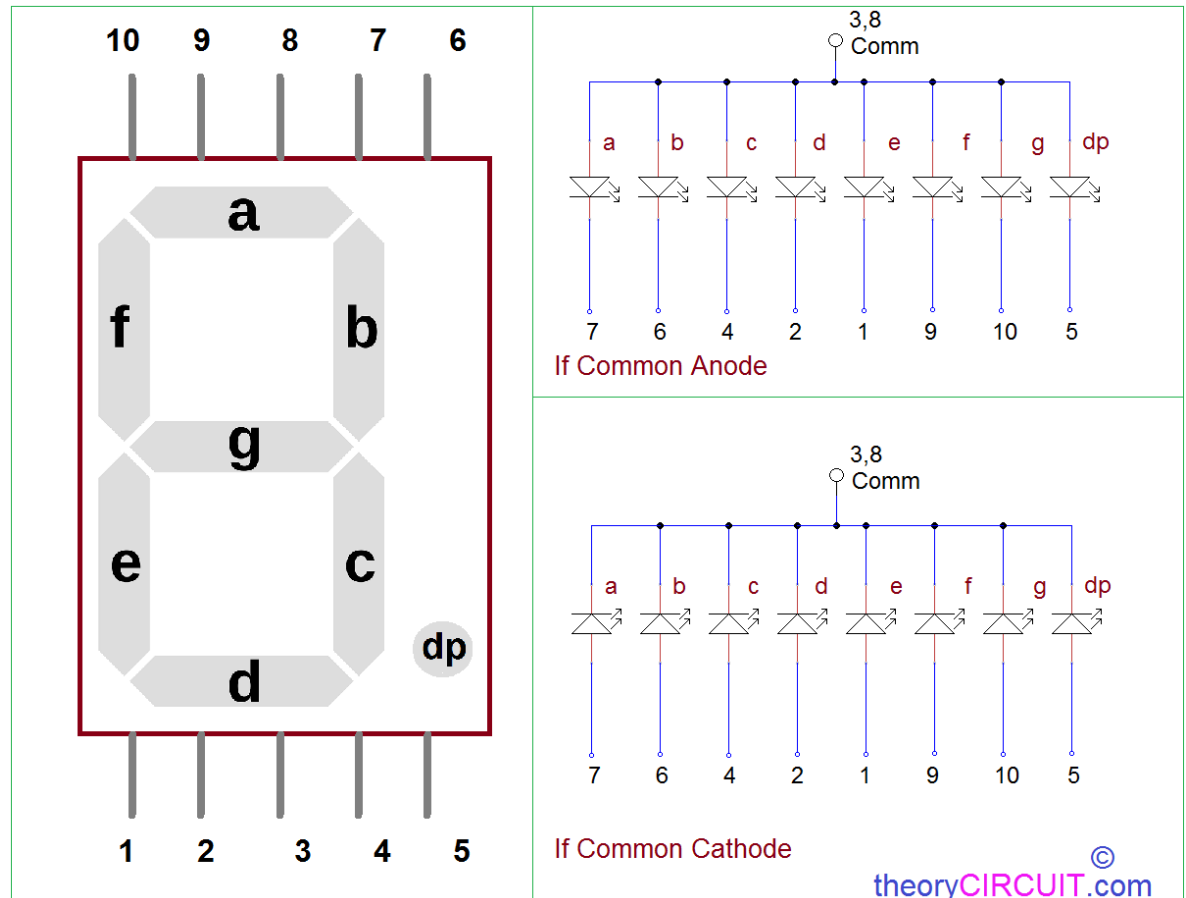
```
void loop() {  
  if ((millis() - start) > PERIOD) {  
    i += 1;  
    if(i>3)i=0;  
    PORTB = LED[i];  
    start = millis();  
  }  
}
```

SEDMO-SEGMENTNI DISPLAY

Napisati program koji broji od 0 do 9 s ponavljanjem, i prikazuje rezultat na sedmosegmentnom LED displeju. Jedna promjena u sekundi.



7 Segment Display Pinout



Rješenje:

```
#define PERIOD 1000

int i = 0;
long start;
int a=1, b=2, c=4, d=8, e=16, f=32, g=64, dp=128;

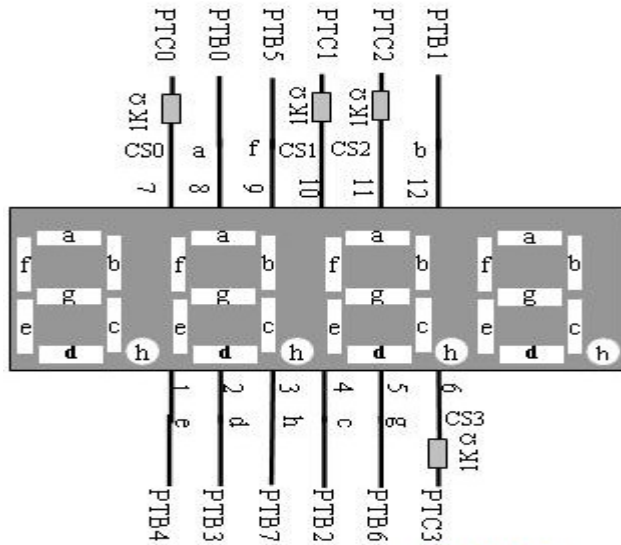
int cifre[]={a+b+c+d+e+f, b+c, a+b+d+e+g, a+b+c+d+g, b+c+f+g,
             a+c+d+f+g, a+c+d+e+f+g, a+b+c, a+b+c+d+e+f+g, a+b+c+d+f+g+dp };

void setup() {
    DDRD |= 0b11111111;
    start = millis();
}

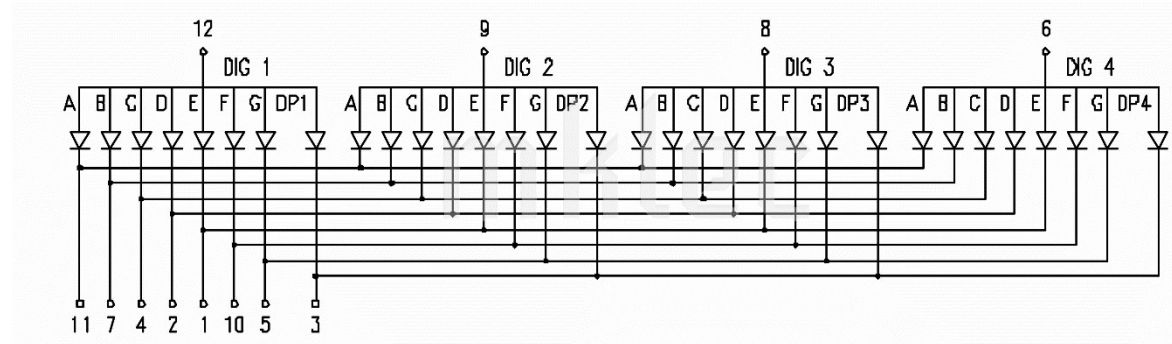
void loop() {
    if ((millis() - start) > PERIOD) {
        PORTD=cifre[i];
        if(++i>9)i=0;
        start = millis();
    }
}
```

ČETVOROCIFARSKI SEDMO-SEGMENTNI DISPLAY

Napisati program koji na četvorocifarskom sedmosegmentnom LED displeju ispisuje '123.4' pet sekundi i '2024' pet sekundi. (4-2-1 poen)



MCU与4连排8段数码管的连接



Rješenje:

```
#define PERIOD 2
```

```
int i = 0;
```

```
long start;
```

```
int a=1, b=2, c=4, d=8, e=16, f=32, g=64, dp=128;
```

```
int cifre[]={a+b+c+d+e+f, b+c, a+b+d+e+g, a+b+c+d+g, b+c+f+g, a+c+d+f+g,  
            a+c+d+e+f+g, a+b+c, a+b+c+d+e+f+g, a+b+c+d+f+g };
```

```
void setup() {
```

```
    DDRD |= 0b11111111;
```

```
    DDRB |= 0b00001111;
```

```
    start = millis();
```

```
}
```

```
void loop() {
```

```
    if ((millis() - start) > PERIOD) {
```

```
        i++;
```

```
        if(i>4)i=1;
```

```
        switch(i){
```

```
            case 1: PORTD=cifre[i]; PORTB=~1; break;
```

```
            case 2: PORTD=cifre[i]; PORTB=~2; break;
```

```
            case 3: PORTD=cifre[i]+dp; PORTB=~4; break;
```

```
            case 4: PORTD=cifre[i]; PORTB=~8; break;
```

```
        }
```

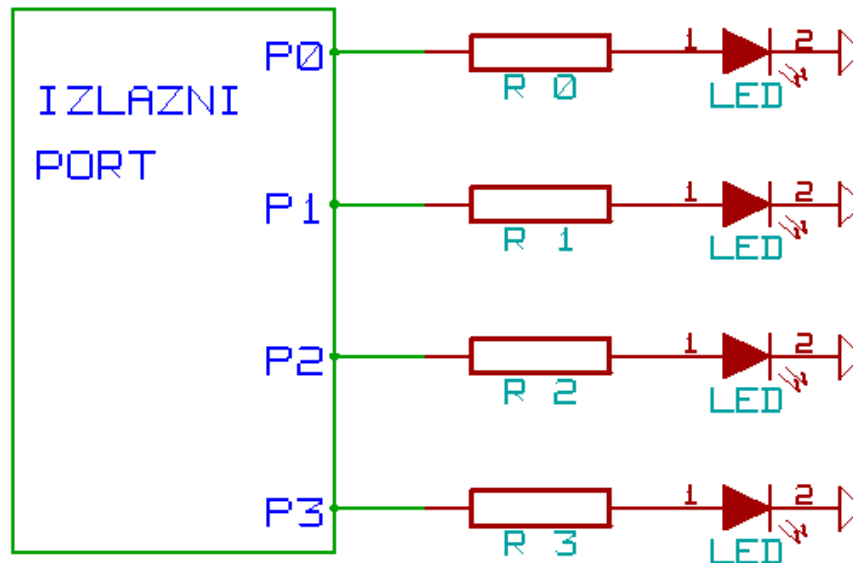
```
        start = millis();
```

```
    }
```

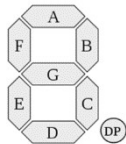
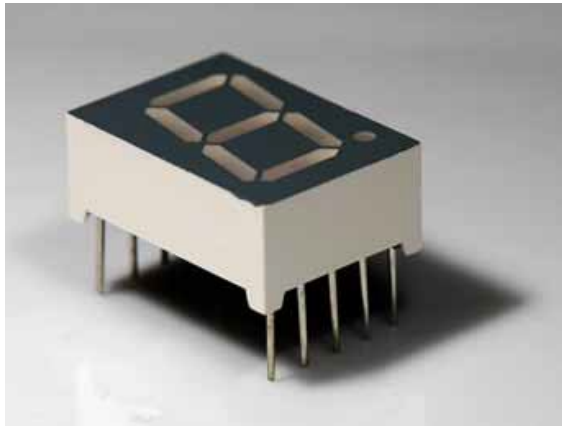
```
}
```

ZADACI ZA VJEŽBU 1

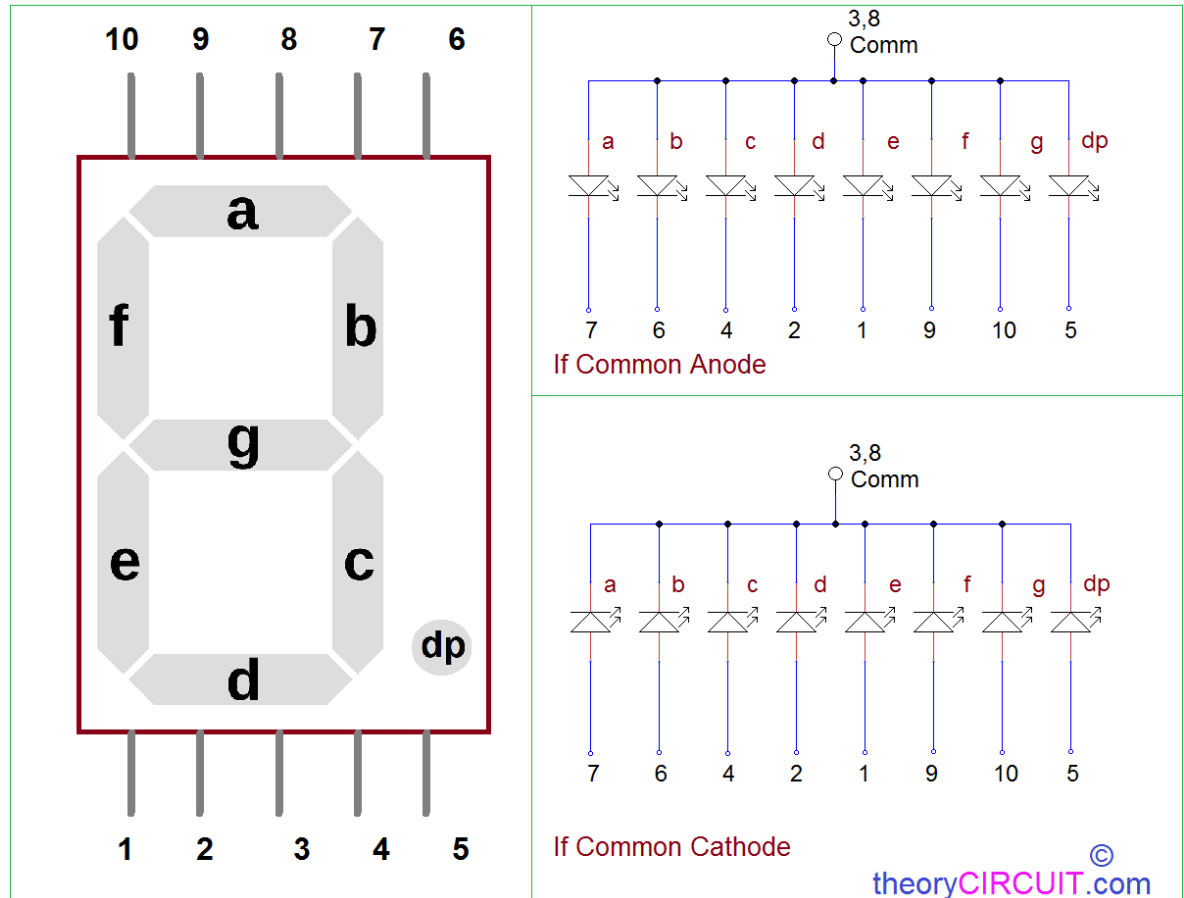
1. „Trčeća tama“ upotrebom 4 LED. Uvijek je samo jedna dioda isključena.
(1 poen)
2. Pomocu 4 LED, u binarnom obliku prikazati vrijednost promjenjive BROJAC. Vrijednost promjenjive brojac se inkrementira svake sekunde.
(2-1 poen)



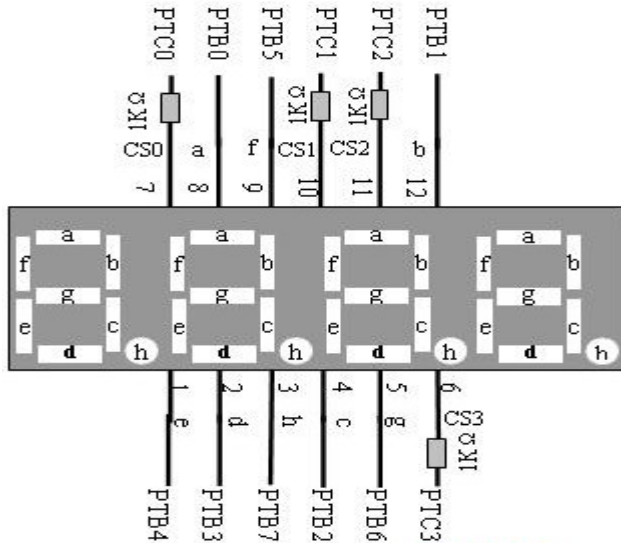
3. Napisati program koji ciklično broji od sljedećim redoslijedom 0, 2, 4, 6, 8, 1, 3, 5, 7, 9, A, b, C, d, E i prikazuje rezultat na sedmosegmentnom LED displeju. Jedna promjena u sekundi. **(3-2 poena)**



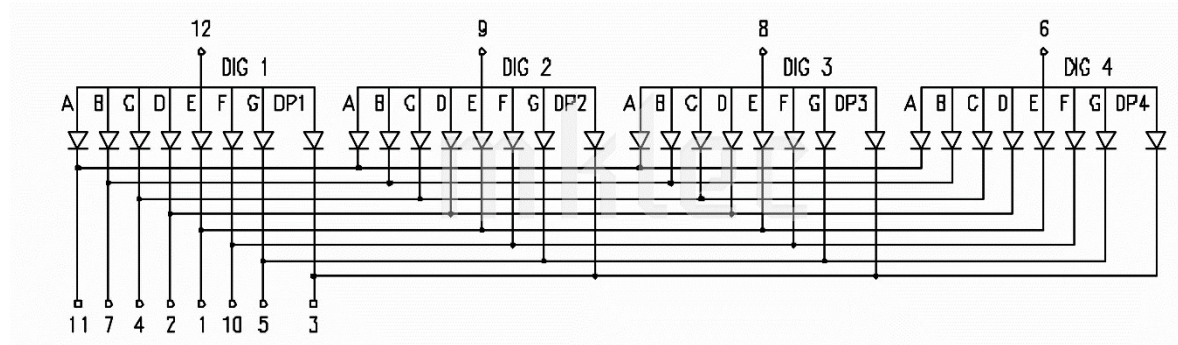
7 Segment Display Pinout

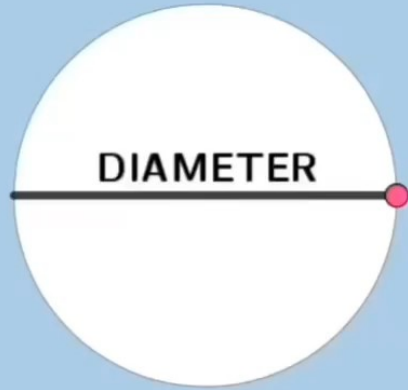


4. Napisati program koji na četvorocifarskom sedmosegmentnom LED displeju ciklično ispisuje '26.02.' pet sekundi i '2025' tri sekunde. (4-3-2 poen)



MCU与4连排8段数码管的连接





Kraj