

# RAČUNARSKÉ PERIFERIE I INTERFEJSI

# O PREDMETU

***Studijski programi za koje se organizuje :***

-Specijalističke studije Elektrotehnike, studijski program Elektronika, Telekomunikacije i Računari (ETR), smjer Elektronika.

***Uslovljenost drugim predmetima:*** Nema formalnih uslova. Podrazumijeva se poznavanje C/C++ jezika.

***Ciljevi izučavanja predmeta:*** Interfejsi (međusklopovi) i periferije (spoljni uređaji) povezuju računare sa spoljašnjom sredinom. Cilj ovog predmeta je da studenti ovladaju znanjem i tehnikama, pomoću kojih će moći da uz pomoć računara (automatski) prikupljaju informacije iz spoljašnjeg svijeta i da upravljaju procesima van računara. Osim teoretskog dijela, značajna pažnja se poklanja praktičnom radu.

***Metod nastave i savladanja gradiva:*** Predavanja, računске vježbe i vježbe u računarskoj učionici / laboratoriji. Učenje i samostalna izrada praktičnih zadataka. Konsultacije.

# O PREDMETU

<b>Sadržaj predmeta:</b>	
Pripremna sedmica	Priprema i upis semestra
I sedmica	Uvod; Upoznavanje sa predmetom, ciljevima i načinom rada;
II sedmica	Mikrokontroleri; Razvojne ploče; Arduino Uno;
III sedmica	Portovi: izlazni, ulazni;
IV sedmica	Analogni ulazi i izlazi
V sedmica	Koračni motori;
VI sedmica	Optički interfejsi: Inkrementalni davači položaja, optički difuzioni senzor, ...;
VII sedmica	<i>I provjera znanja;</i>
VIII sedmica	D/A konverzija; Upravljanje analognim uređajima;
IX sedmica	A/D konverzija: prateća, sukcesivna, paralelna, V/f i f/V konvertori;
X sedmica	Komunikacioni interfejsi: -paralelni, -serijski;
XI sedmica	Priključci (sockets);
XII sedmica	<i>II provjera znanja;</i>
XIII sedmica	Modemi: dial-up, govorni, adsl, gprs; AT komande;
XIV sedmica	Modemi sa integrisanim script interpreterom (Python); GPS;
XV sedmica	Interfejsi u industriji; Vizuelizacija industrijskih procesa;
XVI sedmica	<i>Završni ispit</i>
Završna sedmica	Ovjera semestra i upis ocjena
XVIII-XXI sedmica	Dopunska nastava i popravni ispitni rok

# O PREDMETU

## *Opterećenje studenata na predmetu*

### Sedmično

**6 kredita x 40/30 = 8 časova**

### **Struktura:**

**3 časa predavanja**

**1 čas računskih vježbi**

**4 časa samostalnog rada,  
uključujući konsultacije**

### U toku semestra

**Nastava i završni ispit: (8 časova) x 16 = 128 časova**

**Neophodne pripreme prije početka semestra  
(administracija, upis, ovjera)**

**2 x (8 časova) = 16 časova**

**Ukupno opterećenje za predmet 6x30 = 180 časova**

**Dopunski rad** za pripremu ispita u popravnom ispitnom roku, uključujući i polaganje popravnog ispita od 0 do 30 časova (preostalo vrijeme od prve dvije stavke do ukupnog opterećenja za predmet 150 časova)

### **Struktura opterećenja:**

**128 časova (Nastava)+16 časova (Priprema)+36 časova (Dopunski rad)**

Studenti su obavezni da pohađaju nastavu, rade i predaju sve domaće zadatke, odrade laboratorijske vježbe i obje provjere znanja.

# O PREDMETU

**Literatura:**

Osnovna i pomoćna literatura u elektronskom obliku na [www.etf.ac.me](http://www.etf.ac.me)

Praktični zadaci za laboratorijske vježbe na [www.etf.ac.me](http://www.etf.ac.me)

Z.Mijanović i ostali, »Računarski interfejsi i periferije«, Univerzitet Crne Gore

**Oblici provjere znanja i ocjenjivanje:**

-Bodovi na vježbama 24

-Provjera znanja 26 poena

-Završni ispit 50 poena.

-Prelazna ocjena se dobija ako se kumulativno sakupi najmanje 50 poena.

**Posebnu naznaku za predmet:**

U slučaju da je to potrebno nastava se može izvoditi i na engleskom jeziku.

# Što je mikrokontroler?

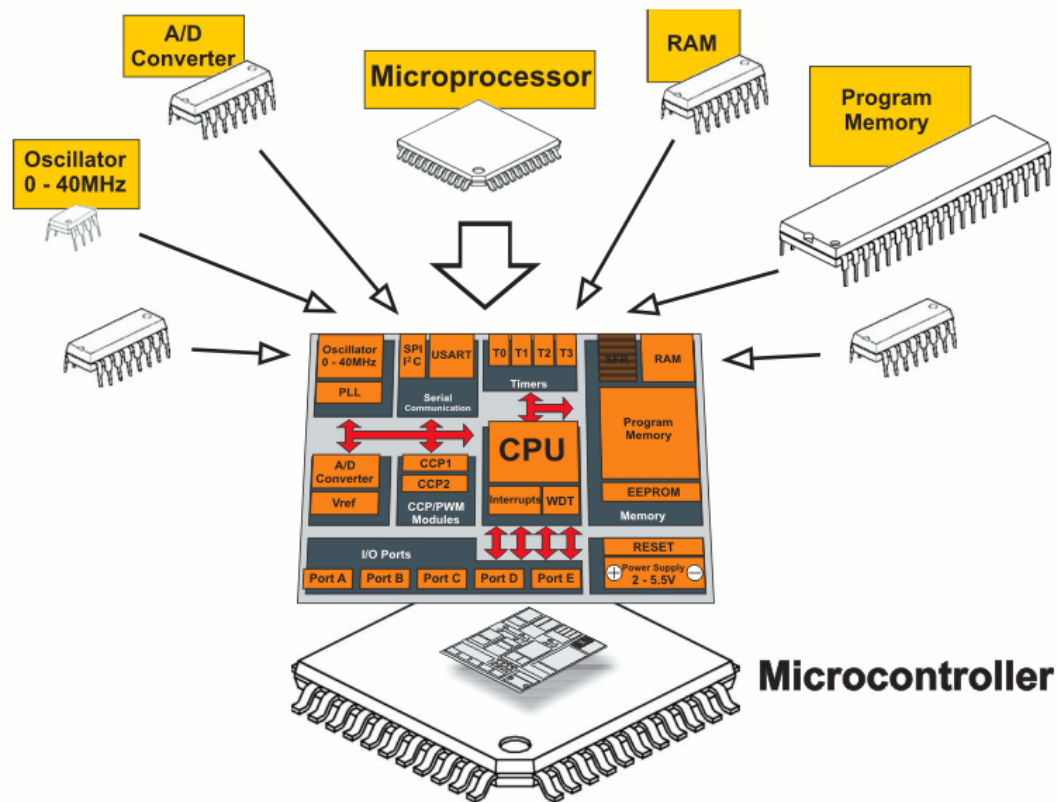
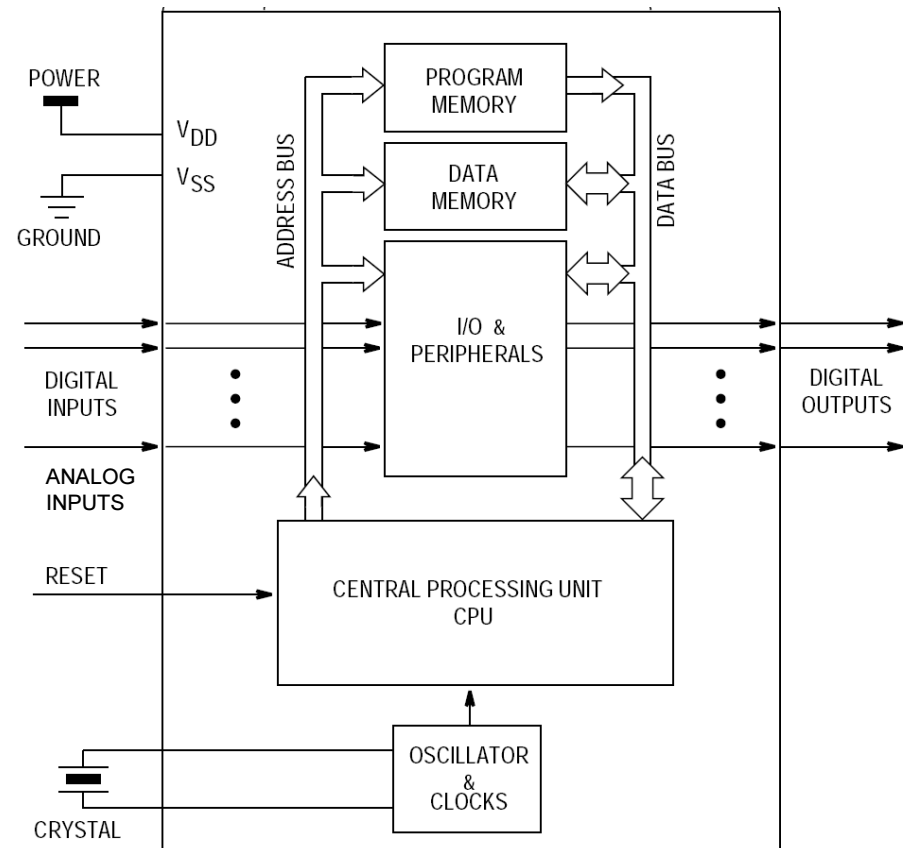
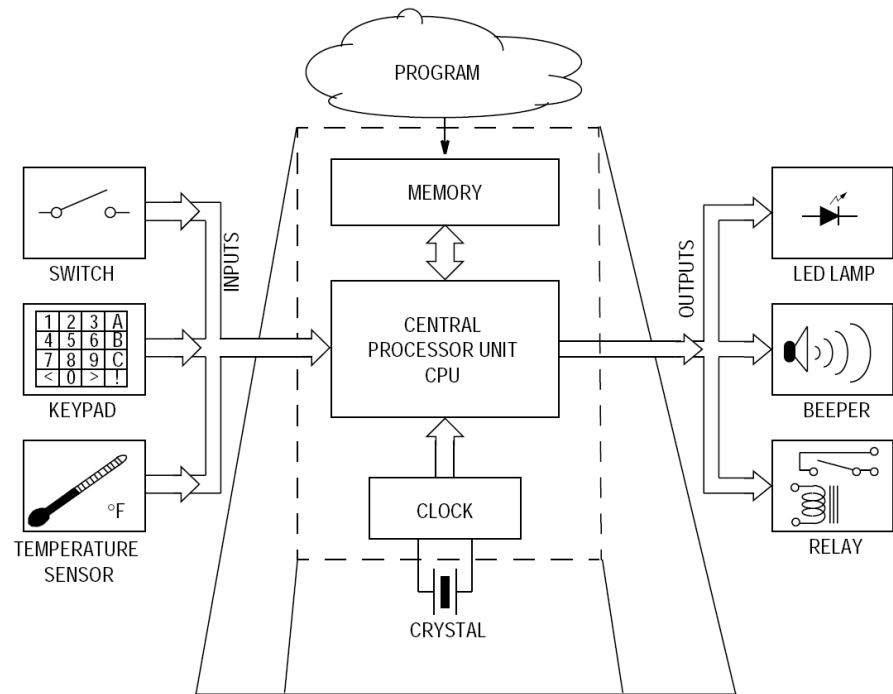


Fig. 0-1 Microcontroller versus Microprocessor

- Mali kompjuter u jednom čipu
  - Sadrži procesor, memoriju, i ulaze/izlaze
- Tipično je „**ugrađen**“ unutar nekih uređaja i kontroliše njihov rad
- Mikrokontroler je često mali i jeftin.

# Što je mikrokontroler?



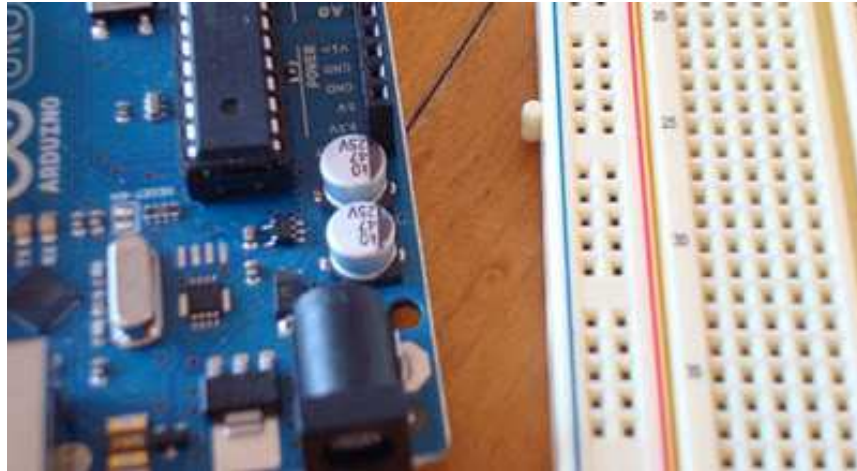
Što je razlika između 'Digital Input' i 'Analog Input'?

# Mikrokontroleri – definicija

- Programeri rade u virtuelnom svijetu.
- Uređaji rade u fizičkom svijetu.
- Kako povezati vituelni i fizički svijet?
- Uvedite mikrokontroler.
- Mikrokontroler je u osnovi mali računar koji posjeduje programabilne ulaze i izlaze opšte namjene.
- Ulazi mogu biti upravljani od strane fizičkog okruženja dok izlazi mogu upravljati fizičkim okruženjem.



# Što je razvojna ploča?



- Štampana matična ploča dizajnirana da olakša rad sa mikrokontrolerom
- Razvojna ploča tipično uključuje:
  - napojno kolo;
  - programerski interfejs;
  - Lako dostupne ulazno/izlazne pinove.

# Arduino – Zvanična definicija

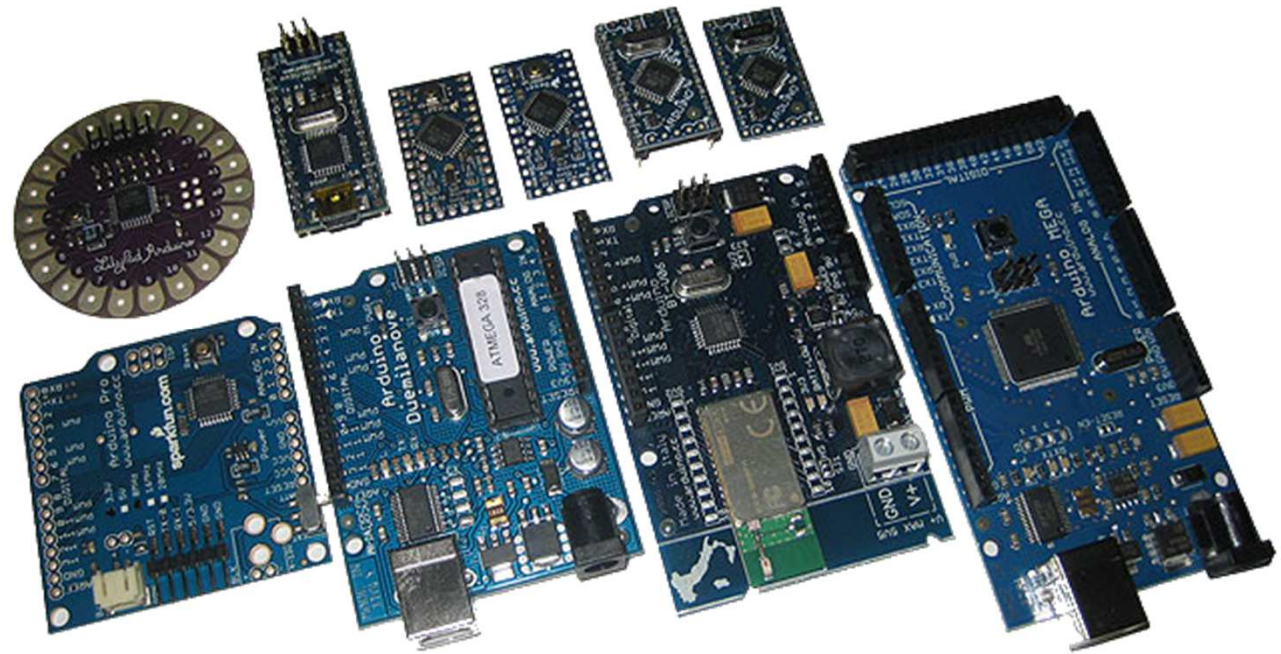
- Uzeto sa zvaničnog web sajta ([arduino.cc](http://arduino.cc)):
  - Arduino je open-source elektronska prototipna platforma zasnovana na fleksibilnom, jednostavnom za upotrebu, hardveru i softveru.
  - Namijenjen je dizajnerima, hobistima, i svima drugima koji su zainteresovani za kreiranje interaktivnih objekata i okruženja.

# Zašto Arduino?

- Bez obzira na razlog, Arduino platforma je postala de fakto standard.
  - Postoji puno realizovanih, dostupnih, projekta koji koriste arduino platformu.
- Teži ravnoteži između jednostavnosti upotrebe i korisnosti.
  - Programski jezici se uglavnom vide kao glavna poteškoća.
  - Arduino C je značajno uproštena verzija C++.
- Nije skup.

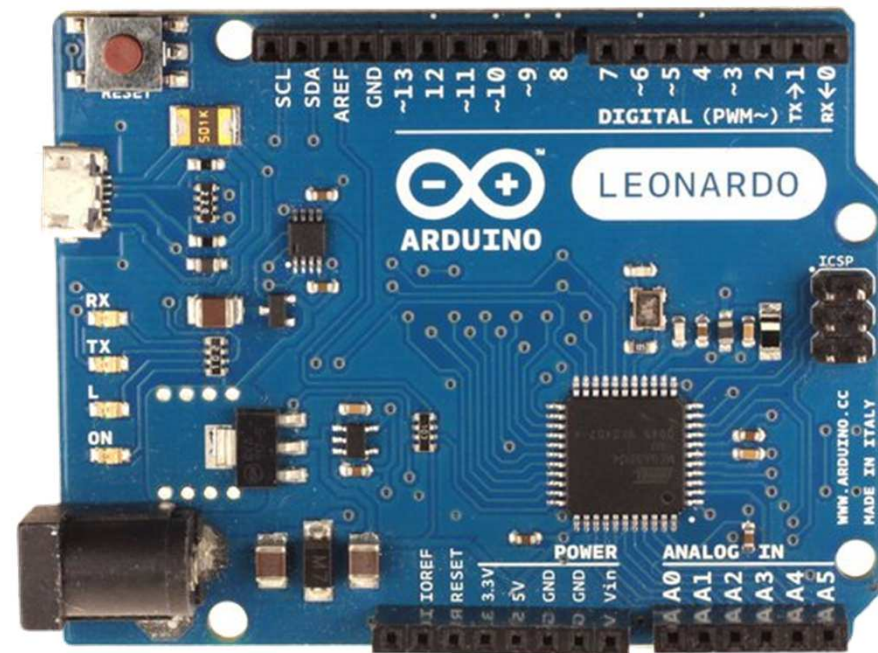
# Tipovi Arduino-a

- Više različitih verzija
  - Broj ulaznih/izlaznih kanala
  - Oblik (gabariti)
  - Procesorska snaga
- Leonardo
- Due
- Micro
- LilyPad
- Esplora
- Uno



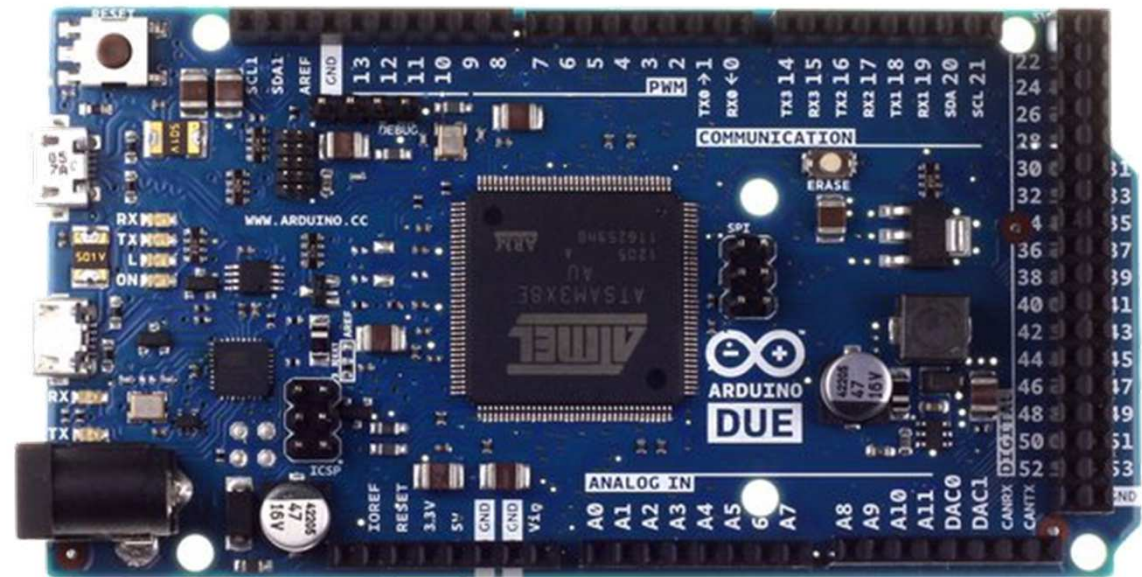
# Leonardo

- U poređenju sa Uno, malo unaprijedjen.
- Koristi ATmega32u4 mikrokontroler koji ima ugrađenu USB komunikaciju
  - Nema potrebe za dodatnim mikrokontrolerom
  - Može se prikazati PC-u kao miš ili tastatura



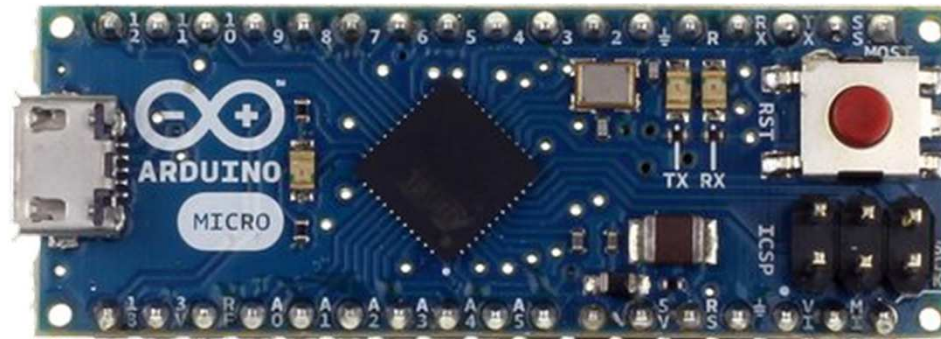
# Due

- Mnogo brži procesor, mnogo više pinova
- Radi na 3.3 volta
- Izgledom sličan Mega



# Micro

- Kad je veličina važna: Micro, Nano, Mini
- Uključuju sve funkcionalnosti Leonardo-a



# LilyPad

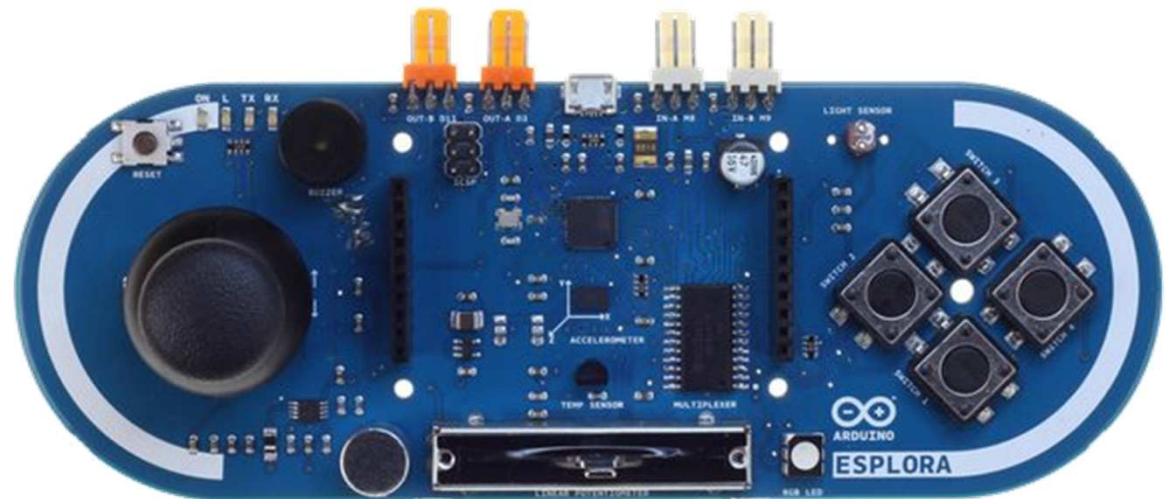
- LilyPad je pogodan za primjenu na odjeći.





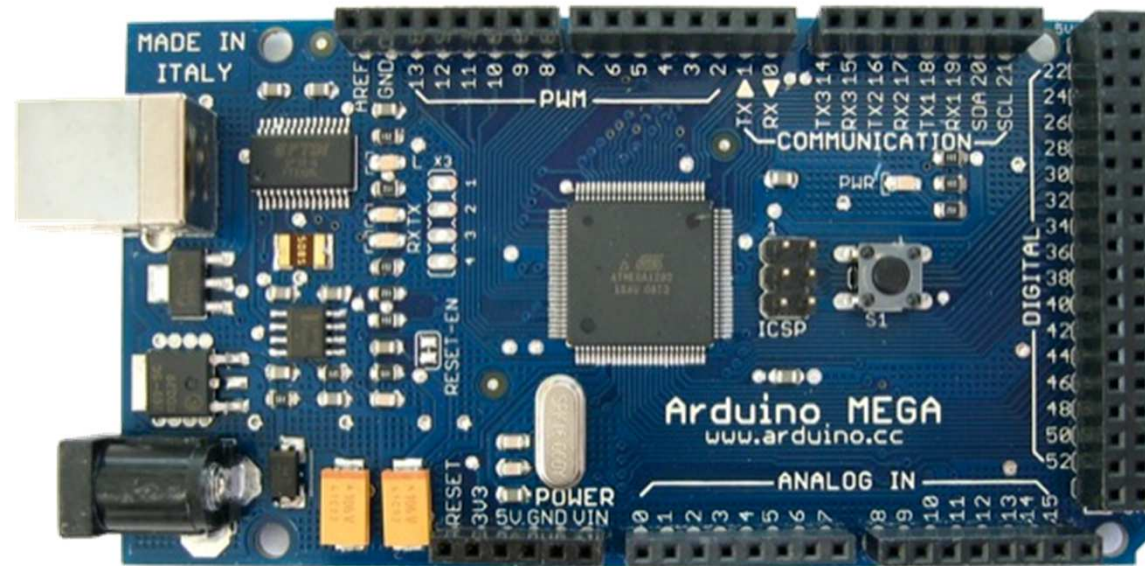
# Esplora

- Game controller
- Sadrži džojstik, četiri tastera, linearni potencijometar (klizač), mikrofon, svjetlosni senzor, senzor temperature, tro-osni akceleromatar.
- Nema standardi set IO pinova.



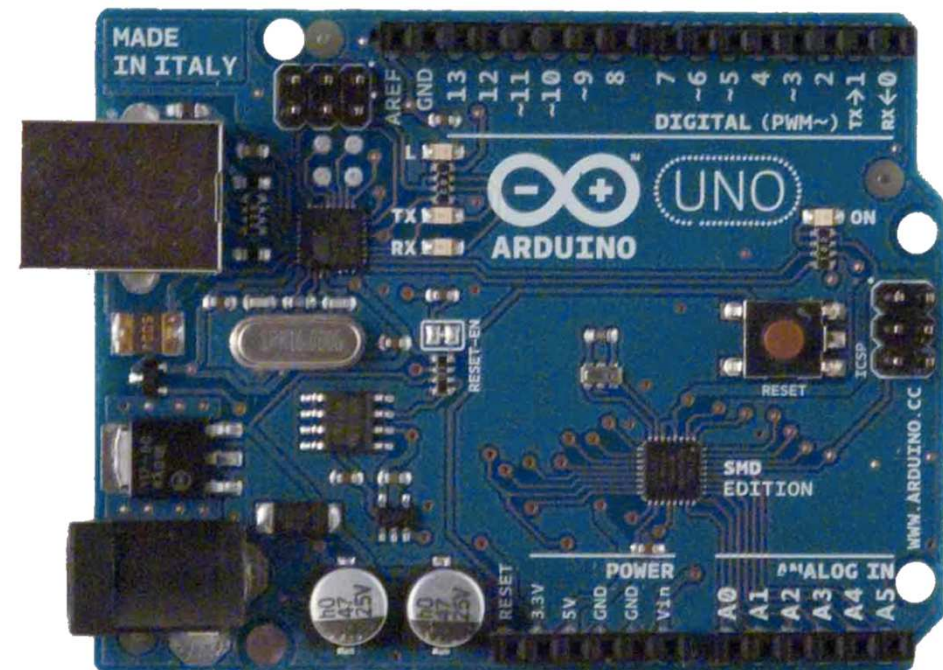
# Mega

- U poređenju sa Uno, Mega:
  - Mnogo više komunikacionih pinova
  - Više memorije

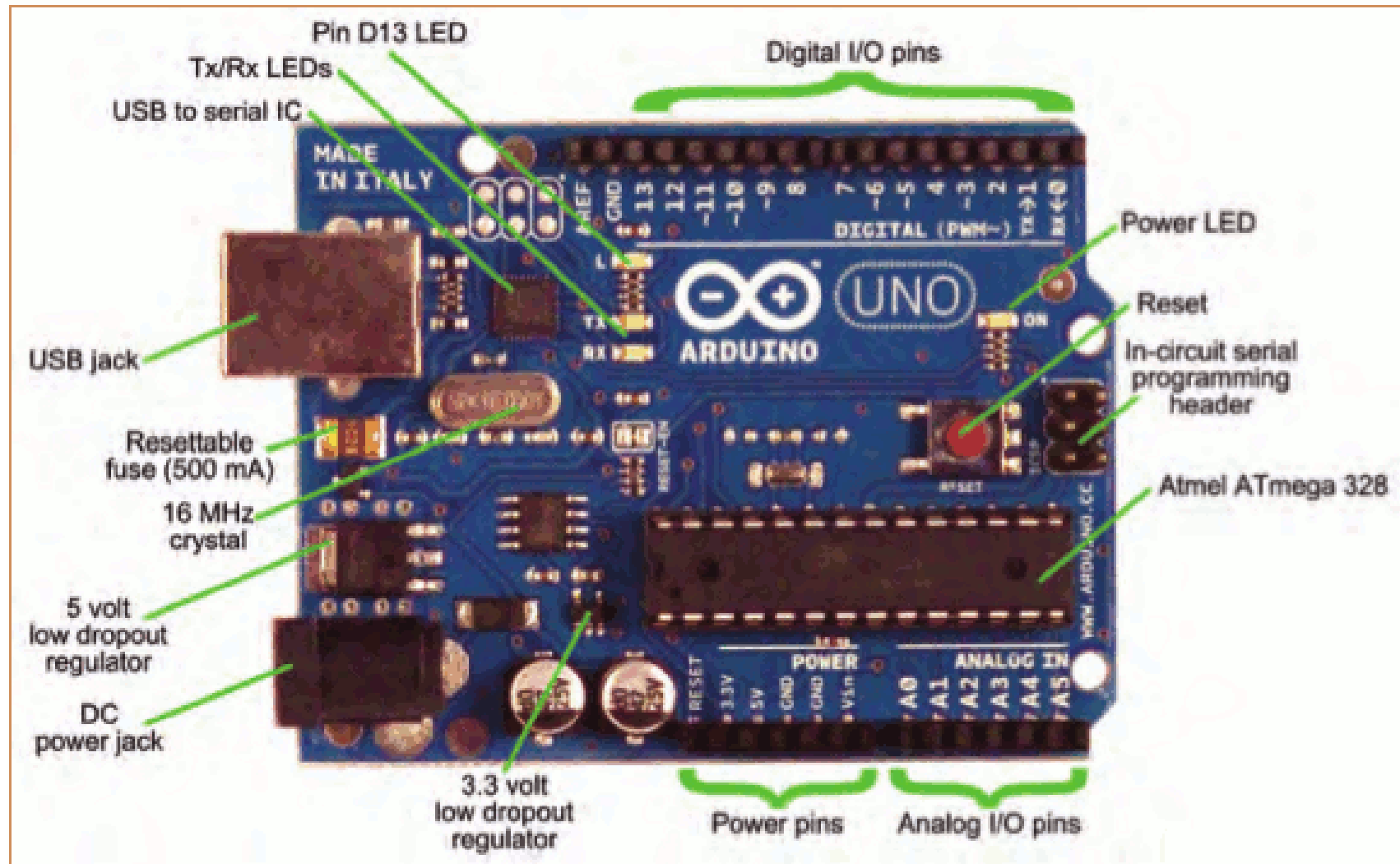


# Arduino Uno

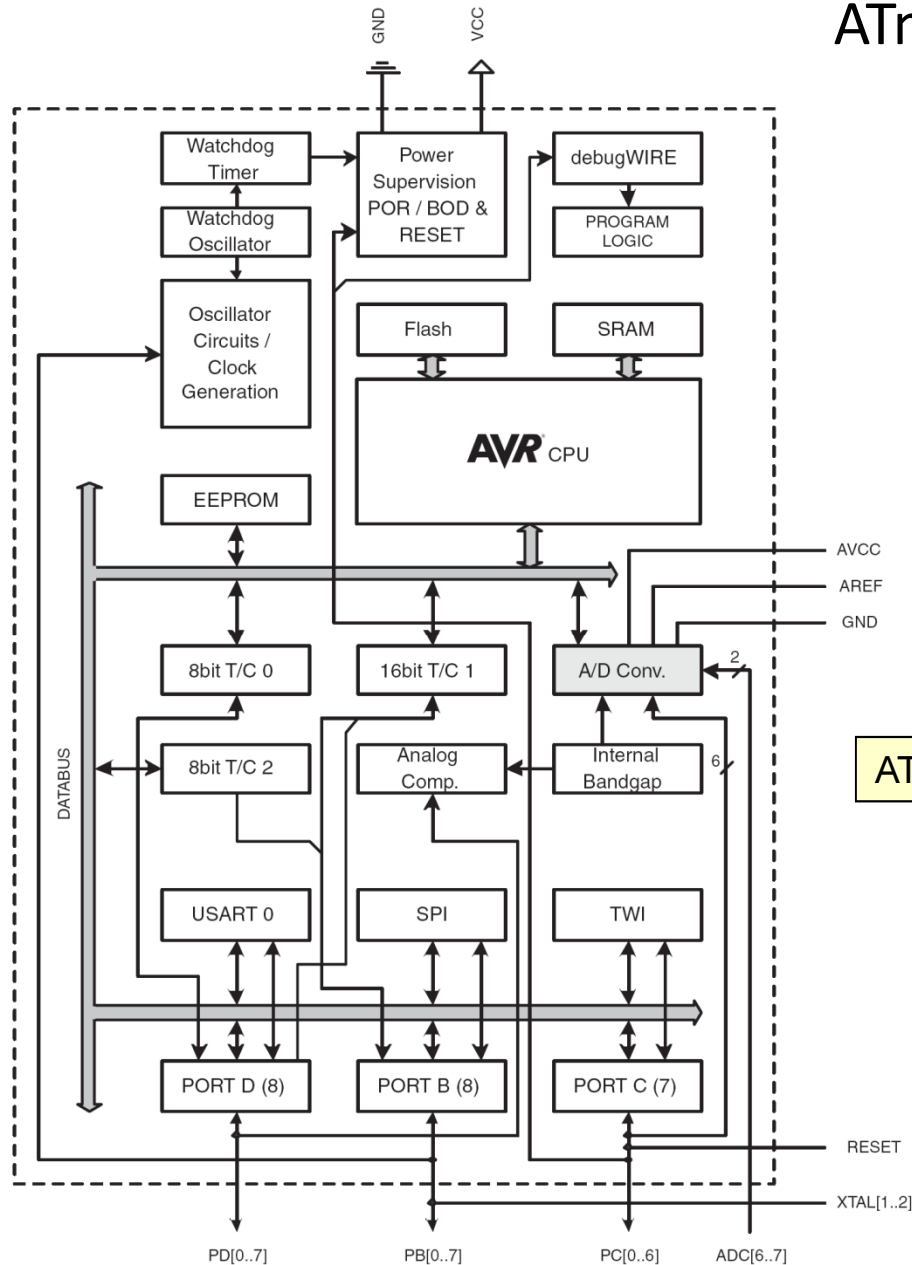
- Pinovi su grupisani u 3 grupe:
  - 14 digitalnih pinova
  - 6 analognih pinova
  - Napajanje
  - Pojavio se 2010



# Arduino Uno razvojna ploča

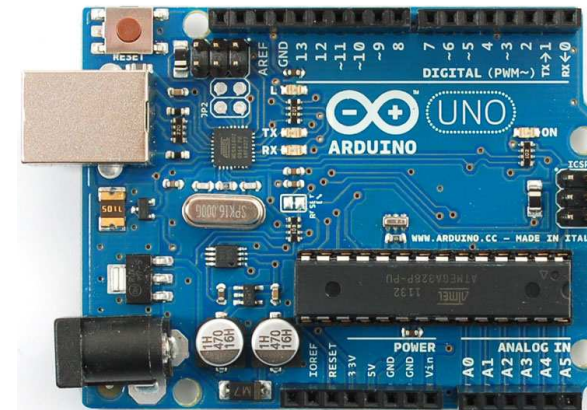


# ATmega328 unutrašnja architektura



(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 (SS/OC1B/PCINT2)
(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)

ATmega328 data sheet pp. 2, 5



# ATmega328 karakteristike

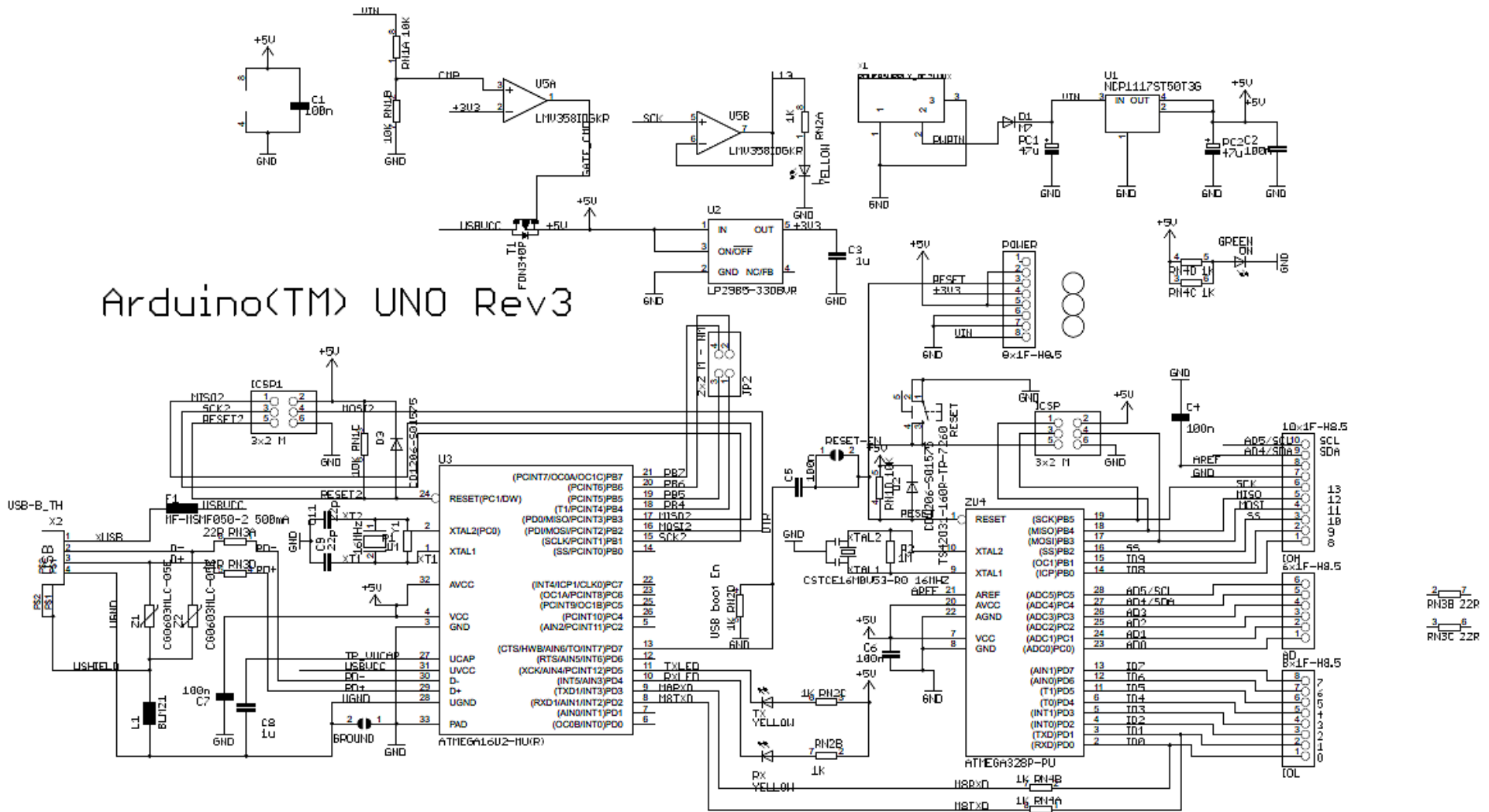
## Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1K Bytes EEPROM
  - 512/1K/1K/2K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C<sup>(1)</sup>
  - Optional Boot Code Section with Independent Lock Bits  
In-System Programming by On-chip Boot Program  
True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package  
Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package  
Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I<sup>2</sup>C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)

ATmega328 data sheet p. 1

[http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P\\_datasheet.pdf](http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet.pdf)

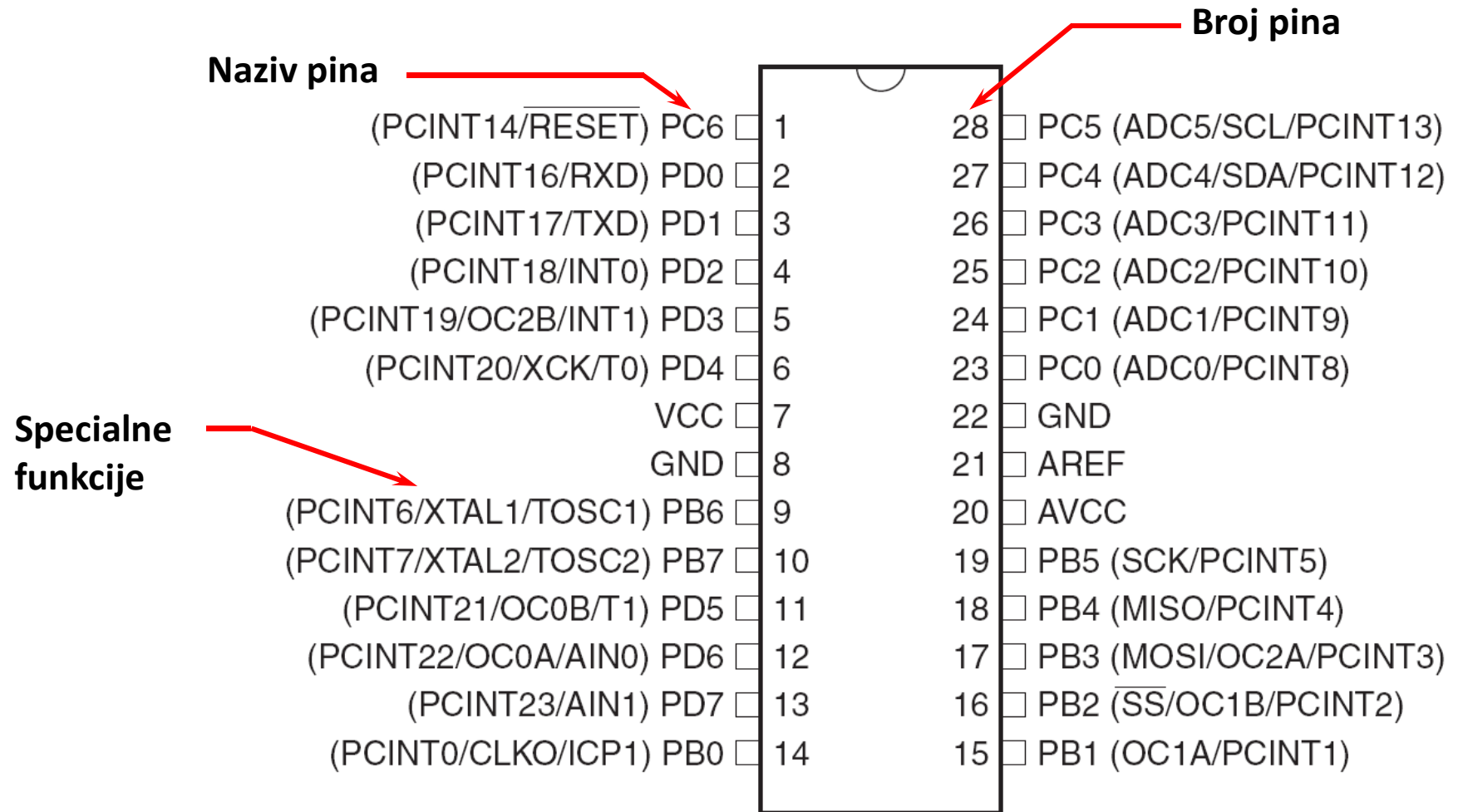
# Arduino Uno – električna šema



Arduino(TM) UNO Rev3

2 7  
R13B 22P  
3 8  
R13C 22P

# ATmega328 Microcontroller





# Absolutni maximumi

## 28.1 Absolute Maximum Ratings\*

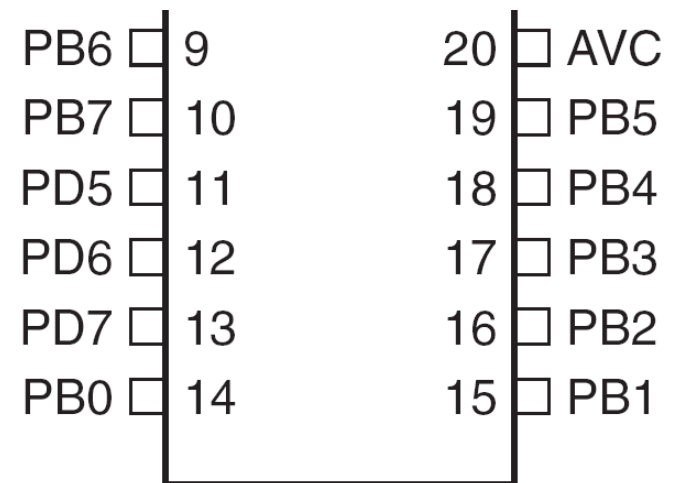
\*NOTICE:

Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Operating Temperature .....	-55°C to +125°C
Storage Temperature .....	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground .....	-0.5V to $V_{CC}+0.5V$
Voltage on $\overline{\text{RESET}}$ with respect to Ground.....	-0.5V to +13.0V
Maximum Operating Voltage .....	6.0V
DC Current per I/O Pin .....	40.0 mA
DC Current $V_{CC}$ and GND Pins .....	200.0 mA

# Microcontrolerski portovi i pinovi

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

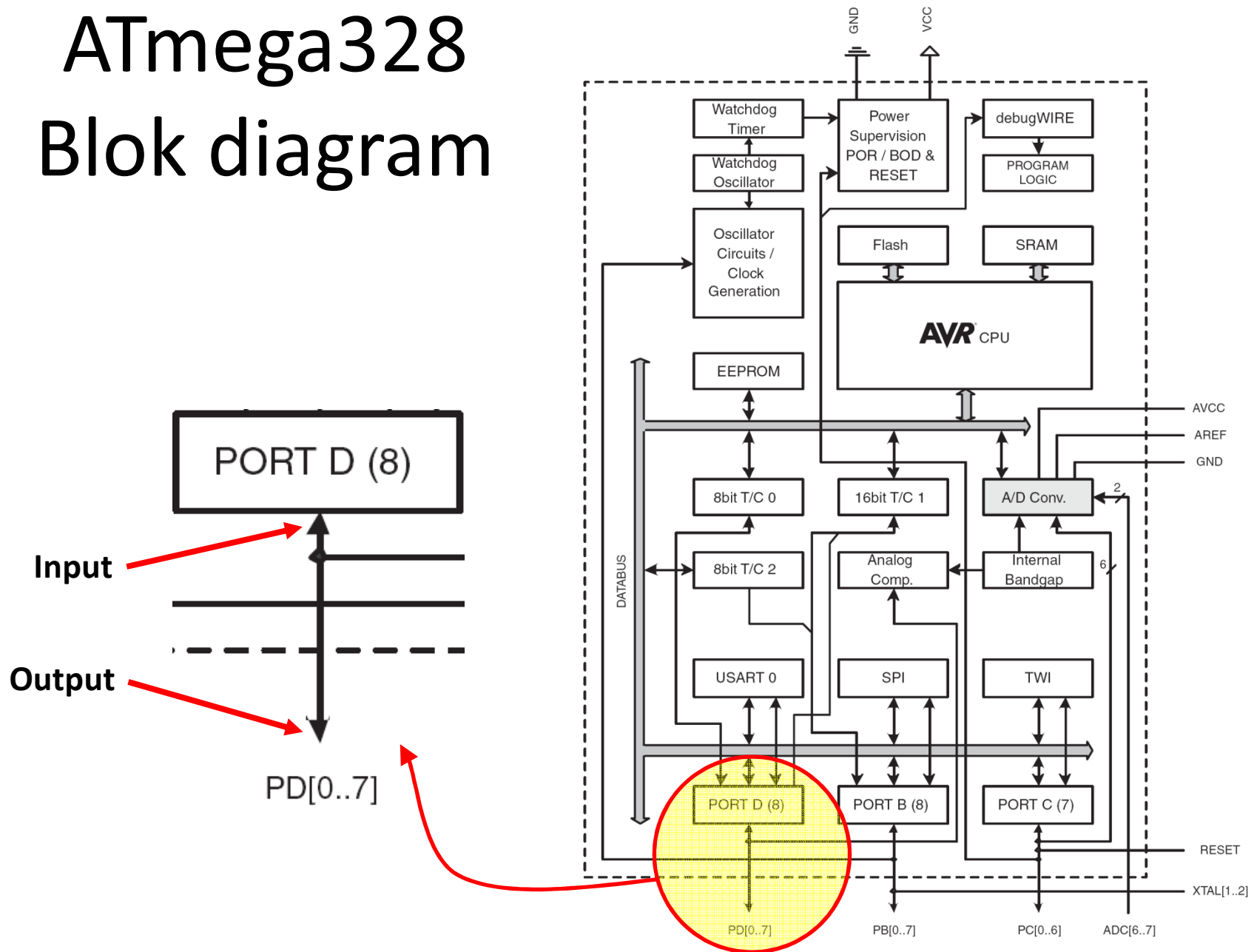


# Port Pin – Usmjerenje podataka

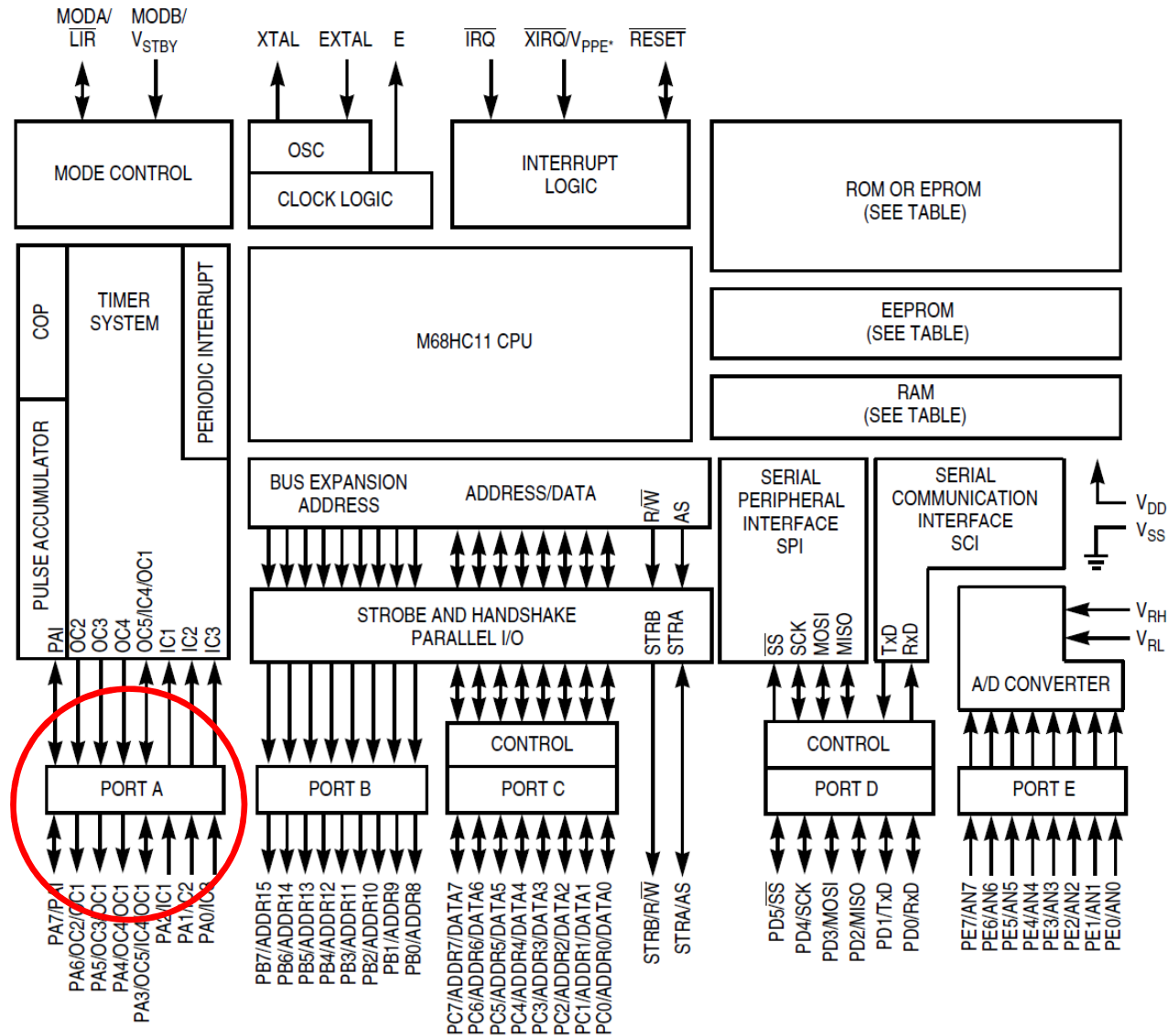
- Ulaz
  - Kada se želi uzeti informacija iz spoljašnjeg svijeta (senzori) u MCU
- Output
  - Kada se želi izmijeniti stanje nečega **izvan** MCU (uključiti ili isključiti motor, itd.)
- Po uključenju napajanja svi pinovi su ulazni.
- Program može mijenjati usmjerenja podataka za svaki pin u svakom trenutku.

# ATmega328

## Blok diagram



# M68HC11 mikrokontroler



# Postavljenje smjera toka podatka za pin

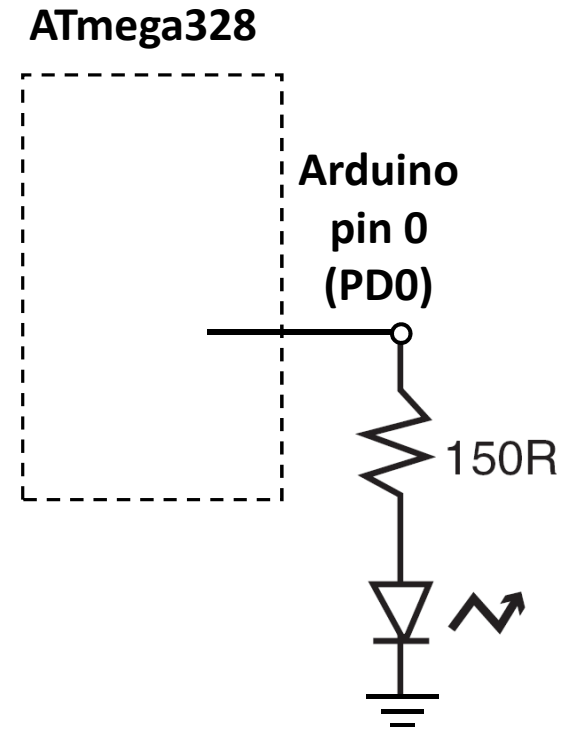
- Arduino
  - `pinMode(pin_no., dir)`
    - Pr. postaviti Arduino pin 3 (PD3) kao izlazni
      - `pinMode(3, OUTPUT);`
  - Napomena: jedan pin u jednom trenutku
    - Predpostavimo da se želi postaviti pinove 3, 5, i 7 (PD3, PD5, i PD7) kao izlazne?
    - Postoji li način da se oni postave istovremeno?
      - 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

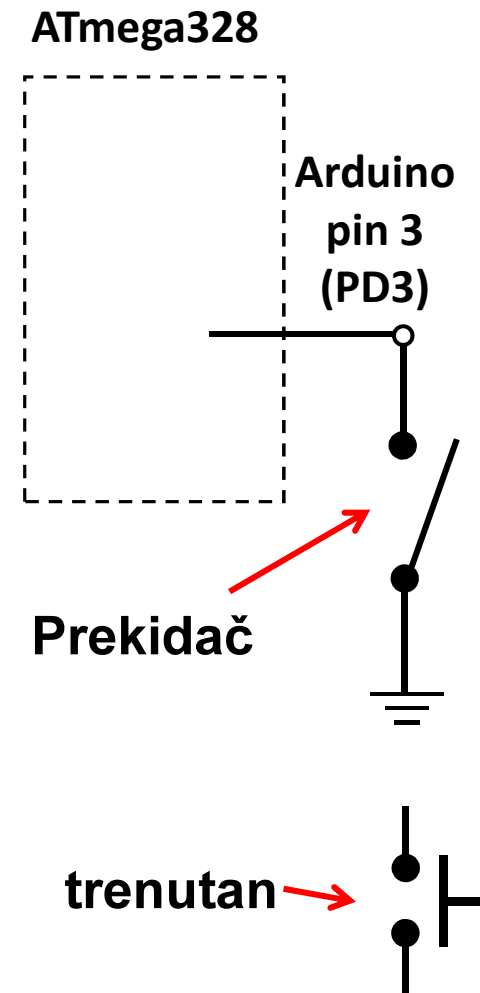
- Uključiti LED, koja je povezana na Arduino pin 0 (PD0) (otpornik!)
  - Koji tok podataka treba biti za pin 0 (PD0)?
    - `pinMode (____, ____)` ;
  - Uključenje LED
    - `digitalWrite (0, HIGH)` ;
  - Isključenje LED
    - `digitalWrite (0, LOW)` ;





# Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor
  - Pr. Senzor pojasa za sjedište u autu
  - Detekcija **stanja prekidača**
    - Koji tok podataka treba biti za Arduino pin 3 (PD3)?
    - `pinMode (____, ____)` ;
    - Koji će biti napon na PD3 kada je prekidač zatvoren?
    - Koji će biti napon na PD3 kada je prekidač otvoren?



# Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor, nastavak.

- Učinimo napon na pinu poznatim uključanjem pull-up otpornika za PD3

- Neka je PD3 ulazni port:

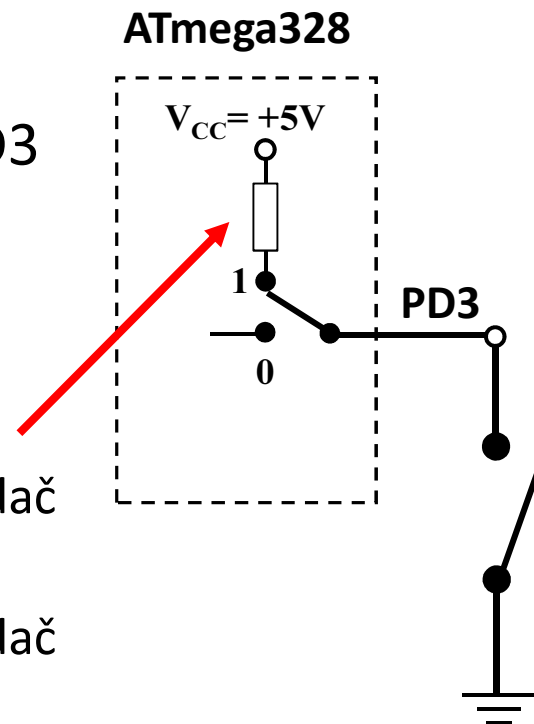
- `digitalWrite(3, HIGH);`

- uključenje “pull-up” otpornika

- `pinMode(3, INPUT_PULLUP);`

- Koji će napon biti na PD3 kada je prekidač otvoren?

- Koji će napon biti na PD3 kada je prekidač zatvoren?

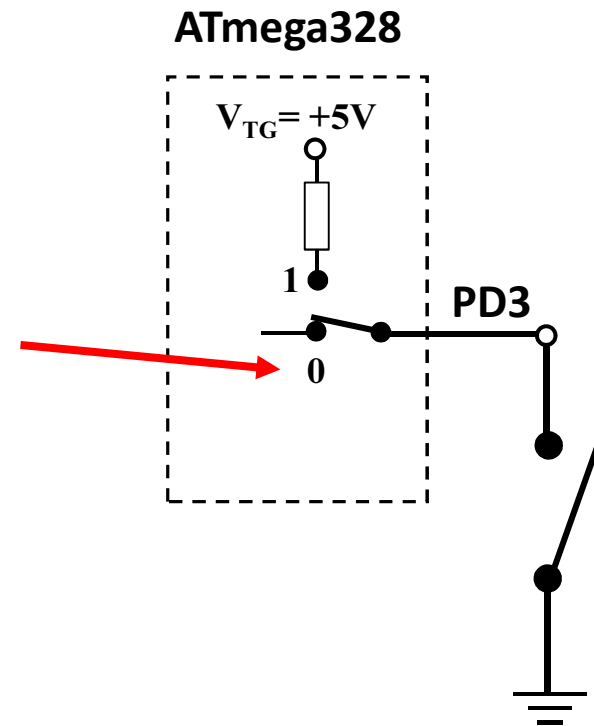


# Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor, nastavak.
  - Za isključenje pull-up otpornika
    - Neka je PD3 ulazni port:

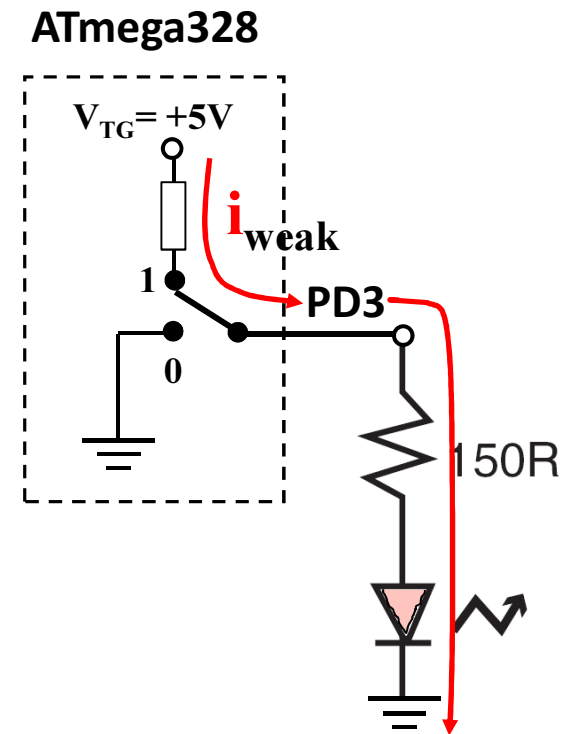
```
digitalWrite(3, LOW);
```

Isključuje "pull-up" otpornik



# Pin kao ulazni + Pull-up otpornik

- Mogućnost 'slabog pogona' kada je pull-up otpornik uključen
  - Pin koji je postavljen kao ulazni sa uključenim pull-up otpornikom može dati malu struju.
    - Zapamtiti ovo!



# I što?

- 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 na 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)	<b>DDB7</b>	<b>DDB6</b>	<b>DDB5</b>	<b>DDB4</b>	<b>DDB3</b>	<b>DDB2</b>	<b>DDB1</b>	<b>DDB0</b>	<b>DDRB</b>
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	

# 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 bit 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 detailji

ATmega328 datasheet, pp. 76-94

Figure 13-2. General Digital I/O<sup>(1)</sup>

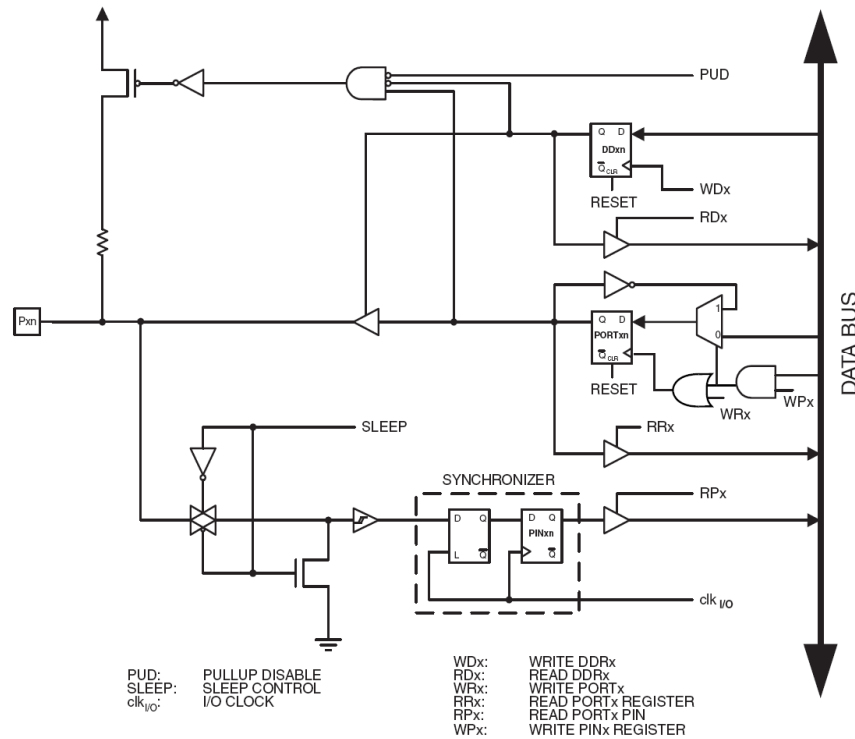
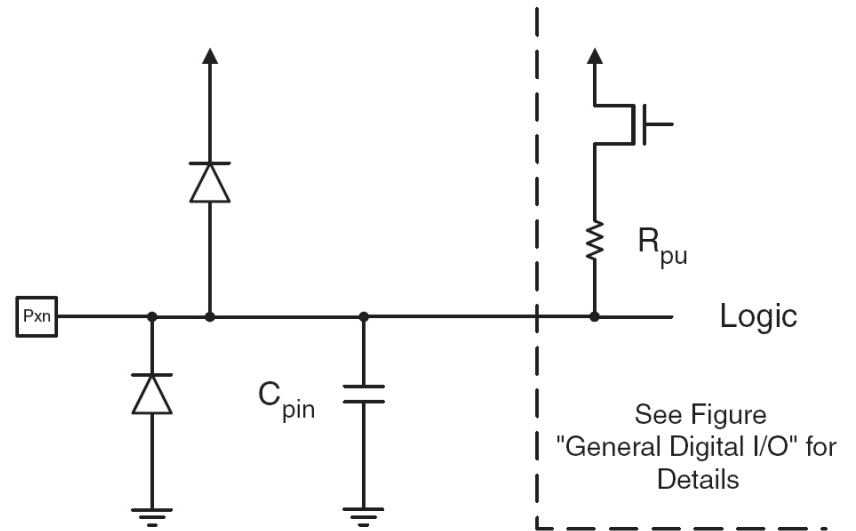


Figure 13-1. I/O Pin Equivalent Schematic



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	

[Jump to bits](#)



# 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

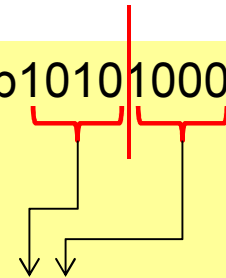
```
DDRD = 0b10101000;
```

ili

```
DDRD = 0xA8;
```

ili

```
DDRD |= 1<<PD7 | 1<<PD5 | 1<<PD3;
```



# Primjer 2

- Postaviti Arduino pinove 0 i 1 (PD0 i PD1) kao ulazne, i uključiti pull-up otpornike

- Arduino pristup

```
pinMode(0, INPUT);  
pinMode(1, INPUT);  
digitalWrite(0, HIGH);  
digitalWrite(1, HIGH);
```

Ili ako je upotrijebljena me106.h:

```
pinMode(PIN_D0, INPUT);  
pinMode(PIN_D1, INPUT);  
digitalWrite(PIN_D0, HIGH);  
digitalWrite(PIN_D1, HIGH);
```

- Alternativni pristup

```
DDRD = 0; // all PORTD pins inputs  
PORTD = 0b00000011;  
ili  
PORTD = 0x03;
```

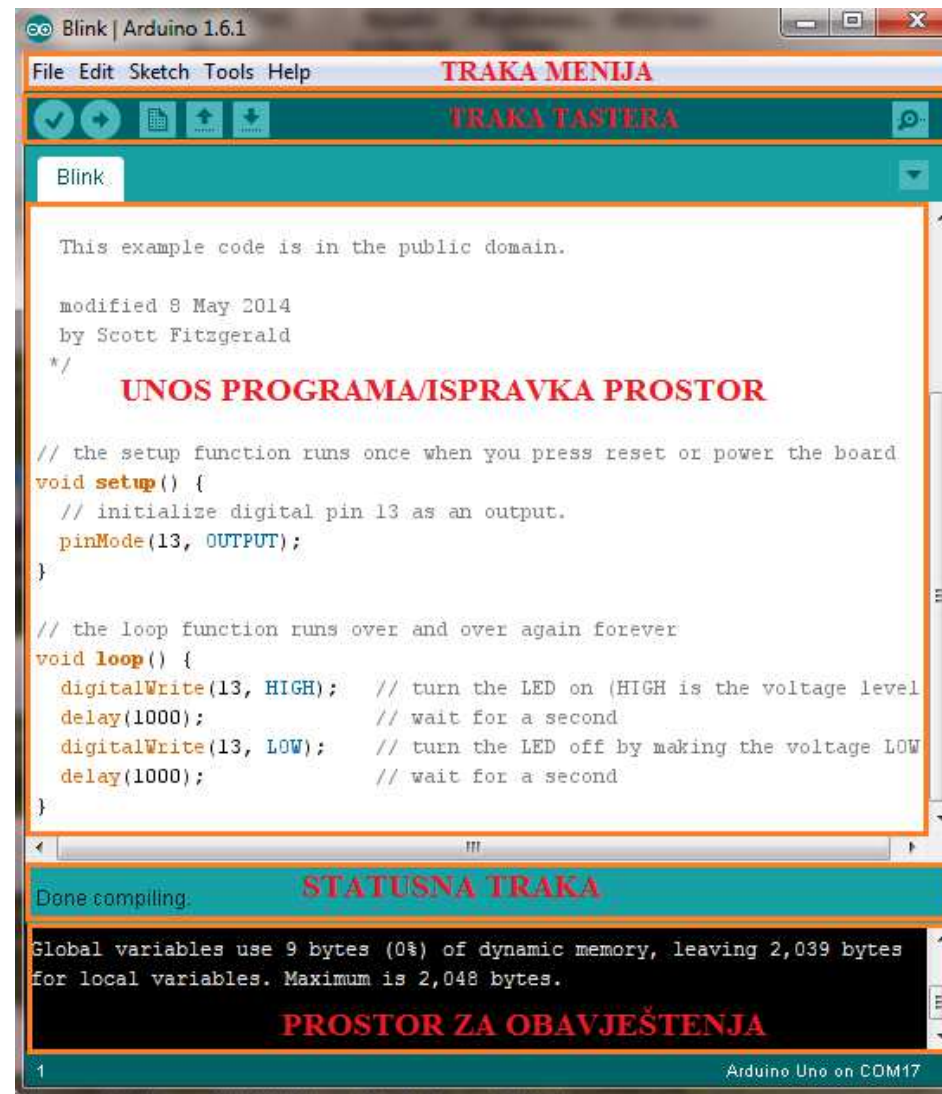
Ili još bolje:

```
DDRD &= ~(1<<PD1 | 1<<PD0);  
PORTD |= (1<<PD1 | 1<<PD0);
```

# Kako startovati?

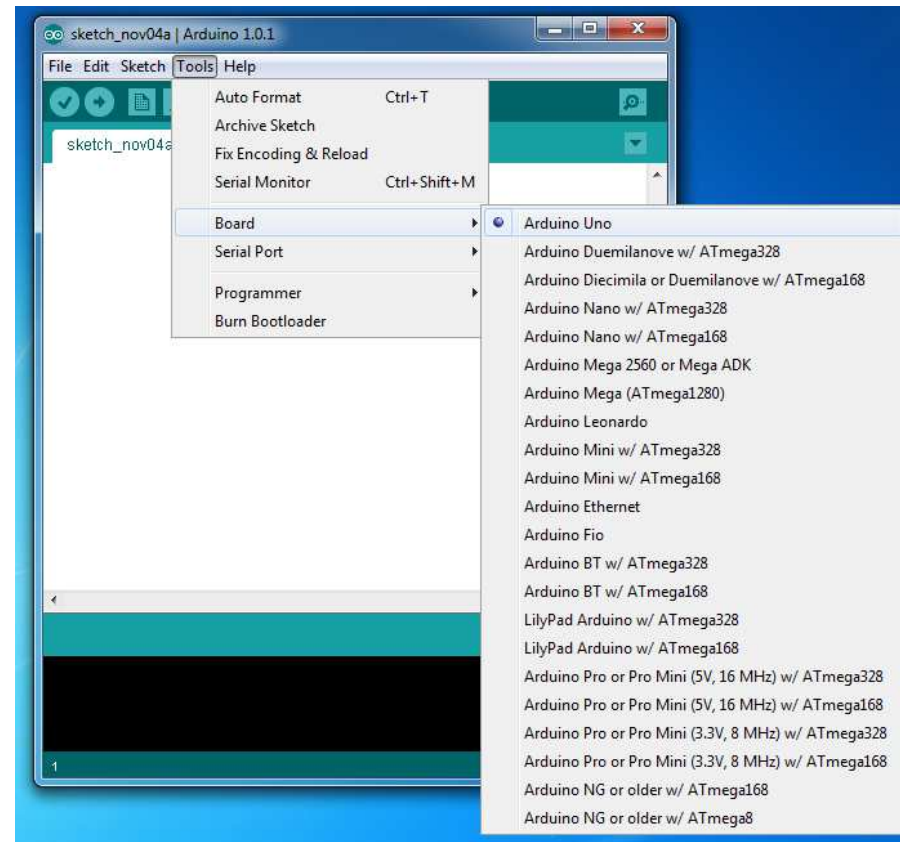
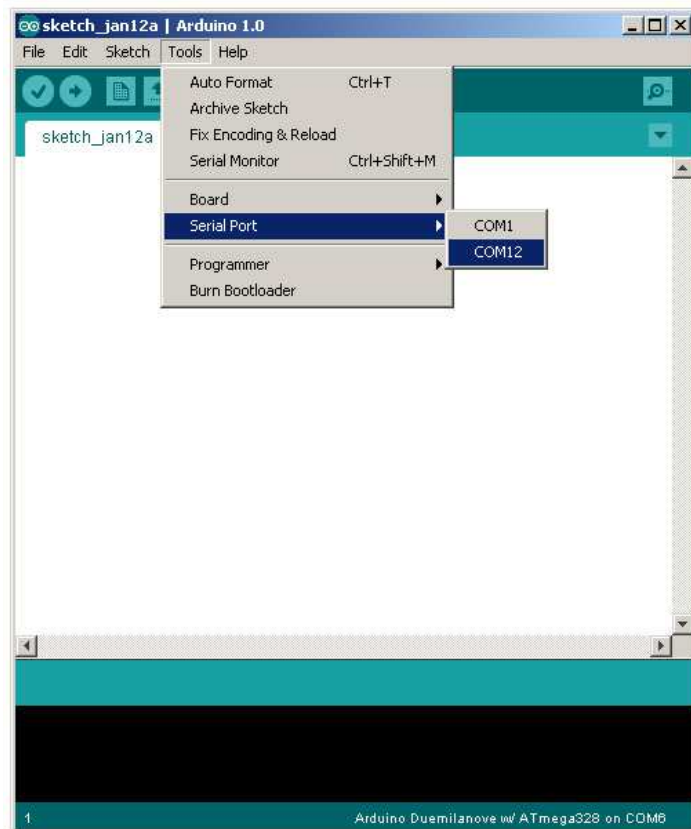
- Posjetite: <http://arduino.cc/en/Guide/HomePage>
  1. Preuzmite & instalirajte Arduino environment (IDE)
  2. Povežite ploču sa računarom pomoću USB kabla
  3. Ako je potrebno, instalirajte dodatne drajvere
  4. Pokrenite Arduino IDE
  5. Selektujte razvojnu ploču
  6. Selektujte serijski port
  7. Otvorite blink primjer
  8. Upišite program u razvojnu ploču
  - ...
  9. Pisanje vlastitog programa
  10. Nerviranje/Debugiranje/Primoravanje da radi
  11. Oduševljenje i neposredno započinjanje novog projekta
  12. (spavanje samo kad se mora)

# Arduino IDE



Pogledajte: <http://arduino.cc/en/Guide/Environment> za više informacija

# Odaberite serijski port i ploču





# Razvoj Arduino programa

- Zasnovan na C++ bez 80% komandi.
- Pregršt novih komandi.
- Programi se nazivaju 'sketches' (skečevi, skice) .
- Skečevi obavezno sadrže dvije funkcije:
  - void setup( )
  - void loop( )
- setup( ) se pokreće prvi i samo jedanput.
- loop( ) se pokreće neprestano, dok se ne isključi napajanje ili se ne učita novi skeč.

# Arduino C

- Arduino skečevi uglavnom upravljaju pinovima na arduino ploči.
- Arduino skečevi su uvijek petlja.
  - `void loop( ) { }` je isto što i `while(1) { }`

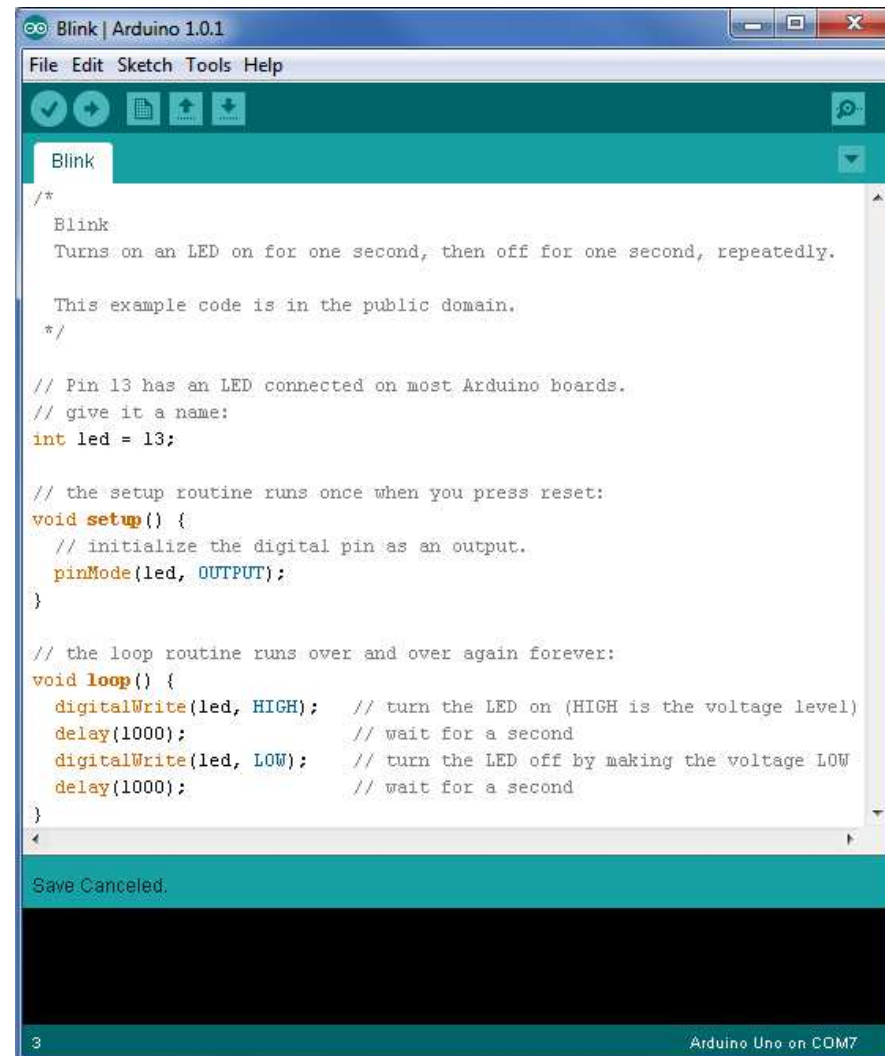


# Arduino tajming

- `delay (ms)`
  - Pauza niekoľko milisekundi
- `delayMicroseconds (us)`
  - Pauza niekoľko mikrosekundi
- Više komandi:  
[arduino.cc/en/Reference/HomePage](http://arduino.cc/en/Reference/HomePage)

# Osobine kompajlera

- Brojni jednostavni skečevi su uključeni u kompajler
- Nalaze se pod opcijom File, Examples
- Kada je skeč napisan, može se upisati u programsku memoriju mikrokontrolera na Arduino štampanoj ploči kroz opcije File, Upload, ili pritiskom na <Ctrl> U



The screenshot shows the Arduino IDE interface with the 'Blink' sketch loaded. The code is as follows:

```
Arduino 1.0.1
File Edit Sketch Tools Help

Blink

/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}

Save Canceled.

3 Arduino Uno on COM7
```

# Arduino C je izveden iz C++

- Ovaj program radi treperenje LED na pinu 13

- avr-libc

```
#include <avr/io.h>
#include <util/delay.h>
```

```
int main(void) {
    while (1) {
        PORTB = 0x20;
        _delay_ms(1000);
        PORTB = 0x00;
        _delay_ms(1000);
    }
    return 1;
}
```



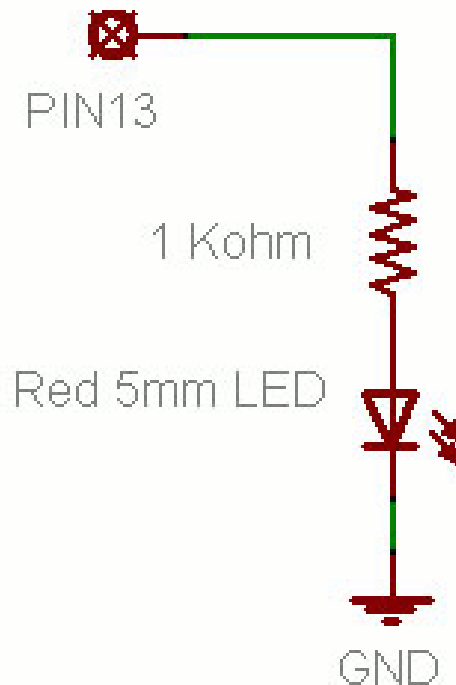
- Arduino C

```
void setup( ) {
    pinMode(13, OUTPUT);
}
```

```
void loop( ) {
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

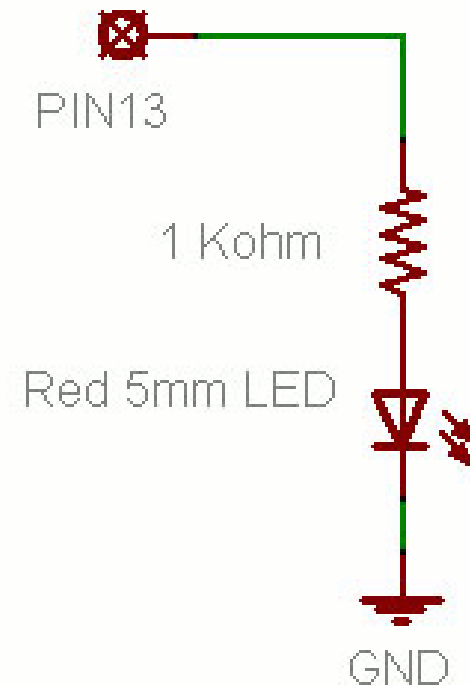
# Prosto elektronsko kolo

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



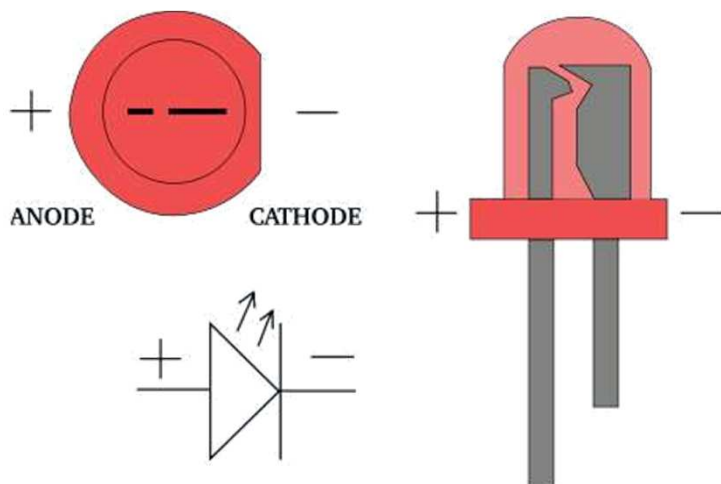
# Osnovno LED kolo

- 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.

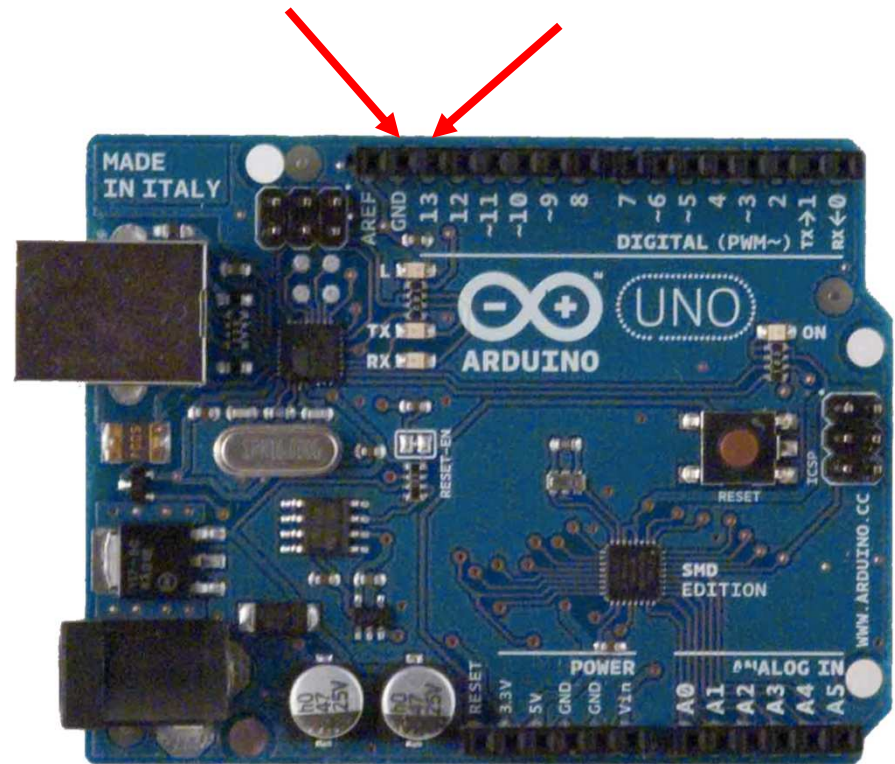


# Blink Skeč (Treperenje)

```
void setup( ) {  
  pinMode(13, OUTPUT);  
}  
void loop( ) {  
  digitalWrite(13, HIGH);  
  delay(1000);  
  digitalWrite(13, LOW);  
  delay(1000);  
}
```

Povežite jedan kraj kola

Povežite drugi kraj kola



# Struktura Arduino programa

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

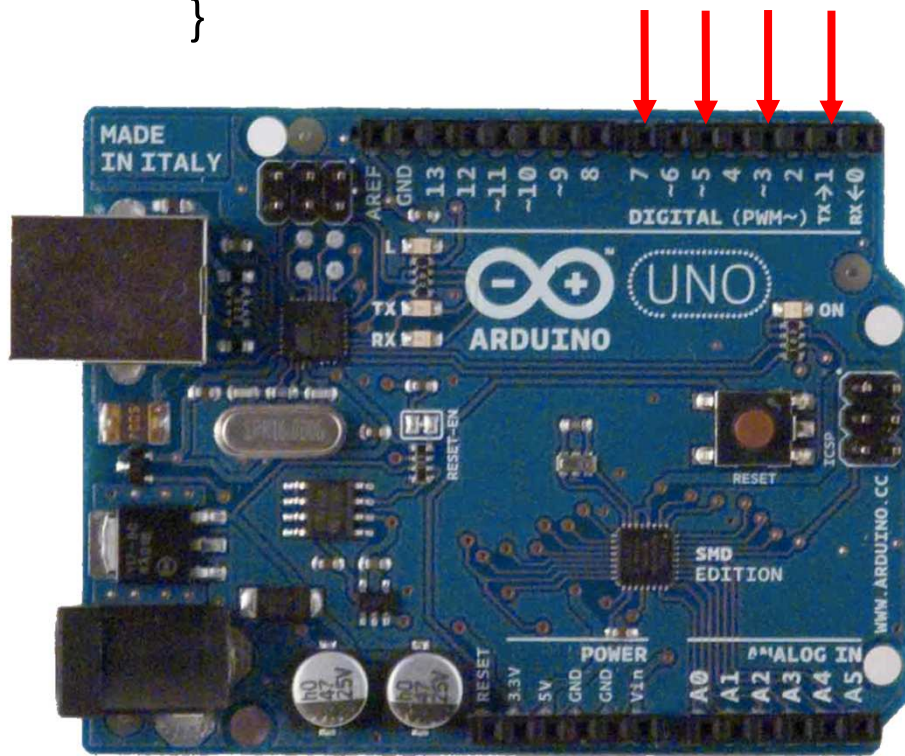
// loop() method runs forever,
// as long as the Arduino has power

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
}
```



# Treperenje 4 LED skeč

```
void setup( ) {  
  pinMode(1, OUTPUT);  
  pinMode(3, OUTPUT);  
  pinMode(5, OUTPUT);  
  pinMode(7, OUTPUT);  
}
```



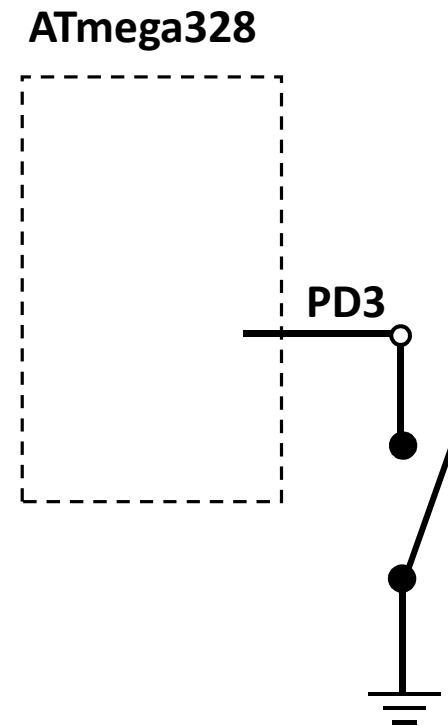
```
void loop( ) {  
  digitalWrite(1, HIGH);  
  delay (200);  
  digitalWrite(1, LOW);  
  
  digitalWrite(3, HIGH);  
  delay (200);  
  digitalWrite(3, LOW);  
  
  digitalWrite(5, HIGH);  
  delay (200);  
  digitalWrite(5, LOW);  
  
  digitalWrite(7, HIGH);  
  delay (200);  
  digitalWrite(7, LOW);  
}
```

# I što?

- Super. Treperi svjetlo. Ništa posebno.
- Obuhvatili smo samo izlazne postove za sada.
- Možemo li upotrijebiti ulaze za detekciju fizičkih pojava?

# Ulazni digitalni pin – Primjer 1

- ‘Očitavanje ulaznog pina’
  - Napisati ćemo nekoliko C linija koda za Arduino u cilju definisanja načina djelovanja kada je pojas vozača u autu vezan (prekidač zatvoren).
    - Ako je pojas vezan, omogućeno je uključenje auta kroz poziv funkcije `start_enable()`.
    - Ako pojas nije vezan omogućeno je uključenje auta kroz poziv funkcije `start_disable()`
  - Napisaćemo najprije psudokod!



# Ulazni digitalni pin – Primjer 1

- ‘Očitavanja pina’

- Pseudokod:

- Postaviti PD3 kao ulazni

- Uključiti PD3 pull-up otpornik

- Očitati napon sa Arduino pin 3 (PIN\_D3)

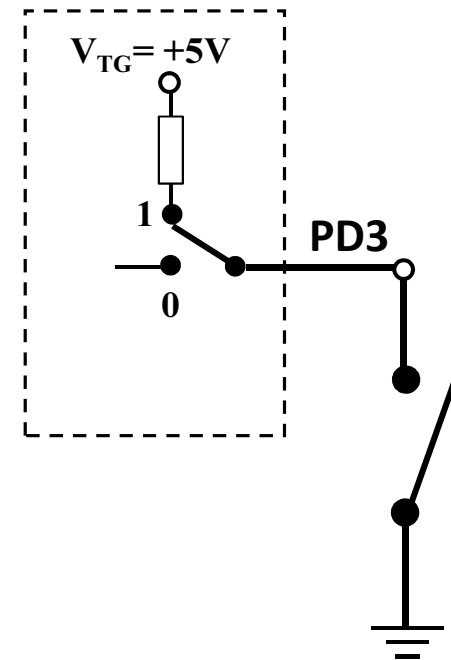
- IF PIN\_D3 napon je LOW (vezan), THEN

- pozovi funkciju start\_enable()

- ELSE

- pozovi start\_disable()

ATmega328



# Ulazni digitalni pin – Primjer 1

- ‘Očitavanja pina’

- Pseudokod:

- Postaviti PD3 kao ulazni

- Uključiti PD3 pull-up otpornik

- Očitati napon sa Arduino pin 3 (PIN\_D3)

- IF PIN\_D3 napon je LOW (vezan), THEN

- pozovi funkciju start\_enable()

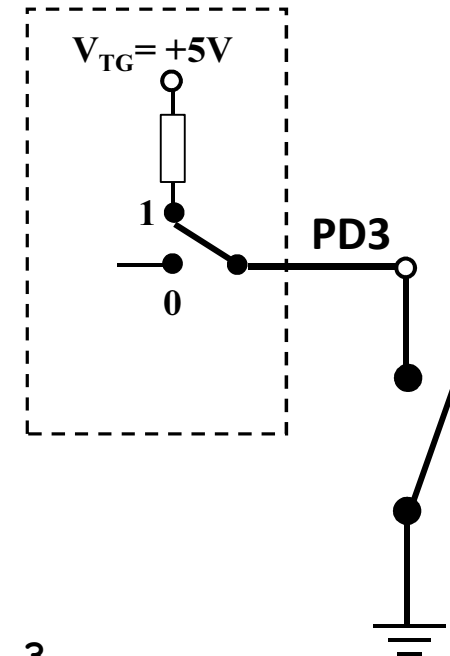
- ELSE

- pozovi start\_disable()

Fragment. Nije cijeli program.

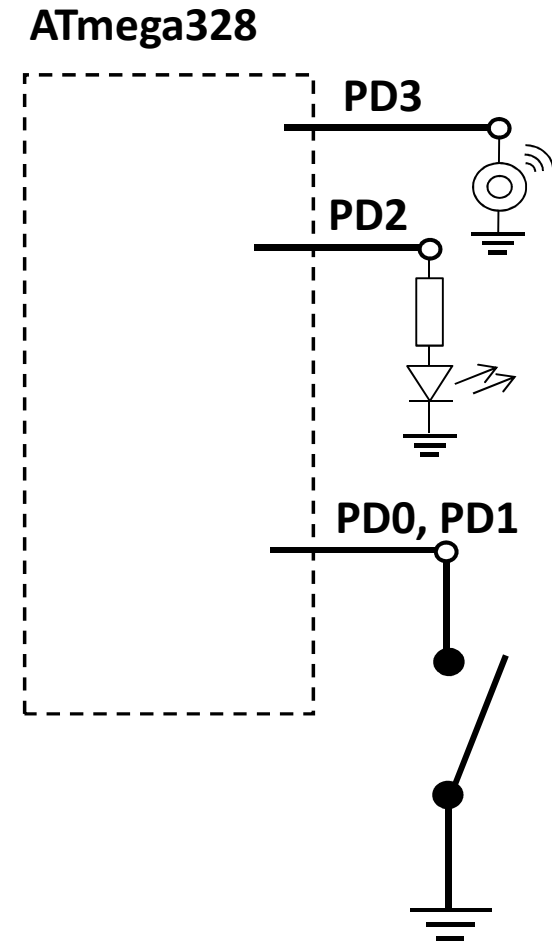
```
#define PIN_SWITCH 3
#define LATCHED LOW
pinMode(PIN_SWITCH, INPUT_PULLUP);
belt_state = digitalRead(PIN_SWITCH);
if (belt_state == LATCHED)
{ ig_enable(); }
else
{ ig_disabled(); }
```

ATmega328



# Ulazni digitalni pin – Primjer 2

- Čitanje sa pina i upisivanje na pin
  - Napisaćemo nekoliko linija C koda za Arduino, s ciljem uključenja LED (PD2) i zvučnog signala (PD3) ako je ključ u bravi (PD0 zatvoren), ali pojas vozača nije vezan (PD1 otvoren)
  - Najprije pseudokod



# Ulazni digitalni pin – Primjer 2

- Pseudokod:

Postavljanje toka podataka za pinove

Postaviti PD0 i PD1 kao ulaze

Uključiti pull-up otpornike za PD0 i PD1

Postaviti PD2 i PD3 kao izlaze

Beskonačna petlja

IF je ključ u bravi THEN

IF ako je pojas vezan, THEN

Isključi zvučni signal

Isključi LED

ELSE

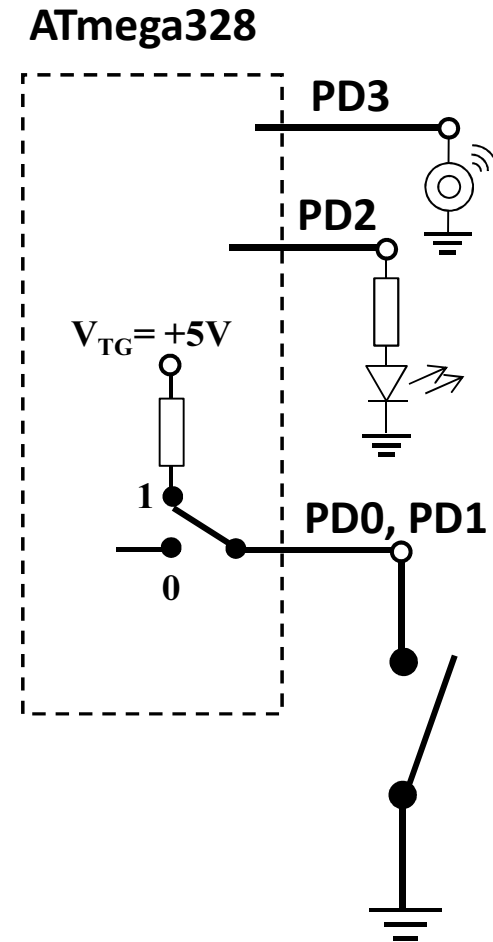
Uključi LED

Uključi zvučni signal

ELSE

Isključi zvučni signal

Isključi LED

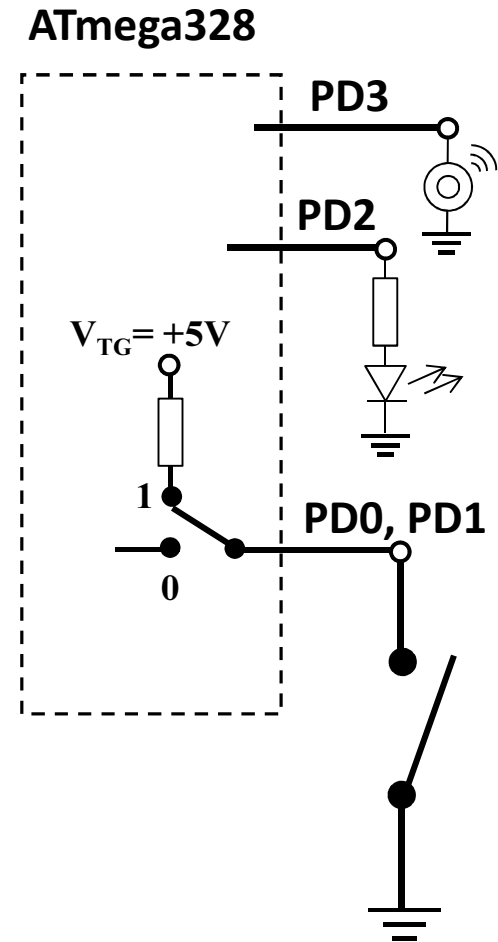


# Ulazni digitalni pin – Primjer 2 (Arduino kod)

```
#define PIN_IGNITION 0
#define PIN_SEATBELT 1
#define PIN_LED 2
#define PIN_BUZZER 3
#define SEATBELT_LATCHED LOW
#define KEY_IN_IGNITION LOW
#define LED_ON HIGH
#define LED_OFF LOW
#define BUZZER_ON HIGH
#define BUZZER_OFF LOW

void setup()
{
  pinMode(PIN_IGNITION, INPUT_PULLUP); // key switch
  pinMode(PIN_SEATBELT, INPUT_PULLUP); // belt latch switch
  pinMode(PIN_LED, OUTPUT); // lamp
  pinMode(PIN_BUZZER, OUTPUT); // buzzer
}

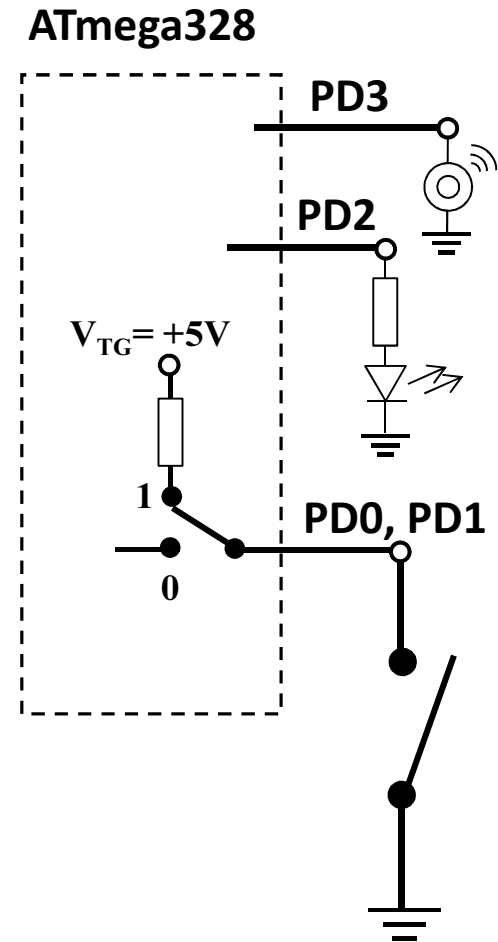
/* see next page for code */
```





## Ulazni digitalni pin – Primjer 2 (Arduino kod)

```
/* see previous page for code before loop() */
void loop()
{
  int key_state = digitalRead(PIN_IGNITION);
  int belt_state = digitalRead(PIN_SEATBELT);
  if (key_state == KEY_IN_IGNITION)
  {
    if (belt_state == SEATBELT_LATCHED)
    {
      digitalWrite(PIN_BUZZER, BUZZER_OFF);
      digitalWrite(PIN_LED, LED_OFF);
    }
    else // key is in ignition, but seatbelt NOT latched
    {
      digitalWrite(PIN_BUZZER, BUZZER_ON);
      digitalWrite(PIN_LED, LED_ON);
    }
  }
  else // key is NOT in ignition
  {
    digitalWrite(PIN_BUZZER, BUZZER_OFF);
    digitalWrite(PIN_LED, LED_OFF);
  }
}
```



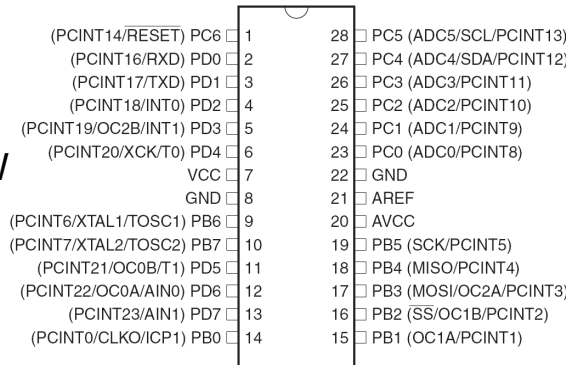
# Ulazni digitalni pin – Primjer 2 (Alternativni kod)

/\* NOTE: #defines use predefined PORT pin numbers for ATmega328 \*/

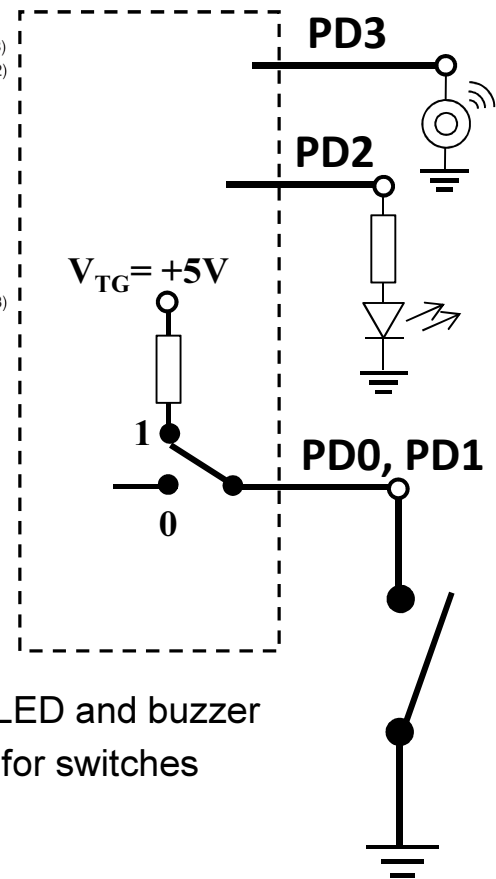
```

#define PIN_IGNITION PD0
#define PIN_SEATBELT PD1
#define PIN_LED PD2
#define PIN_BUZZER PD3
#define SEATBELT_LATCHED LOW
#define KEY_IN_IGNITION LOW
#define LED_ON HIGH
#define LED_OFF LOW
#define BUZZER_ON HIGH
#define BUZZER_OFF LOW
#define _BIT_MASK( bit ) ( 1 << ( bit ) ) // same as _BV( bit )
void setup()
{
    PORTD = 0; // all PORTD pullups off
    DDRD = _BIT_MASK(PIN_LED) | _BIT_MASK(PIN_BUZZER); // LED and buzzer
    PORTD |= _BV(PIN_IGNITION) | _BV(PIN_SEATBELT); // pullups for switches
}

```



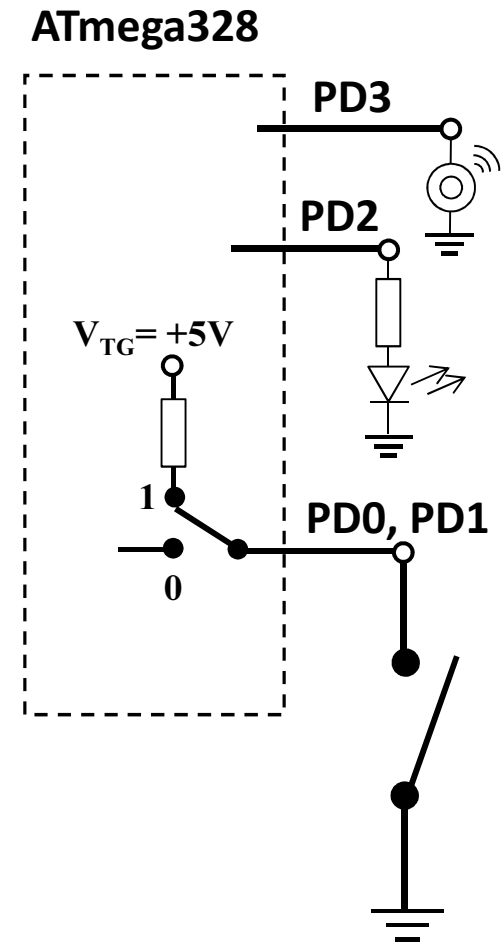
ATmega328



/\* See next page for loop() code \*/

# Ulazni digitalni pin – Primjer 2 (Arduino kod)

```
/* see previous page for setup() code */  
void loop()  
{  
  uint8_t current_PORTD_state, key_state, belt_state;  
  current_PORTD_state = PIND; // snapshot of PORTD pins  
  key_state = current_PORTD_state & _BV(PIN_IGNITION);  
  belt_state = current_PORTD_state & _BV(PIN_SEATBELT);  
  if (key_state == KEY_IN_IGNITION)  
  {  
    if (belt_state == SEATBELT_LATCHED)  
    {  
      PORTD &= ~(_BV(PIN_LED) | _BV(PIN_BUZZER));  
    }  
    else  
    {  
      PORTD |= (_BV(PIN_LED) | _BV(PIN_BUZZER));  
    }  
  }  
  else  
  {  
    PORTD &= ~(_BV(PIN_LED) | _BV(PIN_BUZZER));  
  }  
}
```



# Analogni ulazi

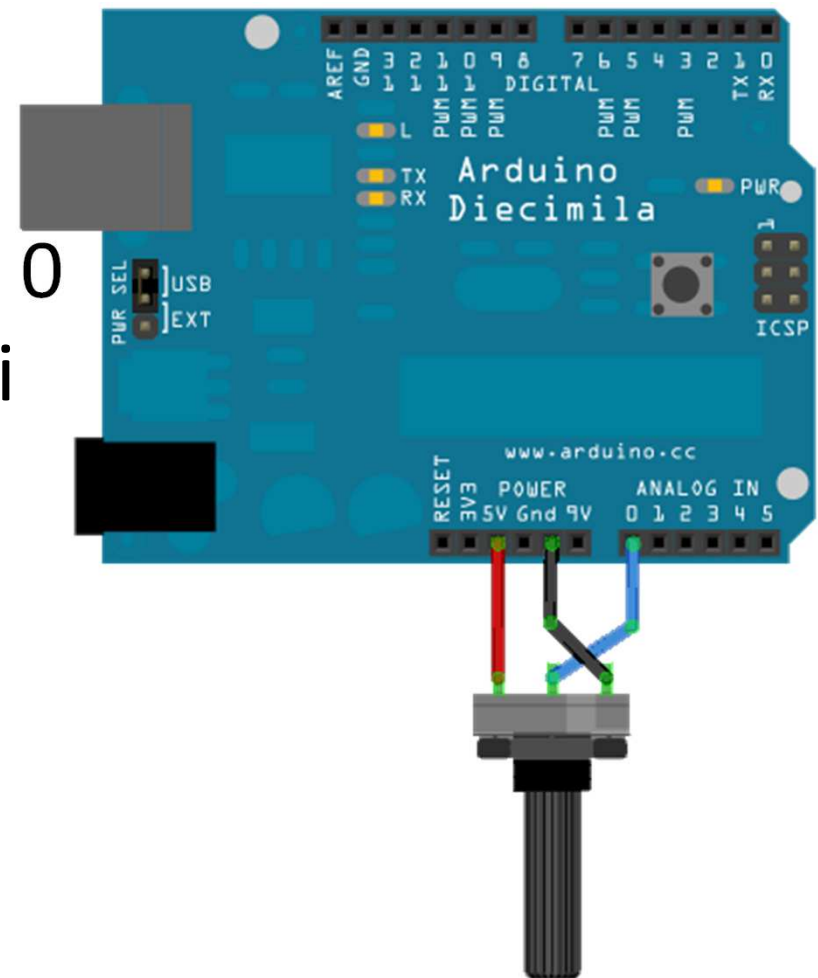
Analogni ulazi Arduina se mogu očitati kao opseg brojeva, u zavisnosti od električnih karakteristika kola.

- 0 to 1023
- .0049 V per digit (4.9 mV)
- Vrijeme očitavanja je 100 $\mu$ s (10,000 puta u secondi)

# Analogni ulaz

- Potenciometar (promjenjivi otpornik) priključen je na analogni pin 0 Arduinoa.
- Vrijednost napona na pinu 0 veoma zavisi od otpornosti potenciometra, odnosno pozicije klizača.

```
napon=AnalogRead(0);
```



# Analogni ulazi-primjene

- Promjenljivi otpornik se može zamijeniti sa senzorom.
- Na primjer foto-otpornik.
  - Zavisno od nivoa osvjetljaja foto otpornika može se:
    - Uključiti LED
    - Pojačati ili smanjiti intezitet sijanja LED (ili LED niza)
- Mnogi senzori su jednostavno promjenljivi otpornici. Otpornost im se mijenja sa promjenom nekih fizičkih karakteristika okoline.

# Senzori

- Senzori mogu biti digitalni ili analogni.
- Obično, senzori koji mjere opseg vrijednosti mijenjaju svoju otpornost.
- Arduino može senzorisati samo napon, ne otpornost.
- U cilju obezbjedjenja napona Arduino, senzori koji mijenjaju svoju otpornost zahtijevaju dodatno, često naponski djelilac.

# Razni senzori

- Temperatura

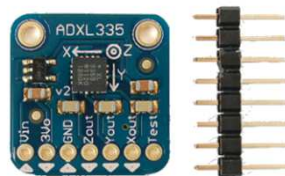


- Ugao

- Pekidači

- Je li korisnik zatvorio prekidač ili pritisnuo taster?

- Akcelerometar

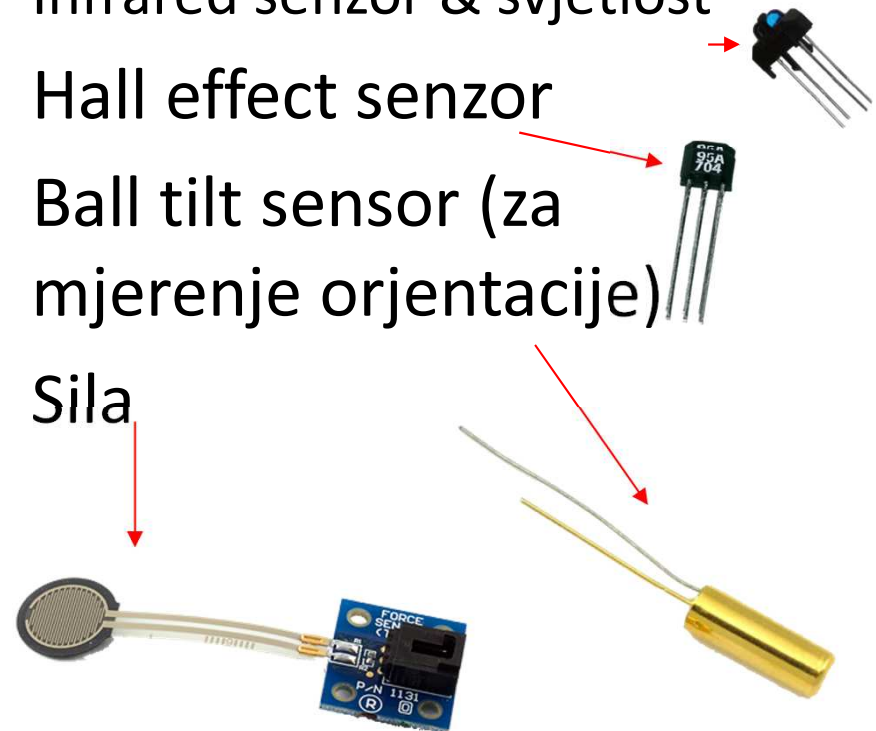


- Infrared senzor & svjetlost

- Hall effect senzor

- Ball tilt sensor (za mjerenje orijentacije)

- Sila





# “Konkurencija” Arduinu

- Alternative Arduino platformi
  - Pinguino – PIC kontroler
  - MSP430 – Texas Instruments;
  - Drugi: korisnički, Teensy, itd.
- Netduino
- Računari
  - Raspberry Pi
  - BeagleBones – TI; ima računar i kontroler

# Netduino

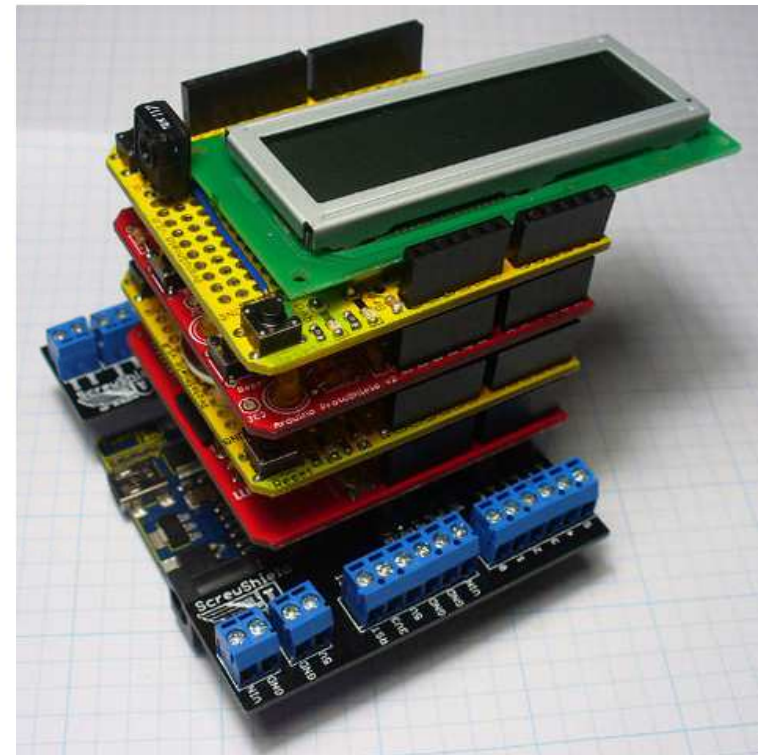
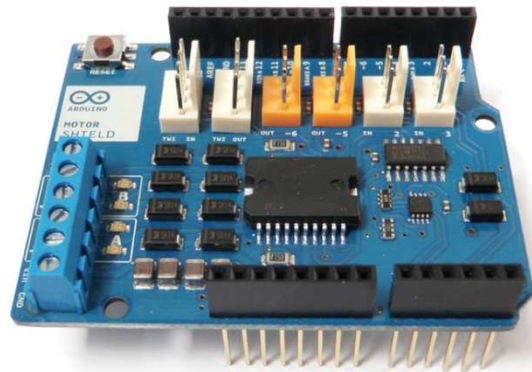
- Mikrokontroler i razvojno okruženje kreirano od strane Microsoft za rad sa .NET Micro Framework.
- DALEKO moćnije razvojno okruženje.
- Razlike
  - Pinovi na Netduino su 3.3V (ne 5V).
  - Netduino ima mnogo brži procesor
  - 60K RAM-a (naspram Uno-vih 2K).
- U velikoj mjeri kompatibilan sa Arduino-om, ali nije potpuno izmjenjiv.

# Raspberry Pi

- Pojednostavljeni računar, ne kontroler
- Koristi Debian Linux
  - Arch Linux ARM, Fedora, FreeBSD, Slackware...
- Programiran sa Python-om
  - BBC BASIC, C, Perl
- Kako je u pitanju računar (ime operativni sistem), ima drugačiju ulogu u projektima.
- Hijerarhija: računar upravlja kontrolerima, kontroleri upravljaju hardverom.

# Šildovi (Dodaci)

- Šildovi su ploče koje se dodaju na Arduino ploču.
- Oni proširuju mogućnosti Arduina.
- Primjeri:
  - Ethernet
  - GPS
  - Motor
  - Prototip
- [shieldlist.org](http://shieldlist.org)

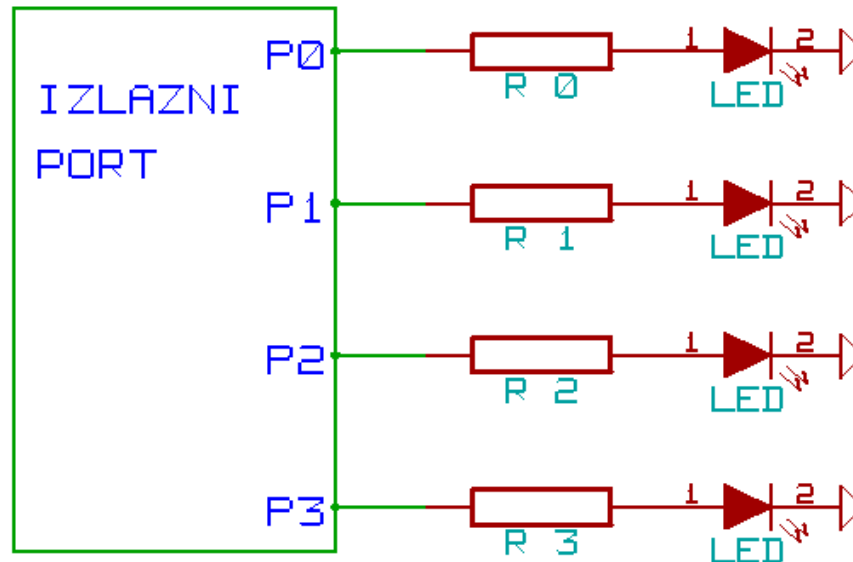


# Zaključak

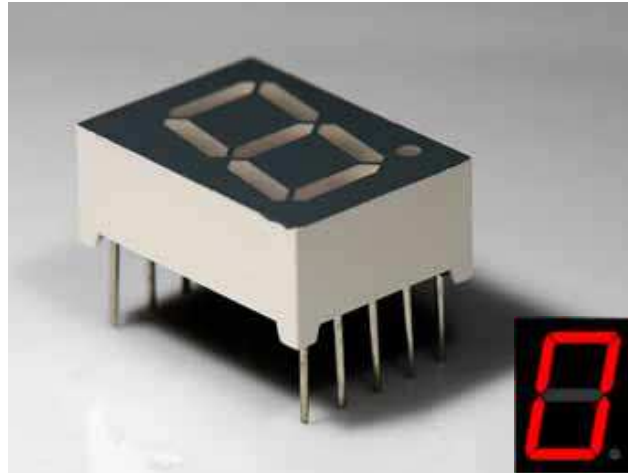
- Arduino platforma predstavlja jeftin način da se uđe u svijet robotike.
- Arduino ima:
  - Brojne korisnike
  - Bogatu online biblioteku kodova i projekata

# Zadaci za vježbu 1

1. Pomocu 4 LED, u binarnom obliku prikazati vrijednost promjenjive BROJAC. Vrijednost promjenjive brojac se inkrementira svake sekunde. **(2 poena)**
2. Trcece svjetlo upotrebom 4 LED. Uvijek je samo jedna dioda uključena. **(3 poena)**



3. Napisati program koji broji od 0 do 9, s ponavljanjem, i prikazuje rezultat na sedmosegmentnom LED displeju. (4 poena)



Pomoć:

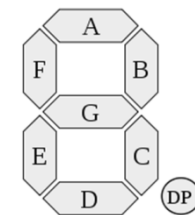
Razmotrite pisanje funkcije:

```
void writeDigit(int n)
```

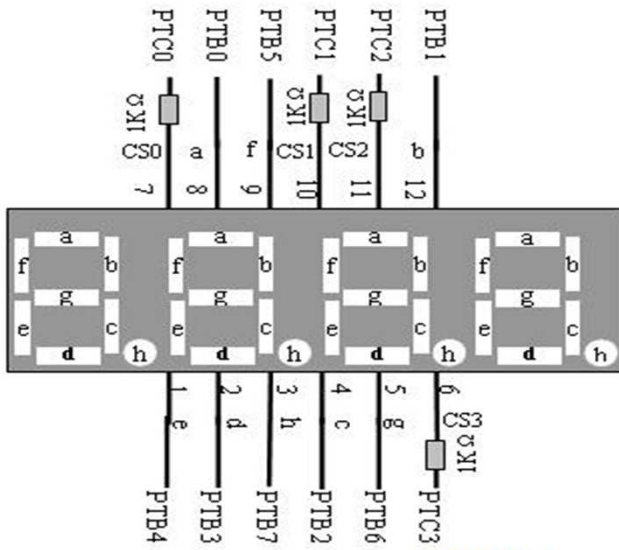
koja ispisuje jednu cifru

Trasformaciona tabela

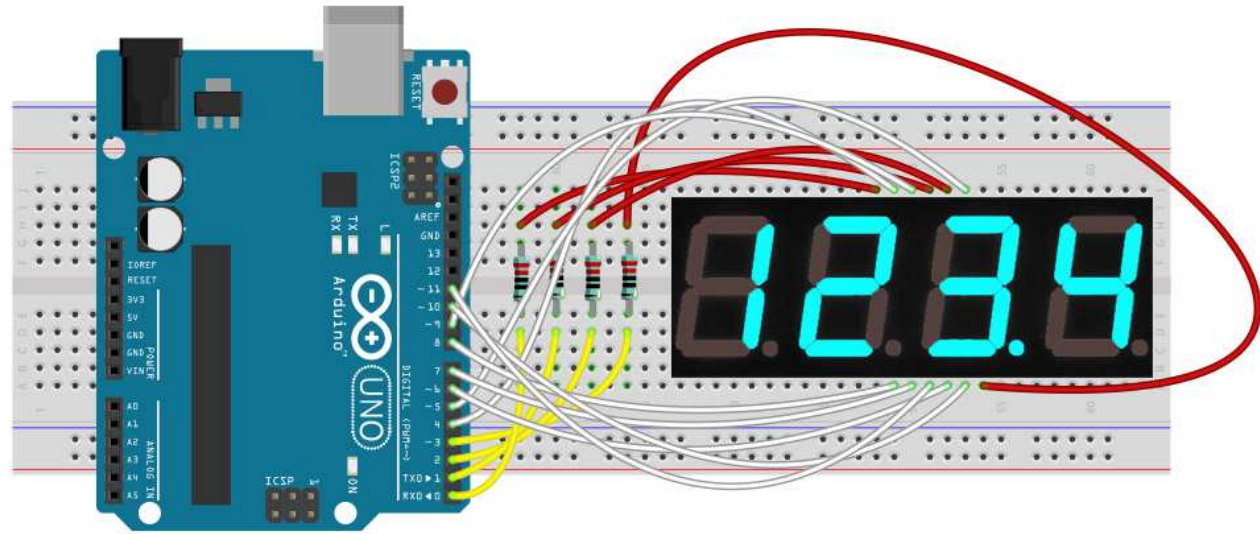
Digit	ABCDEFG	A	B	C	D	E	F	G
0	0x7E	on	on	on	on	on	on	off
1	0x30	off	on	on	off	off	off	off
2	0x6D	on	on	off	on	on	off	on
3	0x79	on	on	on	on	off	off	on
4	0x33	off	on	on	off	off	on	on
5	0x5B	on	off	on	on	off	on	on
6	0x5F	on	off	on	on	on	on	on
7	0x70	on	on	on	off	off	off	off
8	0x7F	on	on	on	on	on	on	on
9	0x7B	on	on	on	on	off	on	on



4. Napisati program koji na četvorocifarskom sedmosegmentnom LED displeju ispisiuje: '123.4'. (5 poena)

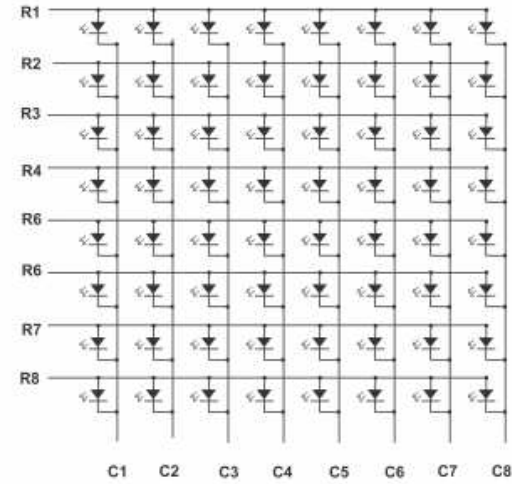
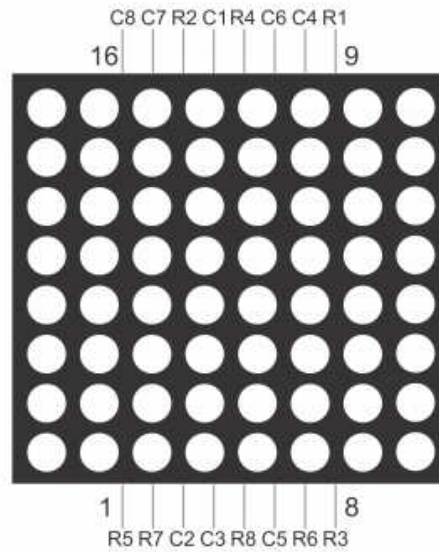


MCU与4连排8段数码管的连接





5. Napisati program koji na 8X8 matrix LED displeju ispisuje kvadrate koji se skupljaju i šire naizmjenično ili srce koje kuca (odnosno pojavljuje se i nestaje). (6 poena)



Kuraj