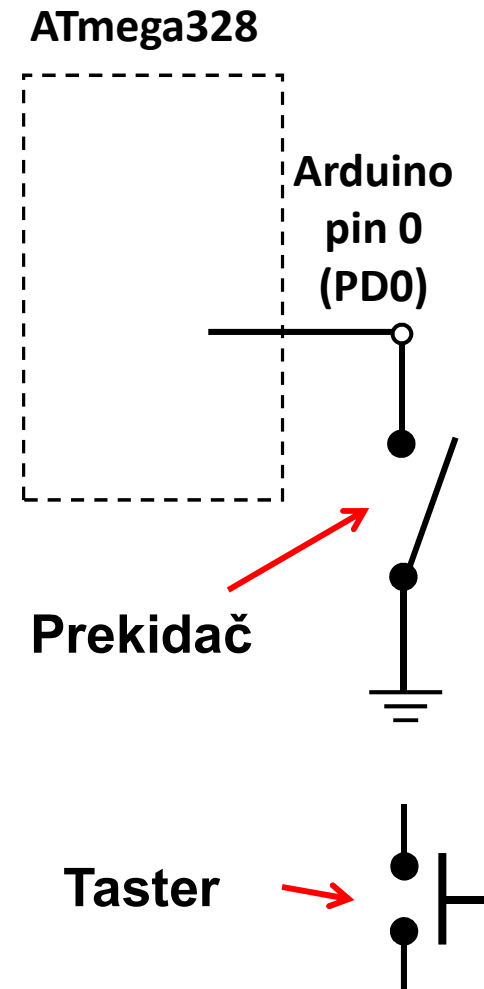


ULAZNI DIGITALNI PORTOVI



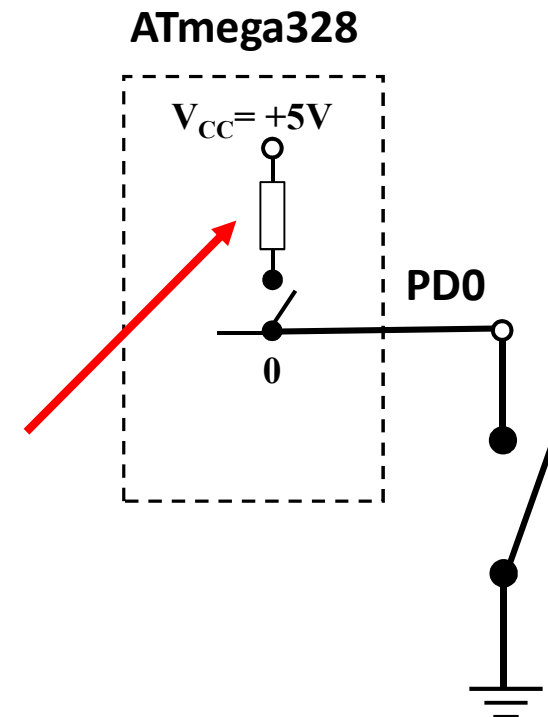
Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor
 - Pr. Senzor pojasa za sjedište u autu
 - Detekcija **stanja prekidača**
 - Koji tok podataka treba biti za Arduino pin 0 (PD0)?
 - `pinMode (__0__, __INPUT__)` ;
 - Koji će biti napon na PD0 kada je prekidač zatvoren?
 - Koji će biti napon na PD0 kada je prekidač otvoren?



Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor, nastavak.
 - Učinimo napon na pinu poznatim uključanjem pull-up otpornika za PD0
 - Neka je PD0 ulazni port:
 - `digitalWrite(0, HIGH);`
uključenje “pull-up” otpornika
 - `pinMode(0, INPUT_PULLUP);`
 - Koji će napon biti na PD0 kada je prekidač otvoren?
 - Koji će napon biti na PD0 kada je prekidač zatvoren?

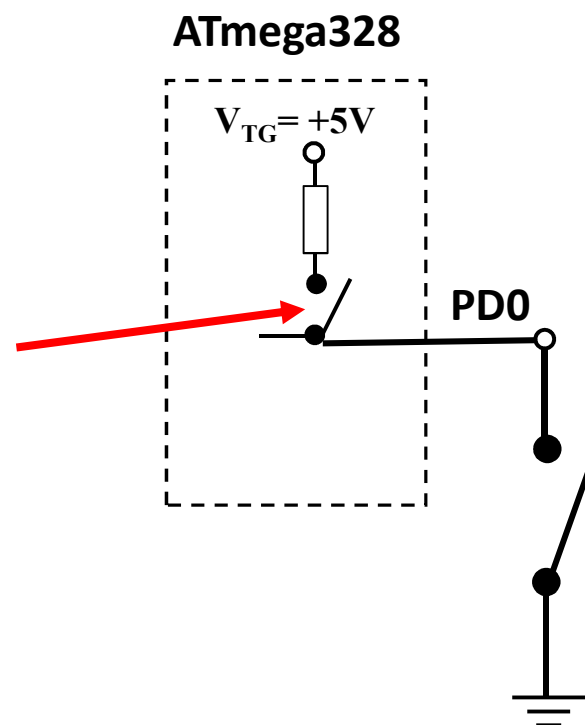


Pin kao ulazni + Pull-up otpornik

- Prekidač kao senzor, nastavak.
 - Za isključenje pull-up otpornika
 - Neka je PD0 ulazni port:

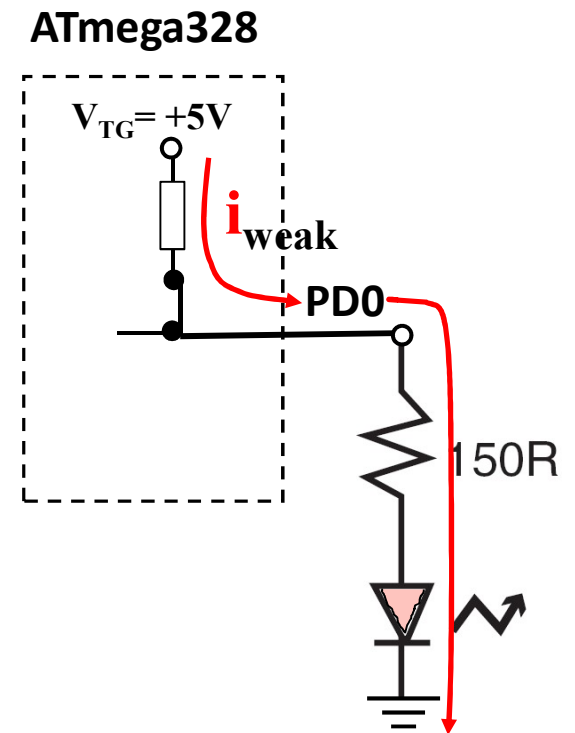
```
digitalWrite(0, LOW);
```

Isključuje "pull-up" otpornik



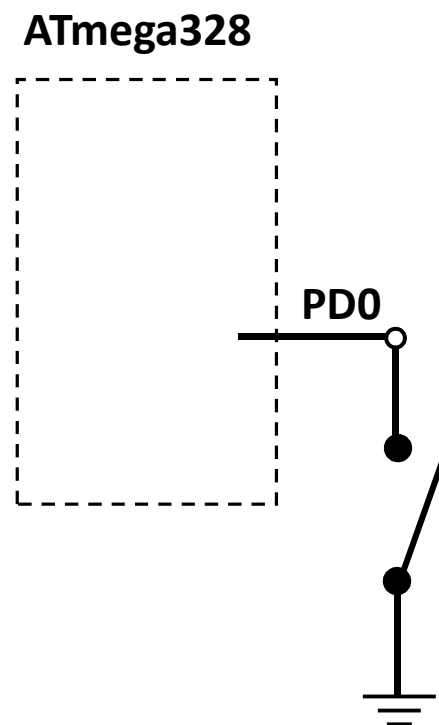
Pin kao ulazni + Pull-up otpornik

- 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!



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 PD0 kao ulazni

- Uključiti PD0 pull-up otpornik

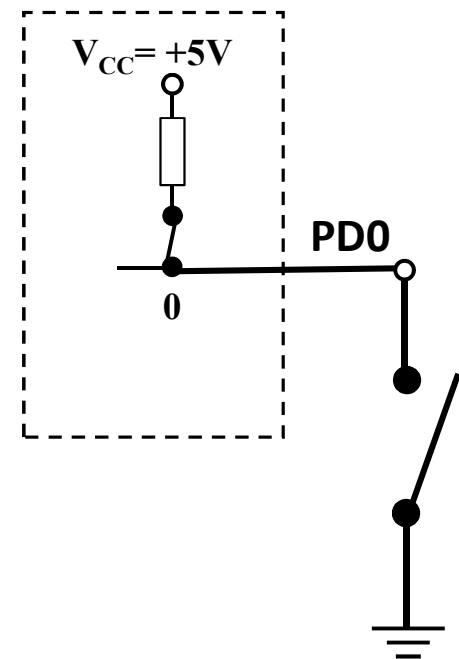
- Očitati napon sa Arduino pin 0 (PIN_D0)

- IF PIN_D0 napon je LOW (vezan), THEN
pozovi funkciju start_enable()

- ELSE

- pozovi start_disable()

ATmega328



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);
```


Ulazni digitalni pin – Primjer 1

- ‘Očitavanja pina’

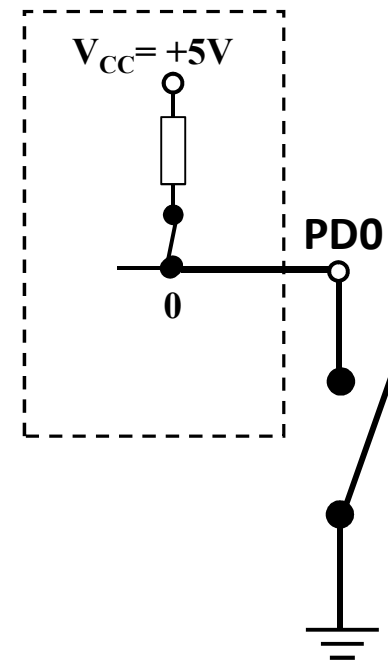
- Pseudokod:

- Postaviti PD0 kao ulazni
 - Uključiti PD0 pull-up otpornik
 - Očitati napon sa Arduino pin 3 (PIN_D0)
 - IF PIN_D0 napon je LOW (vezan), THEN
 - pozovi funkciju start_enable()
 - ELSE
 - pozovi start_disable()

Fragment. Nije cijeli skeč.

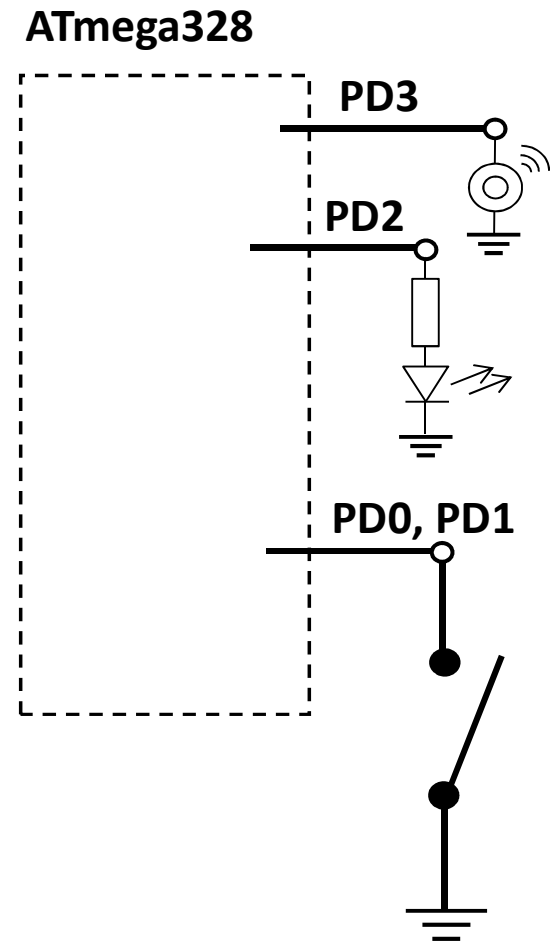
```
#define PIN_SWITCH 0
#define LATCHED LOW
pinMode(PIN_SWITCH, INPUT_PULLUP);
belt_state = digitalRead(PIN_SWITCH);
if (belt_state == LATCHED)
{ ig_enable(); }
else
{ ig_disabled(); }
```

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

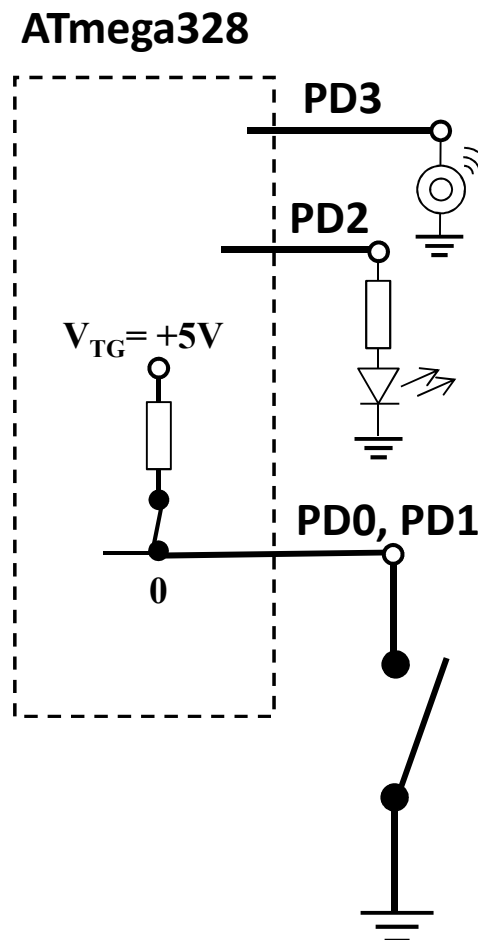
Uključi LED

Uključi zvučni signal

ELSE

Isključi zvučni signal

Isključi LED

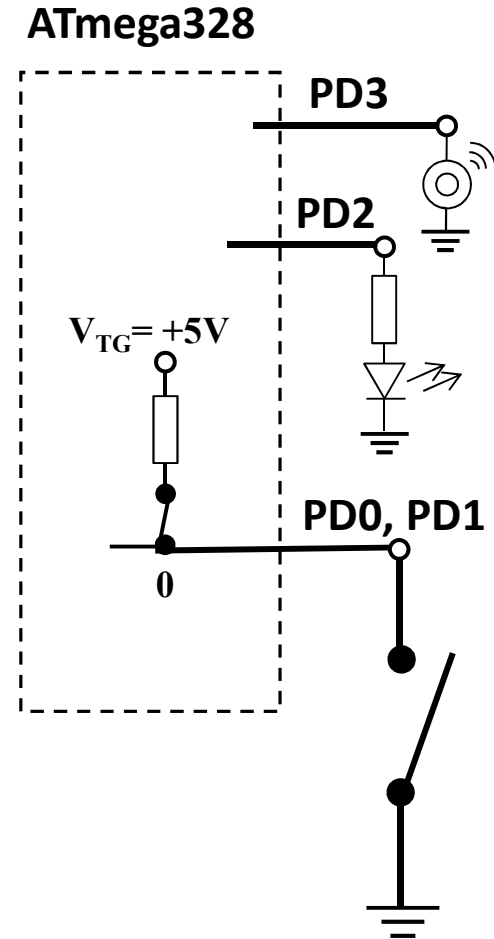


Ulazni digitalni pin – Primjer 2 (Arduino kod)

```
#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

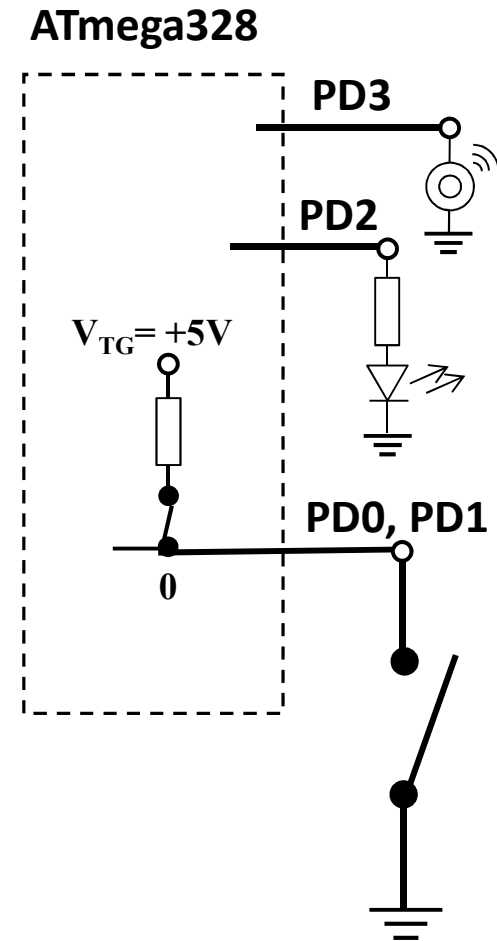
void setup()
{
  pinMode(PIN_IGNITION, INPUT_PULLUP); // key switch
  pinMode(PIN_SEATBELT, INPUT_PULLUP); // belt latch switch
  pinMode(PIN_LED, OUTPUT); // lamp
  pinMode(PIN_BUZZER, OUTPUT); // buzzer
}

/* 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);  
  }  
}
```

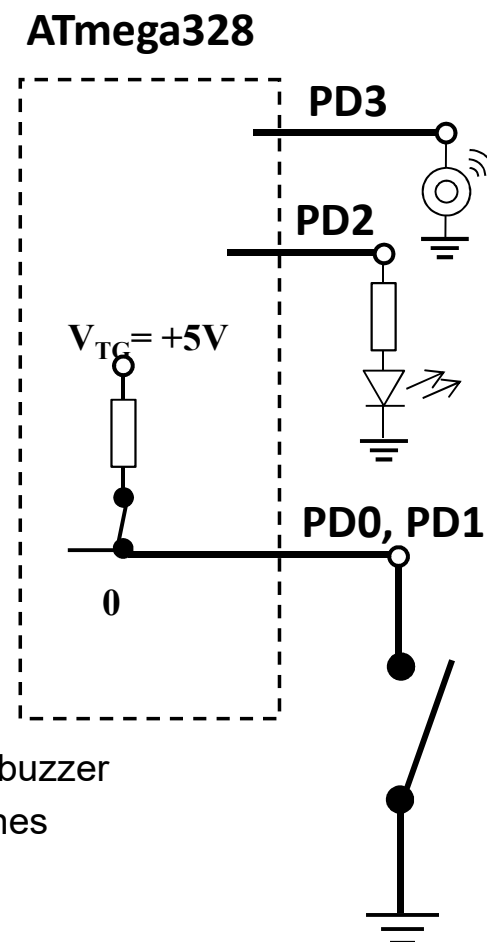
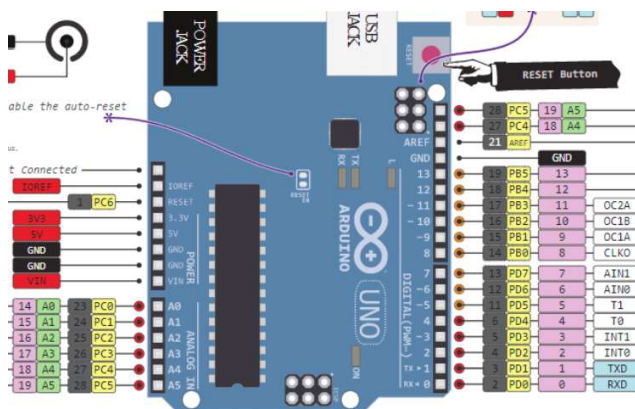


Ulazni digitalni pin – Primjer 2 (Alternativni kod)

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

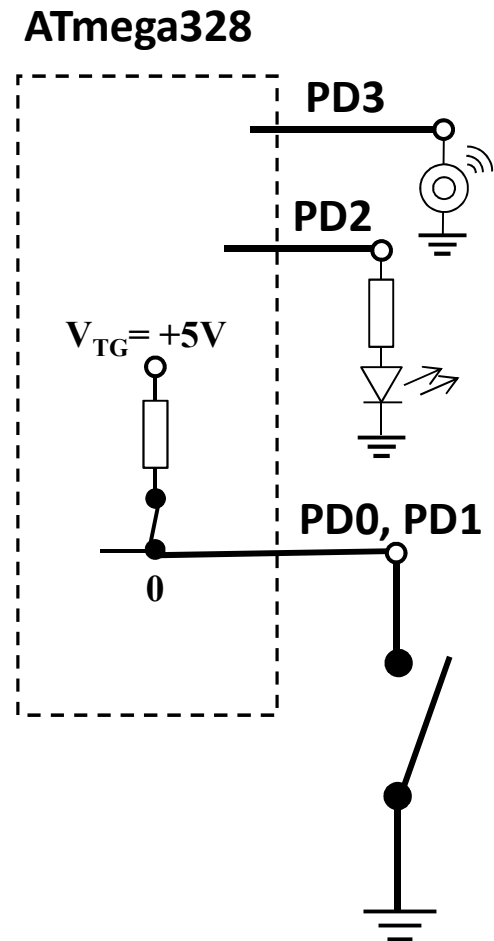
```
#define PIN_IGNITION PD0
#define PIN_SEATBELT PD1
#define PIN_LED PD2
#define PIN_BUZZER PD3
#define SEATBELT_LATCHED LOW
#define KEY_IN_IGNITION LOW
#define LED_ON HIGH
#define LED_OFF LOW
#define BUZZER_ON HIGH
#define BUZZER_OFF LOW
#define _BIT_MASK(bit) ( 1 << (bit) ) // same as _BV( bit)
void 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
}
```

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



Ulazni digitalni pin – Primjer 2 (Alternativni 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));  
  }  
}
```



Serijska komunikacija

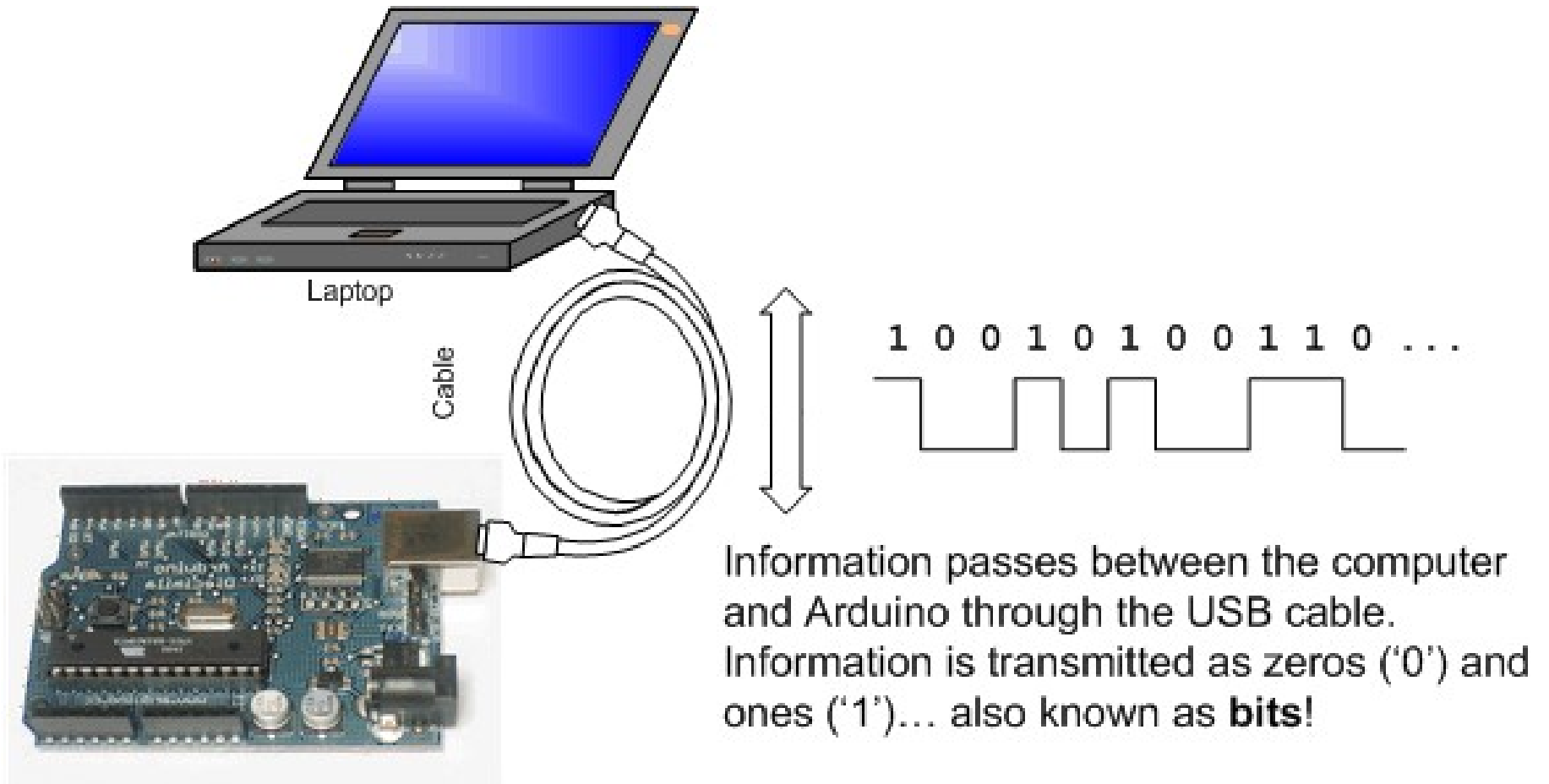
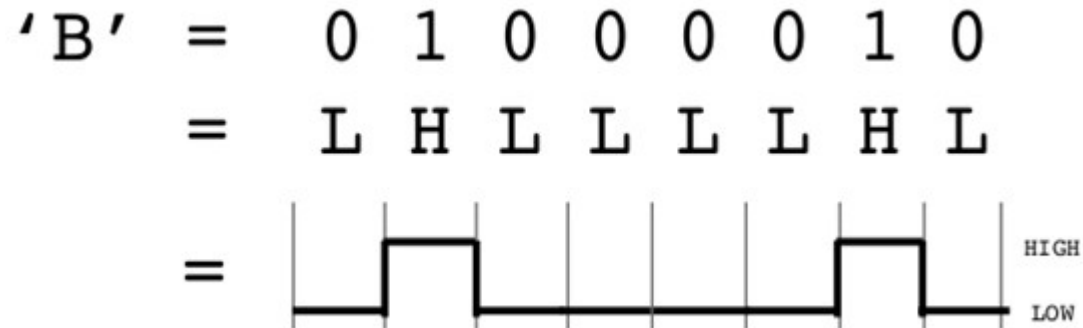


Image from <http://www.ladyada.net/learn/arduino/lesson4.html>

Serijska komunikacija

Serijska- jer su podaci razbijeni na bitove. Svaki bit se šalje jedan za drugim preko jedne žice

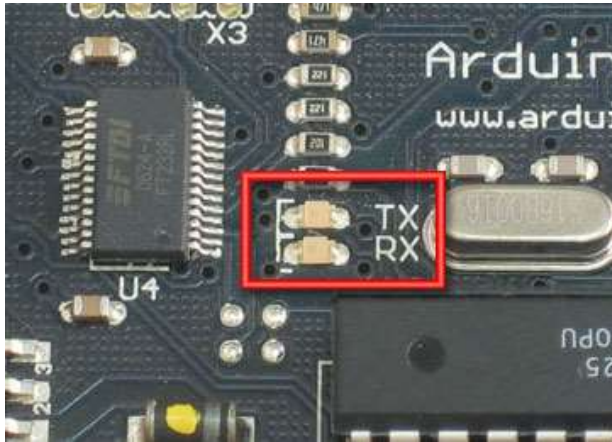
Primjer: ASCII karakter 'B' se šalje kao:



Mijenja se stanje na pinu baš kao kada se upravlja treperenjem LED.

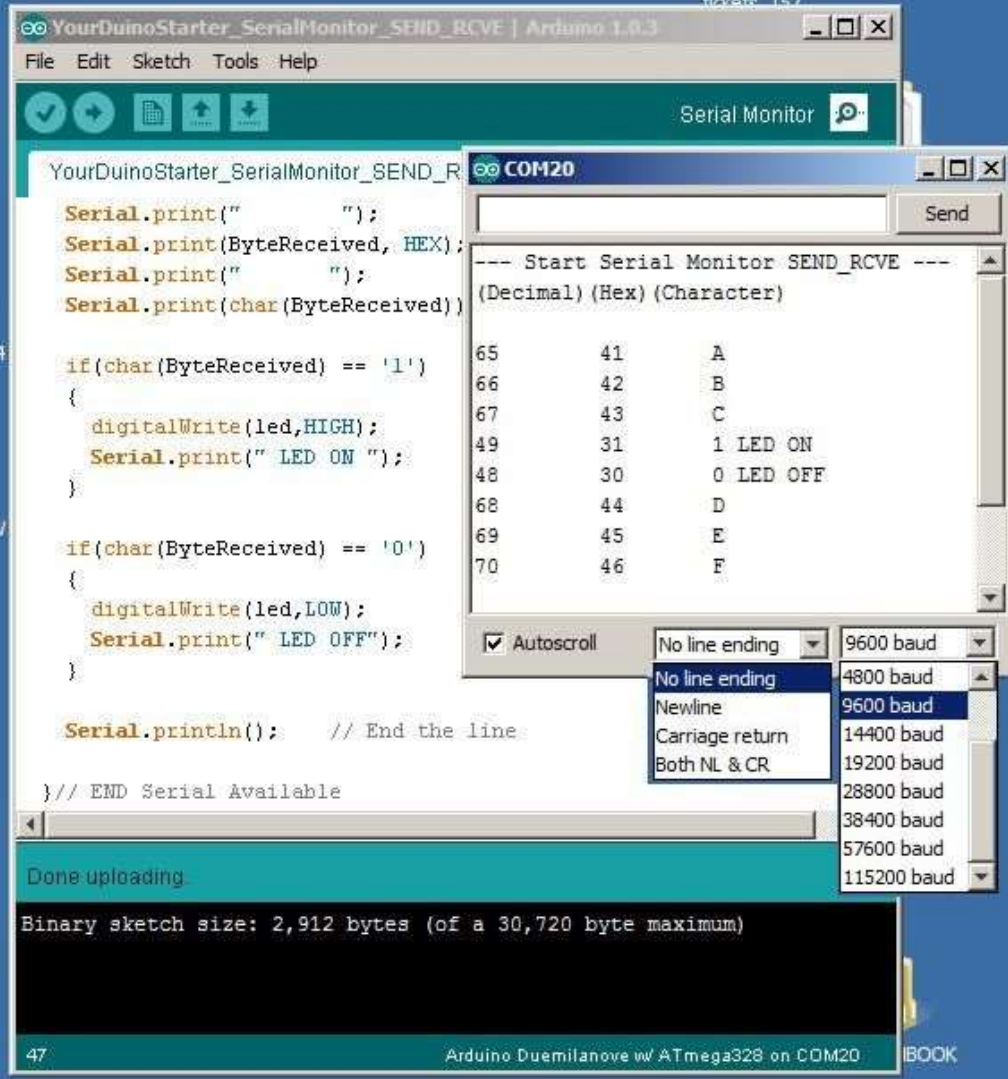
Jedna linija se koristi za slanje i jedna za prijem podataka.

Serijska komunikacija



- ***Kompajliranje*** prevodi program u binarne podatke (jedinice i nule)
- ***Uploading (upisivanje)*** šalje bitove kroz USB kabl do Arduina.
- Dvije LED diode blizu USB konektora trepere dok se podaci prenose
 - **RX** treperi kada Arduino prima podatke
 - **TX** treperi kada Arduino šalje

Serijski monitor



The screenshot displays the Arduino IDE interface. The main window shows a sketch titled "YourDuinoStarter_SerialMonitor_SEND_RCVE" with the following code:

```
Serial.print(" ");
Serial.print(ByteReceived, HEX);
Serial.print(" ");
Serial.print(char(ByteReceived));

if(char(ByteReceived) == '1')
{
  digitalWrite(led,HIGH);
  Serial.print(" LED ON ");
}

if(char(ByteReceived) == '0')
{
  digitalWrite(led,LOW);
  Serial.print(" LED OFF");
}

Serial.println(); // End the line

} // END Serial Available
```

The Serial Monitor window is open, showing the following output:

```
--- Start Serial Monitor SEND_RCVE ---
(Decimal) (Hex) (Character)
65      41      A
66      42      B
67      43      C
49      31      1 LED ON
48      30      0 LED OFF
68      44      D
69      45      E
70      46      F
```

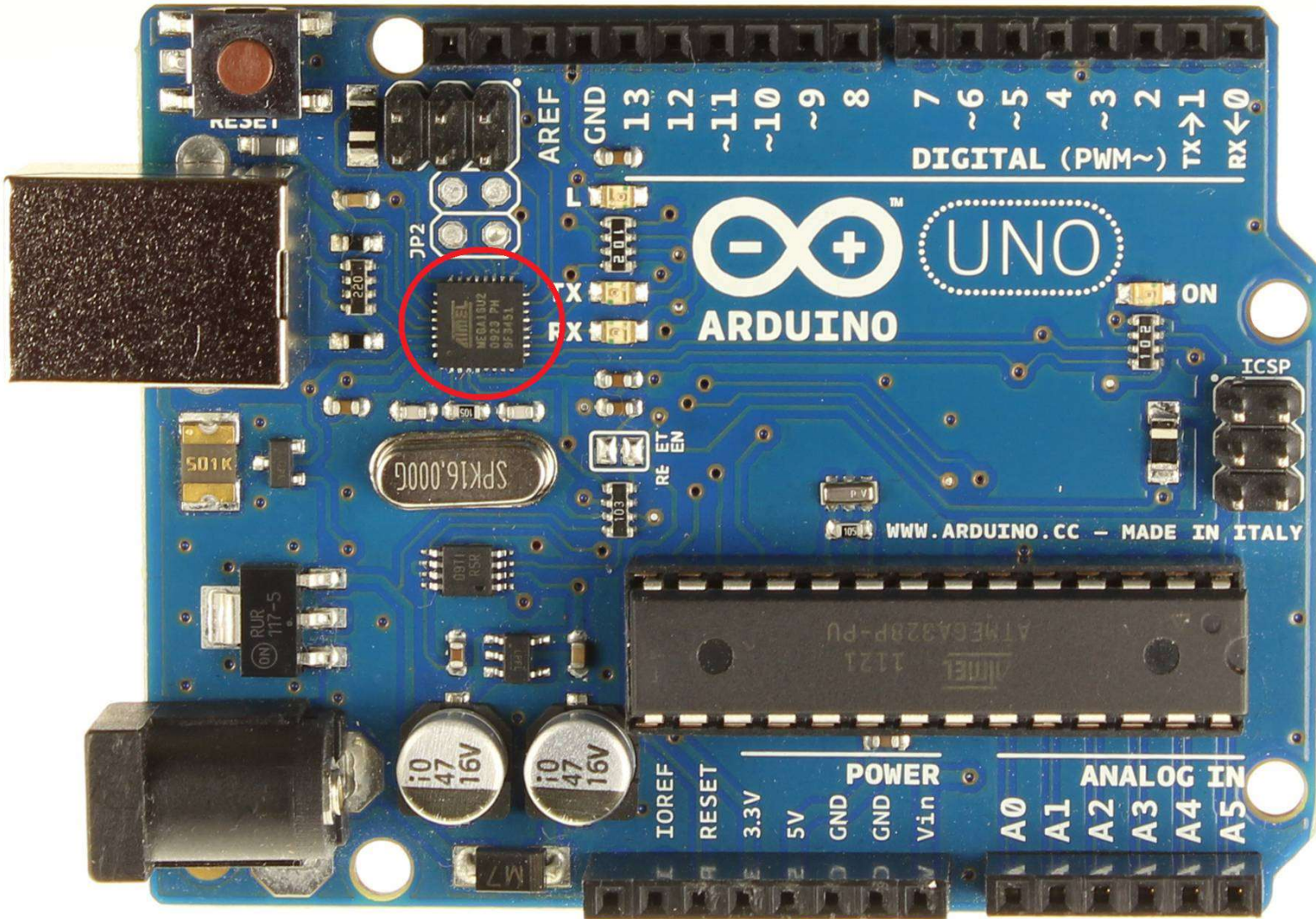
The Serial Monitor window also shows a dropdown menu for line endings and a baud rate dropdown menu. The line ending dropdown is currently set to "No line ending". The baud rate dropdown is currently set to "9600 baud".

At the bottom of the IDE, the status bar shows "47" and "Arduino Duemilanove w/ ATmega328 on COM20".

Osnovne komande

- `Serial.begin()`
 - pr., `Serial.begin(9600)`
- `Serial.print()` or `Serial.println()`,
`Serial.write()`

Serial-to-USB chip



Dva različita komunikaciona protokola

Serijski (TTL):

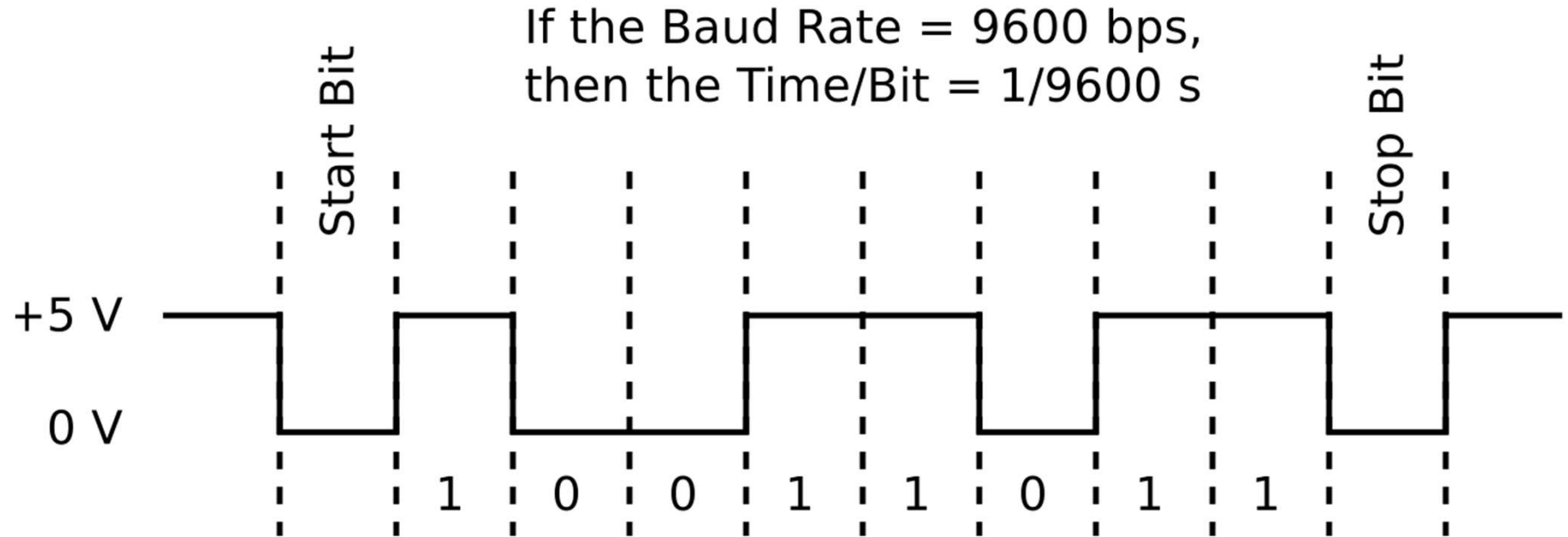


Image from <http://www.fiz-ix.com/2013/02/introduction-to-arduino-serial-communication/>

USB protokol

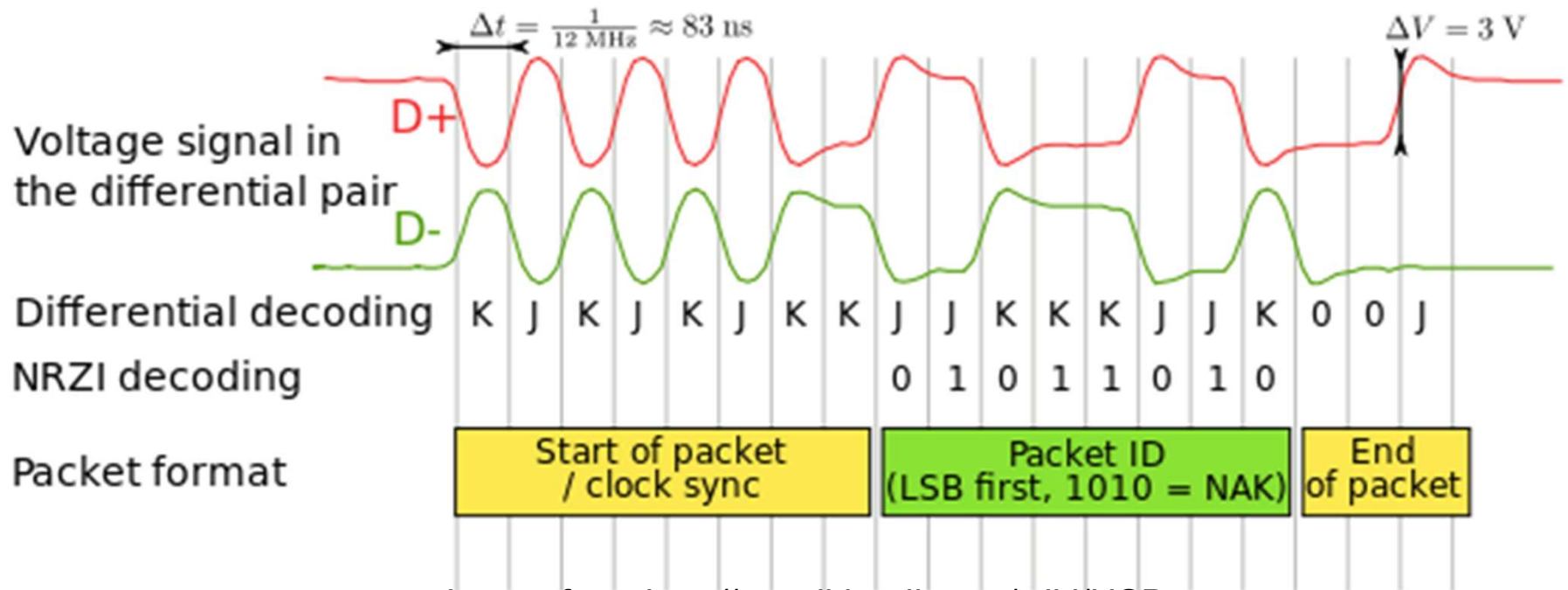


Image from <http://en.wikipedia.org/wiki/USB>

- Puno komplikovaniji

Brojanje koliko puta je pritisnut taster.



```
#define TASTER 3
#define LED 2
int brojPritisaka;
boolean Taster, pTaster;

void setup() {
  // put your setup code here, to run once:
  pinMode(TASTER, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
  Serial.begin(9600);

  //Inicijalizacija promjenljivih
  Taster=digitalRead(TASTER);
  pTaster=Taster;
  brojPritisaka=0;
}

void loop() {
  // put your main code here, to run repeatedly:
  Taster=digitalRead(TASTER);
  if(!Taster && (pTaster!=Taster)){
    brojPritisaka++;
    Serial.println(brojPritisaka);
  }
  if(Taster)digitalWrite(LED, LOW);
  else digitalWrite(LED, HIGH);
  pTaster=Taster;
  delay(2);
}
```


Izbjegavanje upotrebe dužeg čekanja u skeču – upotreba funkcije millis(), micros()

Primjer upotrebe funkcije millis()

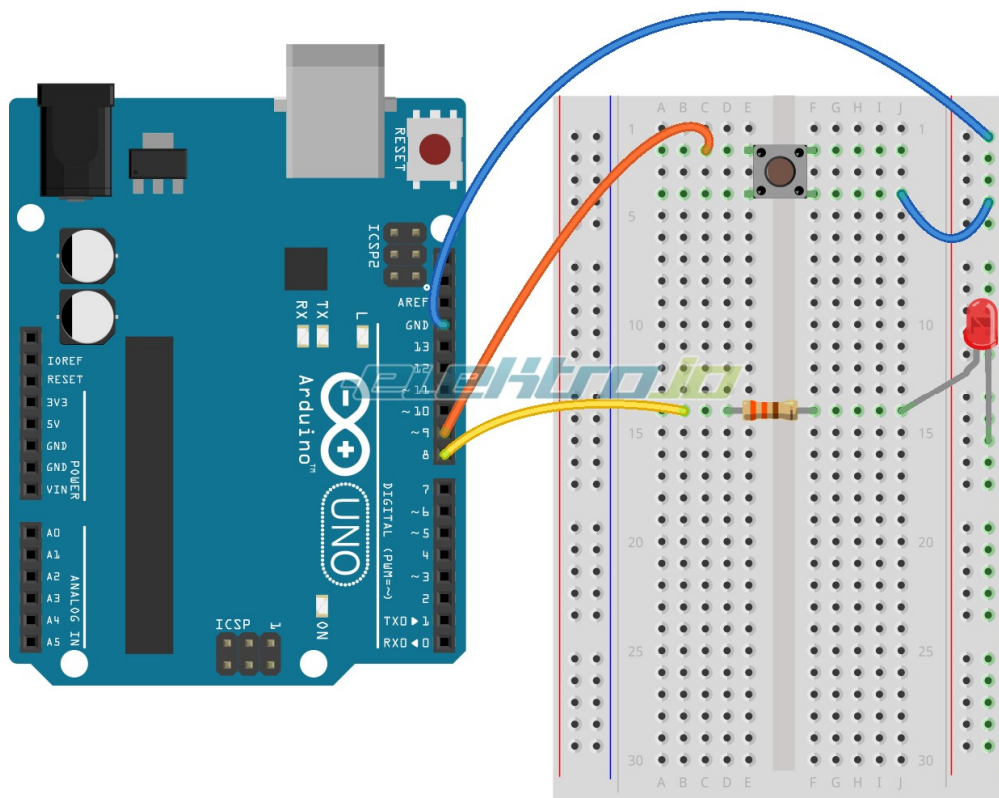
```
unsigned long startMillis; //globalne promjenljive
unsigned long currentMillis;
const unsigned long period = 1000; //vrijednost je u milisekundama
const byte ledPin = 13; //korištenje ugrađene diode LED

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

void loop()
{
  currentMillis = millis(); //dodjela trenutnog „vremena“ promjenljivoj (broj ms od startovanja programa)
  if (currentMillis - startMillis >= period) //ispitivanje da li je period istekao
  {
    digitalWrite(ledPin, !digitalRead(ledPin)); //ako jeste, promjena stanja LED. Korišćenje trika za promjenu stanja
    startMillis = currentMillis; //VAŽNO je ubilježiti startno vrijeme trenutnog stanja LED.
  }
}
```

Zadaci za vježbu

1. Upotrijebiti taster za uključenje i isključenje LED. Svaki pritisak tastera duži od tri sekunde uključuje LED. Pritisak tastera kraći od jedne sekunde isključuje LED. Pritisci tastera trajanja između sekunde i tri sekunde ne mijenjaju stanje LED. **(2-1 poen)**



Made with  Fritzing.org

2. Trčeće svjetlo sa 4 LED. Smjer se određuje prekidačem. Prekidač otvoren - jedan smjer, prekidač zatvoren - drugi smjer. Zaustavlja se pritiskom na taster. Otpuštanjem tastera nastavlja protrčavanje na isti način. Trčeće svjetlo treba reagovati neposredno po promjeni stanja tastera i/ili prekidača (ne treba otrčati do kraja niza, pa tek onda reagovati). **(3-2 poen)**



3. Nivo tečnosti u bazenu. Kao gornji i donji senzor nivoa upotrijebiti obične kratkospojnike, a kao bazen čašu i nešto vode u njoj. Informaciju o nivou tečnosti indicirati na jednocifarkom sedmo-segmentnom displeju, LED diodama i zvučno, na sljedeći način:

- OK nivo - slovo 'O' i uključena zelena LED,
- nizak nivo - slovo 'L' i uključena žuta LED,
- visok nivo - slovo 'H' i uključena crvena LED,
- neispravnost - slovo 'E', isključene sve LED i isprekidani zvučni signal.

(5-4-3 poena)

