LAB VEŽBA 3 – MIKROKONTROLERI

Tema: Cilj ove vežbe je da se studenti kreiranjem nekoliko mikrokontrolerskih programa upoznaju sa postupkom izrade mikrokontrolerskog programa i postupkom programiranja mikrokontrolera. Na raspolaganju je Arduino IDE razvojno okruženje, a dizajn se implementira na Arduino Nano razvojnom sistemu.

Komponente potrebne za vežbu:

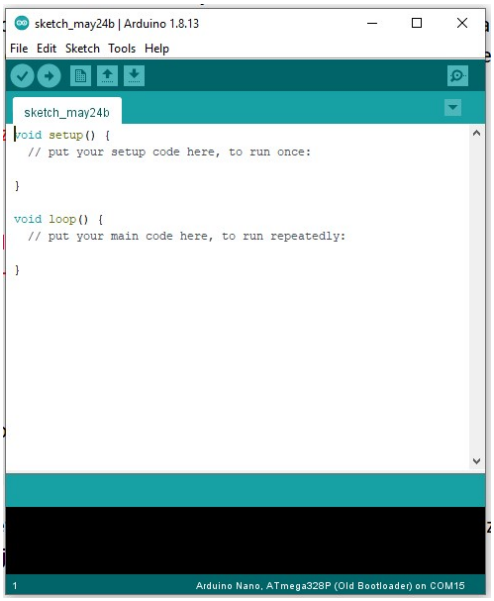
1. 7x otpornik 1kΩ
2. Arduino Nano razvojna ploča
3. 1x taster
4. 1x crvena LED dioda
5. 1x LED displej
6. 1x potenciometar 1kΩ

ZADATAK 1

Arduino IDE je potrebno instalirati sa linka

https://downloads.arduino.cc/arduino-1.8.13-windows.exe

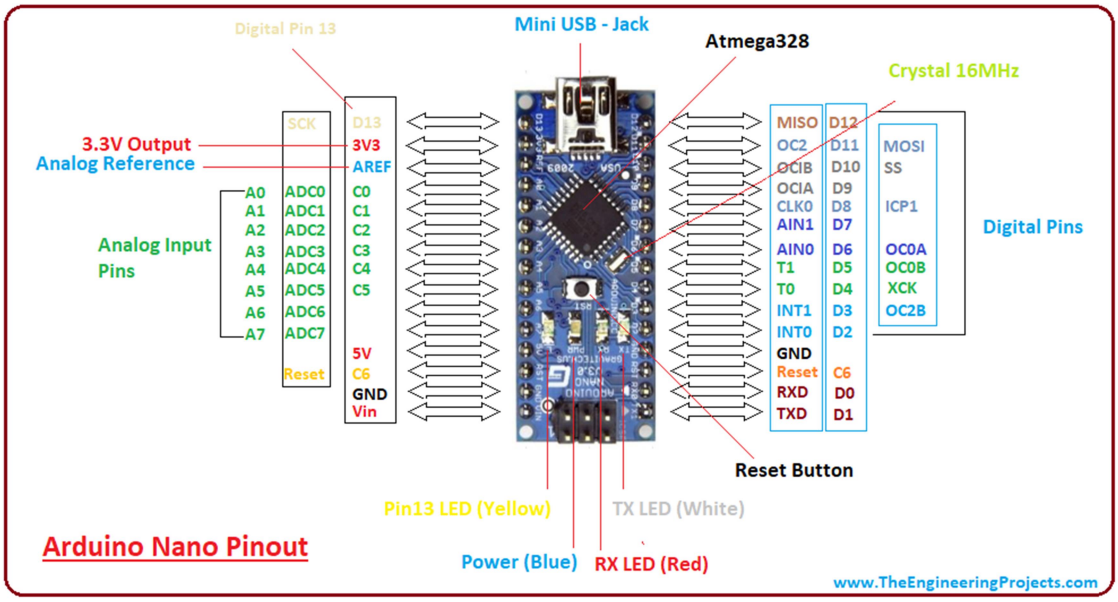
Nakon instalacije i pokretanja Arduino IDE aplikacije otvara se prozor sa praznom novom aplikacijom, koja se u ovom okruženju naziva *sketch* (Slika 1) i čuva se u datoteci sa ekstenzijom *.ino*. Program se može pisati u *C* i *C++* programskom jeziku.



Slika 1. Početni prozor Arduino aplikacije

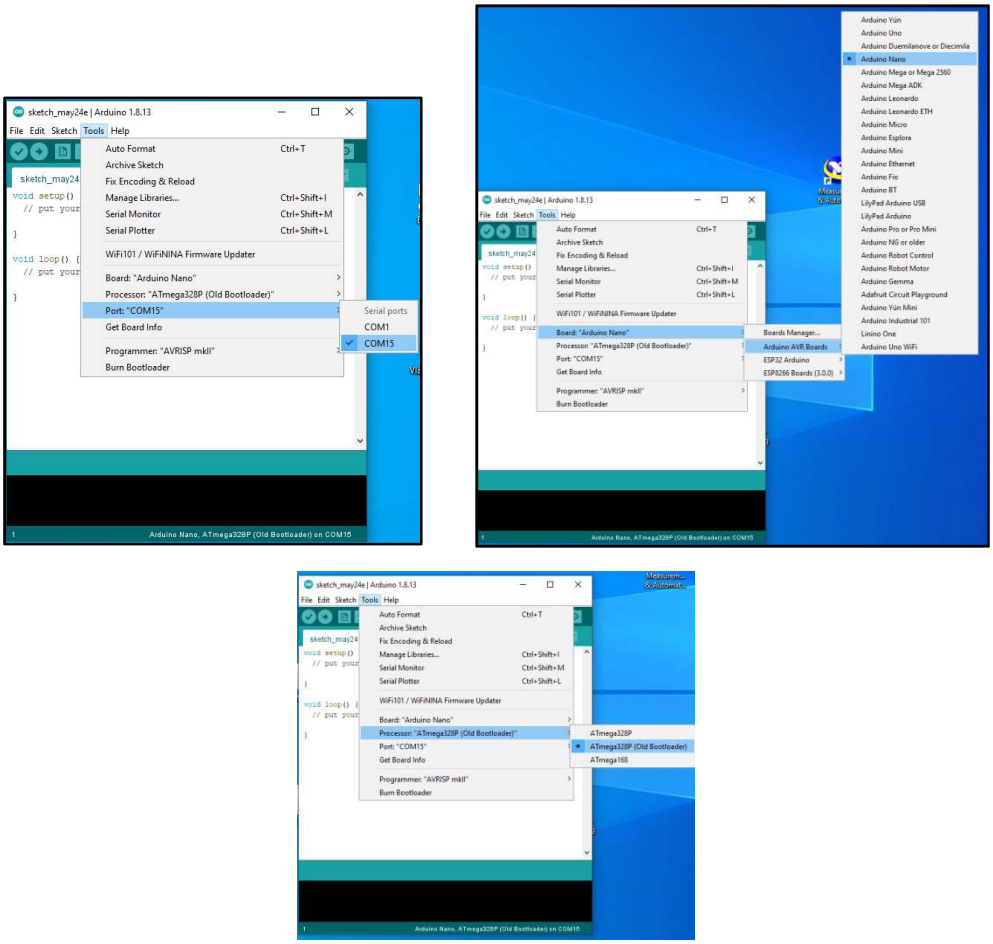
Kako je već rečeno na predavanjima, program je podeljen u dva dela (funkcije). Funkcija *setup* se izvršava na početku programa, samo jednom, dok se funkcija *loop* izvršava neprestano – kad se jednom dođe do kraja funkcije, njeno izvršavanje počinje ispočetka. U funkciju *setup* se smeštaju neophodne naredbe za inicijalizaciju hardvera koja se vrši samo jednom, dok se prava funkcionalnost programa opisuje naredbama u okviru funkcije *loop*.

Funkcionalnost pinova razvojne ploče Arduino Nano prikazana je na slici 2. Funkcionalnost pojedinih pinova biće objašnjena kako se budu koristili za izradu ove vežbe.



Slika 2. Priključci razvojne ploče Arduino Nano

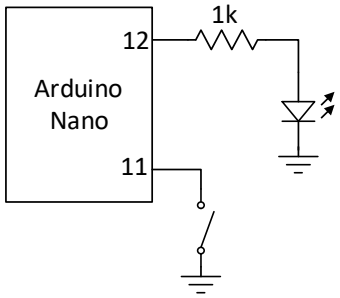
Da bi se program koji se piše pravilno kompajlirao i da bi mogao da se prebaci u programsku memoriju mikrokontrolera, potrebno je podesiti port na koji je povezana razvojna ploča, izabrati odgovarajuću razvojnu ploču, i procesor na razvojnoj ploči.



Slika 3. Podešavanje Arduino IDE

U **prvom delu** zadatka je potrebno napisati program kojim se obezbeduje da je LED dioda uključena za vreme dok je taster pritisnut. Pri pritisku tastera je potrebno uključiti diodu. Ukoliko taster nije pritisnut diodu je potrebno isključiti.

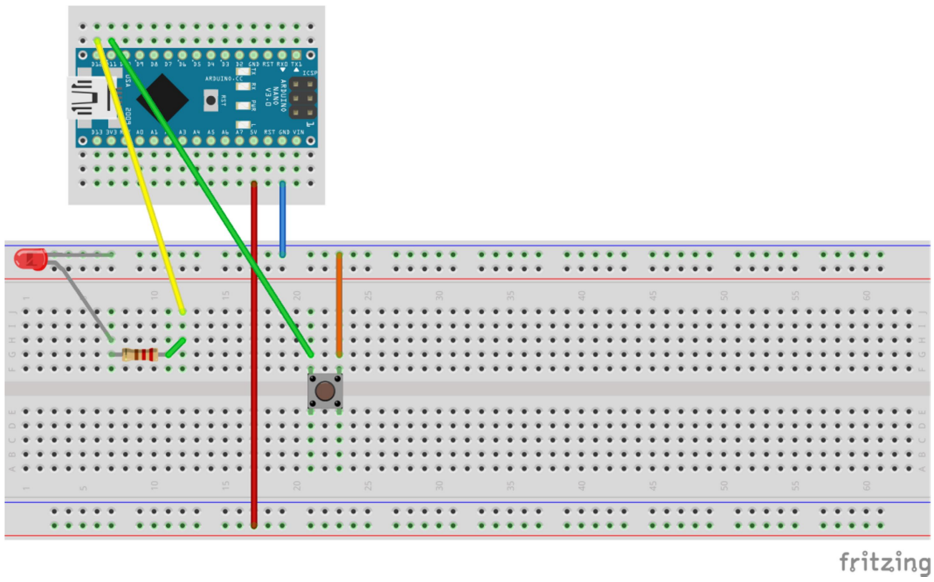
Na slici 4 je prikazana električna šema koju je potrebno povezati. Na jedan od digitalnih ulaza/izlaza Arduino Nano (na primer 12) povezan je taster kao ulaz, dok je na neki drugi digitalni ulaz/izlaz (na primer 11) preko otpornika od 1 kΩ povezana crvena svetleća dioda (LED).



Slika 4. Električna šema uz prvi korak zadatka 1

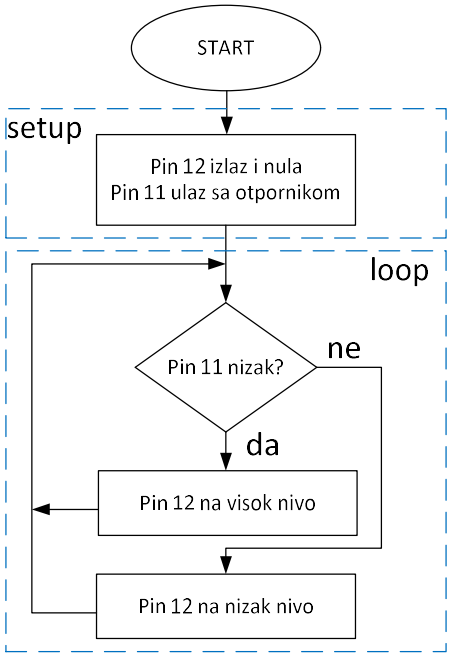
Kada je na digitalnom priključku 12 visok logički nivo kroz diodu će teći struja i ona će svetleti. Kada je na tom priključku nizak logički nivo kroz diodu neće teći struja i ona neće svetleti. Kada je taster pritisnut, na priključku 11 će biti prisutan nizak logički nivo. Kada taster nije pritisnut priključak 11 će biti na visokom logičkom nivou ako je na priključku 11 uključen unutrašnji otpornik povezan na napon napajnja, o čemu je reči bilo na predavanjima.

Povezivanje šeme sa slike 4. prikazano je na slici 5.



Slika 5. Povezivanje šeme uz prvi korak zadatka 1

Dijagram toka programa koji zadovoljava zadatak prikazan je na slici 6.



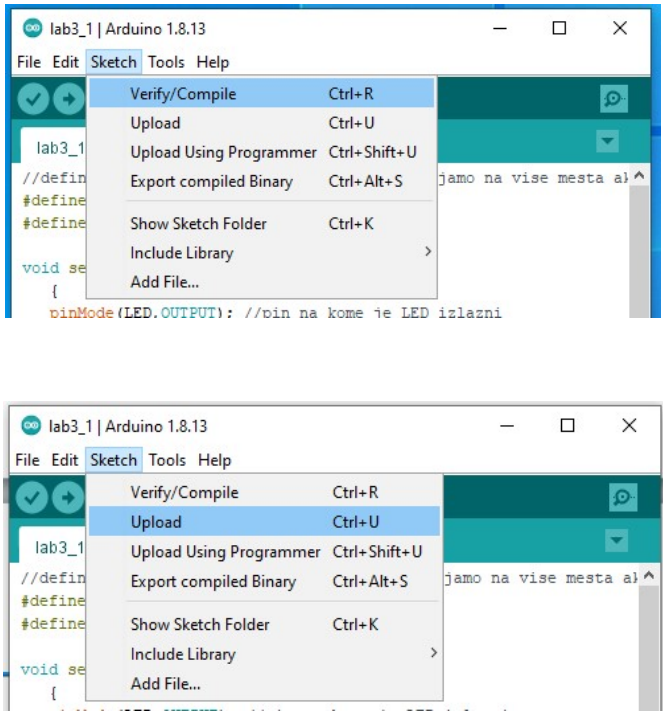
Slika 6. Dijagram toka programa uz prvi korak zadatka 1

U *setup* delu programa se inicijalizuju priključci, 12 kao izlazni i sa početnom vrednošću 0 kako dioda ne bi svetlela, i 11 kao ulazni sa uključenim otpornikom.

U delu koji se ponavlja (*loop*) se proverava da li je taster pritisnut: ako jeste na pin 12 se izbacuje visok nivo kako bi dioda svetlela; ako nije na pin 12 se izbacuje nizak nivo kako dioda ne bi svetlela.

Programski kod koji odgovara dijagramu toka sa slike 6 prikazan je u nastavku. Prekopirajte ga u Arduino IDE, izvršite kompajliranje i prebacivanje na razvojnu ploču kako je pokazano na slici 7 (objašnjeno je i na predavanjima) i verifikujte da program radi ispravno.

```
//definicija prikljucaka, da ne moramo da menjamo na vise mesta ako ih promenimo
#define LED 12
#define BTN 11
void setup()
  {
  pinMode(LED,OUTPUT); //pin na kome je LED izlazni
  digitalWrite(LED,LOW); //pin na kome je LED nula, da ne svetli na pocetku
  pinMode(BTN,INPUT_PULLUP); // pin na kom je taster da bude ulazni sa otpornikom
  }

void loop()
  {
  // put your main code here, to run repeatedly:
  if (digitalRead(BTN)==LOW)//citam prikljucak tastera i proveravam da li je nizak
    digitalWrite(LED,HIGH); //ako jeste
  else
    digitalWrite(LED,LOW); //
  }
```



Slika 7. Kompajliranje i prebacivanje programa na mikrokontroler

U **drugom delu** zadatka je potrebno napisati program kojim se obezbeduje da se svakim pritiskom na taster LED diodi menja stanje. Električna i šema povezivanja su iste kao u prethodnom delu zadatka, prikazane na slikama 4 i 5 respektivno.

Da bi menjali stanje diode na svaki pritisak tastera, potrebno je pamtiti to stanje. Takođe je potrebno pamtiti prethodno stanje tastera, da se ne bi dioda neprestano palila i gasila dok je taster pritisnut. Algoritam izvršavanja programa bi mogao da se opiše na sledeći način: ako je taster trenutno pritisnut, a prethodno nije bio pritistnut, menja se željeno stanje diode. Ako je stanje uključena na priključak 12 se izbacuje visok logički nivo, u suprotnom se izbacuje nizak logički nivo.

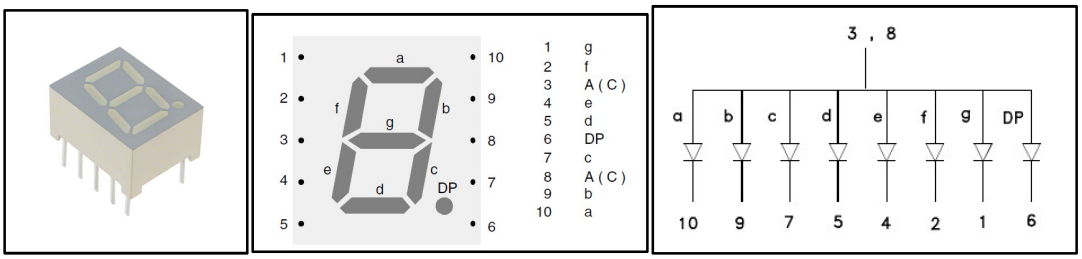
Kako bi se izbegao efekat odskakanja tastera, na kraju svakog prolaska kroz glavni program je pogodno sačekati neko vreme, reda 10-100 ms.

Arduino IDE ima ugrađenu funkciju *delay(x)*, koja uzrokuje čekanje u programu u trajanju od *x* ms. Funkcija koristi 8-bitni Timer/Counter 0 koji iz tog razloga nije dostupan korisniku za korišćenje u druge svrhe.

Programski kod koji odgovara tekstu zadatka prikazan je u nastavku. Prekopirajte ga u Arduino IDE, izvršite kompajliranje i prebacivanje na razvojnu ploču, i verifikujte da program radi ispravno.

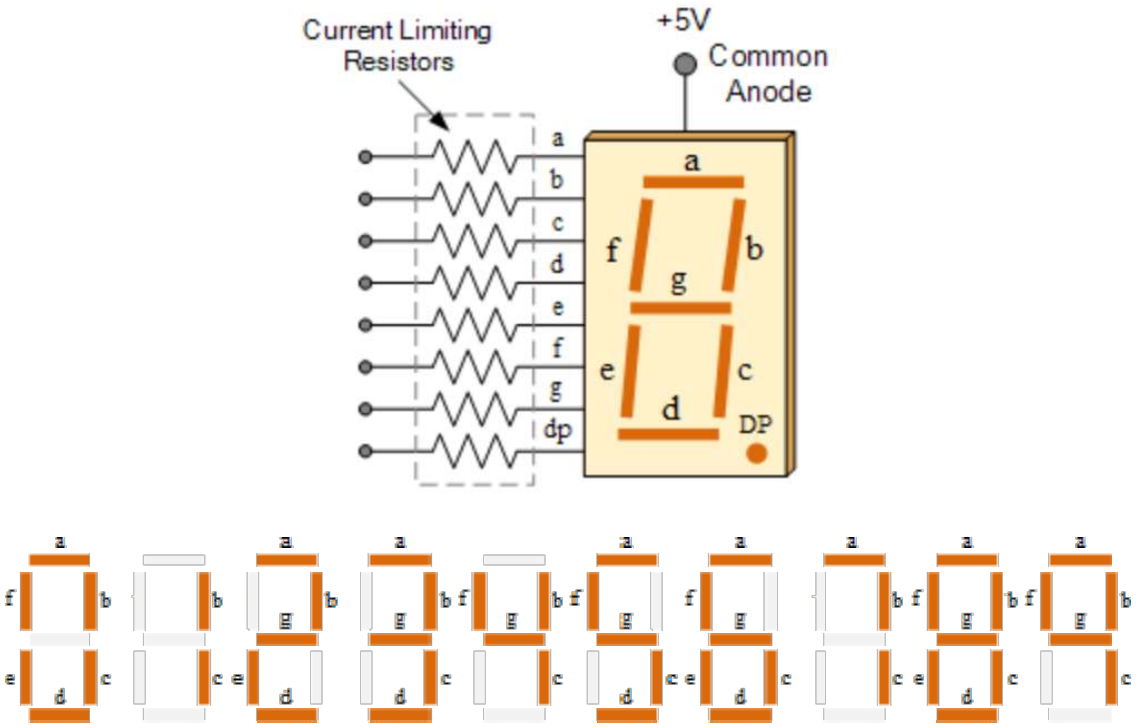
```
#define LED 12
#define BTN 11
char state = 0;//promenljiva koja cuva stanje LED diode, inicijalno iskljucena
char btn_old = HIGH; //promenljiva koja cuva staro stanje tastera, inicijalno nije pritisnut
char btn; // promenljiva u koju se smesta trenutno stanje tastera

void setup()
  {
  pinMode(LED,OUTPUT);
  digitalWrite(LED,LOW);
  pinMode(BTN,INPUT_PULLUP);
  }

void loop()
  {
  btn = digitalRead(BTN); //citanje stanja tastera
  if ((btn_old == HIGH) && (btn==LOW) ) //ako pre nije, a sada jeste pritisnut
    state = 1-state;//promena stanje diode, ako je bilo 0 postaje 1, i obratno
  if (state == 1)//provera zeljenog stanja diode
    digitalWrite(LED,HIGH); //ako je 1 izbaci visok nivo na pin LED
  else digitalWrite(LED,LOW); // ako nije izbaci nizak nivo na pin LED
  btn_old = btn; //zapamti trenutno stanje tastera za sledecu iteraciju
  delay(100);//sacekaj 100ms da bi izbegli ocitavanje odskakanja tastera
  }
```

U **trećem delu** zadatka je potrebno napisati program kojim se obezbeduje da se sa računara kontroliše prikaz na segmentnom LED displeju. Segmentni LED displej je elektronska komponenta u kojoj je integrisano više (u ovom slučaju osam) svetlećih dioda koje imaju zajedničku anodu ili katodu (u ovom slučaju anodu), kako je pokazano na slici 8.



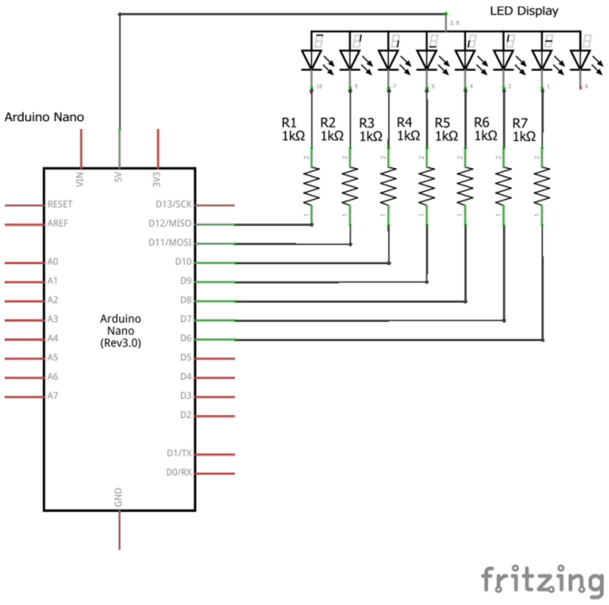
Slika 8. Segmentni LED displej sa zajedničkom anodom

Svaki od segmenata u okviru displeja sa zajedničkom anodom se uključuje/isključuje tako što se na njegovu katodu preko otpornika dovodi visok/nizak naponski nivo, kako je prikazano na slici 9. Kada je prisutan visok logički napon kroz diodu nema struje i ona je isključena, i obratno. Od vrednosti upotrebljenih otpornika zavisi intenzitet struje kroz diodu i svetlosti. Odgovarajućom kombinacijom naponskih nivoa na priključcima na displeju je moguće prikazati sve decimalne cifre, što je takođe prikazano na slici 9. Na primer, ako svetle segmenti a, b i c prikazuje se cifra 1.

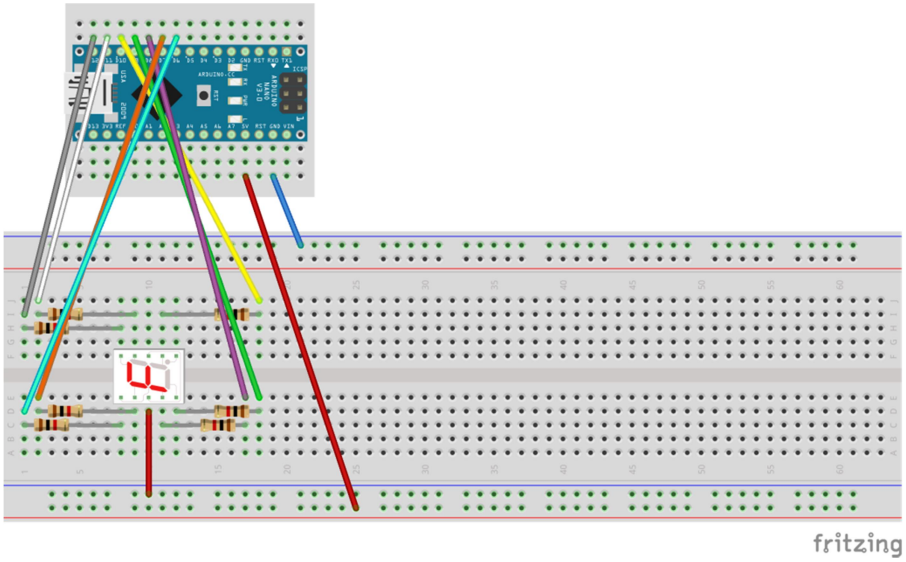


Slika 9. Povezivanje LED displeja sa zajedničkom anodom.

Na slici 10 je prikazana električna šema koju je potrebno povezati. Priključci LED displeja a (10), b (9), c (7), d (5), e (4), f (2) i g (1) povezani su redom na priključke 12, 11, 10, 9, 8, 7 i 6 Arduino Nano, preko otpornika od 1 kΩ. Potrebno je i jedan od priključaka LED displeja 3 i 8 povezati na 5 V.



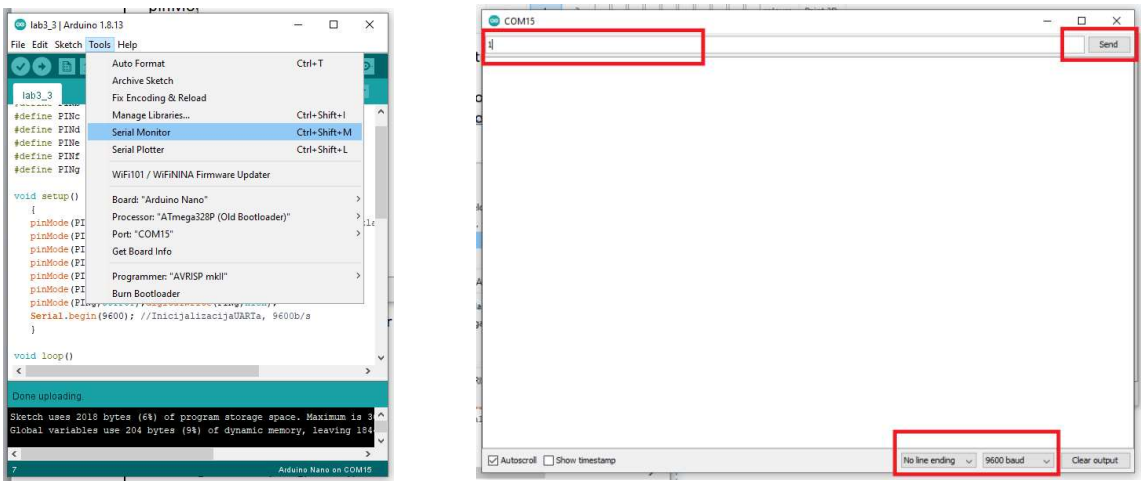
Slika 10. Električna šema uz treći korak zadatka 1



Slika 11. Povezivanje šeme uz treći korak zadatka 1

U okviru Arduino IDE postoje ugrađene funkcije za komunikaciju preko UART periferije. UART periferija je preko USB/UART konvertora povezana na USB priključak ploče, tako da za UART komunikaciju sa računarom nije poterbno nikakvo dodatno povezivanje. Da bi se koristile ugrađene funkcije, poterbno je UART periferiju inicijalizovati na željenu brzinu prenosa, naredbom Serial.begin(*brzina*). Provera iz programa da li postoje primljeni podaci vrši se pozivanjem funkcije Serial.available(), dok se čitanje jednog primljenog podatka vrši funkcijom Serial.read(). Slanje teksta na UART vrši se naredbama Serial.print(podatak) i Serial.println(podatak) – razlika je što druga funkcija šalje i specijalni karakter za prelazak u novi red.

Programski kod u nastavku radi na sledeći način: ako se sa računara pošalje cifra 1, ta se cifra prikazuje na displeju. Ukoliko se pošalje bio koji drugi znak, displej se gasi I računaru se šalje poruka “znak nije definisan”. Prekopirajte ga u Arduino IDE, izvršite kompajliranje i prebacivanje na razvojnu ploču. Nakon toga uključite Serial Monitor u Arduino IDE i preko nejga pošaljite prvo cifru 1, a potom neki drugi znak (Slika 12).



Sika 12. Serial Monitor

```
#define PINa  12
#define PINb  11
#define PINc  10
#define PINd  9
#define PINe  8
#define PINf  7
#define PINg  6

void setup()
  {
  pinMode(PINa,OUTPUT);digitalWrite(PINa,HIGH);//podesavanja izlaza, inicijalno visoki
  pinMode(PINb,OUTPUT);digitalWrite(PINb,HIGH);
  pinMode(PINc,OUTPUT);digitalWrite(PINc,HIGH);
  pinMode(PINd,OUTPUT);digitalWrite(PINd,HIGH);
  pinMode(PINe,OUTPUT);digitalWrite(PINe,HIGH);
  pinMode(PINf,OUTPUT);digitalWrite(PINf,HIGH);
  pinMode(PINg,OUTPUT);digitalWrite(PINg,HIGH);
  Serial.begin(9600); //InicijalizacijaUARTa, 9600b/s
  }

void loop()
  {
  if (Serial.available())//ima li podataka na UARTu?
    {
    switch (Serial.read())//ako ima procitaj jedan bajt i postupi u zavisnosti sta je stiglo
      {
      case '1'://ako je stigao karakter 1 (zato apostrofi), treba da svetle segmenti b i c
        digitalWrite(PINa,HIGH);
        digitalWrite(PINb,LOW);
        digitalWrite(PINc,LOW);
        digitalWrite(PINd,HIGH);
        digitalWrite(PINe,HIGH);
        digitalWrite(PINf,HIGH);
        digitalWrite(PINg,HIGH);
        break;
      default://ako je stiglo nesto sto nije definisano nista ne svetli
        digitalWrite(PINa,HIGH);
        digitalWrite(PINb,HIGH);
        digitalWrite(PINc,HIGH);
        digitalWrite(PINd,HIGH);
        digitalWrite(PINe,HIGH);
        digitalWrite(PINf,HIGH);
        digitalWrite(PINg,HIGH);
        Serial.println("znak ne postoji!");//poruka racunaru
        break;
      }
    }
  }
```

ZADATAK – 5 poena: Modifikovati prethodni program tako da pored cifre 1 prikazuje još dve cifre, x i x+5 , gde je x = **(BrInd % 5)**, **BrInd** je broj indeksa, a **%** predstavlja moduo operaciju, odnosno ostatak pri deljenju. **Ukoliko se kao rezultat dobije x=1, potrebno je prikazati cifre 6 i 0.**

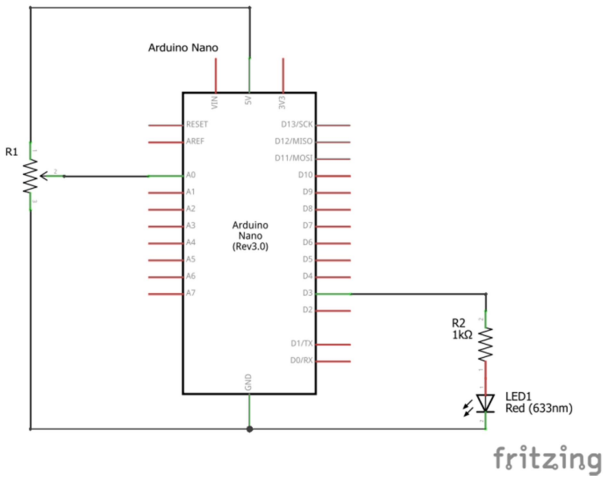
ZADATAK 2

U **prvom delu** zadatka je potrebno napisati program kojim se obezbeduje kontrola osvetljenosti svetleće diode korišćenjem potenciometra, uz istovremeno slanje trenutne vrednosti osvetljaja na računar korišćenjem UART komunikacije.

Električni simbol, izgled i međusobni raspored priključaka korišćenog potenciometra prikazani su na slici 13. Odnos otpornosti između priključaka 1 i 2 i priključaka 2 i 3 se menja obrtanjem osovine potenciometra, ali se ukupna otpornost između priključaka 1 i 3 ne menja. Na ovaj način se, ako se između krajnjih priključaka dovede neka razlika potencijala, može menjati razlika potencijala priključka 2 u odnosu na neki krajnji priključak potenciometra. Na primer, ako je priključak 1 na potencijalu nule a priključak 3 na potencijalu 5 V, potencijal priključka 2 se može menjati u opsegu 0-5 V.

Slika 13.

Na slici 14 je prikazana električna šema koju je potrebno povezati. Na jedan od analognih ulaza Arduino Nano (na primer A0) povezan je potenciometar, dok je na jedan od digitalnih ulaza/izlaza koji ima PWM funkcionalnost (nemaju svi, na primer 3 ima) preko otpornika od 1 kΩ povezana crvena svetleća dioda (LED).

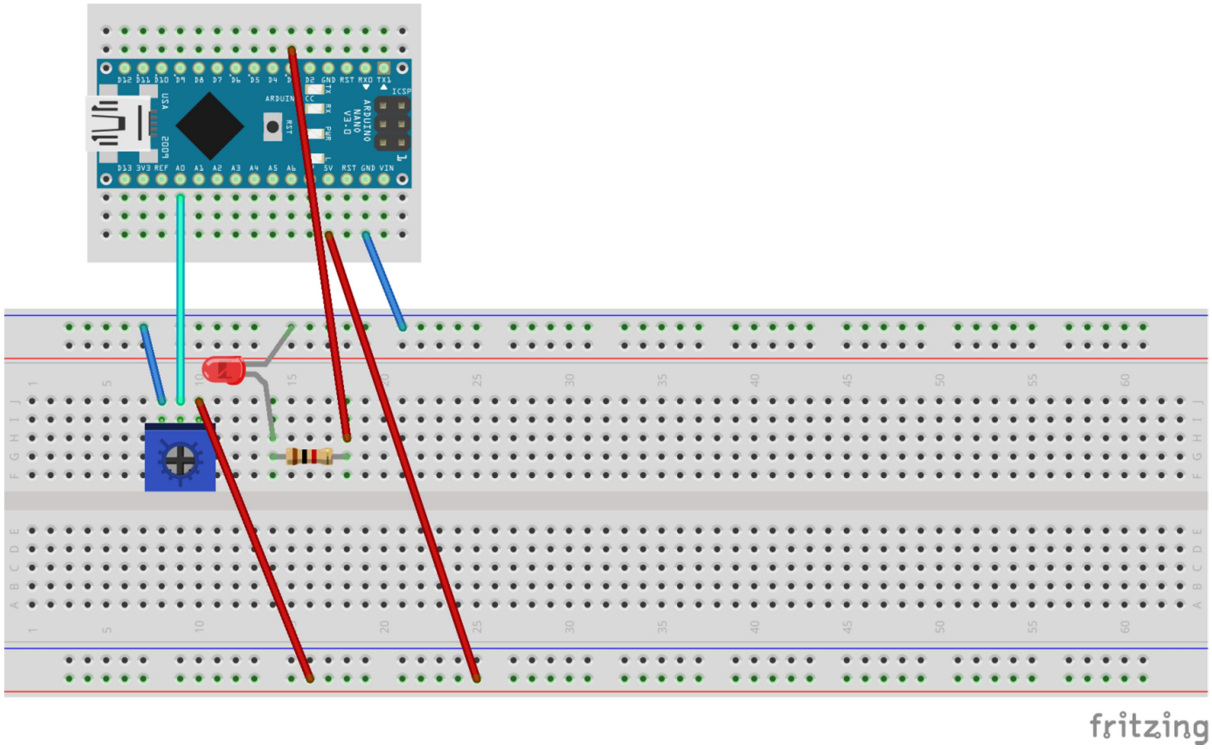


Slika 14. Električna šema uz prvi korak zadatka 2

Podešavanje intenziteta svetlosti diode može se vršiti dovođenjem na nju PWM signala, o kome je bilo reči na predavanjima. Arduino IDE poseduje ugrađenu funkciju za generisanje PWM na određenom priključku, AnalogWrite(pin,vrednost), gde vrednost može imati vrednost između 0 i 255 i definiše trajanje visokog naponskog nivoa PWM signala u odnosu na periodu signala u opsegu 0-100%.

Očitavanje analognog napona se u okviru Arduino IDE vrši pozivanjem funkcije AnalogRead(pin).

Povezivanje šeme sa slike 14. prikazano je na slici 15.

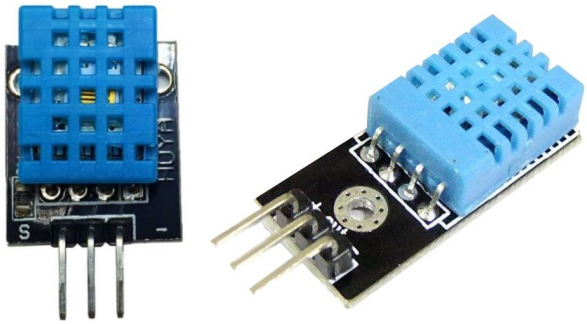


Slika 15. Povezivanje u prvom koraku zadatka 2

Programski kod koji odgovara tekstu zadatka prikazan je u nastavku. Prekopirajte ga u Arduino IDE, izvršite kompajliranje i prebacivanje na razvojnu ploču. Šrafcigerom koji ste dobili u kompletu okrećite osovinu potenciometra do kraja u oba smera (čućete kliktanje kada dođete do kraja) i u Serial Monitoru u okviru Arduino IDE posmatrajte poruke koje mikrokontroler šalje. Istrovremeno posmatrajte šta se dešava sa osvetljajem LED diode.

```
#define ADCIN  A0
#define LED   3
int analog_value;// promenljiva u kojoj se cuva ocitana vrednost sa ADC

void setup()
  {
  Serial.begin(9600);//ukljucivanje UARTa
  pinMode(LED,OUTPUT);digitalWrite(LED,LOW);//podesavanje izlaza
  }

void loop()
  {
  //citanje analogne vrednosti, ADC je 10 bita, opseg 0-1023, zato delimo sa 4
  analog_value = analogRead(ADCIN)/4;
  analogWrite(LED,analog_value);//upis PWMa
  Serial.println(analog_value);//slanje na UART
  delay(500);//cekanje 500ms
  }
```

U **drugom delu** drugom zadatka očitavaćemo temperaturu i vlažnost sa senzora DHT11 pomoću Arduina i slati izmerene vrednosti na računar preko UART komunikacije. U vašim kompletima imate ovaj senzor montiran na pločici, u dve varijante prikazane na slici 16. Obe varijante su funkcionalno iste, samo je različit raspored priključaka. U varijanti na slici 16 levo priključci su s leva na desno redom komunikaciona linija (oznaka **S**), napajanje 5V (bez oznake) i masa (oznaka **–**). U varijanti na slici 16 desno priključci su s leva na desno redom napajanje 5V (oznaka **+**), komunikaciona linija (oznaka **out**) i masa (oznaka **–**).

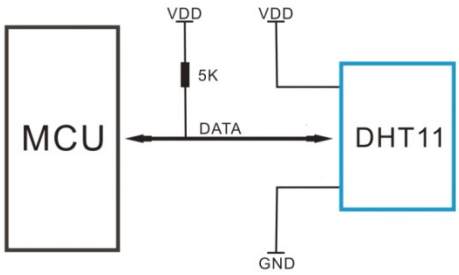


Slika 16. Senzor DHT11

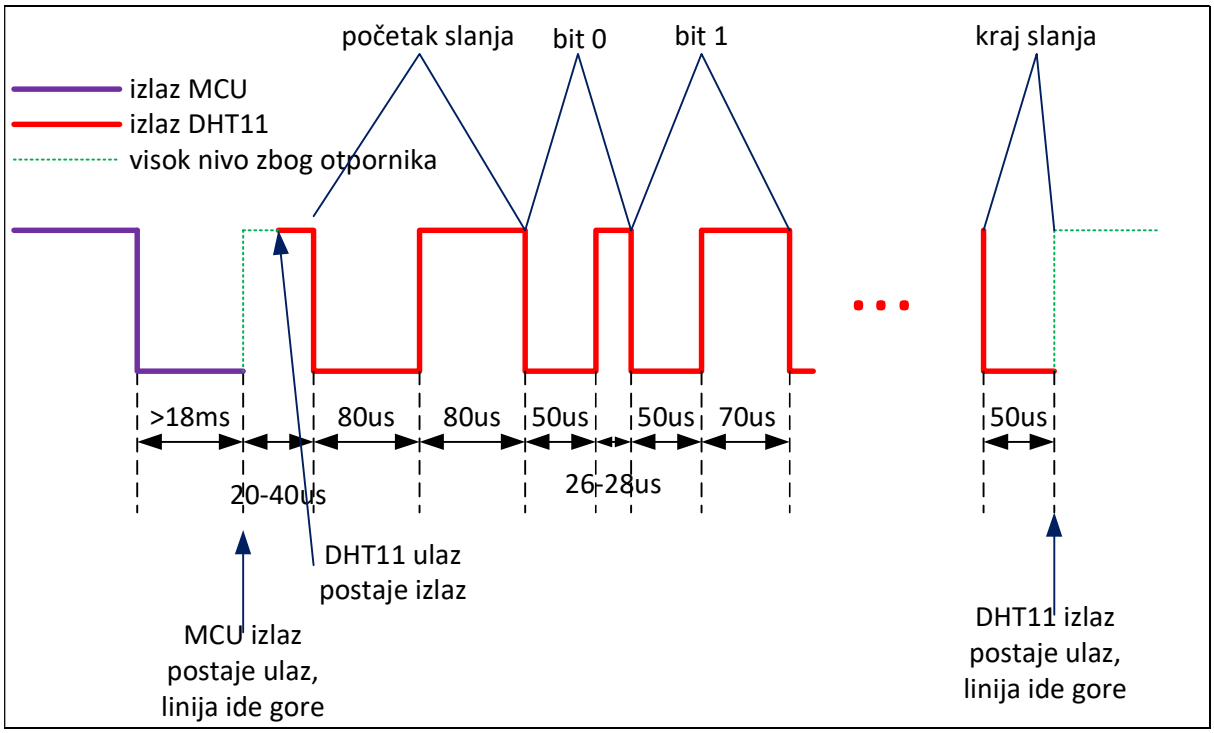
DHT11 sa mikrokontrolerom komunicira na nestadardan način, preko jedne linije, i za to ne postoji posebna periferija u mikrokontroleru. Umesto toga, potrebno je komunikaciju implementirati u softveru.

Kada komunikacije nema linija je na visokom naposkom nivou. Mikrokontroler šalje zahtev za rezultatima merenja senzoru tako što na liniju postavi nizak nivo u trajanju od najmanje 18ms praćen visokim nivoom trajanja 20-40 us, nakon čega mikrokontroler menja smer svog priključka i očekuje informacije od DHT 11. Dok senzor ne počne da šalje linija je na visokom nivou zbog pull-up otpornika koje se nalazi na pločici sa senzorom (Slika 17).

DHT11 početak komunikacije signalizira niskim nivoom na liniji u trajanju od 80 us praćen visokim nivoom istog trajanja. Nakon toga DHT11 šalje 40 bita informacije kodovana vremenski na sledeći način: bit 0 se šalje kao nizak nivo trajanja 50 us praćen visokim nivoom trajanja 26-28 us, dok se bit 1 šalje kao nizak nivo trajanja 50 us praćen visokim nivoom trajanja 70 us. Nakon što se prenese svih 40 bita, DHT11 poslednji put postavlja liniju na nizak nivo u trajanju od 50 us, nakon čega odvaja svoj izlaz od linije, koja ostaje na visokom nivou zbog pull-up otpornika. Vremenski dijagram komunikacije prikazan je na slici 18.

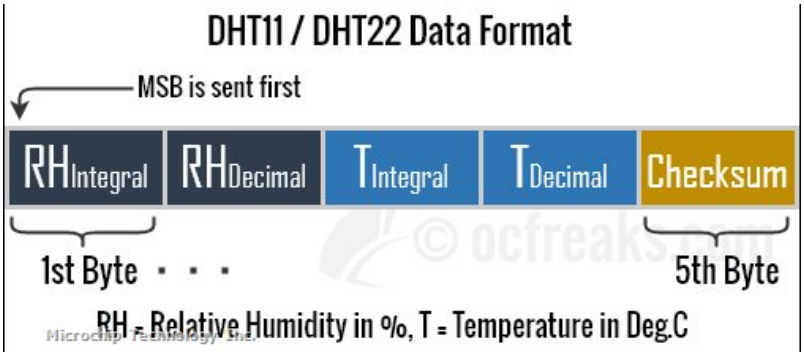


Slika 17. Povezivanje DHT11 i mikrokontrolera



Slika 18. Vremenski dijagram komunikacije između mikrokontrolera i DHT11

40 bita informacije koju DHT11 šalje je organizovano tako (slika 19) da se prvo šalje 8 bita koji predstavljaju ceo broj procenata vlažnosti, zatim 8 bita koji predstavljaju decimalni deo procenata vlažnosti, potom 8 bita koji predstavljaju ceo broj stepena Celzijusovih temperature, pa 8 bita koji predstavljaju 8 bita decimalnog dela stepena Celzijusovih temperature. Poslednjih 8 bita koriste se za proveru da nije došlo do greške u prenosu, čime se nećemo baviti u ovoj vežbi. U svakoj grupi od 8 bita prvo se šalje bit najveće težine. **U ovom zadatku razmatraćemo samo cele vrednosti vlažnosti i temperature.**



Slika 19. Tumačenje podataka sa DHT11

Sa slike 18 se uočava da se očitavanje podataka sa DHT11 svodi na merenje trajanja visokog nivoa koji sledi nakon niskog nivoa konstantnog trajanja od 50 us. Jednostavnije je, pošto nizak nivo ima konstantno trajanje, meriti trajanje između dve susedne promene napona na liniji sa visokog na niski nivo. Ukoliko je to trajanje oko 80 us u pitanju je prenos bita 0, ako je oko 120 us u pitanju je prenos bita 1. Još jednostavnije, može se postaviti prag od recimo 100 us, i ako je trajanje manje od njega u pitanju je bit 0, ako je veće u pitanju je bit 1.

Za merenja vremena reda mikrosekunde u Arduino IDE može se koristi funkcija micros() koja koristi Timer/Counter 0, ali ćemo mi u ovoj vežbi direktno raditi sa registrima 8 bitnog Timer/Countera 2.

Naime, ako brojač broji unapred od 0 sa taktom reda us, njegovim očitavanjem dobijamo precizu informaciju o proteklom vremenu. Registar brojača se zove TCNT2, dok se podešavanja njegovog takta nalaze u kontrolno-statusnom registru TCCR2B, prikazanom na slici 20.

Bit	7	6	5	4	3	2	1	0	
(0xB1)	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	TCCR2B
Read/Write	W	W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Slika 20. Registar TCCR2B

Kako smo rekli na predavanjima, biti CS2(2:0) definišu takt brojača, prema tabeli sa slike 21. Kombinacija 000 zaustavlja brojač. Iz tabele možemo kao željenu kombinaciju izabrati 011, što znači da je takt brojača osnovni takt (16 MHz) podeljen sa 32, odnosno 500kHz. Tada je perioda takta 2 us, što znači da će za impuls trajanja 100 us brojač izbrojati do 50.

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(no prescaling)
0	1	0	clk_{T2S}/8 (from prescaler)
0	1	1	clk_{T2S}/32 (from prescaler)
1	0	0	clk_{T2S}/64 (from prescaler)
1	0	1	clk_{T2S}/128 (from prescaler)
1	1	0	clk_{T2S}/256 (from prescaler)
1	1	1	clk_{T2S}/1024 (from prescaler)

Slika 21. Podešavanje takta za Timer/Counter 2

Da bi brojač brojao unapred potrebno je da se podese biti WGM2(2:0), od kojih je WGM22 u registru TCCR2B, dok su preostala dva, WGM21 i WGM20 u registru TCCR2A (slika 22).

Bit	7	6	5	4	3	2	1	0	
(0xB0)	COM2A1	COM2A0	COM2B1	COM2B0	–	–	WGM21	WGM20	TCCR2A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Slika 23. Registar TCCR2A

Sa slika 21 i 23 se vidi da je inicijalna vrednost po uključenju napajanja WGM2(2:0) = 000, što odgovara brojanju unapred od nule do maksimalne vrednosti (slika 24).

Mode	WGM2	WGM1	WGM0	Timer/Counter Mode of Operation	TOP
0	0	0	0	Normal	0xFF
1	0	0	1	PWM, phase correct	0xFF
2	0	1	0	CTC	OCRA
3	0	1	1	Fast PWM	0xFF
4	1	0	0	Reserved	–
5	1	0	1	PWM, phase correct	OCRA
6	1	1	0	Reserved	–
7	1	1	1	Fast PWM	OCRA

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

Slika 24. Podešavanje načina brojana za Timer/Counter 2

Ostali biti u registrima TCCR2A i TCCR2B nisu od interesa, tiču se generisanja PWM pomoću T/C 2, i njihove inicijalne vrednosti koje su nula odgovaraju našim potrebama.

Praktično, kada je potrebno merenje, u programu je potrebno isključiti takt brojaču preko bita CS2(2:0), upisati početnu vrednost u TCNT2 (to je nula), uključiti takt brojaču preko bita CS2(2:0), i čitanjem registra TCNT2 dobijati informaciju o proteklom vremenu.

Povezati preko protoborda DHT11 i Arduino Nano pločicu. Linija za komunikaciju može biti povezana na bilo koji digitalni priključak Arduino Nano, na primer priključak 4. DHT11 je potrebno osim linije za komunikaciju povezati i 5 V i masu sa Arduino Nano.

Programski kod koji odgovara tekstu zadatka prikazan je u nastavku. Prekopirajte ga u Arduino IDE, izvršite kompajliranje i prebacivanje na razvojnu ploču. U Serial Monitoru u okviru Arduino IDE posmatrajte poruke koje mikrokontroler šalje. Pokušajte da duvate “toplo” u senzor i posmatrajte da li će doći do ikakve promene pokazivanja.

```
#define DHT_PIN 4
unsigned char bits[40];//niz u kome se cuvaju vrednosti bita
unsigned char data_count;//brojac primljenih bita
unsigned char temperature, humidity;

void setup()
  {
  Serial.begin(9600);//konfiguracija UARTa
  }

void loop()
  {
  pinMode(DHT_PIN,OUTPUT);//pin izlazni
  digitalWrite(DHT_PIN,HIGH);//jedinica na izlaz, neaktivno stanje linije
  delay(250);//250ms pre nego sto pocnemo, proizvoljno trajanje
  digitalWrite(DHT_PIN,LOW);//0 na izlazu, da startuje senzor
  delay(20);//cekanje 20ms, po dokumentaciji teba da je >18ms.
  pinMode(DHT_PIN,INPUT);//pin menaj smer na ulazni
  while (digitalRead(DHT_PIN) == HIGH);//cekamo da senzor liniju obori na nulu

 //poceo start od senzora, sada treba za 80us da ide napon gore, pa za 80 us opet dole, to necemo da merimo
  while (digitalRead(DHT_PIN) == LOW);//cekamo da ulaz ode na jedan
  while (digitalRead(DHT_PIN) == HIGH);//kada ulaz opet padne na nulu zavrsio se prvi bit koji govori da pocinje slanje
  TCCR2B = 0;//zaustavljam tajmer pre nego sto upisem u njega za svaki slucaj
  data_count = 0;//brojac primljenih bita stavljam na nulu, treba da stigne 40 bita
  while (data_count<40) // citam 40 bita u petlji
    {
    TCNT2 = 0;//reestujem tajmer
    TCCR2B = 3;//ukjucujem tajmer sa ucestanoscu f/32, jedan inkrement je 2us posto je f = 16MHz
    while (digitalRead(DHT_PIN) == LOW);//cekamo da ulaz ode na jedinicu
    while (digitalRead(DHT_PIN) == HIGH);//pa cekamo da ulaz opet ode na nulu
    TCCR2B=0; //zaustavljam tajmer pre citanja i kasnijeg upisa
    if (TCNT2 <50) bits[data_count] = 0;//ako je proslo manje od 100us bit je 0
    else bits[data_count] = 1;//ako više onda je bit 1
    data_count++;
    }
  //konverzija nula i jedinica u konacne vrednosti, samo celobrojni deo
  humidity = 128*bits[0]+64*bits[1]+32*bits[2]+16*bits[3]
          +8*bits[4]+4*bits[5]+2*bits[6]+bits[7];
  temperature = 128*bits[16]+64*bits[17]+32*bits[18]+16*bits[19]
          +8*bits[20]+4*bits[21]+2*bits[22]+bits[23];
  Serial.print("vlaznost= ");Serial.print(humidity,DEC);Serial.print("% ");
  Serial.print("temperatura= ");Serial.print(temperature,DEC);Serial.println("C");

  delay(2000);//cekanje 2 sekunde do sledeceg merenja
  }
```

ZADATAK – 5 poena: Modifikovati šemu prethodnog zadatka dodavanjem potenciometra i jedne crvene LED diode (uz odgovarajući otpornik). U programu je potrebno dodati očitavanje potenciometra, čiji opseg 0-1023 odgovara temperaturi 0-50 C koja predstavlja prag za uključenje LED diode, ako je temepratura koja se očitava sa DHT11 veća od tog praga. Računaru se pored temperature i vlažnosti šalje i očitana vrednost praga (u C) za uključenje LED diode.