PROJEKTI IZ ELEKTRONIKE

dr. David Rihtaršič

2022-December

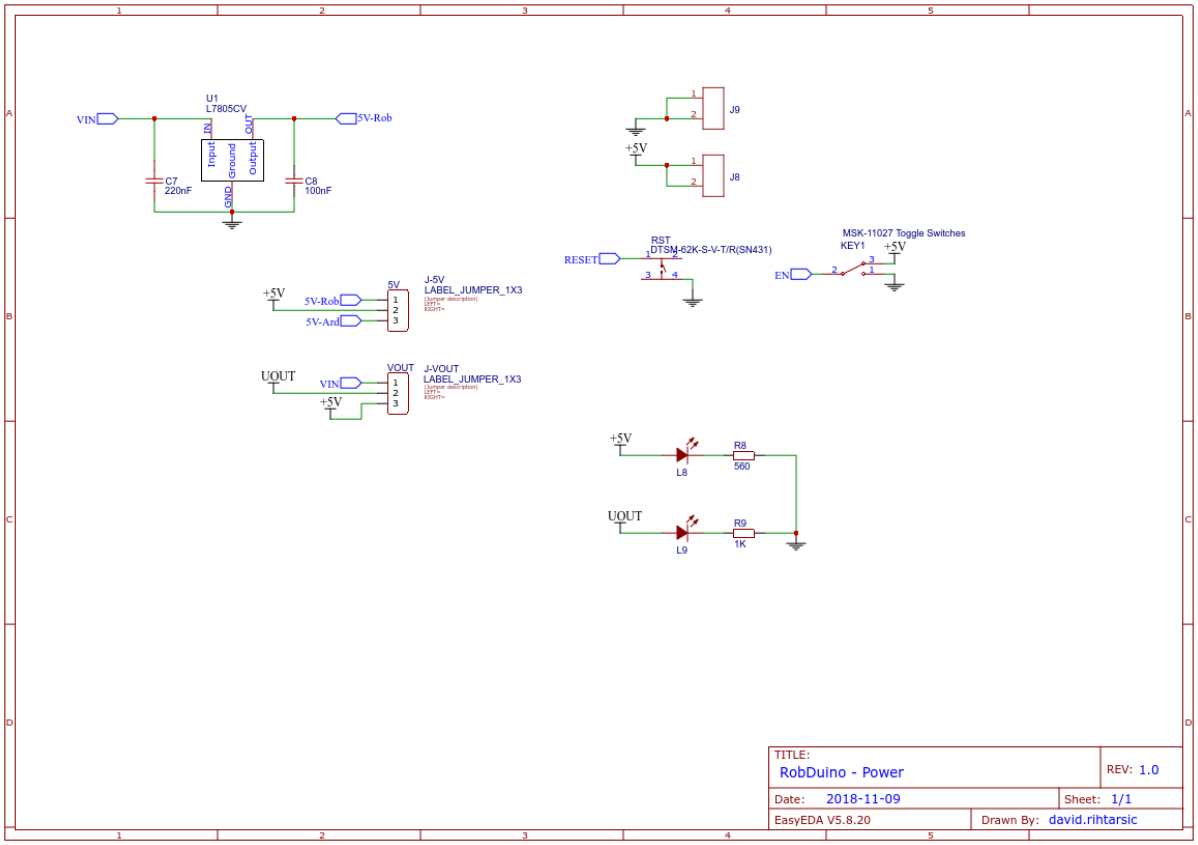
Kazalo

1 NAČRTOVANJEM ELEKTRONSKIH VEZIJ

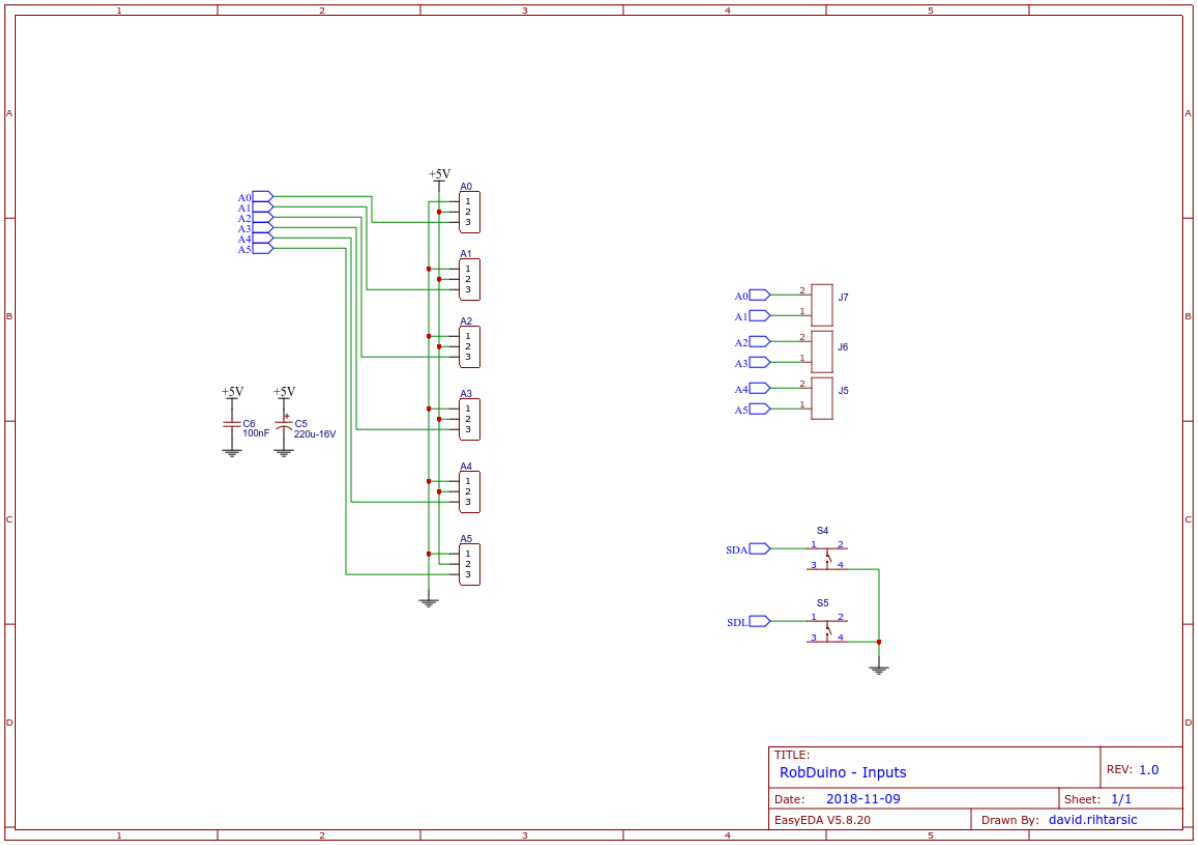
Tudi pri pedagoškem procesu je pomembno, da so vezja narisana nazorno (tako sheme, kot tudi sestavljanje vezja na prototipni ploščici). V ta namen lahko uporabljate različna orodja. Omenili bomo vsaj dva, ki sta prosto-dostopna in bi priporočali njihovo uporabo.

1.1 Stikalne sheme

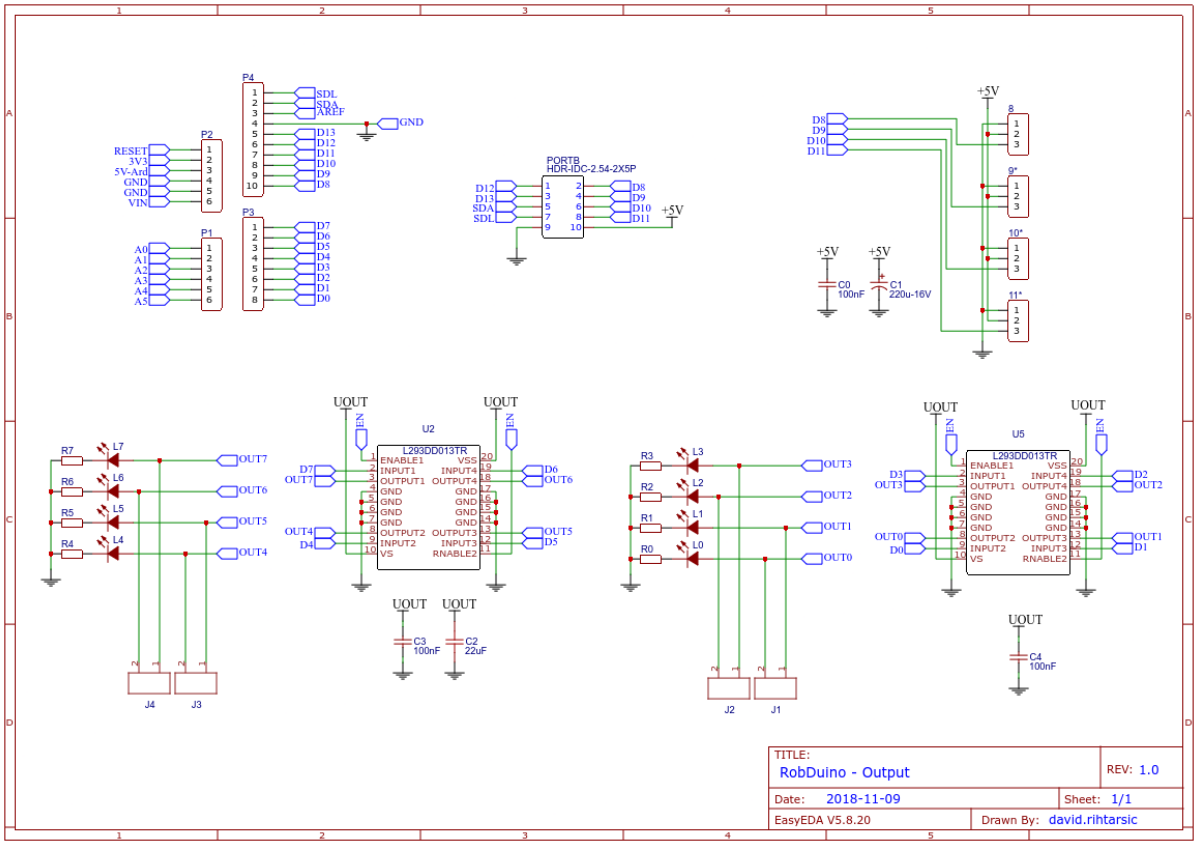
EasyEDA je spletno orodje, ki je namenjeno risanju elektronskih vezij, načrtovanju TIV in izdelavi potrebnih datotek za njihovo izdelavo. Primer stikalne sheme krmilnika RobDuino lahko vidite na sl. 1.1,sl. 1.2 in sl. 1.3.



Slika 1.1: Stikalna shema modula RobDuino - napajalni del.



Slika 1.2: Stikalna shema modula RobDuino - vhodni del.



Slika 1.3: Stikalna shema modula RobDuino - izhodni del..

1.1.1 NALOGA: Stikalne sheme

V programskem orodju EasyEDA narišite shemo astabilnegamultivibratorja, izvozite sliko sheme in j vključite v poročilo.

1.2 Tiskana vezja

1.2.1 NALOGA: Risanje TIV

Za to vezje izrišite TIV in izpišite seznam elektronskih komponent. Izgled TIV izvozite in vstavite v poročilo. Prav tako vstavite seznam komponent.

1.3 Virtualna vezja

Sestavljanje vezij pa na prototipni plošči pa je drugačen proces in za začetnike precej zahteven, zato je priporočljivo, da uporabljate orodja kot so TinkerCAD-Circuits.

1.3.1 NALOGA: Sestavljanje virtualnih vezij

V programskem orodju TinkerCAD-Circuits sestavite vezje na prototipni ploščici in sliko vstavite v poročilo.

2 SVETLOBNI SPOJNIKI

Pred davnimi časi, ko svet še ni slišal za digitalno tehnologijo, se je gospod Samuel Morse domislil, da bi črke kodiral v kratke in dolge pulze. Te pa bi lahko kar najlažje pošiljal od ene točke do druge na najrazličnejše načine … in telekomunikacije so prijokale na svet.

2.1 Vklop porabnika s tranzistorjem

Pogosto moramo porabnike skozi katere tečejo večji tokovi ($I > 500\,mA$) vključiti s tranzistorjem.

2.1.1 NALOGA: Vklop žarnice s tranzistorjem

1. Za pošiljanje Morsejevih znakov uporabite žarnico. Ali nek drug vir z večjo svetilnostjo.
2. Dolge in kratke pulze bomo pošiljali s svetlobnim oddajnikom. Načrtujete ustrezno rešitev (narišite shemo vezja) tako, da bomo s pritiski na tipko vklapljali in izklapljali svetilo (uporabite žarnico [12V in 0,6 A]).
3. Bodite pozorni, na električne omejitve tipke, ki jih najdete v navodilih za uporabo tipke. Načrtujte ustrezno rešitev.

2.2 Komparator napetosti

V kolikor želimo ločiti med dvema napetostnima nivojema, lahko uporabimo komparator napetosti s primerno izbrano (ali celo nastavljivo) referenčno vrednostjo.

2.2.1 NALOGA: Komparator napetosti

1. Izdelajte svetlobni sprejemnik, v katerega boste za zaznavanje osvetljenosti uporabili elektronski element s hitrim odzivom.

2. Analogni signal senzorja modificirajte tako, da boste lahko nedvoumno podajali informacijo (npr. LED svetilo), ki jo je poslal svetlobni oddajnik.
3. Z osciloskopom zajemite časovni potek napetostnega potenciala na vhodnem in na izhodnem priključku komparatorja.

2.3 Schmittov sprožilnik

Iz prejšnje naloge lahko ugotovimo, da je izhodni signal svetlobnega senzorja opremljen z neželeno motnjo. Če v tem primeru želimo ločiti dve različni stanji senzorja, komparator napetosti ni dobra rešitev. Ob prehodu senzorja iz enega stanja v drugega tako dobimo na izhodu komparatorja več prehodov, čeprav se je osvetljenost senzorja spremenila le enkrat. Tako lahko ugotovimo potrebo po histerezi s katero bomo bomo mejno vrednost razmejili na dve mejni vrednosti. Tako ločevanje nam omogoča schmittov sprožilnik.

2.3.1 NALOGA: Komparator napetosti s šmitovim sprožilnikom

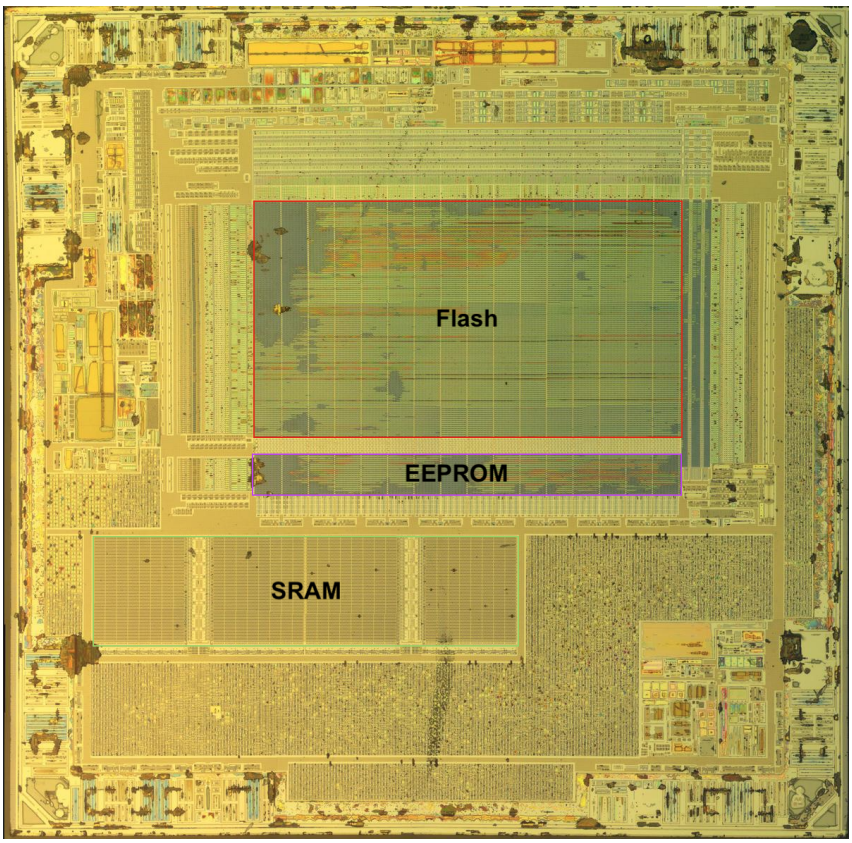
1. Iz grafa prejšnje naloge, ki predstavlja časovno odvisnost napetostnega potenciala vhodnega in izhodnega signala, odčitajte nove mejne vrednosti ($U_{h1}\ in\ U_{h2}$).
2. Komparator napetosti iz prejšnje naloge zamenjajte s komparatorjem napetosti s šmitovim sprožilnikom (narišite stikalno shemo).
3. Z osciloskopom zajemite časovni potek napetostnega potenciala na vhodnem in na izhodnem priključku komparatorja napetosti s šmitovim sprožilnikom.

3 UPORABA MIKROKRMILNIKOV

Za projekte, ki vključujejo programabilno elektroniko, pogosto uporabljamo že izdelane krmilnike iz družine Arduino. Na teh vezjih lahko najdemo mikrokrmilnike proizvajalca Atmel. Najbolj pogosto uporabljena krmilnika (Arduino Uno in Arduino NANO) temeljita na mikrikrmilniku Atmega328p. Blokovna shema tega mikrokrmilnika je prikazana na sl. 3.2.

3.1 Shema mikrokrmilnika ATmega238

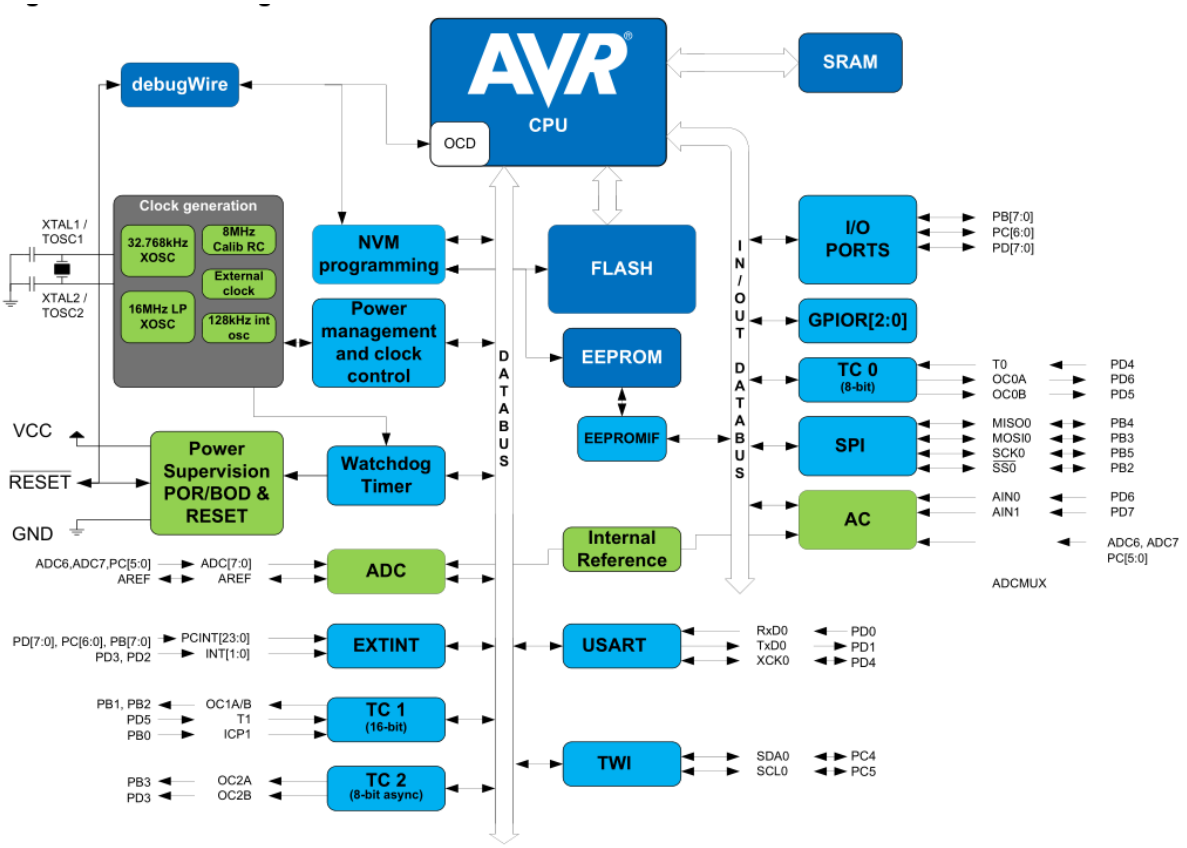
Mikrokrmilniki so integrirana vezja, z zelo kompleksno notranjo strukturo. Sestavlja jih na milijone tranzistorjev, ki s povezavami in ostalimi osnovnimi elementi sestavljajo smiselne logične sklope. Povečana slika dejanskega integriranega vezja mikrokrmilnika ATmega238 je na sl. 3.1.



Slika 3.1: Slika vezja mikrokrmilnika ATmega238 na plošči SiO2

Iz sl. 3.1 je nemogoče razbrati posamezne dele integriranega vezja. Opazimo lahko le večje enake sklope, ki so namenjene spominskim funkcijam. Še bolj podrobno sliko pa si lahko ogledate na povezavi ATmega238-SiO2.

Pri tako kompleksnih vezjih je bolj smiselno, da posamezne logične sklope predstavimo z blokovno shemo. Tako shemo lahko najdemo v navodilih za uporabo mikrokrmilnika ATmega238 in je prikazana na sl. 3.2.



Slika 3.2: Blokovna shema mikrokrmilnika ATmega238.

3.1.1 NALOGA: Glavni deli krmilnika

1. Na blokovni shemi označi pomembne dele krmilnika in zapiši njihov glavni namen (funkcijo). Glavni sestavni deli so: generator delovnega takta (ura), centralno procesna enota, začasni (delovni) spomin, trajni spomin, vhodno-izhodne enote, komunikacijske enote …

Mikrokrmilniki na krmilnikih Arduino so že opremljeni s programom (angl. »boot loader«), ki poskrbi za ustrezno prepisovanje programske vsebine, ki jo računalnik pošlje preko USB vodila. Tako lahko enostavno programiramo mikrokrmilnike, ki so na ploščah krmilnikov Arduino.

3.1.2 NALOGA: Osnovne nastavitve in testni program

1. Iz programskega okolja Arduino IDE prepišite nastavitve programatorja ter izpolni tbl. 3.1:

Tabela 3.1: Nastavitveni parametri programatorja.

Nastavitveni parameter	Vrednost nastavitvenega parametra
Board (Plošča)	
~~Procesor~~ (Mikrokrmilnik)	
Port (vrata)	
Programmer (programator)	

2. Iz primerov, ki so vključeni v programskem okolju Arduino IDE izberite Blink.ino in ga preskusite.

3.2 Programiranje krmilnikov Arduino

Za programiranje mikrokrmilnika skrbi odprtokodna programska koda - avrdude, ki se izvaja v ozadju programskega okolja Arduino IDE. Proces programiranja lahko bolj natančno spremljamo tako, da vključimo

```
File -> Preferences:

Shov verbose output during: [x] compilation [x] upload
```

3.2.1 NALOGA: AVRDUDE - program za prenos strojne kode

1. Prepišite ukazno vrstico programa avrdude za premos strojne kode in
2. opišite pomen parametrov.

Več o parametrih lahko najdete na spletni strani avrdude

3.3 Uporaba vhodno-izhodnih priključkov na kmilniku Arduino

3.3.1 NALOGA: Shema krmilnika Arduino NANO

1. Oglejte si shemo krmilnika Arduino NANO in

2. skušaj ugotoviti na kateri priključek IO enote je priključena LED, ki je na krmilniku.
3. Izpolni tbl. 3.2

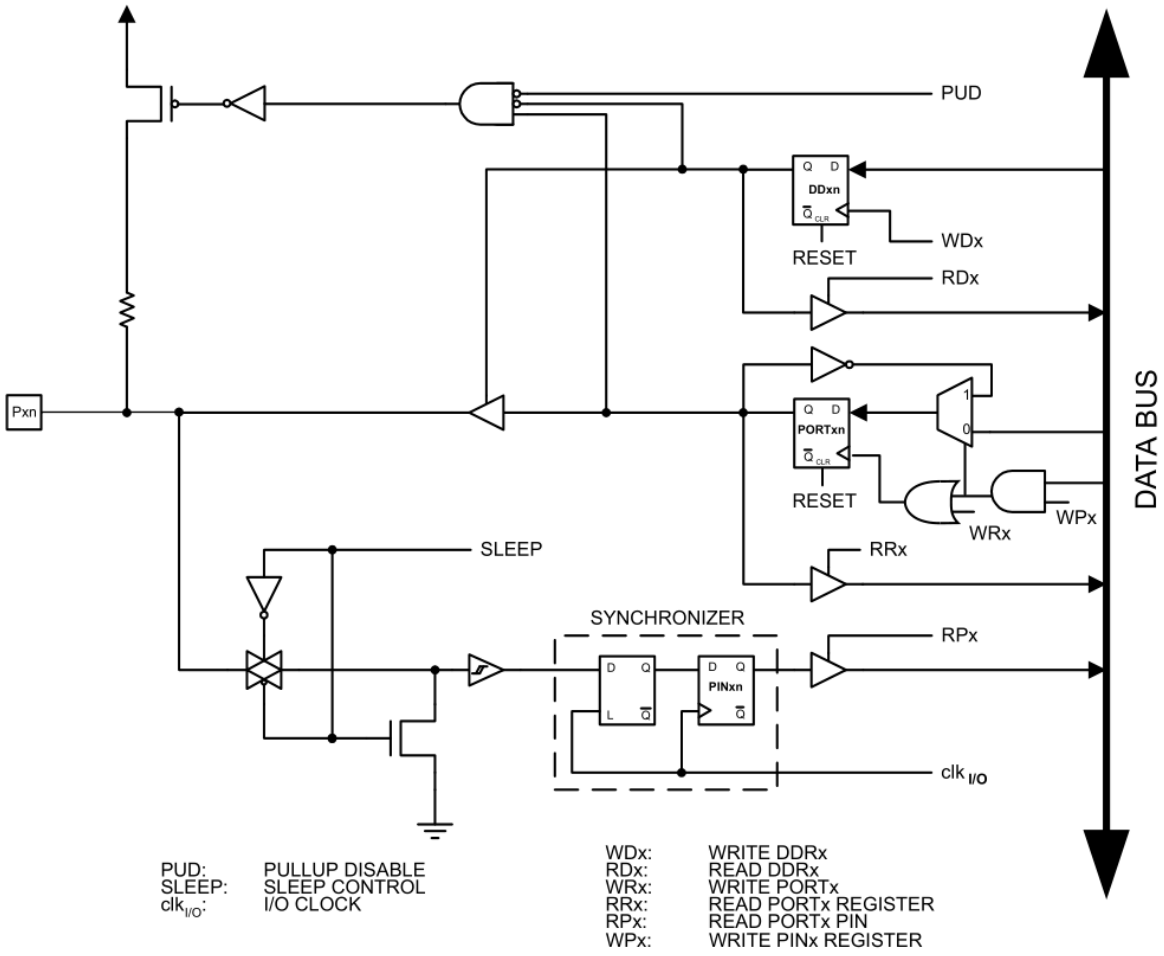
Tabela 3.2: Razporeditev priključkov na krmilniku in mikrikrmilniku.

Funkcija krmilnika	Arduino NANO[a]	ATmega238[b]
LED na plošči		
TxD		
RxD		

[a] Oznaka priključka na krmilniku Arduino NANO.
[b] Oznaka priključka na mikrokrmilniku ATmega238.

3.4 Vhodno-izhodne enote mikrokrmilnika

Vhodno-izhodne enote mikrokrmilnika so dvo-smerne z možnostjo nastavitve upora proti napajanju. Stikalna shema enega priključka na neki vhodno-izhodni enoti je prikazana na sl. 3.3.



Slika 3.3: Shema priključka n na vhodno-izhodni enoti x.

Čeprav je shema na sl. 3.3 nekoliko bolj kompleksna, lahko ugotovimo, da za nastavitev delovanja vhodno-izhodne enote potrebujemo le dva signala:

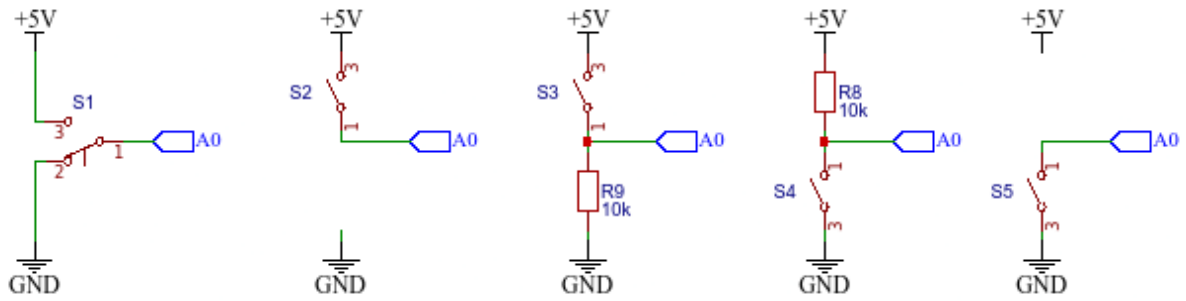
- WDx - (DDRx) - za določanje vhodne oz. izhodne funkcije priključka
- WPx - (PORTx) - za določanje vrednosti logičnega stanja na priključku x.

3.4.1 NALOGA: Krmilni registri mikrokrmilnika

1. Preoblikuj program Blink.ino tako, da boš krmilil vklop in izklop LED z nastavitvijo krmilnikh registrov.
2. Predstavite programsko kodo.

3.5 Branje napetostnega potenciala

Branje napetostnega potenciala je najbolj pogosto uporabljeno pri vzorčenju pritiska tipke. Poglejmo si tak primer.



Slika 3.4: Različne možnosti priključitve tipke na digitalni vhod.

3.5.1 NALOGA: Branje napetostnega potenciala

1. Sestavite prve tri različice priključitve tipke na digitalni vhod krmilnika A0 in preskusite spodnji program.
2. Razložite (ne-)delovanje.

```
void setup(){
    pinMode(13, OUTPUT);
    pinMode(A0, INPUT);
}

void loop(){
    if ( digitalRead(A0) == HIGH)
        digitalWrite(13, HIGH);
    else
        digitalWrite(13, LOW);
}
```

3.5.2 NALOGA: Uporaba upora proti napajanju

1. Spremenite delilnih napetosti tipka - upor tako, da bo upor vezan proti napajanju in nato spremenite program tako, da bo delovanje ostalo enako prejšnji nalogi. Priložite stikalno

shemo vezja in preoblikovan program.
2. Odstranite upor in preverite delovanje vezja. Kaj opazite, razložite delovanje.
3. Vključite notranji upor proti napajanju, ki se nahaja v mikrokrmilniku ob vsakem vhodno-izhodnem priključku (glej sl. 3.3). Priložite programsko spremembo.

3.6 Prehodni stiki stikal

Fizični preklopni elementi (kot je tipka) imajo tudi fizikalne lastnosti kot so trdota, trdnost, prožnost, elastičnost… Zaradi vseh teh lastnosti je preklop tipke iz enega položaja v drugega lahko tudi nekoliko nepredvidljiv.

3.6.1 NALOGA: Večkratni preklopi

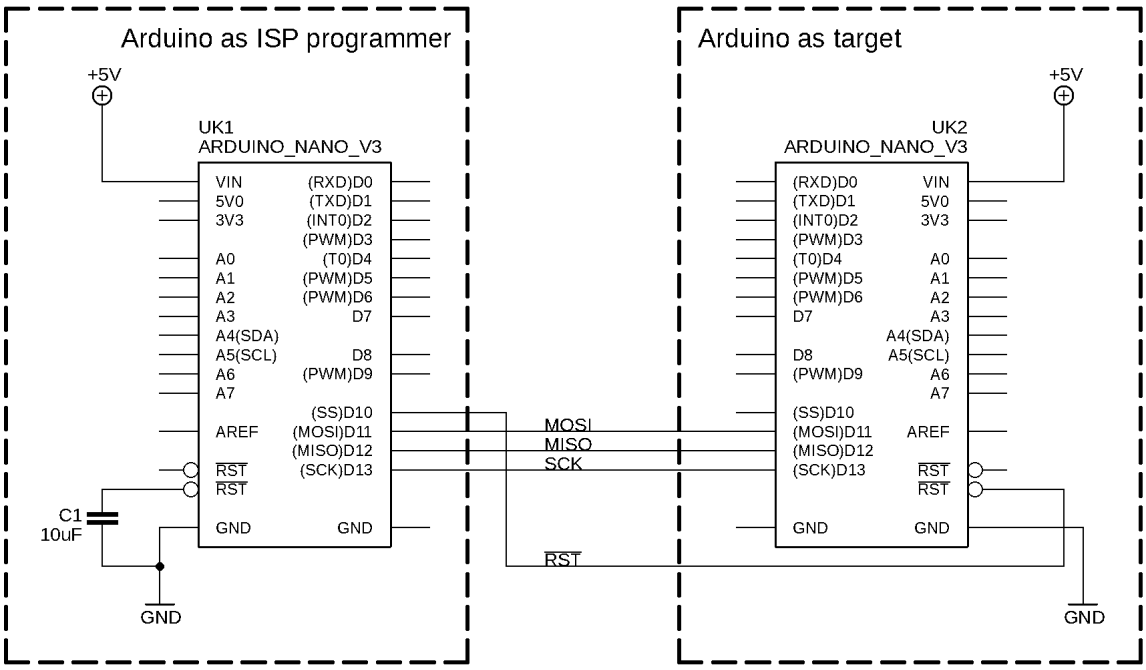
1. Preskusite spodnji program in opišite njegovo delovanje.
2. Opišite zaznane težave.
3. Z osciloskopom ujemite enega od prehodov (0->1 ali 1->0) in
4. problem rešite:
    - programsko ter
    - elektronsko. Obe rešitvi dokumentirajte.

```
void setup()
{
    const int LED = 13;
    pinMode(LED, OUTPUT);
    pinMode(A0, INPUT_PULLUP);
}

int i=0;

void loop()
{
    int tipka_je_pritisnjena = !digitalRead(A0);
    if (tipka_je_pritisnjena)
    {
      i++;
      while (tipka_je_pritisnjena)
        tipka_je_pritisnjena = !digitalRead(A0);
      int i_je_liho = i % 2;
      if (i_je_liho)
        digitalWrite(13, HIGH);
      else
        digitalWrite(13, LOW);
    }
}
```

3.7 PROGRAMIRANJE MIKRMILNIKOV

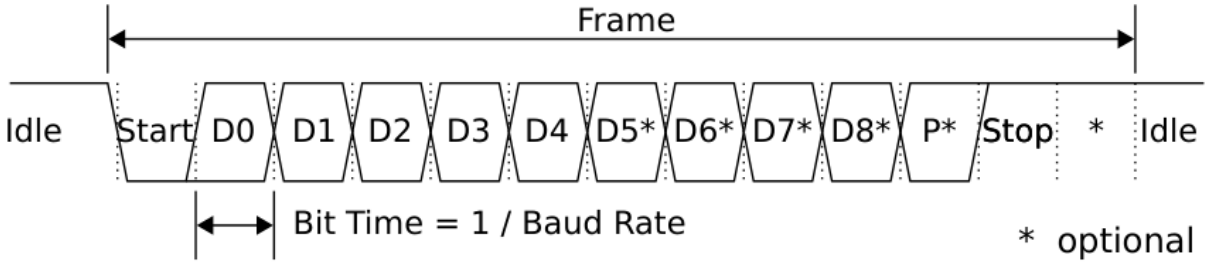


Slika 3.5: Shema povezave krmilnika Arduino nano kot programator s krmilnikom Arduino nano kot ciljno vezje.

4 KOMUNIKACIJSKI VMESNIKI

4.1 Serijska komunikacija UART

Nekaj več o UART komunikaciji si lahko preberete vsepovsod na svetovnem spletu. Ker jo uporabljamo že več kot pol stoletja, lahko rečemo, da sodi med osnovne komunikacijske protokole.



Slika 4.1: Časovni potek napetosti na kominikacijski povezavi.

4.1.1 NALOGA: Osnovni parametri UART protokola

1. Preučite UART protokol in prerišite časovni potek signala (teoretično).
2. Na teoretičnem primeru komentirajte:

- pomen napetostnega signala,
- kako si sledijo podatkovne informacije,
- označite start in stop bit,
- izračunajte dolžino trajanja enega bita pri $baud = 9600b/s$.

Komunikacija UART je tako razširjena, da jo vključujejo v skoraj vse programabilne elektronske komponente in Arduino NANO ni nobena izjema. Mikrokrmilnik ATmega328p vsebuje enoto za komunikacijo UART in je dostopna na priključkih 0 (Rx) in 1 (Tx). Preko te enote lahko pošiljamo/sprejemamo podatke drugih zunanjih naprav.

4.1.2 NALOGA: Uporaba serijske UART komunikacije

1. Preučite shemo za krmilnik Arduino NANO in poiščite priključka za UART komunikacijo.
2. Preskusite naslednji program mikrokrmilnika za pošiljanje nekega besedila računalniku in odziv spremjajte v serijskemu oknu programa ArduinoIDE:

```
1  void setup() {
2        Serial.begin(9600);
3  }
4  void loop() {
5    Serial.println("Pozdravljen svet.");
6    delay(1000);
7  }
```

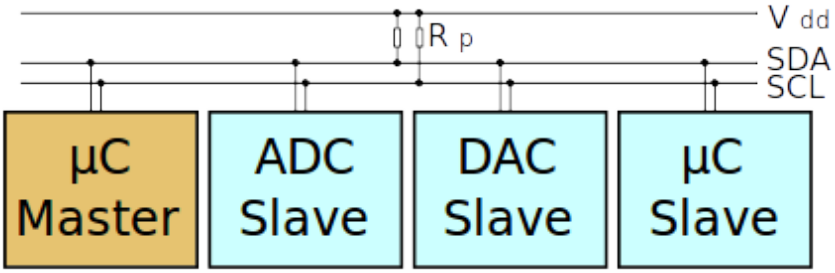
4.1.3 NALOGA: Časovni potek napetosti UART komunikacije

1. Z osciloskopom posnemite napetostni signal pošiljanja enega samega znaka in
2. parametre primerjajte s teoretičnimi vrednostmi komunikacije.
3. Na grafu U(t) označite logične vrednosti posameznih bitov in označite njihovo funkcijo (start bit, stop bit in položaj bita D0..D7).
4. Iz grafa U(t) odčitajte poslano podatkovno vrednost in jo primerjajte z ASCII tabelo.

```
1    Serial.print("M");
```

4.2 I2C komunikacija

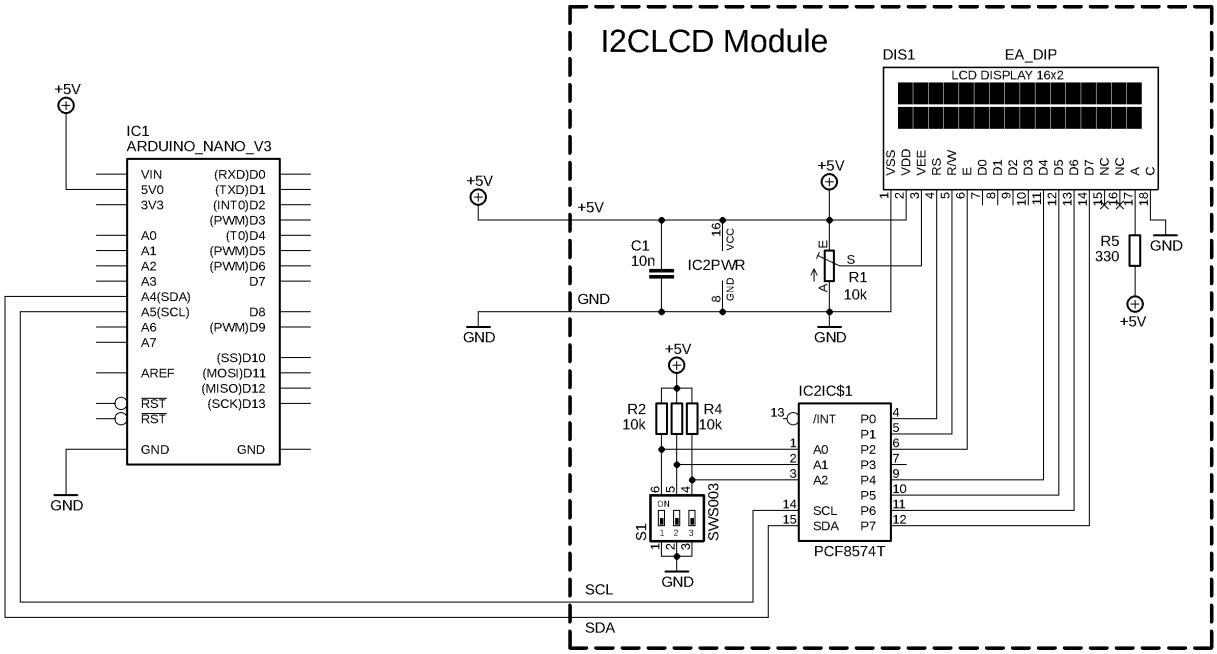
Komunikacija lahko poteka tudi na drugačne načine, na primer med več napravami. Ena takih komunikacij je t.i. I2C komunikacija. Več o tej komunikaciji si lahko preberemo na wikipediji o I2C podatkovnem vodilu



Slika 4.2: Na I2C vodilo priključene naprave.

V primeru, ki ga prikazuje sl. 4.2 je glavna naprava označena kot »master«, ki bo v našem primeru Arduino NANO. Ostale naprave pa so »podložniki«. Vsak od njih mora imeti svoj naslov in mora zanj glavna naprava vedeti, saj le tako lahko vzpostavi komunikacijo z njim (podobno kot IP številke v TCP/IP omrežju).

Naslove podložnikov včasih lahko nastavimo ročno na podložniku ali pa so zapisani že v sami napravi podložnika. Slednjo situacijo si lahko ogledamo na primeru LCD z I2C vodilom.



Slika 4.3: Priključitev LCD-ja na I2C vodilo.

4.2.1 NALOGA: Priključitev I2C LCD-ja

1. Priključite LCD z I2C vodilom na Arduino NANO tako, kot prikazuje sl. 4.3 in s programom, ki ga najdete na Arduino strani, ugotovite njegov naslov, ter ga zapišite : _____

Ker je sam protokol komunikacije bolj zapleten, bomo v ta namen uporabljali knjižnico LiquidCrystal_I2C.h. Knjižnico lahko namestimo v Arduino IDE tako:

```
Sketch -> Include Library -> Manage Libraries

Filter your Serch... : LiquidCrystal I2C (by Frank de Brabander)
```

Ta knjižnica vsebuje podobna imena funkcij, kot jih uporabljamo pri objektu Serial za serijsko komunikacijo (npr: Serial.print).

4.2.2 NALOGA: Izpis na LCD

1. Preskusite naslednji program in
2. nastavite primeren kontrast LCDja.
3. Vezju dodajte še potenciometer, s katerim bomo lahko poljubno nastavljali napetostni potencial in ga merili na priključku A0.
4. Spremenite program tako, da boste izpisovali na LCD izpisovali napetost in ne ADC vrednost.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C Lcd(0x27, 16, 2);
int adcValue = 0;
const int POT_PIN = A0;
void setup() {
  pinMode(POT_PIN,INPUT);
  Lcd.init();
  Lcd.init();
  Lcd.backlight();
}
void loop() {
  adcValue = analogRead(POT_PIN);
  Lcd.clear();
  Lcd.print("potenciometer:");
  Lcd.setCursor(0,1);
  Lcd.print(adcValue);
  delay(200);
}
```

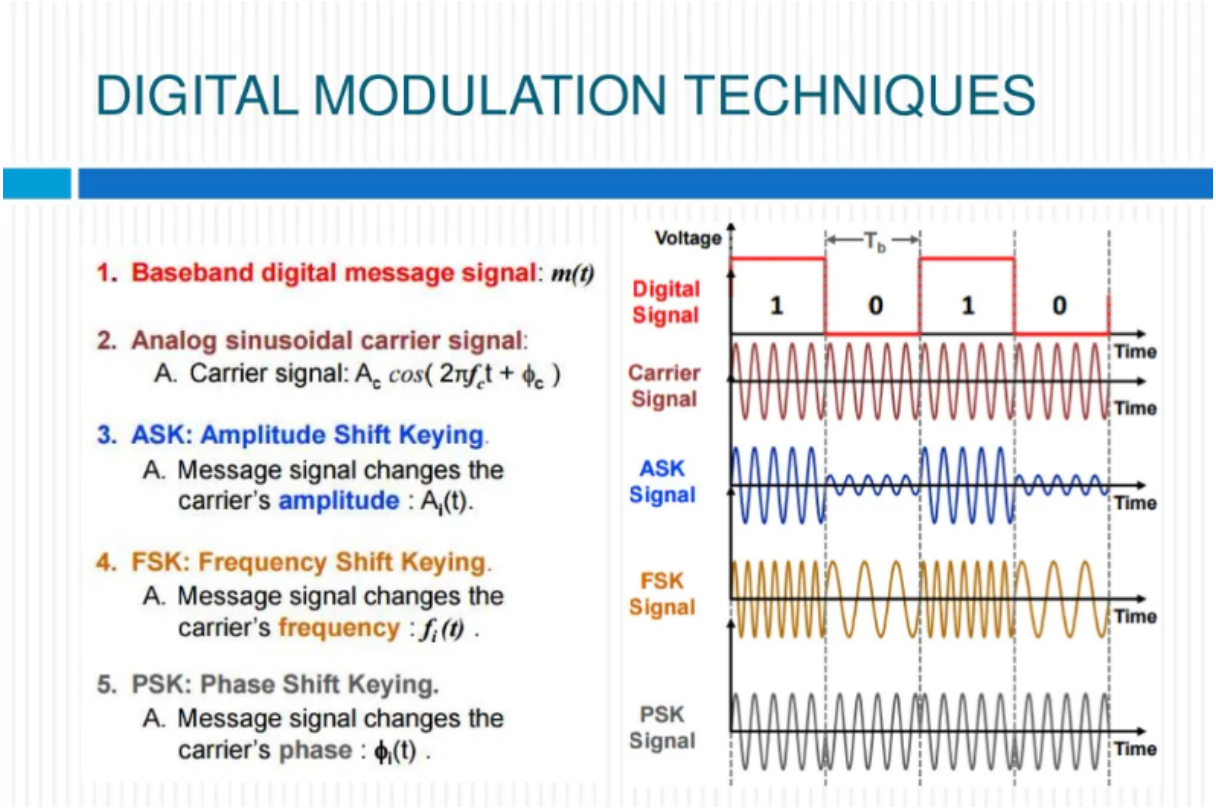
5 MODULACIJA

Pri prenosu podatkov z električnimi veličinami (U(t)) od ene naprave do druge se pogosto soočamo s kakovostjo prenesene informacije. Pri tem nas zanima kolikšen del električne količine (npr. napetosti) dejansko pripada poslani informaciji in kolikšen del drugim EM motnjam v vezju (angl.: Signal to noise ratio). Pri reševanju tega problema si pomagamo z modulacijo podatkovnih signalov.

Modulacija je proces pri katerem nekoliko spremenimo nekatere lastnosti dobro definiranega periodičnega signala (nosilni signal). Spremenjene lastnosti nosilnega signala pa so neposredno odvisne od same informacije.

Sprejemanje moduliranih signalov je tako lahko bistveno bolj selektivno. Že s preprostimi R-C elektronskimi siti lahko dosežemo bistveno boljšo kvaliteto signala.

Primeri moduliranih signalov so vidni na sl. 5.1.



Slika 5.1: Primeri zarličnih modulacijskih tehnik.

5.1 Amplitudna modulacija

Amplitudno modulacijo uporabljamo na primer pri prenosu informacije o pritisnjeni tipki na daljinskem upravljalniku za televizijo, radio, tv-box… Ko pritisnemo neko tipko, elektronika v daljinskem upravljalniku to zazna in pridobi kodo tipke. Kodo tipke nato amplitudno moduliramo z nosilnim signalom in jo preko svetlobne komunikacije pošljemo do televizije. Zelo pogosto je za frekvenco nosilnega signala izbrana frekvenca 38 kHz, saj le-te ne najdemo v okolici. Za sprejemanje in demodulacijo teh signalov pa največkrat uporabimo elektronsko integrirano vezje kot je na primer TSOP32230[1].

5.1.1 NALOGA: Pošiljanje IR signala

1. Splošni opis naloge: Po IR komunikaciji pošljite vaše ime in priimek. S krmilnikom Arduino NANO generirajte UART signal, ki bo vseboval vaše podatke o vašem imenu in priimku.

[1] https://www.vishay.com/docs/82489/tsop322.pdf

Načrtujte tudi elektroniko za generirajte periodičnega nosilnega signala. Nato podatke amplitudno modulirajte in z IR diodo pošljite do sprejemnika. V poročilo dodajte shemo vezja in izpis programske kode.

Dekompozicija problema:

1. Preučite dokumentacijo o integriranem vezju TSOP32230 in zapišite bistvene lastnosti vezja v tbl. 5.1:

Tabela 5.1: Pomembnejše latsnoti IV TSOP32230.

Lastnost integr. vezja	vrednost	Enota
razporeditev nožic:		
frekvenca nosilnega signala:		
minimalni čas trajanja logične enice (tmin):		
valovno dolžino svetlobne pri max občutljivosti:		

2. Narišite časovno odvisnost izhodnega signala (U(t)) od prejete svetlobne informacije za integrirano vezje TSOP32230.
3. Iz t_{min} izračunajte hitrost UART komunikacije ($baud$):
4. Za oddajno diodo boste imeli na voljo diodo z oznako LD271-L. Iz dokumeta s podatki prepišite pomembnejše lastnosti in jih vpišite v tbl. 5.2:

Tabela 5.2: Lastnosti diode LD271.

Lastnosti LED LD271	Vrednost	Enota
valovno dolžino svetlobe, ki jo oddaja:		
kolensko napetost (Uk):		
nazivni tok (I0):		
kratkotrajen maksimalen tok (Imax):		

5. Z astabilnim-multi-vibratorjem generirajte nosilni signal.
6. Skonstruirajte vezje za amplitudno modulacijo, ki jo lahko realizirate z eno od digitalnih operacij (IN, ALI, NE-ALI, EKSKL.-ALI).

5.2 Frekvenčna modulacija

Čeprav modulacijo signalov uporabljamo predvsem v digitalni obdelavi podatkov za potrebe različnih komunikaciji, pa je modulacija razširjena tudi na druga področja. Pogosto na ta način moduliramo tudi odzive različnih senzorjev.

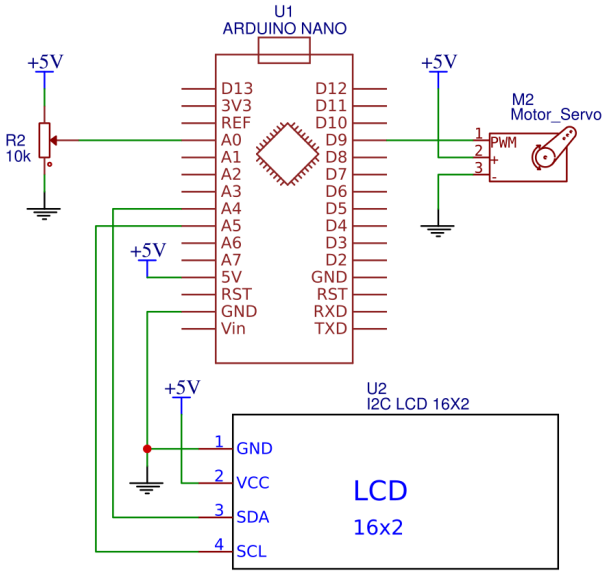
5.2.1 NALOGA: Frekvenčna modulacija fizikalnih količin

1. Sestavite astabilni-multivibrator s časovno konstanti, ki je odvisna od RC člena.
2. Na mesto upora R vstavite element, katerega upornost je odvisna od neke fizikalne količine (temperature, osvetljenosti …). Z osciloskopom posnemite časovno odvisnost napetosti (U(t)) pri različnih fizikalnih pogojih.
3. Astailni-multivibrator lahko sestavite s krmilnikom Arduino NANO in RC členom, ki nudi nekaj več dodatnih možnosti uporabe.

```
//   +--[R2k2]--+---||---+
//   |          | 100uF  |
// [A5]       [A7]     [GND]
#define PIN_C A7
#define CAPACITANCE_uF 100      // Kapaciteta kond. v uF
#define PIN_R A5
#define LED 13
#define SPR 12
#define LO_TRSH 400
#define HI_TRSH 700
unsigned long startTime;
void charge_cycle(){
  startTime=millis();
  digitalWrite(LED, 1);
  digitalWrite(SPR, 0);
  digitalWrite(PIN_R, 1);
  int C_ADC = analogRead(PIN_C);
  while (C_ADC < HI_TRSH) C_ADC = analogRead(PIN_C);
}
void discharge_cycle(){
  digitalWrite(LED, 0);
  digitalWrite(SPR, 1);
  digitalWrite(PIN_R,0);
  int C_ADC = analogRead(PIN_C);
  while (C_ADC > LO_TRSH ) C_ADC = analogRead(PIN_C);
}
void printTime(){        // Čas je prikazan v ms -> milisekundah
  unsigned long timePeriod = millis() - startTime;
  Serial.println(timePeriod);
}
void setup() {
  pinMode(PIN_C, INPUT);
  pinMode(PIN_R, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(SPR, OUTPUT);
  Serial.begin(115200);
}
void loop() {
  charge_cycle();
  discharge_cycle();
}
```

5.3 Pulzno-širinska modulacija

V uvodu v PWM modulacijo smo omenili krmiljenje servo motrjev. Ker se ti motorji zelo pogosto uporabljajo v modelarstvu je prav, da jih pobližje spoznamo. Oglejte si primer uporabe funkcije krmilnika Arduino NANO - servo.write().



Slika 5.2: Stikalna shema priključitve servo-motorja na krmilnik Arduino NANO.

5.3.1 NALOGA: Krmiljenje servo-motorja

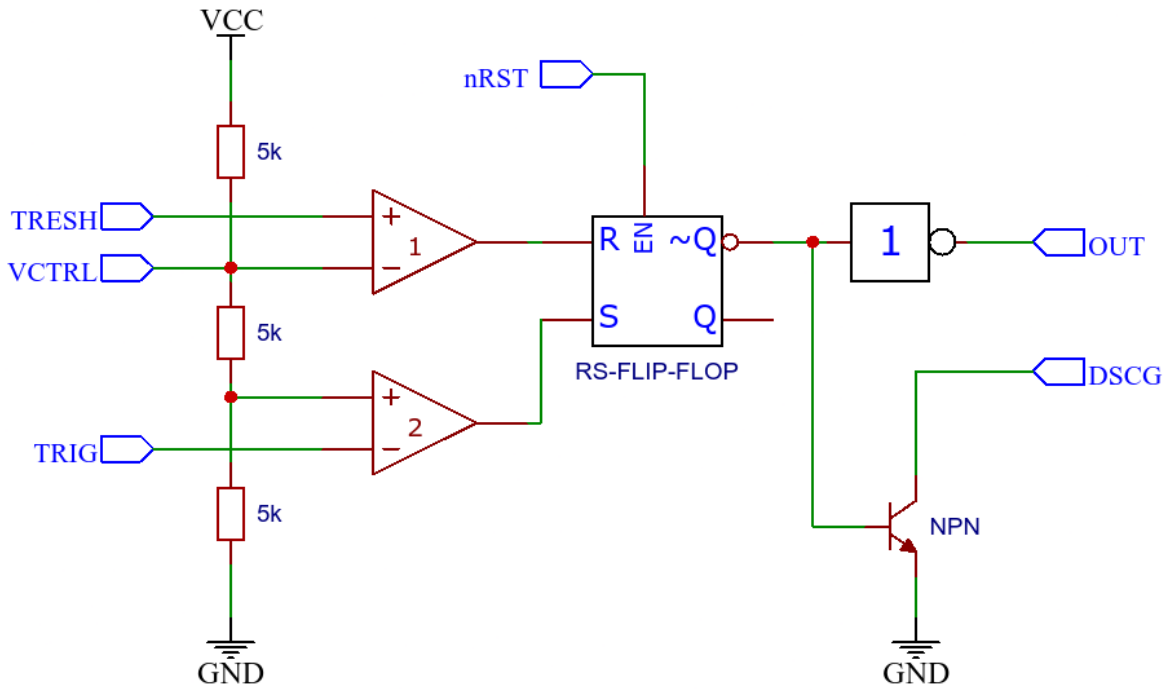
1. Na krmilnik priključite potenciometer s katerim boste lahko nastavljali poljubno napetost med 0 V in 5 V.
2. Nato priključite še servo-motor in ga krmilite s primernim programom tako, da bo motor spreminjal svojo orientacijo gredi glede na položaj potenciometra.
3. Z osciloskopom posnemite PWM signal za krmiljenje servo-motorja pri različnih položajih potenciometra.

6 INTEGRIRANO VEZJE 555

Tako imenovani “časovnik 555” je integrirano vezje (angl. Integrated circuit - IC), ki ga uporabljamo v različnih aplikacijah, kjer želimo generirati časovno odvisne napetostne pulze. To nam pride prav za izgradnjo časovnih zakasnitev, oscilatorjev itd.

Čeprav so ga na trg dali že leta 1972 (pred 45 leti), je zaradi nizke cene in enostavne uporabe še danes pogosto uporabljeno IC. Pravzaprav je IC 555 najbolj priljubljeno vezje nasploh. Pravjo, da še niso napisali knjige za elektroniko, v kateri nebi bilo projekta s tem vezjem, zato naj tudi ta skripta ne bo izjema.

6.1 Zgradba integriranega vezje 555



Slika 6.1: Blokovna shema integriranega vezja 555.

6.1.1 NALOGA: Blokovna zgradba integriranega vezja

Na spletnih straneh poiščite kako je zgrajeno vezje 555[a], prerišite shemo vezja in poskušajte razumeti njegovo delovanje. Odgovorite po čem je to vezje dobilo svoje ime.

[a] https://www.ti.com/lit/ds/symlink/ne555.pdf

6.2 Uporaba integriranega vezja 555

6.2.1 NALOGA: 555 kot RS-flip-flop

Integrirano vezje 555 uporabite kot R-S flip-flop. Na izhod integriganega vezja priključite svetlečo diodo, ki bo nakazovala stanje spominske celice. Narišite stikalno shemo vezja.

6.2.2 NALOGA: 555 kot Schmitov sprožilnik

Integrirano vezje 555 uporabite kot schmittov sprožilnik. Na vhod priklučite potenciometer, s katerim lahko poljubno izbirate potencial in hkrati opazujte izhodno stanje sprožilnika. Na izhod integriganega vezja priključite svetlečo diodo, ki bo nakazovala stanje schmittovega sprožilnika.

Odvisnost izhodnga potenciala napetosti od vhodnega - U2(U1) prikažite tudi na osciloskopu tako, da prikažete obe krivulji.

Narišite stikalno shemo vezja.

6.2.3 NALOGA: 555 kot Astabilni-multivibrator

Integrirano vezje 555 uporabite kot astabilni-multivibrator tako, da boste lahko nanj priključili svetlečo diodo, ki jo boste videli utripati. Narišite shemo vezja in preverite izhodni napetostni signal z osciloskopom.

6.2.4 NALOGA: 555 kot Monostabilni-multivibrator

Integrirano vezje 555 zvežite v način monostabilnega-multivibratorja tako, da ko boste s pritiskom na tipko sprožili en sam pulz, ki bo trajal približno 3 s.

Narišite stikalno shemo vezja.

6.2.5 NALOGA: 555 kot generator trikotniškega signala

Integrirano vezje 555 uporabite v vlogi generatorja periodičnega signala s trikotniško obliko. Generator lahko naredite tako, da najprej sestavite: - tokovni vir s PNP tranzistorjem, - ki naj polni kondenzator C1. - Naboj na kondenzatorju naj se izprazni, preko 7. priključka integriganega vezja 555, - ko napetost doseže 2/3 napajalne napetosti.

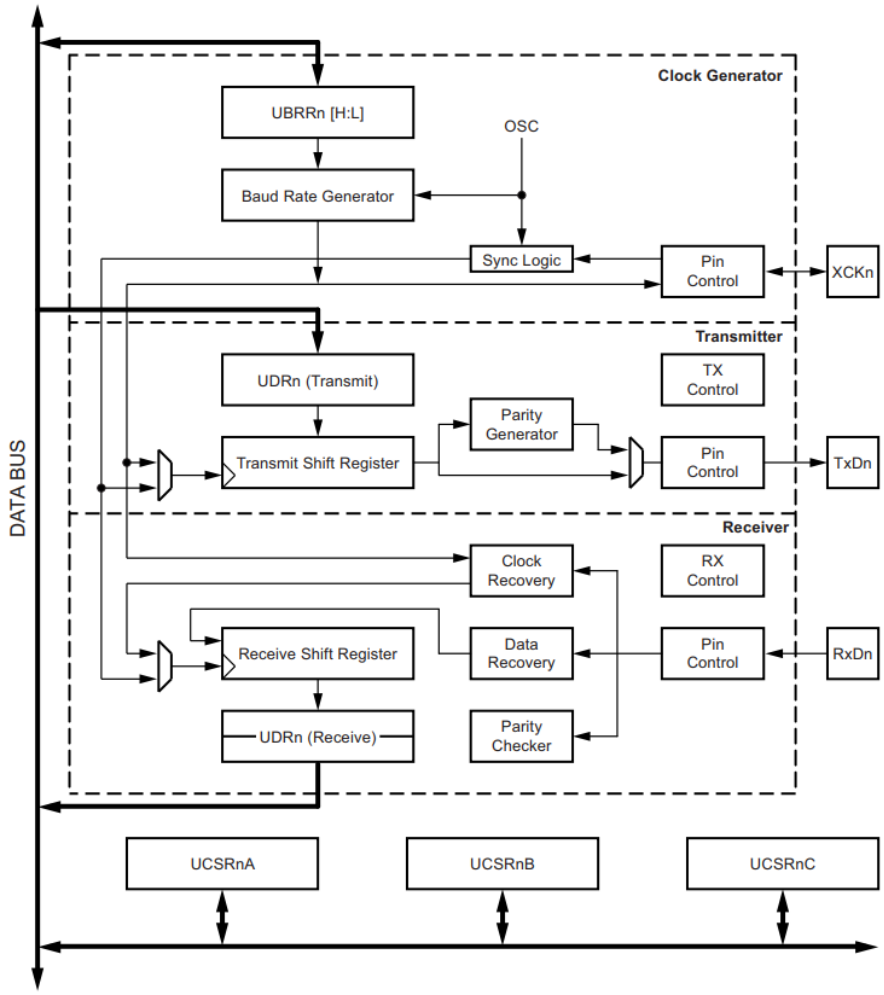
Narišite stikalno shemo in izhodni signal preverite z osciloskopom.

7 SEKVENČNA VEZJA

7.1 D-flip-flop

D-flip-flop (D-ff) je pogosto uporabljen predvsem v dveh elektronskih funkcijah: - kot nastavitneni/pomnilni register in - kot pomikalni register.

D-ff je v obeh funkcijah uporabljen v periferni enoti USART mikrokrmilnika ATmega328. Enota USART skrbi za serijsko komunikacijo UART, ki poteka po protokolu RS232. Blokovni prikaz oddajnega dela USART enote prikazuje sl. 7.1.



Slika 7.1: Blokovni prikaz sestava USART periferne enote krmilnika ATmega328.

Oddajni del (Transmitter) USART enote je v splošnem sestavljen iz pomnilnega registra UDRn v katerega shranimo podatek, ki ga želimo poslati. Ta podatek se nato premakne v pomični register `Transmit Shift Register` (TSR). Nazadnje se prične faza pomikanja podatka proti serijskemu izhodu pomičnega registra. Med to fazo lahko z novim podatkom nastavimo pomični register UDRn. A ponovno nastavljanje registra TSR med samim pomikanjem TSR registra ni mogoče.

7.1.1 NALOGA: Pomikalni register

V simulacijskem programu (SimulIDE) načrtujte: - pomnilni register UDRn in - pomični register TSR.

Oba registra naj boste zgrajena iz D-ff in naj vsebujeta 8 pomnilnih celic (8-bitni register). Krmiljenje napetosti prožitvenih signalov kot so `Enable` in `Clock` lahko zagotovite z napetostnimi viri (5V - on/off).

7.2 T-flip-flop

T-flip-flop (T-ff) je pogosto uporabljen v sekvenčnih vezjih kot: - dvojiški števnik in - kot delitelj osnovnega urinega takta.

Kot slednja funkcija (deljitelj osnovnega takta) je tudi različica sekvenčnega vezja uporabljena v periferni enoti USART (glej sl. 7.1 - `Clock Generator`). Glede na to, da lahko z zaporedno vezavo T-ff delimo osnovni takt le s koeficienti: 2, 4, 8, 16 … tudi razloži, zakaj so pri oddajanju podatkov po UART vodilu na voljo frekvence: 19200, 9600, 4800, 2400 …

7.2.1 NALOGA: Dvojiški števnik

V simulacijskem programu (SimulIDE) sestavite: - 4-bitni dvojiški števnik (kaskadno vezani T-ff), - izhodne signale (D3, D2, D1 in D0) priključite na vhod - integrirano vezje 74HC4511 (BCD -> 7 seg. LED) in izhode le teh - povežite na 7-segmentni LED prikazovljnik.

8 KRMILJENJE IN REGULACIJA

8.0.1 NALOGA: Krmiljenje moči DC motorja

Načrtujete in sestavite vezje za krmiljenje moči enosmernega motorja. Vezje izvedite z uporabo krmilnaka Arduino nano, ki ga opremite s potenciometrom za nastavljanje želene vrednosti. S funkcijo `analogWrite(pin, val)` krmilite izhod za krmiljenje motorja. Narišite stikalno shemo vezja in podajte celotno programsko kodo krmilnika.

8.0.2 NALOGA: Regulacija osvetljenosti

Načrtujete in sestavite vezje za regulacijo osvetljenosti. Podajte stikalno shemo vezja in vezje tudi sestavite. Če boste nalogo rešili z uporabi krmilne elektronike, predložite tudi program. # OPERACIJSKI OJAČEVALNIKI

8.1 LM358

- single supply
- low power consumtion
- low input offset 3-7mV
- common mode ground?
- dva op. amp. v enem ohišju
  - LM324 (enake karakteristike, le 4x op.amps)

8.2 TL071

- designed for dual supply voltages
- low total harmonic distortion 0.003% (audio aplications)
- very low input baise curent (1pA)

8.3 LMV358

- rail-to-rail
- low inout offset 1mA
- low input baise current (10pA)
- supply-voltage 2.5V - 5.5V
- dva op.amp v enem ohišju
  - LMV324 (anke karakteristike, le da so 4x)

8.4 LM741

- ? zakaj, ko pa je…

8.5 RC4558

- low noise

8.6 NE5532

- low noise
- in audio applications

8.7 OP071

- low noise
- low offset votage (60 uV)
- načrtovan za bi-polatno napajanje
- nekoliko večji vhodni tok (1nA)