USB diferencijalni voltmetar

Sičanica, Goran

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

USB diferencijalni voltmetar

Diplomski rad

Goran Sičanica

Osijek, 2016.

ETFOS
ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 30.06.2016.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Goran Sičanica
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 632 - R, 23.10.2013.
Mentor:	Doc.dr.sc. Davor Vinko
Sumentor:	
Predsjednik Povjerenstva:	Doc.dr.sc. Vanja Mandrić-Radivojević
Clan Povjerenstva:	Doc.dr.sc. Tomislav Matić
Naslov diplomskog rada:	USB diferencijalni voltmetar
Znanstvena grana rada:	**Elektronika (zn. polje elektrotehnika)**
Zadatak diplomskog rada:	Za informacije o temi javiti se mentoru na mejl: davor.vinko@etfos.hr
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 3 Jasnoća pismenog izražavanja: 2 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	30.06.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

ETFOS
ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

IZJAVA O ORIGINALNOSTI RADA

Osijek, 07.07.2016.

Ime i prezime studenta:	Goran Sičanica
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 632 - R, 23.10.2013.
Ephorus podudaranje [%]:	11

Ovom izjavom izjavljujem da je rad pod nazivom**: USB diferencijalni voltmetar**

izrađen pod vodstvom mentora Doc.dr.sc. Davor Vinko

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Potreba za instrumentima koji mjere viskom točnošću tj. osjete jako male promjene napona je jako velika. Primjenjuju se u različitim poljima kao što je senzorika, audio tehnika, mjerna tehnika, komunikacijska tehnika i niz drugih polja. Diferencijalna tehnika mjerenja napona je jedna od najkorištenih i najtočnih metoda mjerenja napona. U toj tehnici voltmetar se koristi razlikom između poznatoga napona (refentnoga napona) i nepoznatoga napona, kako bi ustanovio vrijednost nepoznatoga napona. Analogni instumenti koji mjere napon su stvar prošlosti i jako se rijetko više koriste. Računalo koje radi u digitalnom svijetu ne može primiti analogni signal kao takav. Već ga se mora prilagoditi pomoću analogno-digitalnih pretvornika. Pretvornik LTC2400 je analogno-digitalni pretvornik koji mjeri napon pomoću diferencijalne tehnike i analogni signal pretvara u digitalni pomoću delta sigma modulacije. Pretvornici s delta sigma modulacijom uveliko su zamijenili pretvornike sa sukcesivnom aproksimacijom koji su bili najrašireniji do početka primjene delta sigma pretvornika. Pretvornici sa sukcesivnom aproksimacijom su ograničeni do 18 bitne rezolucije, a pretvornici koji rade s delta sigma modulacijom postižu rezoluciju od 18 do 24 bita. Osim prednosti u većoj rezoluciji ovi pretvornici su se nametnuli i povoljnijom cijenom.

1.1. Zadatak i struktura rada

Ovaj rad ima za cilj dizajnirati i napraviti sklop koji učitava naponski analogni signal u rasponu od 0 do 5 V i pretvoriti ga u digitalni oblik te prikazati rezultat, zatim analizirati točnost dobivenoga sklopa. Sklop se sastoji dvije glavne komponente. Prva je analogno-digitalni pretvornik sa rezolucijom od 24 bita koji radi pomoću delta sigma modulacije. Riječ je o proizvodu LTC2400 tvrtke Linear Tehnology. Druga je mikroupravljač Atmega16 iz obitelji AVR tvrtke Atmel. Atmega16 prima 32 bitni binarni niz od LTC2400, i izdvaja 24 bitni dio koji predstavlja mjerenje analognog signala. Taj 24 bitni niz se obrađuje kako bi se dobio iznos ulaznog analognog signala. Nakon toga mikroupravljač taj iznos putem UART sučelja šalje na USB sučelje računala pomoću USB/serial adaptera, ili na LCD display koji se može priključiti na sklop.

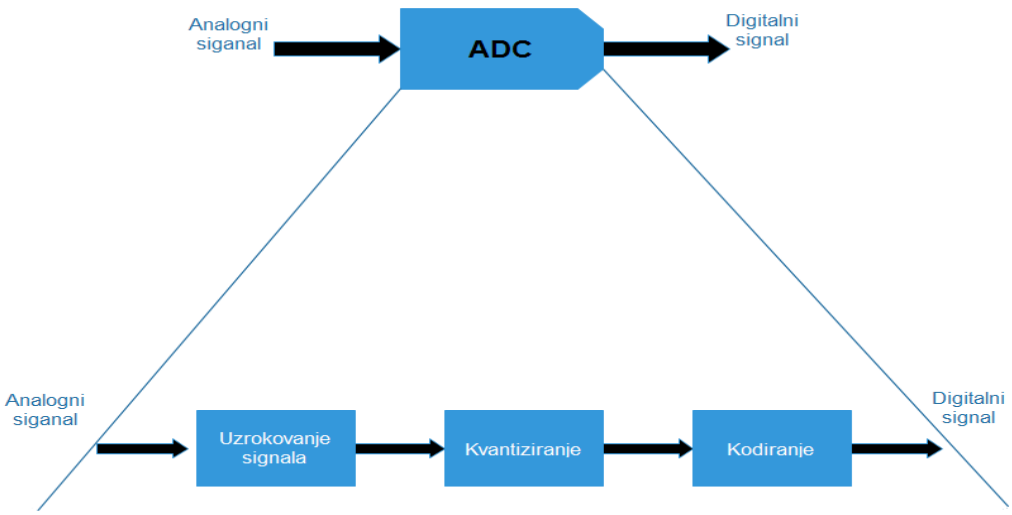
Struktura rada podijeljena na šest cjelina. Nakon uvoda, druga cjelina nosi naziv Stanje tehnike i govori o analogno-digitalnim pretvornicima, opisuje njihov princip rada i daje nam uvid u pretvornike s delta sigma modulacijom. Treća cjelina govori o dizajniranju i izradi sklopa, od protoboarda do krajne PCB pločice. Četvrta cjelina opisuje kod koji je implementiran na Atmega16 upravljača koji upravlja cijelim procesom. Peta cjelina prikazuje rezultate mjerenja i analizira ih. Zaključak i korištena literatura se nalazi u zadnjoj šestoj cjelini.

2. STANJE TEHNIKE

2.1. Uloga i način rada analogno-digitalnih pretvornika

Signali u stvarnom svijetu su isključivo analogni (npr. zvuk, svjetlo, sila, tlak, temperatura...). Analogni signal je kontinuiran u vremenu tj. neprekidan, dok digitalni signal nije stalan u vremenu i prikazuje se pomoću binarnih stanja nula i jedinica. Kako bi se analogni signal iz stvarnog svijeta u analogno-digitalnom pretvorniku mogao prepoznati tj. mjeriti njegov iznos, potrebno ga je prilagoditi „digitalnom svijetu“ koji se temelji na binarnim brojevima tj. znamenka nula i znamenka jedan, koje predstavljaju dva stanja. Analogno-digitalni pretvornik (skrać. ADC ili A/D) pretvara analogni signal u digitalni signal. Kako bi se pretvorba mogla izvršiti treba definirati frekvenciju uzrokovanja analognog signala (engl. *sample rate).* Nijedan pretvornik ne može izvršiti pretvorbu trenutno. Potrebno vrijeme za izvšenje pretvorbe zove se vrijeme pretvorbe (engl. *conversation time*) .

Pretvorba signala se vrši u tri osnovna koraka, kao što je prikazano na slici 2.1. Prvi korak je uzrokovanje signala, signal se diskretizira po vremenu frekvencijom $f_s = \frac{1}{T_s}$. Drugi korak je kvantiziranje signala, signal se diskretizira po amplitudi, čime se dobije konačan broj diskretnih stanja. Zadnji korak je kodiranje, što znači da se pojedinim diskretnim vrijednostima pridružuje digitalni kod.

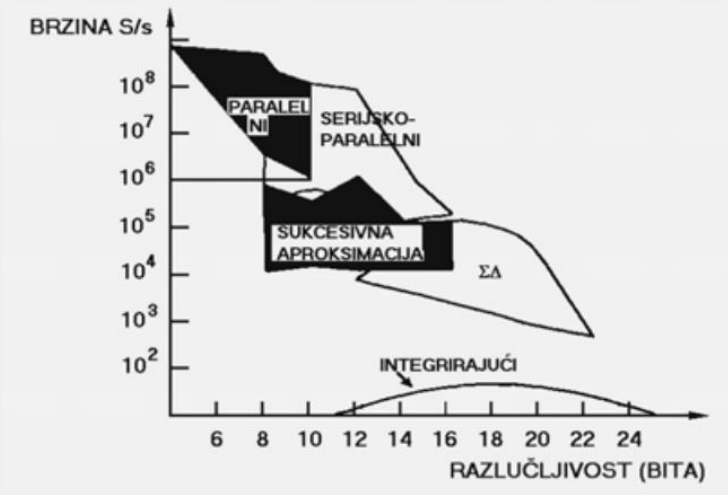


Slika 2.1. Koraci ADC pretvorbe

2.2. Vrste analogno-digitalnih pretvornika

Prema načinu rada razlikujemo sljedeće pretvornike: pretvornici sa sukcesivnom aproksimacijom, integracijski pretvornici, paralelni (engl. *flash*) pretvornici i cjevovodni (engl. *pipeline*) pretvornici. Na slici 2.2. prikazane su pojedine vrste pretvornika i njihova ograničenja u smislu brzine obrade i rezolucije.

Pretvornici s paralelnim načinom rada su najbrži po odzivu, ali zato im je rezolucija tj. preciznost najniža. Pretvornici koji rade na principu sigma delta modulacije imaju jako veliku rezoluciju, međutim brzina im je niža od paralelnih i serijskih pretvornika.



Slika 2.2. Brzina prevorbe i rezolucija pojedinih vrsta pretvornika [1]

ADC pretvornike možemo podijeliti i prema fekvenciji uzrokovanja signala. Na Nyquistove pretvornike i pretvornike s naduzrokovanjem.

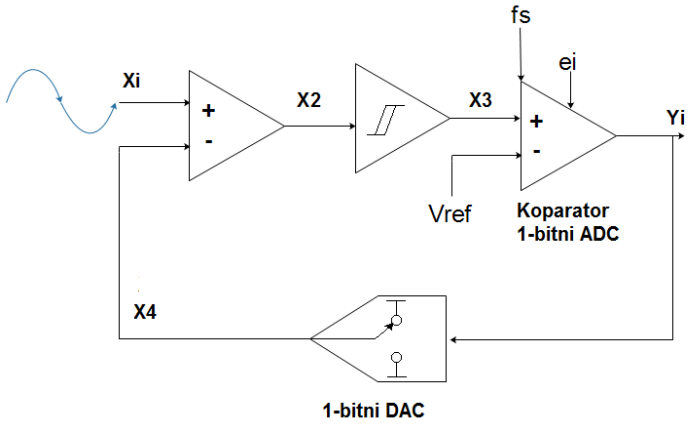
Nyquistovi pretvornici uzrokuju signal frekvencijom koja je najmanje dva puta veća od najveće frekvencije cjelokupnog spektra ulaznog analognog signala. Pretvornici s naduzrokovanjem uzrokuju signal sa N puta višom frekvencijom $F_S = N * f_s$ od najveće frekvencije ulaznog analognog signala.

Delta sigma pretvornici pripadaju grupi pretvornika koji koriste naduzrokovanje.

2.3. Način rada pretvornika s delta sigma modulacijom

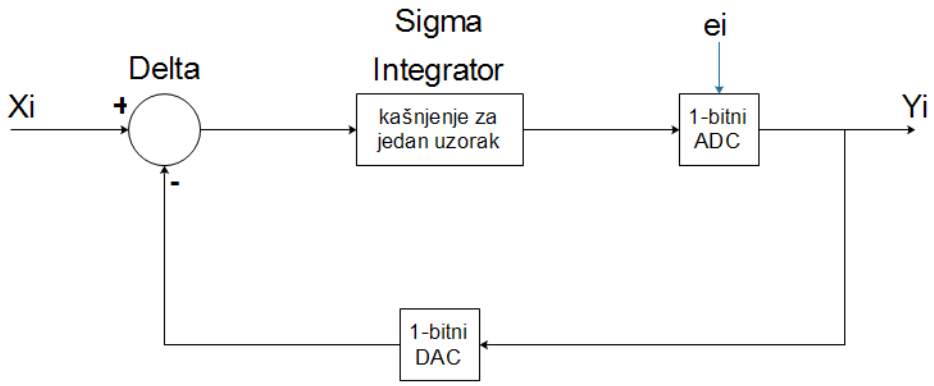
Kod sigma delta pretvornika , za razliku od konvencionalnih pretvornika, ne kvantizira se amplituda uzrokovanog signala, već se kvantizira promjena amplitude između dva uzroka što predstavlja diferencijalnu kvantizacijsku tehniku. Taj pristup ima smisla kod prirodnih signala iz razloga što postoji veza između dva susjedna uzroka. Najprije se kodira analogni signal koristeći visoku frekvenciju delta sigma modulacije, zatim se signal provlači kroz digitalni filter s decimacijom. Pomoću delta sigma modulacije koja koristi visoku frekvenciju uzrokovanja postiže se promjenjen izgled spektralne gustoće šuma. Na taj način, snaga šuma se potiskuje iz promatranog spektralnog područja u kojem se nalazi ulazni signal u područje viših frekvencija koje se zatim otklanjaju pomoću digitalnog filtra.

Na slici 2.3. prikazan je delta sigma modulator prvoga reda pomoću operacijskih pojačala. Diferencijalno operacijsko pojačalo oduzima ulazni signal od signala dobivenog povratnom vezom, a integrator akumulira razliku. Komperator predstavlja jedno bitni ADC gdje se pojavljuje kvantizacijski šum. U ADC ulazi akumulacijska razlika ulaznog signala i povratne veze, što znači da se kvantizira promjena amplitude, a ne sama amplituda uzrokovanog ulaznoga signala.



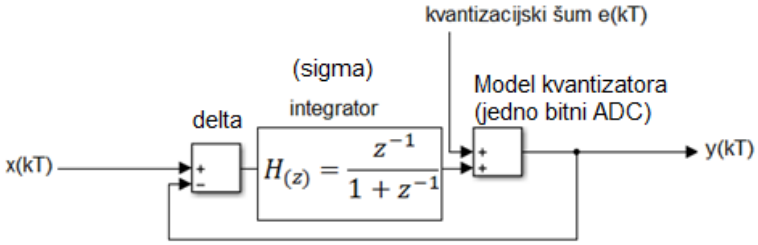
Slika 2.3. Delta sigma modulator prvoga reda

To možemo modelirati kao što je prikazano na slici 2.4. delta predstavlja razliku ulaznog signala i signala iz povratne veze, sigma je integrator koji kasni za jedan uzorak i jedno bitni ADC gdje se pojavljuje kvantizacijski šum. Kako bi se digitalni signal s izlaza mogao uspoređivati s ulaznim analognim signalom, u povratnoj vezi se nalazi jedno bitni digitalno-analogni pretvornik.



Slika 2.4. Model delta sigma pretvornika

Modulator se sastoji od sumatora koji predstavlja delta dio i integratora koji predstavalja sigma dio. Jedno bitni ADC možemo prikazati kao sumator s dodatkom šuma uslijed kvantizacije. Radi jednostavnije analize prijenosne funkcije su postavljene u Z domeni što je prikazano na slici 2.5.



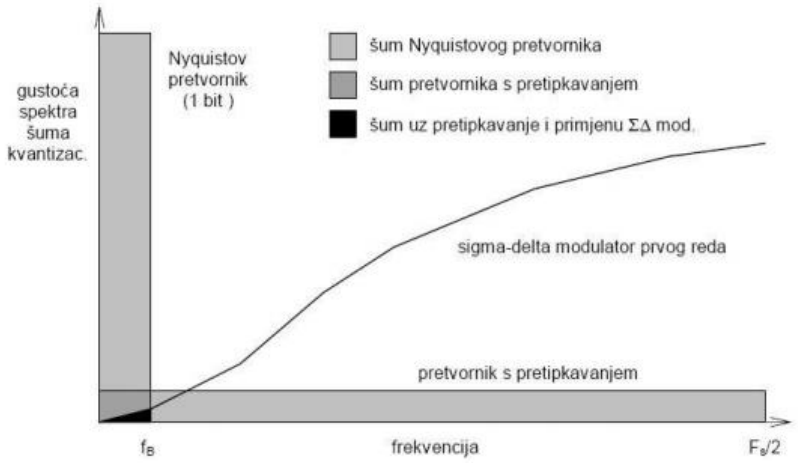
Slika 2.5. Model delta sigma pretvornika u Z domeni

Prijenosna funkcija signala (*STF*) formula 2-1 se ponaša kao niskopropusni filtar. Signal se nalazi na malim frekvencijama, kako je frekvencija pretipkavanja velika tako da prolazi direktno na izlaz što je vidljivo iz prijenosne funkcije signala. Prijenosna funkcija šuma (*NTF*) formula (2-2) se ponaša kao visokopropusni filtar.

$$STF = \frac{Y_{(z)}}{X_{(z)}} = \frac{H_{(z)}}{1+H_{(z)}} = \frac{\frac{z^{-1}}{1+z^{-1}}}{1+\frac{z^{-1}}{1+z^{-1}}} = \quad z^{-1} \tag{2-1}$$

$$NTF = \frac{Y_{(z)}}{E_{(z)}} = \frac{1}{1+H_{(z)}} = \frac{1}{1+\frac{z^{-1}}{1+z^{-1}}} = \quad 1 - z^{-1} \tag{2-2}$$

Kvantizacijski šum koji inače djeluje konstatno za vrijeme pretvorbe prolazi kroz visokopropusni filtar tako da se ne pojavljuje na niskim frekvencijama. Takvom prijenosnom funkcijom se postiže pomak šuma kvantizacije u pojas visokih frekvencija. S time se šum kvantizacije u pojasu signala značajno smanjuje. Šum koji je potisnut u spektar viših frekvencija će se naknadno ukloniti digitalnom filtracijom. Kao što je prikzano na slici 2.6. šum se kod pretvornika s pretipkavanjem i delta sigma modulacijom potiskuje u pojas visokih frekvencija.

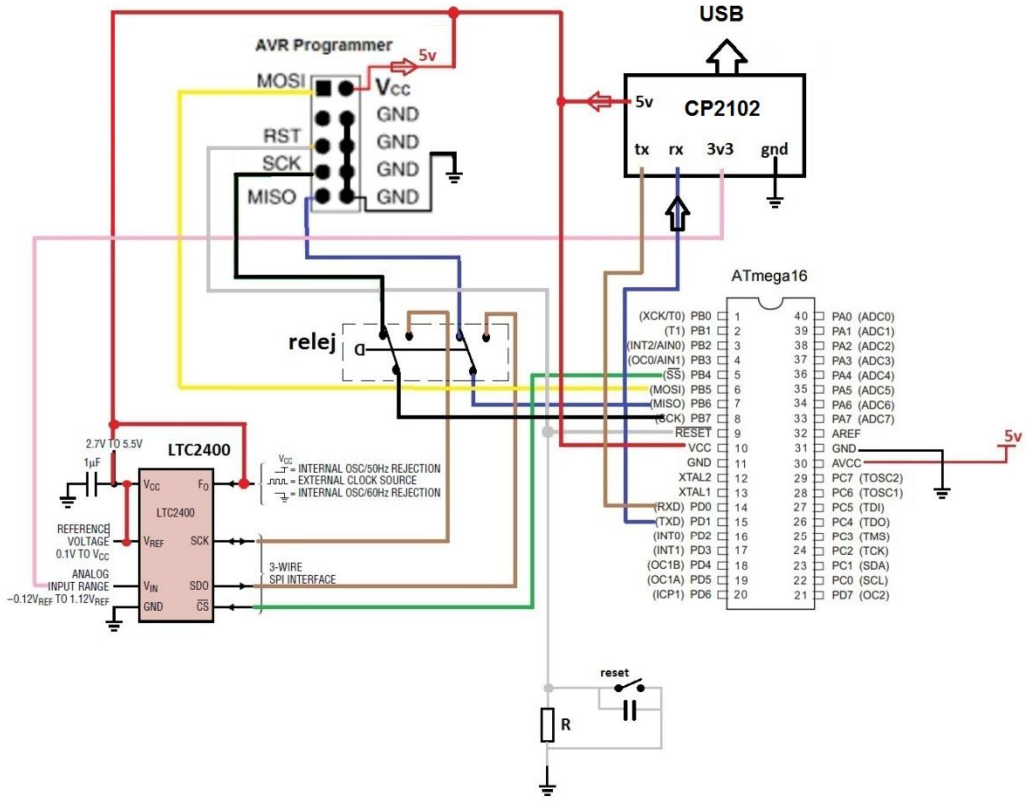


Slika 2.6. Uobličenje spektra šuma kvantizacije sigma-delta modulacijom [2]

3. PLANIRANJE I RAZVOJ SKLOPOVLJA

3.1. Sklop na razvojnoj pločici

Komponente ovoga prototipa su povezane preko razvojne pločice (engl. *breadboard*) koja omogućava jednostavno i brzo povezivanje komponenti. Testni prototip je sastavljen od pet komponenti. To su LTC2400, mikroupravljač, AVR programator, releja i USB/UART mosta. Na slici 3.1. je prikazano, kako su te komponente međusobno povezane. Radi se o konceptualnoj shemi, koja prikazuje na koji su način komponente povezane kao i točan položaj pojedinih pinova svake komponente.

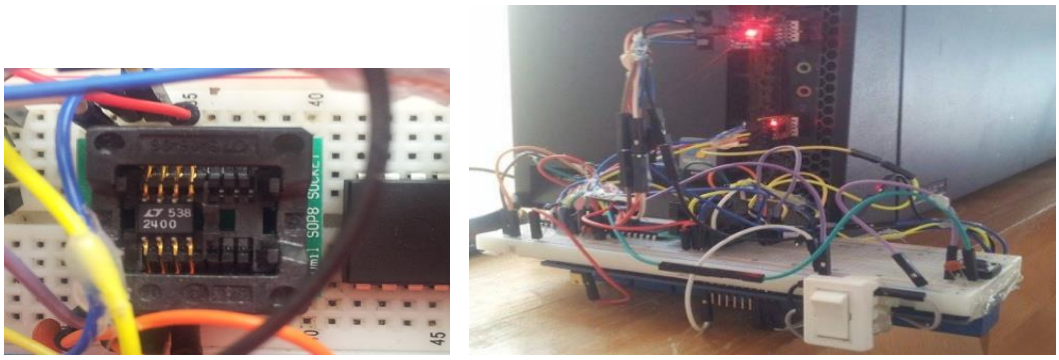


Slika 3.1. Shema prototipa

Na slici 3.2. prikazan je prototip. Kako se LTC2400 proizvodi jedino u S8 pakiranju potrebno je njegove pinove prilagoditi razvojnoj pločici. Kao što se vidi na slici 3.2.a ADC-a

pretvornik leži u adapteru. Prototip ima dva releja koji se uključuju na bijeli prekidač vidljiv na slici 3.2.b sklop na protoboardu.

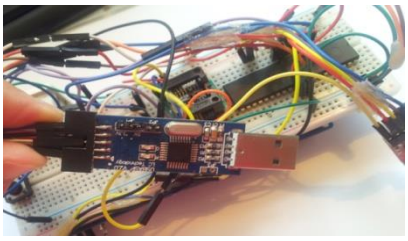
Zadatak releja kada se uključe jest da pinove od mikroupravljača koji su išli na pretvornik prespoje na avr programator. Radi se o pinovima *SCK* i *MISO* koje također koristi i pretvornik. Releji olakšavaju programiranje i ponovno puštanje sklopa u rad, tako što se može brzo provjeriti ispravnost koda. Preko USB/UART mosta podaci se šalju iz mikroupravljača putem UART komunikacije na USB port računala.

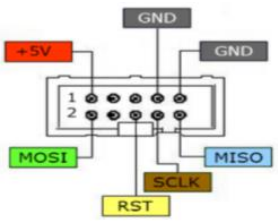


a) LTC2400 u adapteru b) Sklop na protoboardu

Slika 3.2. Slika prototipa

USBasp adapter slika 3.3. je open source i open hardware avr programator, koji nije primjenjiv u industriji kao npr. AVRISP. USBasp je jako povoljan programator i može se kupiti i za 3 dolara. Glavni problem osim što nemaju nekakve opcije poput podupiranja svih ISP protokola kao AVRISP i to što dolazi sa starim firmeware-om. Tako da se nije mogao povezati s Atmega16 mikroupravljačem. Kao što je bio slučaj kod mene. Postavio sam noviji firmware za što je bio potreban dodatni programator. Na sljedećoj slici prikazani su pinovi od USBasp adaptera.

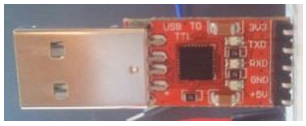




a) USBasp b) Pinovi USBasp-a

Slika 3.3. Slika USBasp

USB/UART MOST (CP2102) slika 3.4. služi kako bi slali s UART na USB podatke prikazan je na slici, korišten je samo pin za primanje podataka *RXD* kako nije bilo slanja podataka od računala prema mikroupravljaču. A Njegov pin 3V3 s naponom 3.3 V je koristio za provjeru rada pretvornika.



Slika 3.4. CP2102 adapter

3.2. Analogno-digitalni pretvornik LTC2400

LTC2400 je niskonaponski delta sigma ADC s trožičnim serijskim sučeljem i ima osam pinova. Dolazi u S8 pakiranju i to u dva oblika LTC2400C i LTC2400l. LTC2400C je namjenjen za rad pri temperaturnom opsegu od 0 °C do 70 °C, dok LTC2400l ima veći temperaturni opseg između -45 °C do 85 °C. Pri lemljenju sklopa treba paziti da temperatura ne prelazi 300 °C u vremenu od 10 sekundi, kako ne bi došlo do oštećenja integriranih krugova unutar sklopa.

Glavna odlika pretvornika uz 24 bitnu rezoluciju jest niska cijena u odnosu na druge pretvornike. Rezolucija od 24 bitna ukazuje na kvantizacijski korak tj. najmanji napon koji pretvornik registrira. I za 24 bitni pretvornik od 0 do 5 V dobiva se na slijedeći način.

$$\frac{Raspon\ mjerenoga\ napona}{2^N} = \frac{5}{2^{24}} = 0{,}000000298 \qquad (3\text{-}1)$$

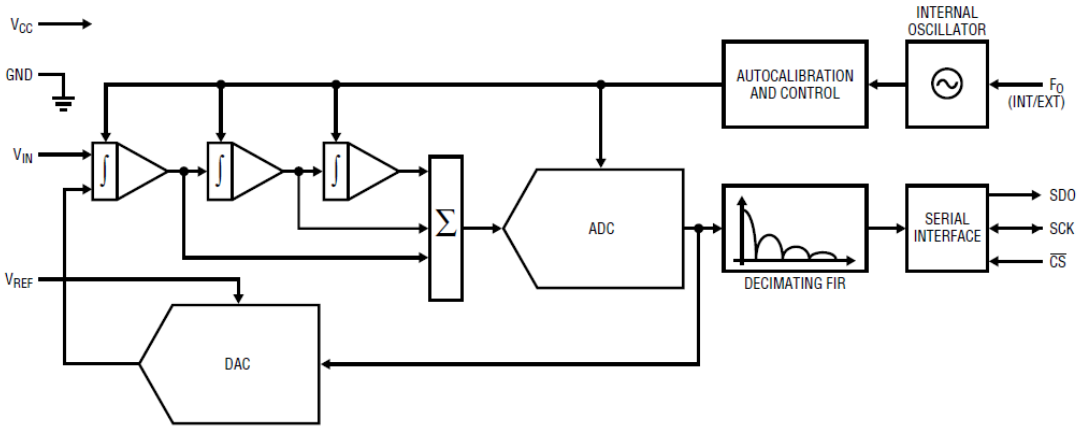
Pretvornik osjeti promjenu napona od 298nV kao što je izračunato formulom 3-1. Ovi podaci su vezani za idealan 24 bitni pretvornik, kakav naravno ne postoji. Proizvođač u

podatkovnom listu navodi performanse pretvornika koje su detaljno prikazane i analizirane u ovome poglavlju. Sklop dobiven ovim radom će znatno odstupati od rezultata iz podatkovnog lista s obzirom na nelinearnosti primjenjenih komponenti, nelinearnost referentnoga ulaznog signala, elektrostatičnih smetnjih i elektromagnetskih smetnji koje djeluju na sklop. Pretvornik pomoću referentoga napona koji je priključen na pinu V_{REF} dobiva vrijednost ulaznoga naponskog signala.

Kao i svaki analogno-digitalni pretovornik i LTC2400 se može primjeniti u raznim rješenjima. Neka rješenja u kojima se primjenuje zbog svoje visoke točnosti mjerenja su: mjerenje težine, direktno mjerenje temperature, analizator plinova, digitalni voltmetar, akvizicija podataka, upravljanje u industrijskim procesima...

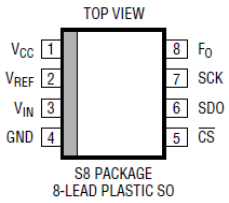
3.2.1. Blokovska shema pretvornika i opis pinova

LTC2400 osim digitalnog filtera integiranog na samom čipu ima i oscilator visoke točnosti. To eliminira potrebu za vanjskim jedinicama kao što su kristali ili oscilatori. Pretvornik šalje podatke frekvencijom 50 ili 60 Hz ±2% s 110 dB do 130 dB. Za svaki pretvorbeni ciklus obavlja kalibraciju odstupanja od pune skale. Kalibracija nema efekta na pretvorbeni ciklus. Prednost kontinuirane kalibracije je osiguranje visoke stabilnosti s obzirom na pogrešku pomaka, promjene napona napajanja i promjene temperature. Na slici 3.5. nalazi se blokovski prikaz pretvornika, vidimo da se radi o modulatoru trećeg reda (tri integratora). Osim izlaznoga signala u DAC (digitalno-analogni pretvornik) koji se nalazi u povratnoj vezi ulazi i refentni napon.



Slika 3.5. Blokovska shema pretvornika [3]

Položaj pinova je prikazan slikom 3.6. Pretvornik se napaja na *Vcc* pinu koji je prvi pin s lijeve strane. V_{REF} pin je ulazni pin i na njega se priključuje referentni napon sklopa. Ulazni napon koji se mjeri priključuje se na V_{IN} pin. Zadnji pin na lijevoj strani čipa je *GND* gdje se priključuje uzemljenje. Prvi pin s desne strane je *Fo* koji je dvosmjeran pin i predstavlja frekvencijski kontrolni pin. Ispod njega je pin *SCK* koji je dvosmjerni pin te na njega priključujemo serijski brojač. Sljedeći pin je *SDO* koji je izlazni pin, preko kojeg izlaze digitalni podaci dobiveni mjerenjem ulaznog signala. Zadnji pin na desnoj strani čipa je *CS* pin preko kojega se upravlja operacijskim stanjem pretvornika.



Slika 3.6. Položaj pinova pretvornika [3]

Na *Vcc* pin se priključuje napon od 2.7 do 5.5 V, također je preporučeno spojiti taj pin sa tantalum kondezatorom 10 μF i paralelno s njime keramički kondezator 0.1 μF koji su s druge strane uzemljeni. Referentni naponski signal treba biti u rasponu od 0.1 do *Vcc*. Raspon ulaznog analognog pina je od -0.125•V_{REF} do 0.125•V_{REF} za V_{REF}>2.5V. A maksimalan raspon ulaznog pina je od 0.3 V do *Vcc* +0.3 V. Tako, ako je referentni napon 5 V možemo priključiti na ulazni pin napon od -0.625 do 5.625 V. Ako *Vcc* padne ispod 2.2 V pretvornik se automatski uvodi u interno reset stanje. Ova odlika garantira integritet rezultata pretvorbe. Ako je *Vcc* iznad kritičnog praga, pretvornik napravi interni power-on-reset (*POR*) signal u trajanju 0.5 ms. *POR* signal očisti sve interne registre. Nakon *POR* signala pretvornik počinje novi pretvorbeni ciklus i prati normalni slijed stanja opisan na slici 3.7. Preporučeni raspon za referentni napon je od 100 mV do Vcc.

Pretvornik ima interni oscilator koji može raditi na frekvenciji slanja rezultata od 50 ili 60 Hz. Ako želimo da pretvornik šalje rezultate frekvencijom od 50 Hz tada pin *Fo* priključujemo na *Vcc* napon, a ukoliko želimo da je ta frekvencija 60 Hz priključujemo pin na uzemljenje. Ukoliko se koristi vanjski oscilator s frekvencijom f_{EOSC} preko formule $f_{EOSC}/2560$ dobije se frekvencija slanja.

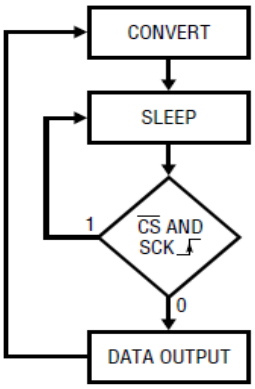
SCK (engl. *Serial Clock*) pin je dvosmjeran pin. Kada se koristi interni mod, tada je *SCK* pin izlazni pin za serijsko sučelje za vrijeme izlaza digitalnih podataka na *SDO* pinu iz pretvornika. Kada se koristi vanjski serijski brojač, *SCK* pin je ulazni pin za vanjsko serijsko sučelje. U internom modu se automatski aktivira interni pull up.

CS (engl. *Chip Select*) pin je digitalni ulaz koji je aktivan na nisku naponsku razinu, omogućava slanje podataka sa *SDO* pina i budi pretvornik iz stanja mirovanja. Nakon svake pretvorbe pretvornik se vraća u stanje mirovanja (sleep mode) i ostaje u njemu sve dok je *CS* pin na visokoj naponskoj razini. U slučaju promjene logičke razine s nule na jedan *CS* pina za vrijeme izlaza podataka iz *SDO* pina taj će prijenos prekinuti. Pretvornik će započeti novu pretvorbu i rezultat držati spremljen dok *CS* pin opet ne dođe u stanje logičke nule.

SDO pin može biti u tri stanja digitalnog izlaza. Za vrijeme slanja podataka koristi se kao serijski podatkovni izlaz. Kada je *CS* u stanju logičke jedinice (*CS = Vcc*) *SDO* pin je u stanju visoke impedancije. Za vrijeme pretvorbe i stanja mirovanja može koristiti za prikaz statusa pretvorbe. Status pretvorbe se može promatrati s povlačenjem *CS* u stanje logičke nule.

3.2.2. Pretvorbeni ciklus

Pretvornik operacijski krug slika 3.7. počinje s pretvorbom zatim ide u stanje mirovanja. U stanju mirovanja ostaje sve dok je na *CS* pinu logička jedinica. Rezultat pretvorbe ostaje u tzv. posmačnom registru i zbog toga je pretvornik u stanju mirovanja. Kada na *CS* pin dođe logička nula, pretvornik počinje slati rezultat pretvorbe. Nema kašnjena rezultata pretvorbe. Izlazni podatak odgovara pretvorbi koja je tek izvedena. Rezultat pretvorbe šalje se vani na *SDO* pin pod kontrolom takta serijskog brojača koji je priključen na pinu *SCK*. Slanje podataka je završeno kada je 32 bitni niz poslan na *SDO* pin ili ako je *CS* pinu došla logička jedinica. U oba slučaja pretvornik pokreće novu pretvorbu i cijeli se ciklus ponavlja.



Slika 3.7. Pretvorbeni ciklus pretvornika [3]

3.2.3. Perfomanse pretvornika

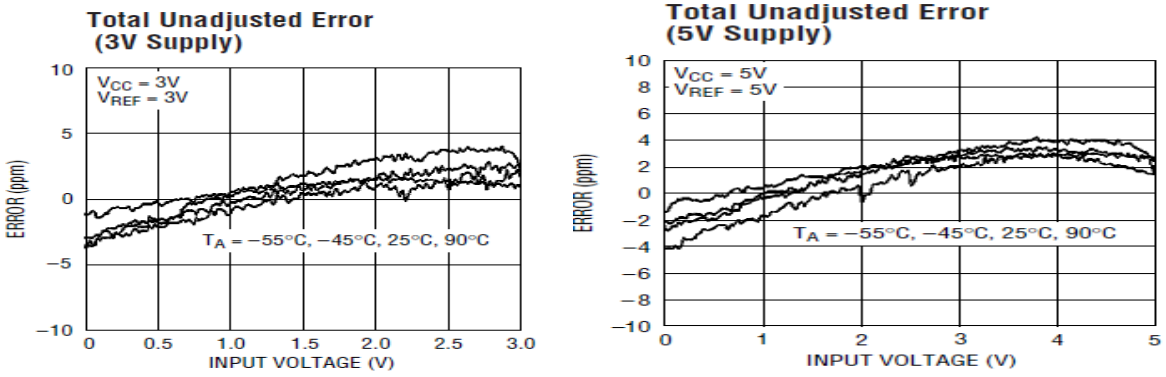
U tablici 3.1 koju daje proizvođač u podatkovnom listu dane su bitne karakteristike pretvornika. Točke u tablici označavaju karakteristike za cijeli operacijski raspon temeperature, polja bez točkih su karatkeristike za temperaturu $T_A = 25\ °C$.

Rezolucija uređaja bez nedostatka koda pri uvjetima od $0.1 \leq V_{REF} \leq V_{CC}$ je 24 bita, ta vrijednost je garantirana dizajnom, a ne subjektom tesitiranja navodi se kao napomena. Integrirana nelinearnost pretvornika za napon od 5V je uobičajno 4 do 15 ppm od V_{REF}-a. Pogreška pomaka pri uvjetu $2.5 \leq V_{REF} \leq V_{CC}$ je uobičajno 0.5 do 15 ppm od V_{REF}-a. Pogreška pri punom opsegu pri uvjetu $0.1 \leq V_{REF} \leq V_{CC}$ je do 2 ppm od V_{REF}-a. U normalnom modu šalje pri frekvenciji od 60 Hz $\pm 2\%$ je od 110 do 130 dB.

Tablica 3.1. Perfomanse pretvornika [3]

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Resolution (No Missing Codes)	$0.1V \le V_{REF} \le V_{CC}$, (Note 5)	●	24			Bits
Integral Nonlinearity	V_{REF} = 2.5V (Note 6) V_{REF} = 5V (Note 6)	● ●		2 4	10 15	ppm of V_{REF} ppm of V_{REF}
Offset Error	$2.5V \le V_{REF} \le V_{CC}$	●		0.5	2	ppm of V_{REF}
Offset Error Drift	$2.5V \le V_{REF} \le V_{CC}$			0.01		ppm of V_{REF}/°C
Full-Scale Error	$2.5V \le V_{REF} \le V_{CC}$	●		4	10	ppm of V_{REF}
Full-Scale Error Drift	$2.5V \le V_{REF} \le V_{CC}$			0.02		ppm of V_{REF}/°C
Total Unadjusted Error	V_{REF} = 2.5V V_{REF} = 5V			5 10		ppm of V_{REF} ppm of V_{REF}
Output Noise	V_{IN} = 0V (Note 13)			1.5		μV_{RMS}
Normal Mode Rejection 60Hz ±2%	(Note 7)	●	110	130		dB
Normal Mode Rejection 50Hz ±2%	(Note 8)	●	110	130		dB
Power Supply Rejection, DC	V_{REF} = 2.5V, V_{IN} = 0V			100		dB
Power Supply Rejection, 60Hz ±2%	V_{REF} = 2.5V, V_{IN} = 0V, (Notes 7, 15)			110		dB
Power Supply Rejection, 50Hz ±2%	V_{REF} = 2.5V, V_{IN} = 0V, (Notes 8, 15)			110		dB

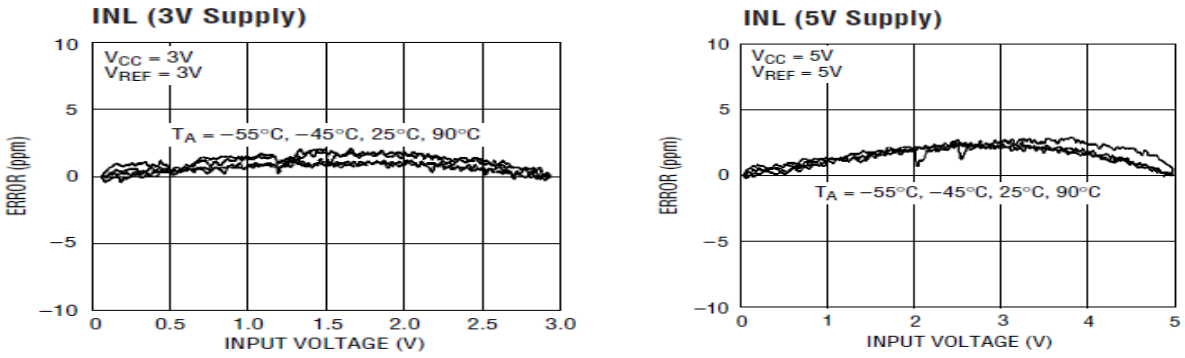
Na slici 3.8. prikazana su dva dijagrama koji predstavljaju ukupnu grešku. Na slici 3.8.a napajanje i referentni napon iznosi 3 V, a na slici 3.8.b iznosi 5 V. Primjećujemo da kod oba napona napajanja greška je najmanja pri polovici ulaznoga napona, također se primjećuje da pri radu na većim temperaturama greška raste do ± 4 ppm.



a) Napajanje 3 V b) Napajanje 5 V

Slika 3.8. Ukupna greška [3]

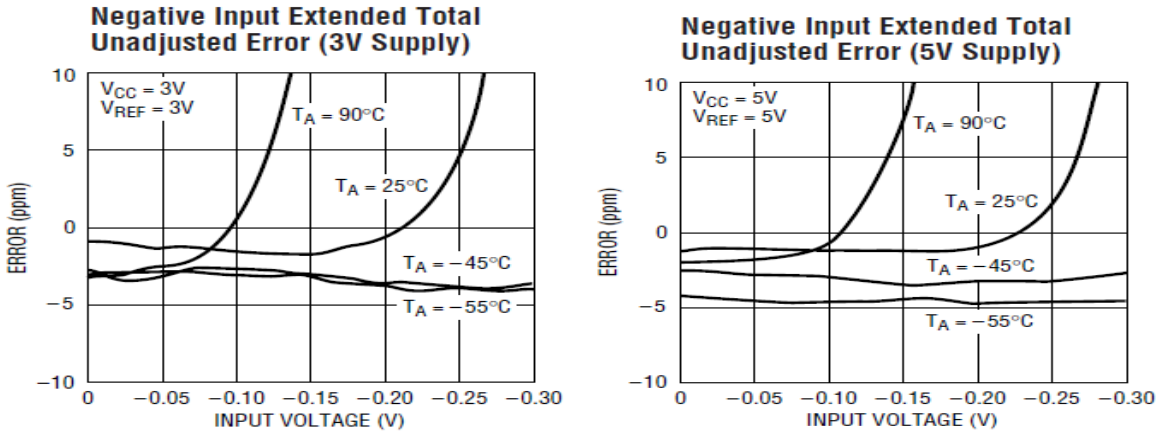
Na slici 3.9. također su prikazana dva dijagrama koji prikazuju integriranu nelinearnu (*INL*) grešku za različite radne temperature. Na slici 3.9.a napajanje i referentni napon iznosi 3 V, a na slici 3.9.b iznosi 5 V. Primjećujemo da je *INL* najveći pri polovici ulaznoga napona, također je ovisan i o temperaturi rada. Razlika *INL*-a je reda 2 ppm za raspon temperatura od -55 °C do 90 °C. Kod obe vrijednosti ulaznoga napona *INL* ima slične vrijednosti i ne prelazi 4 ppm.



a) Napajanje 3 V b) Napajanje 5 V

Slika 3.9. integrirana nelinearna greška pretvornika [3]

Ukupna greška uslijed negativnog ulaznog signala za različite radne temperature je prikazana na slici 3.10. Na slici 3.10.a napajanje i referentni napon iznosi 3 V, a na slici 3.10.b iznosi 5 V. Bez obzira na ulazni mjereni naponski signal glavni faktor povećanja greške je temperatura. Za više temperature greška eksponencijalno raste.

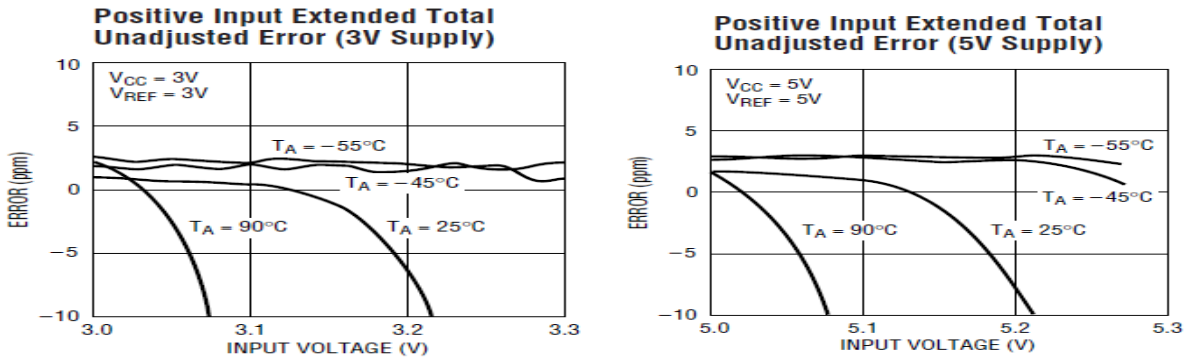


a) Napajanje 3 V b) Napajanje 5 V

Slika 3.10. Ukupna greška usljed negativnog ulaznog signala [3]

Ukupna greška za napone ulaza koje premašuju referentni naponski ulaz za različite radne temperature prikazana je na slici 3.11. Na slici 3.11.a napajanje i referentni napon iznosi 3 V, a na slici 3.11.b iznosi 5 V. Glavni faktor povećanja greške je temperatura. Za više temperature

greška eksponencijalno raste. Dok je za niske temperature kao -45 °C i -55 °C greška je relativno ustaljena.

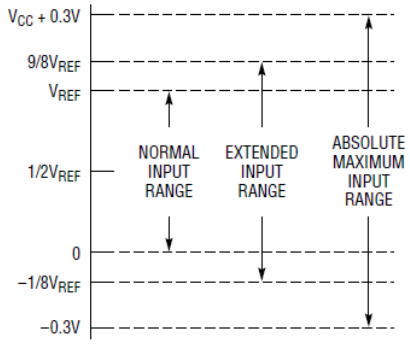


a) Napajanje 3 V b) Napajanje 5 V

Slika 3.11. Ukupna greška usljed negativnog ulaznog signala [3]

3.2.4. Ulazni naponski signal

Pretvornik je u stanju prilagoditi sustav za ulazne napone izvan normalnoga opsega. Kao što je prikazano na slici 3.12., pojedini rasponi ulaznog signala, normalni, prošireni i apsolutno maksimalni. Normalni raspon je od 0V do V_{REF}. Prošireni ulazni raspon je od 0,125 • V_{REF} do 1.125 • V_{REF}. A apsolutno maksimalni naponskim rasponom je od – 0.3V do (Vcc + 0.3V). Iza toga raspona aktivira se zaštitni ulazni ESD-ovi (engl. *Electrostatic Discharge*) i pogreška uslijed curenja ulazne struje se naglo povećava.



Slika 3.12. Rasponi ulaznoga signala [3]

Ulazni signal se priključuje na V_{in} pin raspon mu iznosi od –300 mV do *Vcc* + 300 mV. Kako bi se ograničila struja može se dodati otpornik do 5 kΩ u seriju s V_{in} pinom.

Kod izrade je bitno da se otpornik postavi što bliže pinu V_{in} kako bi parazitski kapacitet bio što manji. Uvođenjem otpornika u seriju stvara se ovisnost temperature i pogreške pomaka zbog curenja ulazne struje. 1nA curenja ulazne struje će prouzročiti 1 ppm offset error na 5 kΩ ako je $V_{REF} = 5\ V$. Ta greška ima značajnu temperaturnu ovisnost.

3.2.5. Format izlaznih podataka

Pretvornik šalje serijski podatkovini niz duljine 32 bita. Prva 4 bita su status bitovi i označavaju predznak, raspon ulaznog napona i stanje pretvorbe. Sljedeća 24 bita su rezultat pretvorbe. Najznačajniji bit *MSB* (engl. *Most Signification Bit*) je prvi izlazni bit, a zadnja četiri bita su najmanje značajna bita *LSB* (engl. *L*east Significant *Bit*). *LSB* bitovi su bitovi koji mogu biti uključeni u rezultat kao prosjek ili odbačeni bez gubitka rezolucije. U tablici 3.2. su prikzani statusni bitovi.

Bit 31 je prvi izlazni bit koji je indikator završetka pretovrbe (EOC bit engl. *End Of Conversion*). Taj bit je dostupan na pinu *SDO* za vrijeme pretvorbe i kada je pin CS u stanju logičke nule. Dok je u logičkoj jedinici za vrijeme pretvorbe i ide u stanje logičke nule kada je pretvorba završena. 30 bit (drugi izlazni bit) je *DMY* bit i uvijek je u stanju logičke nule.

29. bit (treći izlazni bit) je indikator predznaka. Ako je na pinu V_{in} napon veći od nula volti, 29 bit je u stanju logičke jedinice. U suprotnome ako je na pinu V_{in} negativni napon 29 bit je u stanju logičke nule.

28. bit (četvrti izlazni bit) je indikator proširenog ulaznog raspona (extended input range EXR). Ako je ulaz unutar normalnoga ulaznog raspona $0 \leq V_{in} \leq V_{REF}$, taj je bit u stanju logičke nule. Ako je ulaz izvan normalnog ulaznog raspona $V_{in} > V_{REF}$ ili $V_{in} < 0$, taj je bit u stanju logičke jedinice.

Tablica 3.2. Satusni bitovi [3]

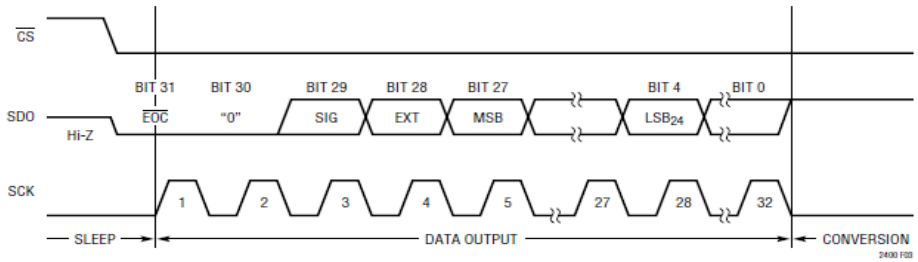
Table 1. LTC2400 Status Bits

Input Range	Bit 31 $\overline{EOC}$	Bit 30 DMY	Bit 29 SIG	Bit 28 EXR
$V_{IN} > V_{REF}$	0	0	1	1
$0 < V_{IN} \leq V_{REF}$	0	0	1	0
$V_{IN} = 0^+/0^-$	0	0	1/0	0
$V_{IN} < 0$	0	0	0	1

27. bit (peti izlazni bit) je najznačajniji bit *MSB*. Bitovi od 27-4 su 24 bitni rezultat pretvorbe..

4. bit je najmanje značajan bit *LSB*, dok su bitovi od 3 do 0 dodatni *LSB* bitovi koji mogu biti uključeni u prosjek ili odbačeni bez gubitka rezolucije.

Podaci se šalju vani sa *SDO* pina pod kontrolom serijskog brojača *SCK* prikazano na slici 3.13. Kada je *CS* u stanju logičke jedinice, *SDO* ostaje u visokoj impedanciji i ignorira se svaki takt serijskoga brojača od integriranoga izlanog shift registra.



Slika 3.13. Format izlaznih bitova [3]

Kako bi došlo do izlaza podataka, *CS* se mora spustiti na logičku nulu. Indikator završetka pretovrbe (*EOC*), uočava da je *CS* spušten u logičku nulu i u realnom vremenu se *EOC* mijenja sa stanja jedinice u nulu po završetku pretvorbe. 31 bit zahvaćen je s prvim rastućim bridom serijskog brojača *SCK*. 30 bit je poslan vani s padajućim bridom od serijskog brojača *SCK*. Zadnji bit (Bit 0) je pomaknut na padajući brid 31 otkucaja brojača. Na padajući brid 32 *SCK* otkucaja, *SDO* prelazi u stanje logičke jedinice, indicirajući novi ciklus pretvorbe. Tablica 3.3. pokazuje izlazne podatke za određene vrijednosti ulaznog signala u odnosu na referentni napon.

Tablica 3.3. Format izlaznih podataka [3]

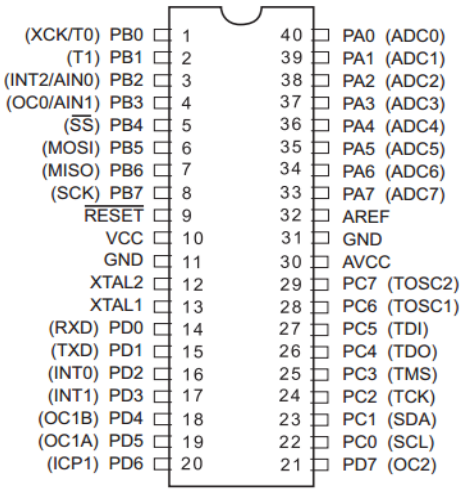
Table 2. LTC2400 Output Data Format

Input Voltage	Bit 31 $\overline{EOC}$	Bit 30 DMY	Bit 29 SIG	Bit 28 EXR	Bit 27 MSB	Bit 26	Bit 25	Bit 24	Bit 23	...	Bit 4 LSB	Bit 3-0 SUB LSBs*
$V_{IN} > 9/8 \cdot V_{REF}$	0	0	1	1	0	0	0	1	1	...	1	X
$9/8 \cdot V_{REF}$	0	0	1	1	0	0	0	1	1	...	1	X
$V_{REF} + 1LSB$	0	0	1	1	0	0	0	0	0	...	0	X
V_{REF}	0	0	1	0	1	1	1	1	1	...	1	X
$3/4V_{REF} + 1LSB$	0	0	1	0	1	1	0	0	0	...	0	X
$3/4V_{REF}$	0	0	1	0	1	0	1	1	1	...	1	X
$1/2V_{REF} + 1LSB$	0	0	1	0	1	0	0	0	0	...	0	X
$1/2V_{REF}$	0	0	1	0	0	1	1	1	1	...	1	X
$1/4V_{REF} + 1LSB$	0	0	1	0	0	1	0	0	0	...	0	X
$1/4V_{REF}$	0	0	1	0	0	0	1	1	1	...	1	X
$0^+/0^-$	0	0	1/0**	0	0	0	0	0	0	...	0	X
–1LSB	0	0	0	1	1	1	1	1	1	...	1	X
$-1/8 \cdot V_{REF}$	0	0	0	1	1	1	1	0	0	...	0	X
$V_{IN} < -1/8 \cdot V_{REF}$	0	0	0	1	1	1	1	0	0	...	0	X

*The sub LSBs are valid conversion results beyond the 24-bit level that may be included in averaging or discarded without loss of resolution.
**The sign bit changes state during the 0 code.

3.3. Osobine i uloga mikroupravljača Atmega16 u radu sklopa

Atmega16 je proizvod tvrtke Atmel pripada obitelji AVR upravljača. Radi se 8 bitnome mikroupravljaču s *RISC4* arhitekturom. Različite tipovi obitelji AVR se međusobno razlikuju veličinom flash memorije (memorija programa), veličinom podatkovne memorije (*EEPROM*), veličinom interne podatkovne memorije (*SRAM*), brojem portova, analogno-digitalnih pretvornika, broja UART, timera i ostaloga. Neke osobine upravljača: maksimalna taktna frekvencija je 16 MHz, 16Kbyte *Flash* memorije, 512 kByte *EEPROM* memorije, 1 kByte interni *SRAM*, 2x8 Bit Timer/Counter, 1x16 Bit Timer /Counter, 8x10 Bit A/DC i jedno sučelje za UART komunikaciju. Upravljač raspolaže s 40 pinova kao što je prikazano na slici 3.14. Pinovi port-a A su korišteni za slanje podataka kako bi se na LCD-u prikazao rezultat. A pinovi port-a C kao kontrolni pinovi LCD-a. Pojedini pinovi porta B za programiranje upravljača (pinovi PB5 do PB10). Za komunikaciju s pretvornikom pinovi PB4, PB6 i PB7. Za UART komunikaciju pin PD0.



Slika 3.14. Pinovi Atmega16 mikroupravljača [4]

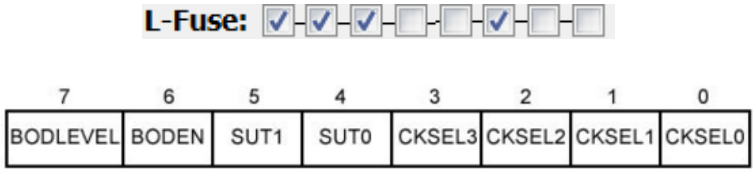
Uloga mikroupravljača Atmega16 je da preuzime 32 bitni niz od LTC2400 pomoću SPI komunikacije. Taj niz obrađuje tj. izvlači 24 bitni niz i preračunava ga u izmjereni napon. Vrijednost izmjerenog napona šalje na serial/USB adapter putem UART komunikacije i na LCD1602 koji je integriran na sklopu. Atmega16 je postavljen na 8 Mz. Kod Atmel AVR mikroupravljača programirati fuse bit znači ga postaviti na nulu, preporučljivo je koristit AVR Fuse Calculator kao na slijedećoj stranici www.engbedded.com/fusecalc. Kako bi se ispravno podesili fuse bitovi. Atmega16 raspolaže s 16 fuse bita tj. 2 fuse byte-a. Prvi byte označen s H-fuse ima sljedeće bitove kao na slici 3.15.

Slika 3.15. H-fuse bitovi

OCEDN bit je inicijalno na jedinici. Služi za pronalaženja grešaka na čipu preko JTAG sučelja. *JTAGEN* bit je inicijalno na nuli. Služi za aktiviranje JTAG sučelja. *SPIEN* bit je inicijalno na nuli, ako se deaktivira taj bit tj. postavi na jedinicu ne može se više mikroupravljača programirati preko ISP-a, može se programirati samo s adapterom koji mikroupravljač paralelno programira

kao što je STK500. *CKOPT* bit označava pojačivač oscilatora. Ako je oscilator previše udaljen od mikroupravljača može se dogoditi da signal oslabljen dođe. Aktiviranjem bita se pojačava taj oslabljeni signal, ali s tim se povećava i potrošnja energije. *EESAVE* bit inicijalno nije aktivan, ako se aktivira s svakim Chip-Erase se briše i *EEPROM*. *BOOTSZ1* i *BOOTSZ2* bitovi označavaju veličinu Bootloader-a. Veličina Bootloader-a je između 128 i 1024 riječi i može se pomoću ova dva bita odrediti. *BOOTRST* izbor reset vektora, inicijalno je deaktiviran. Ako je aktiviran nakon reseta izvođenje počinje od bootloader-a, a ne od adrese 0x00 kao što je kad je deaktiviran. U ovome slučaju aktivirao sam od H-bitova *JTAGEN*, *SPIEN*, *BOOTSZ1* i *BOOTSZ2*.

BODLEVEL bit inicijalno nije aktiviran i reset se napravi kod napona 2.7 V. Kada je aktiviran reset se dogodi na 4V. *BODEN* bit inicijalno nije aktivan, treba se aktivirati ako se koristi *EEPROM*, kako bi se spriječilo brisanje iz memorije uslijed veće varijacije napona. *SUT1* i *SUT0* označavaju vrijeme pokretanja mikroupravljača, inicijalno nisu aktivan, *SUT0* je aktivan, čime se zadaje maksimalno vrijeme od 65ms pokretanja mikroupravljača. *CKSEL* bitovima se određuje izvor takta mikroupravljača. U ovome slučaju aktivni su *SUT0*, *CKSEL3*, *CKSEL1* i *CKSEL0*. Ova kombinacija *CKSEL* bitova označava da se koristi kalibrirani interni *RC* oscilator s frekvencijom takta 8 MHz. Na slici 3.16. prikazano je kako su postavljeni L-fuse bitovi.



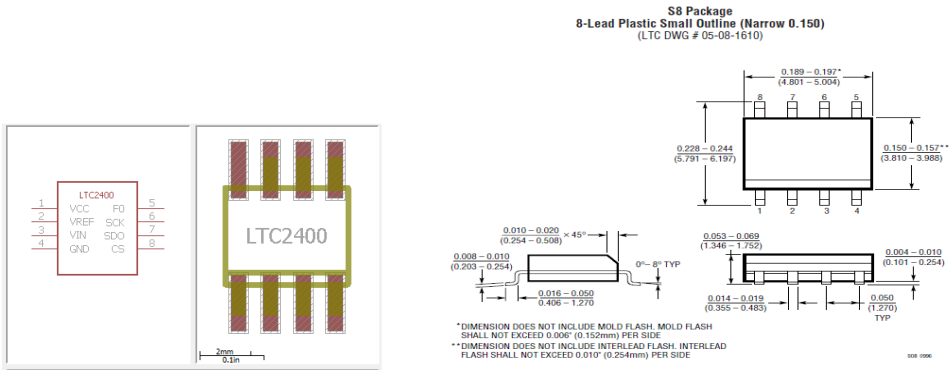
Slika 3.16. L-fuse bitovi

3.4. Dizajniranje pločice

Programski alat Eagle omogućava izradu sheme spoja i dizajniranje PCB pločice. Na shemi PCB pločice se nalaze sve komponente koje koristimo u shemi spoja i međusobno su povezane kao i na shemi spoja. Potrebno je povezati komponente s vodovima koje će biti na PCB pločici. Vodovi se mogu ručno nacrtati ili pomoću Autorouter alata. Za kompleksnije sheme preporučeno je ručno crtanje vodova. Mnoge elektroničke komponente su uključene u Eagle

biblioteku (engl. *library*) gotovo sve, međutim pretvornik LTC2400 nije uključen. Iz tog razloga je isti nacrtan i dodan u biblioteku. Crtanje sheme, a pogotovo crtanje vodova PCB pločice je dugotrajan proces u kojem se uvijek može naći bolji način. Kod dizajniranja je, kako bi se dobila kvalitetna shema spoja i dobro postavljeni vodovi PCB pločice, vrlo važno iskustvo. Kada se nacrtaju vodovi među komponentama CNC strojem se vrši obrada pločice.

Na slici 3.17.a je prikazan je simbol u shemi spoja i izgled komponente na PCB pločici. Simbol se nije nalazio u bibliotekama koje pruža Eagle aplikacija, tako da je bilo potrebno nacrtati simbol i izgled na pločici. Kod crtanja simbola potrebno je obratiti pozornost da se pinovi točno povežu s komponentom koja ide na pločicu. Također je potrebno obratiti pozornost da izgled komponente na PCB pločici odgovara stvarnoj veličini komponente. Dimenzije komponente i razmake među pinovima daje proizvođač u podatkovnome listu prikazane na slici 3.17.b.



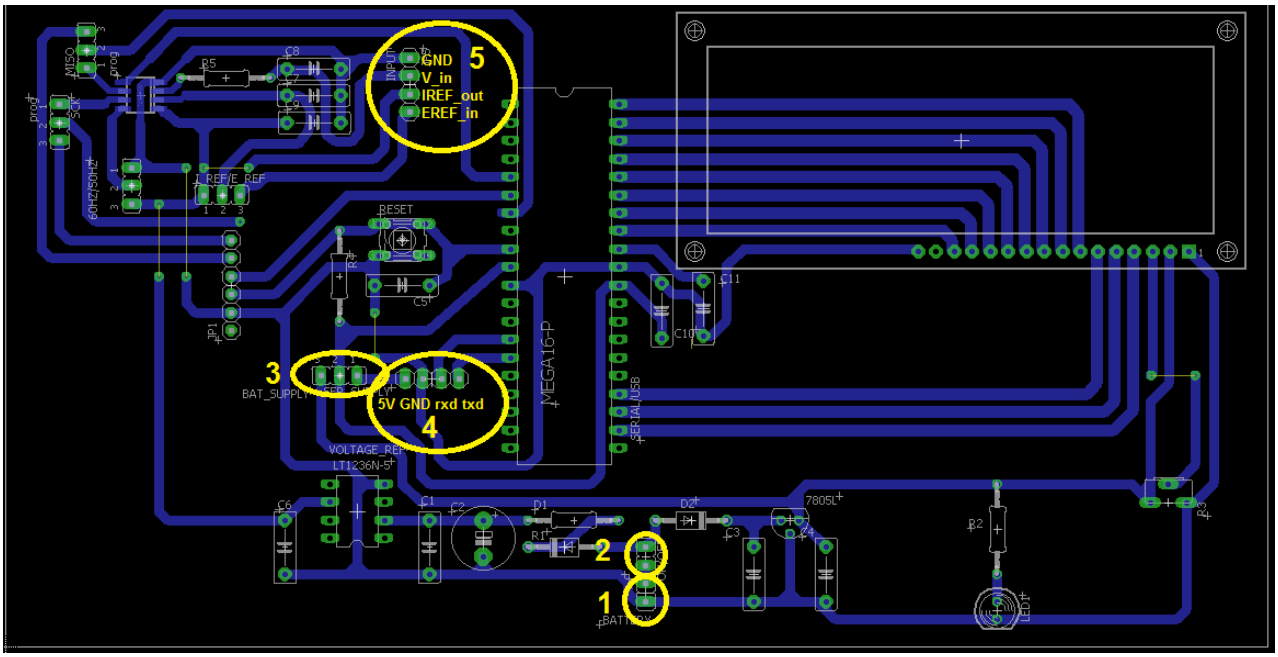
a) LTC2400 u Eeaglu b) Dimenzije LTC2400

Slika 3.17. Pretvornik LTC2400

3.4.1. Shema spoja sklopa i ožičenje pločice

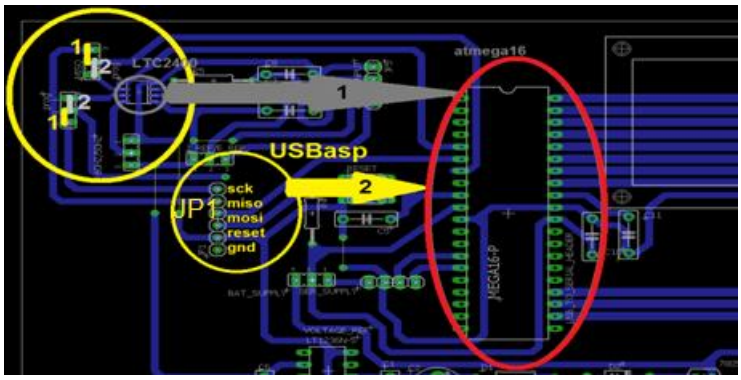
Cijeli sklop se napaja pomoću baterije koja se priključuje izvod koji je označen s brojem 1 na slici 3.18. Nakon njega imamo *ON/OFF* izvod na kojeg možemo priključiti prekidač za paljenje odnosno gašenje sklopa, označen je s brojem 2. Dok je s brojem 3 označen izvod *SER_SUPPLY/BAT_SUPPLY* s kojim možemo isključiti napajanje s baterije i uključiti napajanje mikroupravljača s USB porta računala. Ako spojimo pin lijevi i srednji pin toga izvoda mikroupravljač će se napajati s USB porta, a ako spojimo srednji i desni pin mikroupravljač će se napajati preko baterije. To jest iz reguliranog napona koji dolazi od baterije. Baterija treba biti od

9 do 30 V. Serial/USB izvod označen s brojem 4 koristi se za slanje podataka na USB/serial adapter pomoću UART komunikacije. S brojem 5 označen je izvod *INPUT*. Na GND od izvoda *INPUT* spajamo niži potencijal od naponskoga signala kojega mjerimo. Na pin *V_in* spajamo viši potencijal naponskoga signala kojega mjerimo. Pin *IREF_out* dolazi s internoga preciznoga referentnoga naponskog sklopa i moguće ga je provjeriti vanjskim instrumentom. Pin *EREF_in* služi za spajanje vanjskog referentnoga napona. Pin *V_in* je potrebno što bliže postaviti otporniku *R5*, kako bi se smanjilo utjecaj smetnji na taj vod. Također na slici 3.18 vidimo da je otpornik *R5* postavljen neposredno do pretvornika i pina V_{in} kako bi smetnje usljed duljine voda bile minimalne.



3.18. Položaj pojedinih izvoda na pločici

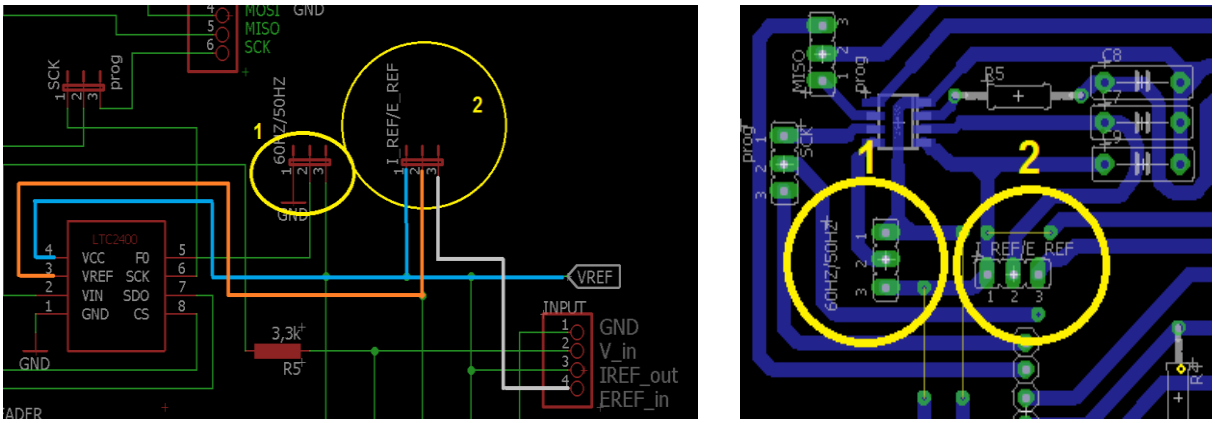
JP1 izvod prikazan na slici 3.19. predstavlja ulaz za USBasp programtor, pinovi *SCK* i *MISO* koriste programator i pretvornik. Kako nije moguće istovremeno programirati upravljač i vršiti pretvorbu koriste se jumperi. Postavljanjem jumpera biramo želimo li da pretvornik komunicira s mikroupravljačom ili programator. Na slici 3.19. je prikazano ako su izvodi spojeni prema 1 onda komunicira mikroupravljač s programatorom, ako su izvodi spojeni prema 2 onda mikroupravljač komunicira s LTC2400.



3.19. JP1 izvod

Na slici 3.20.a prikazan je izvod *60Hz/50Hz* koji je označen s brojem jedan. Ako pin srednji pin spojen s pinom lijevim tj. uzemljenjem pretvornik će slati rezultat s frekvencijom 60 Hz. Ako je srednji spojen s desnim pretvornik će slati rezultat frekvencijom od 50 Hz. Na slici 3.20.b prikazano je gdje se nalazi izvod 60Hz/50Hz na PCB pločici također označeno s brojem jedan.

Osim izvoda *60Hz/50Hz* na slici 3.20. prikazan je i izvod *I_REF/E_REF* označen s brojem dva. Ako je spojen pin lijevi i srednji pin pretvornik će se napajati s internoga napajanja koji je ujedno i referentni napon pretvornika. Ako spojimo srednji i desni pin (na *I_REF/E_REF*) tada trebamo na pinu *IREF_in* (koji se nalazi na *INPUT* izvodu) dovesti eksterni napon koji je referentni napon pretvornika. A pretvonik će se napajati i dalje preko internoga napona. Na slici 3.20.b prikazan je položaj *I_REF/E_REF* izvoda i izvoda *60Hz/50Hz* na PCB pločici.



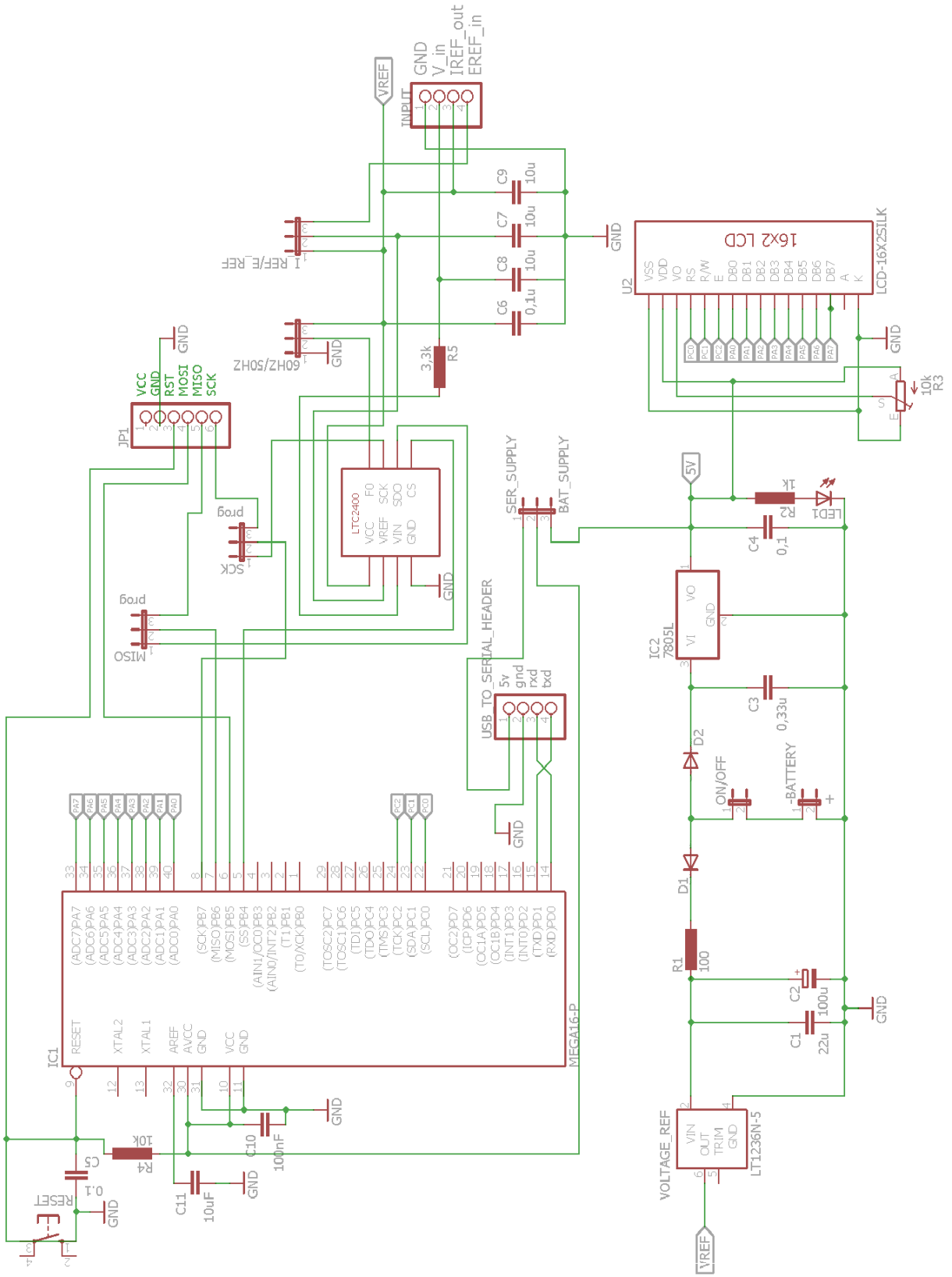
a) Izvodi 60/50Hz i I_REF/E_REF na shemi b) Izvodi 60/50Hz i I_REF/E_REF na pločici

Slika 3.20. Izvodi 60/50Hz i I_REF/E_REF

Na slici 3.21. prikazana je cjelokupna elektro shema sklopa. Diode *D1* i *D2* sprečavaju povratne struje u bateriju i štite sklop u slučaju zamjene polariteta napajanja. Sklop ima tipkalo za resetiranje pretvornika, te jednu led diodu koja signalizira da je prisutan napon napajanja s baterije. Otpornik R3 služi zs podešavanje kontrasta LCD-a

IC2 napaja mikroupravljač i LCD1602. Dok LT1236N-5 predstavlja interni referentni napon i napon napajanja pretvornika. Ako se koristi vanjski referentni napon za pretvornik, onda napon s LT1236N-5 je napon napajanja pretvornika.

U ovoj izvedbi iako prvobitno planirano nije korišten LT1236N-5 nego komponenta 7805L, kao što je IC2. 7805L je također naponski regulator sa manjom točnošću nego LT1236N-5. Mjerenja su rađenja sa vanjskim referentnim izvorom koji je točniji i od LT1236N-5 komponente. 7805L je mnogo povoljnija komponenta od LT1236N-5.



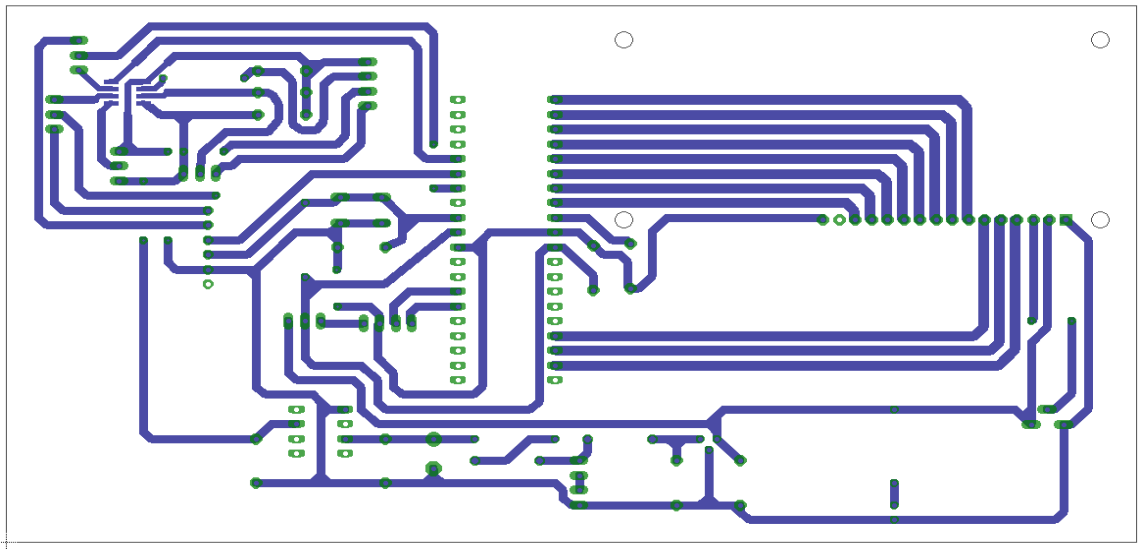
Slika 3.21. Električna shema sklopa

Tablica 3.4. Popis korištenih komponenti

Broj komponente	Naziv	Opis	Vrijednost	Količina
1.	Atmega16	Mikroupravljač		1
2.	LTC2400	ADC pretvornik		1
3.	7805L	Regulator napona	5V	2
4.	LCD1602	Display		1
5.	Reset	Tipkalo		1
6.	R1	Otpornik	100 Ω	1
7.	R2	Otpornik	1 K Ω	1
8.	R3	Promjenivi otpornik	Do 10 KΩ	1
9.	R4	Otpornik	10 KΩ	1
10.	R5	Otpornik	3.3 KΩ	1
11.	C1	Kondenzator	22 µF	1
12.	C2	Elektrolitski kondenzator KKKondenzator	100 µF	1
13.	C3	Kondenzator	0.33 µF	3
14.	C4, C5,C6	Kondenzator	0.1 µF	1
15.	C7,C8,C9,C11	Kondenzator	10 µF	4
16.	C10	Kondenzator	100 nF	1

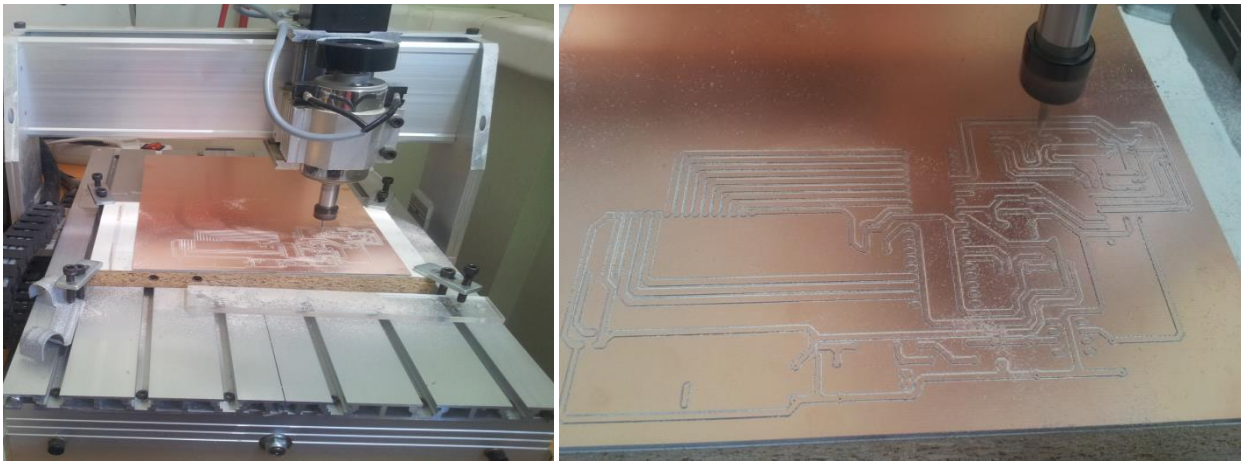
3.4.2. Izrada pločice

Prednji strana pločice tj. vodovi koje CNC treba napraviti prikazane su na slici 3.22.



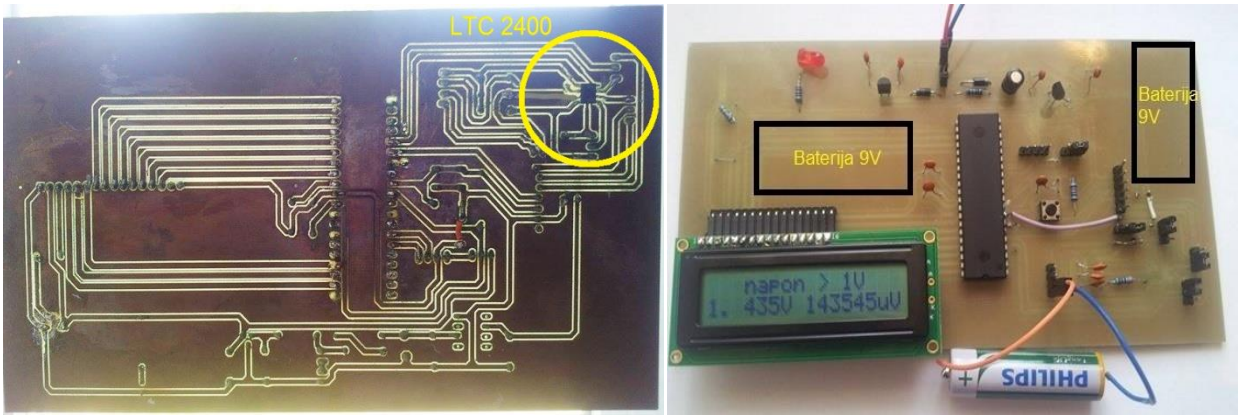
Slika 3.22. Vodovi za CNC stroj

Na slici 3.23. je prikazano kako CNC stroj pravi vodove koji su prije u eaglu dizajnirani. Stroj ima točnošću pozicije od 0.05 mm, a mehaničku rezoluciju 0.0025 mm po koraku.



Slika 3.23. Izrada pločice

Na slici 3.24.b je prikazana gotova pločica s zalemljenim komponentama. Ostavljeno je prostora za dvije baterije od 9 V s kojim bi se instrument napajao. Baterije se mogu spojiti serijski ili paralelno. Preporučeno je da baterije budu u serijskome spoju kako bi napon iznosio 18 V. Sklopu je potrebno najmanje 9 V za napajanje, u paralelnome spoju baterija taj bi se napon brže spustio ispod 9 V nego u serijskome spoju. Na slici 3.24.a prikazana je donja strana sklopa s LTC2400-om.



a) Donja strana sklopa b) Gornja strana sklopa

Slika 3.24. Izgled sklopa

4. RAZVOJ PROGRAMSKOG KODA SKLOPA

Programski kod je pisan u C jeziku, razvijen u programskom okruženju Atmel Studio 6. Sastoji se od sedam datoteka. To su main.c, spi.h, spi.c, uart.h, uart.c, lcd.h, alati.h . Programiranje je izvršeno preko khazama AVR programmer aplikacije. Interni oscilator mikroupravljača je u biblioteci alati.h postavljen na 8 MHz, kako su postavljeni i fuse bitovi.

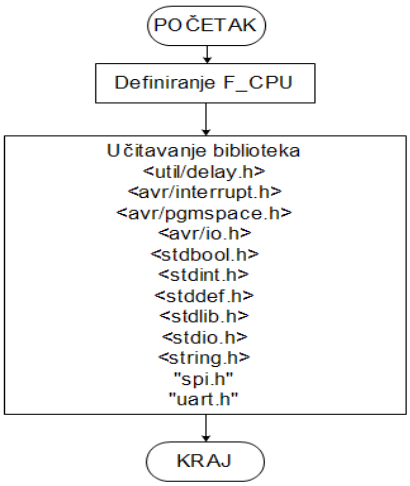
4.1. Način rada koda

Napisani kod mjeri napon od 0 do 5 V s 24 bitnom rezolucijom prepoznaje ako napon je negativan i šalje vrijednost na LCD u mili i mikro voltima. Preko UART komunikacije šalje na USB port računala također izmjerenu vrijednost kojiu prikazuje u mili i mikro voltima. Kako pretvornik ima 16777216 koraka (kvantizacijskih koraka) za napon od 0 do 5 V. Prikazuje se koji je kvantizacijski korak mjerenoga napona i binarna vrijednost mjerenoga napona. U slučaju zamjene polariteta prikazuje se poruka o pogrešnome polaritetu.

SPI komunikacija omogućava sinkroni prijenos podataka između pretvornika i mikroupravljača. Način komunikacije definiran je u spi.h i spi.c. UART komunikacija služi se s dva cirkularna spremnika za primanje i slanje podataka na serijsko sučelje upravljača. Interrupt se generira kada UART komunikacija završi primanje ili slanje. Datoteke uart.h i uart.c je napisao Peter Fleury-a i izvorno su primjenjive za niz Atmega upravljača. U prilogu se nalazi kod aplikacije.

4.2. Dijagram toka alati.h

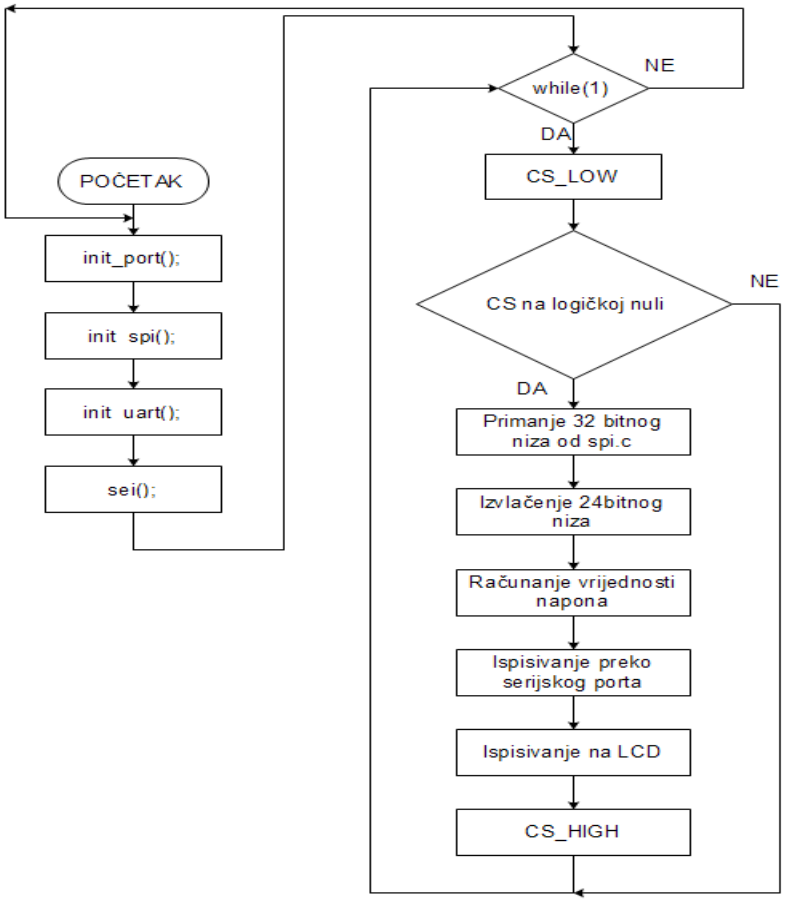
Na slici 4.1. je prikazan dijagram toka alati.h biblioteke. Alati.h sadržava biblioteke koje koriste druge c datoteke.



Slika 4.1. Dijagram toka alati.h biblioteke

4.3. Dijagram toka main.c

Main.c je glavna datoteka njezin dijagram toka prikazan je na slici 4.2. Najprije se vrši inicijalizacija porta za SPI komunikaciju, inicijalizacija SPI komunikacije, inicijalizacija UART komunikacije, pokretanje interrupt routine. Spušta se CS na nisku razinu preuzima se 32 bitni niz od pretvornika preko SPI komunikacije. Izvlači se 24 bitni niz koji predstavlja mjerenu vrijednost, koji se pretvara u vrijednost napona. Nakon toga se šalje ta vrijednost na txd pin mikroupravljača putem uart komunikacije i na LCD display. Nakon pretvorbe se generira interrupt podaci se spremaju u UDR registar (engl. *Uart Data Registar*) i šalju ili primaju na preko txd i rxd pina. Brzina UART komunikacije je postavljena na 9600 Baud-a.



Slika 4.2. Dijagram toka main.c datoteke

4.4. Dijagram toka spi.h i spi.c

Na slici 4.3.a vidimo dijagram toka spi.c datoteke. Najprije definiramo nižu i višu riječ koje imaju zajedno 32 mjesta i služe za primanje podatka iz pretvornika.

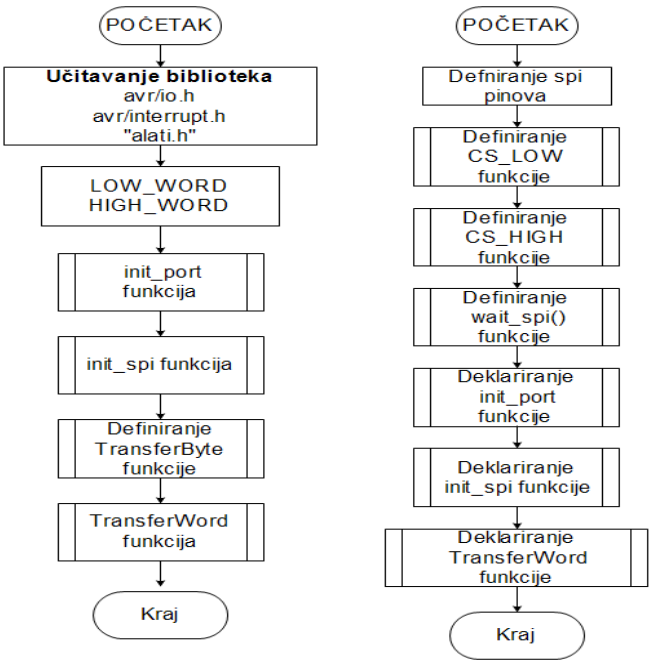
Smjerovi signala pojedinih pinova postavljeni su kako bi se mogla komunikacija odvijati. Što je definirano s init_port funkcijom koja se nalazi u spi.c datotetci. Pinovi *SCK,CS* postave kao izlazni pinovi, a *MISO* kao ulazni pin. *MISO* označava da master prima podatke slave šalje. Pin *CS* se postavi u logičku jedinicu, kako ne bi mikroupravljač primao podatke prije vremena.

Inicijalizacija SPI komunikacije omogućila se u main.c datoteci i u spi.c datoteci definiraju se pojedini bitovi *SPCR* i *SPSR* registara u spi_init funkciji. *SPCR* je osam bitni spi kontrolni registar. Bitovi *SPE,MSTR,SPR0* su postavljeni na jedan, a bitovi *CPOL*, *CPHA* su postavljeni na nulu. *SPE* (engl. *SPI Enable*) bit mora biti postavljen u jedinicu kako bi se omogućila bilo koja

SPI operacija. *MSTR* (engl. *Master/Slave select*) jedinicom je označeno da mikroupravljač ima funkciju mastera. *SPR0* (engl. *SPI Clock Rate Select*) bitovi označavaju brzinu SPI komunikacije u odnosu na frekvenciju oscilatora. Postavljen je bit *SPR0* na jedan, a *SPR1* ostaje na nuli što daje brzinu SPI komunikacije $f_{osc}/16$. Kako je frekvencija mikroupravljača 8 MHz, brzina SPI komunikacije je 500 kHz. Bit *CPOL* (engl. *Clock Polarity*) označava polaritet brojača i postavljen je u nulu. Što znači da je takt u nuli za vrijeme stanja mirovanja, a promjenu na jedinicu broji kao rastući brid. Bit *CPHA* (engl. *Clock Phase*) označava fazu brojača i postavljen je na nulu. Što znači da se podaci učitavaju na rastući brid.

SPSR (engl. *SPI Status Registar*) je statusni registar i njegovi bitovi su u init_spi funkciji postavljeni na nulu.

Funkcija TransferByte učitava po osam bitova u *SPDR* (engl. *SPI Data Registar*) i šalje ih dalje funkciji TransferWord, zatim prima novi osam bitni niz. Tako se u funkciji TransferWord tvori 32 bitni niz koji je pretvornik poslao. U spi.h biblioteci slika 4.3.b se deklariraju funkcije CS_LOW, CS_HIGH, init_port, init_spi kako bi se mogle pozvati u main.c datotetci.

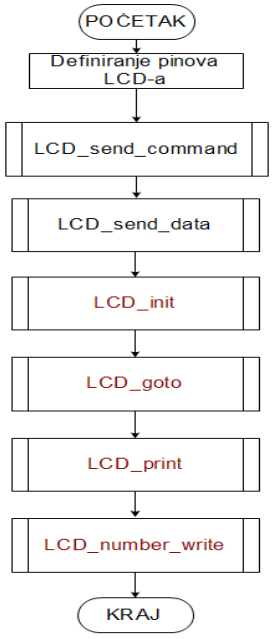


a) Dijagram toka spi.c b) Dijagram toka spi.h

Slika 4.3. Dijagram toka spi

4.5. Dijagram toka lcd.h

Na slici 4.4. vidimo dijagram toka LCD.h datoteke koja se poziva u main.c datoteku. Kao i na prototipu podatkovni pinovi LCD-a su spojeni na na port A mikroupravljača. A kontrolni pinovi LCD-a na port C mikroupravljača. Definirano je na samome početku lcd.h datoteke. U funkciji LCD_send_command se nalaze naredbe za aktiviranje i deaktiviranje LCD-a, te naredbe za gašenje *RW* i *RS* pina. LCD_send_data funkcija postavlja kontrolne pinove da se podaci mogu prikazati. LCD_init funkcija postavlja LCD u 8 bitni mod slanje naredbe 0x38 preko LCD_send_command, zatim 0x01 briše sve s LCD-a, 0x06 mod za unos s auto inkrementacijom bez pomicanja. Funkcija LCD_goto postavlja kursor na mjesto koje smo zadali, kako bi funkcija LCD_print ispisala string sa toga mjesta. Funkcijom LCD_number _write se broj pretvara u string i ispisuje se.



Slika 4.4. Dijagram toka LCD.h

5. FUNKCIONALNOST SKLOPA

Sklop u realnom vremenu prikaziva rezultat pretvorbe na LCD-u i šalje na USB port računala. U slučaju pogrešno spojenoga polariteta sklop daje poruku o tome. Prespajanjem određenih jumpera možemo izabrati da li da pretvornik radi na 50 ili 60 Hz. Također pomoću prespajanja jumpera možemo podesiti da li da koristimo interni referentni naponski signal ili eksterni za pretvornik. Mjeri napone i iznad 5 V, ali je mjerna pogreška veća. Na sklop se može spojiti djelitelj napona pomoću otpornika tako da se mogu i veći naponi priključiti.

5.1. Mjerenja pomoću referentnoga naponskoga signala

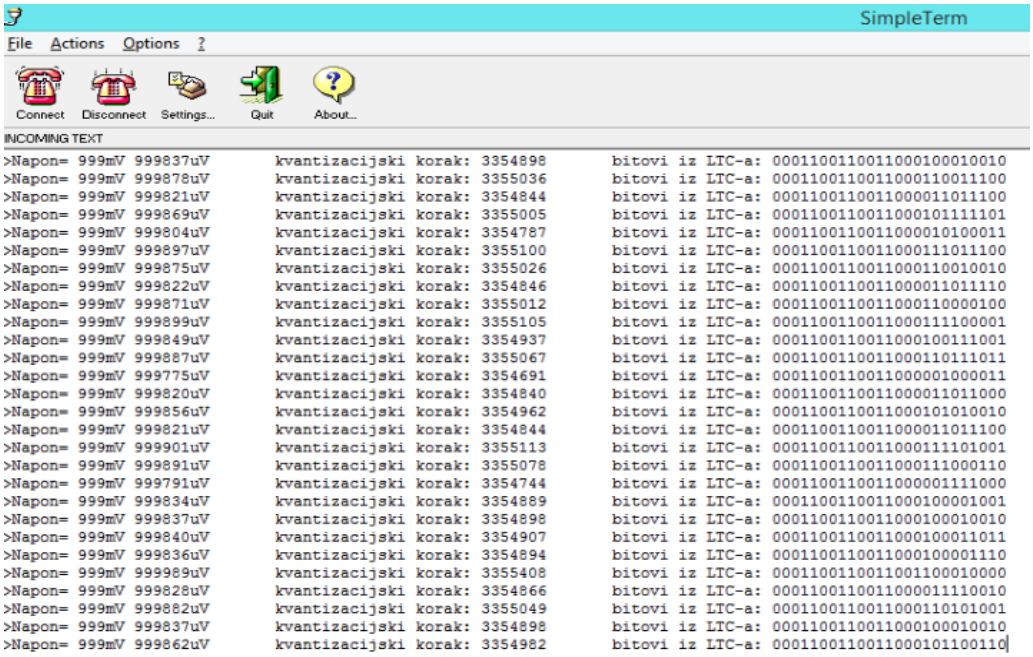
Mjerenja su rađena pomoću napajanja sklopa i referentnoga napona. Na slici 5.1.a prikazan je Agilent 34661A instrument s kojim je baždaren Referentni napon. Osim baždarenja refrentnoga napona instrumentom su baždareni i ulazni naponi koji su mjereni. Na slici 5.1.b prikazan je izvor napajanja koji služi za referentni napon i napon kojega mjerimo.

Kako bi rezultati bili vjerodostojni napravljeno je 20 mjerenja za svakih 100 mV. Očitavanje mjerenoga rezultata se vršilo svake sekunde. Refentni napon iznosi 5V, temperatura u laboratoriju 25 ℃. U prilogu se nalaze sva mjerenja.

a) Agilent 34661A b) Izvor referentnoga napona i mjernoga napona

Slika 5.1. Laboratorijska oprema

Mjereni napon očitava se pomoću SimpleTerm aplikacije koja se povezuje na USB port. Kao na slici 5.2. vidimo da je prikazan izmjerni napon u mili voltima i mikro voltima, broj kvantizacijskih koraka te binarni zapis izmjerenoga napona.



Slika 5.2. Prikaz rezultata mjerenja na računalu

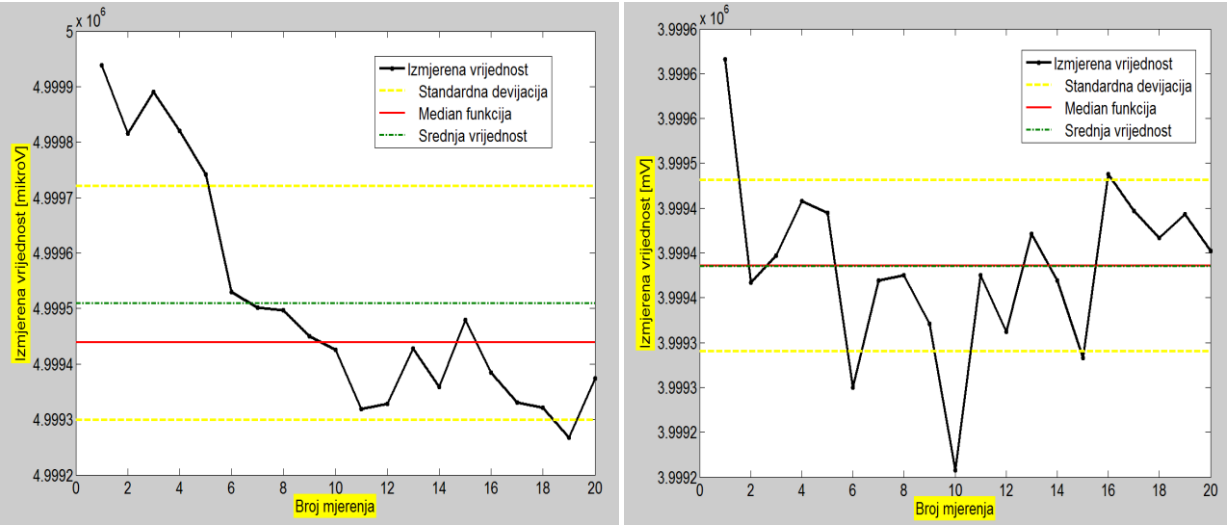
Tablica 5.1. prikazuje sva mjerenja, izračunatu srednju vrijednost za svih 20 mjernih uzoraka, grešku, median funkciju i standardnu devijaciju nad mjerenjima. Greška je dobivena razlikom točne vrijednosti (izmjerene s Agilent 34661A instrumentom) i srednjom vrijednošću svih uzoraka. Vidimo da se greška kreće od 0 do 1100 µV. Prosječna greška sklopa iznosi 314 µV. Ako izbacimo iz te kalkulacije dvije greške koje jedino puno odstupaju od drugih grešaka. Što bi se moglo pripisati pogrešci pri mjerenju. Prosječna greška sklopa iznosi 285 µV. Rezultat je jako dobar ako uzmemo u obzir da na ulazni V_{in} pin je spojen otpornik od 3.3 kΩ koji dodatno unosi smetnju i utjecaj svih ostalih smetnjih. Kao što je nestabilnost mjerenog napona koji također oscilira, pad napona na vodičima do ulaza u pretvornik i pad napona na kontaktima.

Median funkcija nad izmjerenim vrijednostima napona vrši sortiranje uzoraka i uzima srednji uzorak. Standardna devijacija pokazuje koliko puno osciliraju izmjerene vrijednosti od srednje vrijednosti.

Tablica 5.1. Rezultati mjerenja

R. br.	Ulazni napon Mjerenje[mV]	Srednja vrijednost Izmjerenoga napona [µV]	Greška = |(ulazni napon – srednja vrijednost)| [µV]	Median [µV]	Standardna Devijacija [µV]	
1.	5	4.9714e+03	28.6	4973	65.66	
2.	20	1.9959e+04	41	19956	62.95	
3.	60	6.0062e+04	62	60029	158.70	
4.	100	1.0004e+05	40	100065	111.35	
5.	200	1.9998e+05	20	1.9998e+05	54.86	
6.	300	2.9994e+05	60	2.9993e+05	62.34	
7.	400	3.9993e+05	70	3.9992e+05	51.65	
8.	500	4.9993e+05	70	4.9992e+05	51.35	
9.	600	5.9990e+05	100	5.9990e+05	47.15	
10.	700	6.9983e+05	170	699817	50.56	
11.	800	7.9985e+05	150	7.9985e+05	67.59	
12.	900	8.9991e+05	88	8.9991e+05	75.91	
13.	1000	9.9986e+05	140	9.9984e+05	46.27	
14.	1100	1.0998e+06	200	1099828	57.24	
15.	1200	1.1998e+06	200	1199837	41.32	
16.	1300	1.2999e+06	100	1.2998e+06	60.52	
17.	1400	1.3999e+06	100	1399907	57.52	
18.	1500	1.5000e+06	0	1.4999e+06	68.08	
19.	1600	1.5998e+06	200	1599816	62.01	
20.	1700	1.6998e+06	200	1699847	49.32	
21.	1800	1.7999e+06	100	1.7999e+06	47.77	
22.	1900	1.8998e+06	200	1.8998e+06	49.77	
23.	2000	1.9999e+06	100	1.9999e+06	38.08	
24.	2100	2.0999e+06	100	2099922	95	
25.	2200	2.1989e+06	1100	2199782	2171	
26.	2300	2.2996e+06	400	2.2998e+06	723.5	
27.	2400	2.3998e+06	200	2.3998e+06	94.02	
28.	2500	2.4996e+06	400	2.4996e+06	60.77	
29.	2600	2.5999e+06	100	2599914	78.13	
30.	2700	2.6998e+06	200	2699735	65.47	
31.	2800	2.7999e+06	100	2.7998e+06	955.20	
32.	2900	2.8998e+06	200	2.8999e+06	211.42	
33.	3000	2.9994e+06	600	2999743	1.2388e+03	
34.	3100	3.0996e+06	400	3099737	472.41	
35.	3200	3.1997e+06	300	3.1998e+06	165.30	
36.	3300	3.2995e+06	500	3299449	698.87	
37.	3400	3.3995e+06	500	3.3995e+06	100.30	
38.	3500	3.4995e+06	500	3.4995e+06	73.88	
39.	3600	3.5995e+06	500	3.5994e+06	180.61	
40.	3700	3.6995e+06	500	3699508	76.79	
41.	3800	3.7998e+06	200	3799759	96.91	
42.	3900	3.8993e+06	700	3.8993e+06	972.31	
43.	4000	3.9994e+06	600	3999386	95.22	
44.	4100	4.0994e+06	600	4099361	100.85	
45.	4200	4.1995e+06	500	4199486	56.4100	
46.	4300	4.2994e+06	600	4299428	87.89	
47.	4400	4.3990e+06	1000	4398972	199.67	
48.	4500	4.4994e+06	600	4499404	170.13	
49.	4600	4.5993e+06	700	4599319	65.14	
50.	4700	4.6993e+06	700	4699355	141.13	
51.	4800	4.7995e+06	500	4799521	128.15	
52.	4900	4.8996e+06	400	4899621	82.31	
53.	5000	4.9995e+06	490	4.9995e+06	210	

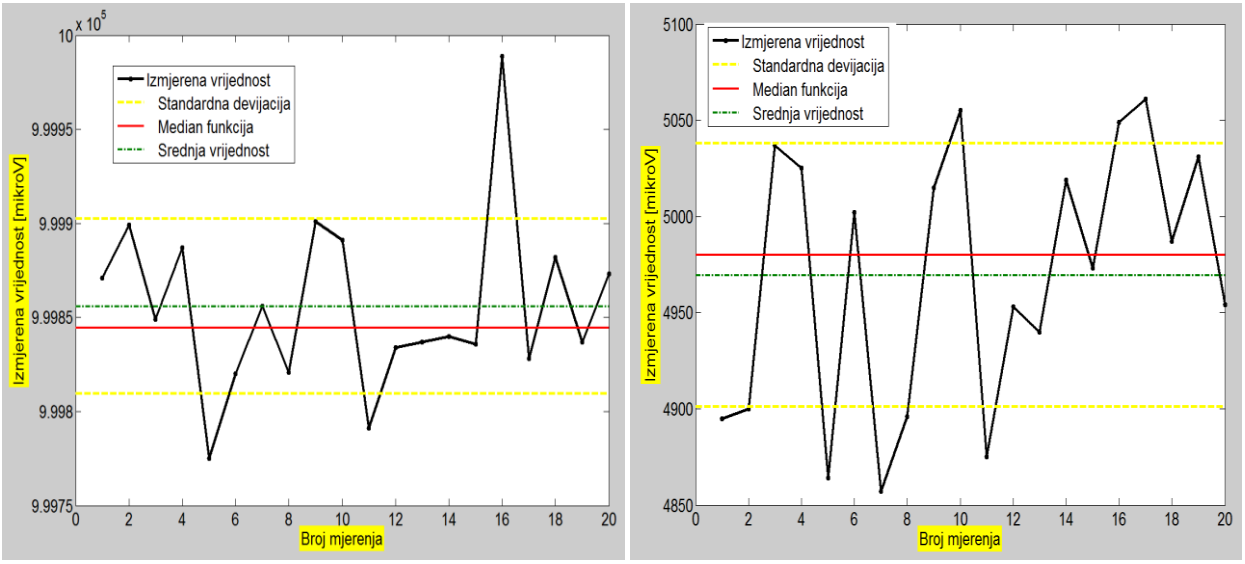
Na slikama 5.3. su prikazani dijagrami mjerenja za različite vrijednosti ulaznoga naponskoga signala. Dijagrami su napravljeni pomoću 20 uzoraka mjerenja. Možemo uočiti srednju vrijednost, standardnu devijaciju i median vrijednost izmjerenoga napona. Za 5 volti ulaznoga naponskog signala slika 5.3.a srednja vrijednost odstupa 490 µV od ulaznoga napona, a standardna devijacija iznosi 210 µV. Kod ulaznoga napona od 4 V slika 5.3.b odstupa 600 µV, a standardna devijacija 95 µV.



a) Ulazni napon 5 V b) Ulazni napon 4 V

Slika 5.3. Mjerenja 5V i 4V

Na slici 5.4.a prikazan je izmjereni napon za ulazni napon od 1 V. Na slici 5.4.b za ulazni napon od 5 mV. Vidimo da pri mjerenju 1 V šesnaesto mjerenje naglo odstupa od ostalih izmjerenih vrijednosti. Može se zaključiti da je to zbog djelovanje okoline jer su sva druga mjerenja unutar određenih granica. Takvi špicevi ulaze u računanje greške i utječu na ukupnu efektivnu rezoluciju sklopa. Za mjerena od 5 mV nema takvih špiceva tako da je i standardna devijacija dosta niska i iznosi 28 µV.



a) Ulazni napon 1 V b) Ulazni napon 5 mV

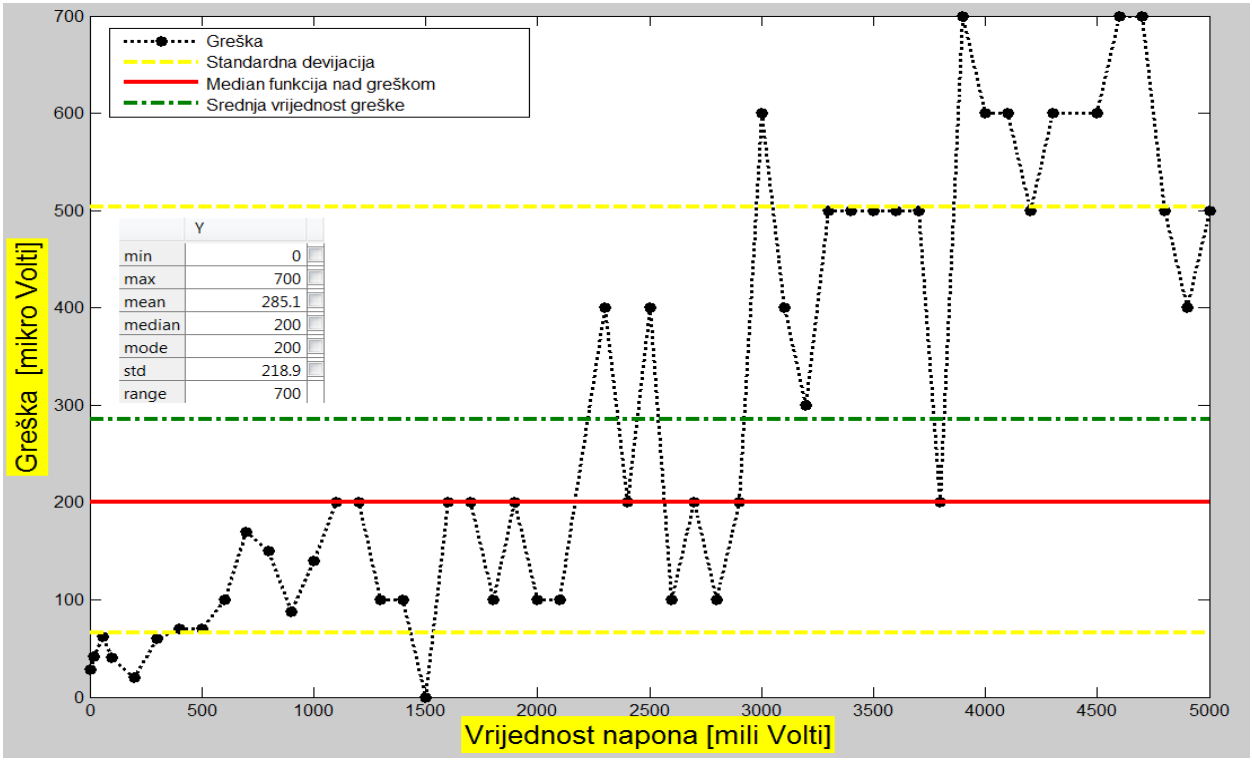
Slika 5.4. Mjerenja 5V i 4V

5.2. Vrednovanje rezultata

S obzirom na sve nelinearnosti korištenih komponentih, smetnje koje djeluju na sklop i oscilacije ulaznoga signala, padove napona do pretvornika možemo reći da greška sklopa iznosi 285 µV, maksimalna greška 700 µV, a minimalna 0 µV. Greška za cijeli raspon od 0 do 5 V iznosi 285 µV, što iznosi 57 ppm prema (5-1) formuli.

$$\frac{Greška\ u\ voltima}{raspon\ u\ voltima} = \frac{0.000285}{5} = 0.000057;\ 0.000057\ \times 1000000 = 57\ ppm \quad (5\text{-}1)$$

Na slici 5.5. prikazan je dijagram kretanja greške, vidimo da za manje napone greška je manja. Kod mjerenja 1500mV postigli smo da srednja vrijednost mjerenih vrijednosti odgovara ulaznom naponu, čime je greška nula. Standardna devijacija greške za sva mjerenja 218.9 µV.



Slika 5.5. Dijagram kretanja greške

6. ZAKLJUČAK

S obzirom na nisku cijenu sklopa reda 20$ dobili smo instrument koji točno mjeri veličine reda mili volti. Ukupna greška odstupanja od ulaznoga napona iznosi 57 ppm. Treba uzeti u obzir da je sklop napravljen s najpovoljnijim komponentama čija je nelinearnost velika. Sa stabilnim ulaznim naponom i boljim dizajnom za što je potrebno ogromno praktično znanje mogu se postići još točnija mjerenja sklopa. Naponi reda desetaka μV su toliko mali da se sami induciraju u vodovima kroz elektromagnetsko polje koje se nalazi u prostoru. Zaključujem da je jako teško iskoristiti više od 20 bita od ukupnih 24 bita koji LTC2400 daje na raspolaganje. Ovaj sklop je gotovo u svim mjerenima grešku imao u istome smjeru. Što ukazuje da uz kalibraciju koja se može napraviti programski mogao imati daleko bolju točnost.

LITERATURA

[1]
http://www.riteh.uniri.hr/zav_katd_sluz/zvd_elek/elektrotehnika/nastava/svel/ai/Download/06-ADP_I_dio.pdf (pristup ostvaren 25.05.2015.)

[2]
L. Celić, M. Đondraš, P. Gajšak, M. Vidović, M. Zorica, Povećanje razlučivosti oblikovanjem šuma kvantizacije, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, Zavod za elektroničke sustave i obradu informacija, siječanj, 2010., Zagreb

[3]
http://cds.linear.com/docs/en/datasheet/2400fa.pdf (pristup ostvaren 25.05.2015.)

[4]
http://www.atmel.com/images/doc2466.pdf (pristup ostvaren 25.05.2015.)

SAŽETAK

U ovome diplomskom radu je izveden voltmetar koji mjeri napon od 0 do 5V. Napon se utvrđuje difrencijalnom tehnikom, koja uspoređuje referentni napon s nepoznatim naponom. Instrument se sastoji od dva dijela od ADC pretvornika i mikroupravljača Atmega16. Rezultat mjerenja mikroupravljač šalje na LCD i na USB port računala pomoću UART komunikacije. Za ADC pretvorbu korišten je LTC2400 pretvornik koji ima 24 bitnu rezoluciju i radi pomoću delta sigma modulacije. Instrument realiziran je na PCB pločici s internim referentnim naponom. Analiziran je programski dio sklopa i pojedine funkcije sklopa. Utvrđenja je greška sklopa koja iznosi 57 ppm. Sklop kao takav i s relativnom niskom cijenom reda 20$ primjeniv je u niz rješenjima, npr. senzorici, komunikacijskoj tehnici, mjerenjima težine, temperature, tlaka i sličnoga.

ABSTRACT

The purpose of this Master Thesis was to design and build a voltmeter for the measurement in the range between 0 and 5V. The voltage is determined by a differential method, which compares a reference voltage with an unknown voltage. The instrument consist of two parts, an ADC converter and a microcontroller Atmega16. The resulting measurement is then sent to a PC USB port through UART communication, and shown ond the voltmeter´s display. For the AD Converter, a LTC2400 converter has been used with a 24 bit resolution and using an delta sigma moudulation. The instrument has been realised on a PCB board with internal reference voltage. Some functions and the embedded software have been analysed. The error of this instrument is 56.7 ppm. The device, with a low production cost of about 20$ is applicable in a variety of technical solutions, i. e. sensors, communication technique, measuring weight, temperature and pressure, etc.

ŽIVOTOPIS

Goran Sičanica je rođen 2.4.1988. godine u Žepču. Četiri razreda osnovne škole je pohađao u Saveznoj Republici Njemačkoj, a osnovnu školu je završio u Žepču, Bosna i Hercegovina. Nakon osnovne škole upisao je elektrotehničku školu u Žepču. Elektrotehničku školu je završio 2006. godine, te je 2008. godine upisao stručni studij računarstva na Fakultetu strojarstva i računarstva u Mostaru. Po završetku stručnog studija, upisao je razlikovnu godinu na Elektrotehničkom fakultetu u Osijeku. Završetkom razlikovne godine upisao je Diplomski studij računarstva, smjer Procesno računarstvo.

PRILOG

Izvorni kod main.c

```
#include "alati.h"
#include "lcd.h"
int n, c, k;
uint32_t d_24bit = 0;
uint32_t d_32bit = 0;
uint32_t napon;
uint16_t napon_lcd;
char string;
static FILE mystdout  = FDEV_SETUP_STREAM(uart_putchar, NULL, _FDEV_SETUP_WRITE); // uključuje printf()
int main(void)
{
        stdout = &mystdout;
        init_port();
        init_spi();
        _delay_ms(150);
        init_uart( UART_BAUD_SELECT(9600UL,8000000UL) );
        sei();
        _delay_ms(100);
        printf(" \n reset");

        while(1)
        {
                CS_LOW;

                if (!(PINB & (1 << CS)))
                {
                        d_32bit = TransferWord(0x00000000); // 32 Bit
                        d_24bit = ((((d_32bit & 0x1fffffff)  >> 4))); // 24 bit
                        napon_lcd = d_24bit  * (5000.0 / 16777216);
                        napon = d_24bit  * (5000000.0 / 16777216);

                        if ((napon > 0) && (napon < 5300000))
                        {
                                printf("\n\r>Napon= %.umV %.luuV       kvantizacijski korak: %.lu
bitovi iz LTC-a: ", napon_lcd, napon, d_24bit );
                                for (c = 24; c >= 0; c--)
                                {
                                        k = d_24bit >> c;

                                        if (k & 1)
                                        printf("1");
                                        else
                                        printf("0");
                                }
                        }
                        else if (napon > 9000000)
                        {
                                printf("\n\r pogrešan polaritet ");
                        }

                        LCD_init();

                        if (napon > 1000000)
                        {
                                LCD_goto(1,4);
                                print("napon > 1V");
                                LCD_goto(2,9);
                                number_write(napon);
                                LCD_goto(2,15);
                                print("uV");
                                LCD_goto(2,1);
                                number_write(napon_lcd);
                                LCD_goto(2,2);
                                print(".");
                                LCD_goto(2,3);
                                number_write(napon_lcd);
                                LCD_goto(2,3);
                                print(" ");
```

```
                        LCD_goto(2,7);
                        print("V");
                }
                else if (napon < 1000000)
                {
                        LCD_goto(1,4);
                        print("napon < 1V");
                        LCD_goto(2,1);
                        number_write(napon_lcd);
                        LCD_goto(2,4);
                        print("mV");
                        LCD_goto(2,9);
                        number_write(napon);
                        LCD_goto(2,15);
                        print("uV");
                }

                        CS_HIGH;
                }
                _delay_ms(650);
        }
}
int uart_putchar(char c, FILE *stream)
{
        uart_putc(c);
        return 0;
                                                }
```

Izvorni kod spi.h

```
#define CS      4
#define MOSI    5
#define MISO    6
#define SCK     7
#define CS_LOW  {PORTB &=~(1<<CS);}
#define CS_HIGH {PORTB |= (1<<CS);}

#define wait_spi()      while(!(SPSR & (1<<SPIF)))

// SPI  init
void init_port(void);
void init_spi(void);

uint32_t TransferWord(uint32_t dat);
```

Izvorni kod spi.c

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include "alati.h"

#define LOW_WORD(x)             ((uint16_t) ((x) & 0xffff))
#define HIGH_WORD(x)    ((uint16_t) ((x) >> 16))


void init_port(void)
{
        DDRB  =  (0<<MISO) | (1 << CS) | (1 << SCK);
        PORTB |= (1 << CS);
}

void init_spi(void)
{

        SPCR = (1 << SPE) | (1 << MSTR) |  (1 << SPR0)| (0<<CPOL) | (0<<CPHA);
        SPSR = 0;
        _delay_ms(100);

}

uint8_t TransferByte(uint8_t data)
```

```
{
        SPDR=data;
        wait_spi();
        return data=SPDR;

}

uint32_t TransferWord(uint32_t data)
{
        uint16_t r_L = 0;
        uint16_t r_H = 0;
        uint16_t Data_L = 0;
        uint16_t Data_H = 0;
        uint32_t data_return = 0;
        r_H = HIGH_WORD(data);
        r_L = LOW_WORD(data);

        Data_H = (TransferByte((r_H>>8) & 0x00FF))<<8;
        Data_H |= (TransferByte(r_H & 0x00FF));

        Data_L = (TransferByte((r_L>>8) & 0x00FF))<<8;
        Data_L |= (TransferByte(r_L & 0x00FF));

        data_return = ((uint32_t) Data_H << 16 ) | Data_L;


        return data_return;
}
```

Izvorni kod uart.h

```
#ifndef UART_H
#define UART_H
#define UART_BAUD_SELECT(baudRate,xtalCpu) ((xtalCpu)/((baudRate)*16l)-1)

/** Size of the circular receive buffer, must be power of 2 */
#ifndef UART_RX_BUFFER_SIZE
#define UART_RX_BUFFER_SIZE 2
#endif
/** Size of the circular transmit buffer, must be power of 2 */
#ifndef UART_TX_BUFFER_SIZE
#define UART_TX_BUFFER_SIZE 2
#endif

/* test if the size of the circular buffers fits into SRAM */
#if ( (UART_RX_BUFFER_SIZE+UART_TX_BUFFER_SIZE) >= (RAMEND-0x60 ) )
#error "size of UART_RX_BUFFER_SIZE + UART_TX_BUFFER_SIZE larger than size of SRAM"
#endif

/*
** high byte error return code of uart_getc()
*/
//#define UART_FRAME_ERROR      0x0800              /* Framing Error by UART       */
//#define UART_OVERRUN_ERROR    0x0400              /* Overrun condition by UART   */
//
#define UART_BUFFER_OVERFLOW  0x0200              /* receive ringbuffer overflow */
//
#define UART_NO_DATA          0x0100              /* no receive data available   */


/*
** function prototypes
*/

/**
   @brief   Initialize UART and set baudrate
   @param   baudrate Specify baudrate using macro UART_BAUD_SELECT()
   @return  none
*/
extern void init_uart(unsigned int baudrate);


/**
```

```
 *  @brief   Get received byte from ringbuffer
 *
 * Returns in the lower byte the received character and in the
 * higher byte the last receive error.
 * UART_NO_DATA is returned when no data is available.
 *
 *  @param   void
 *  @return  lower byte:  received byte from ringbuffer
 *  @return  higher byte: last receive status
 *           - \b 0 successfully received data from UART
 *           - \b UART_NO_DATA
 *             <br>no receive data available
 *           - \b UART_BUFFER_OVERFLOW
 *             <br>Receive ringbuffer overflow.
 *             We are not reading the receive buffer fast enough,
 *             one or more received character have been dropped
 *           - \b UART_OVERRUN_ERROR
 *             <br>Overrun condition by UART.
 *             A character already present in the UART UDR register was
 *             not read by the interrupt handler before the next character arrived,
 *             one or more received characters have been dropped.
 *           - \b UART_FRAME_ERROR
 *             <br>Framing Error by UART
 */
extern unsigned int uart_getc(void);


/**
 *  @brief   Put byte to ringbuffer for transmitting via UART
 *  @param   data byte to be transmitted
 *  @return  none
 */
extern void uart_putc(unsigned char data);


/**
 *  @brief   Put string to ringbuffer for transmitting via UART
 *
 *  The string is buffered by the uart library in a circular buffer
 *  and one character at a time is transmitted to the UART using interrupts.
 *  Blocks if it can not write the whole string into the circular buffer.
 *
 *  @param   s string to be transmitted
 *  @return  none
 */
extern void uart_puts(const char *s );


/**
 * @brief    Put string from program memory to ringbuffer for transmitting via UART.
 *
 * The string is buffered by the uart library in a circular buffer
 * and one character at a time is transmitted to the UART using interrupts.
 * Blocks if it can not write the whole string into the circular buffer.
 *
 * @param    s program memory string to be transmitted
 * @return   none
 * @see      uart_puts_P
 */
extern void uart_puts_p(const char *s );

/**
 * @brief    Macro to automatically put a string constant into program memory
 */
#define uart_puts_P(__s)       uart_puts_p(PSTR(__s))
#endif // UART_H
```

Izvorni kod uart.c

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include "uart.h"

/*
 *  constants and macros
 */

/* size of RX/TX buffers */
#define UART_RX_BUFFER_MASK ( UART_RX_BUFFER_SIZE - 1)
#define UART_TX_BUFFER_MASK ( UART_TX_BUFFER_SIZE - 1)

#if ( UART_RX_BUFFER_SIZE & UART_RX_BUFFER_MASK )
#error RX buffer size is not a power of 2
#endif
#if ( UART_TX_BUFFER_SIZE & UART_TX_BUFFER_MASK )
#error TX buffer size is not a power of 2
#endif




 #if   defined(__AVR_ATmega16__)
/* ATmega with one USART */

#define ATMEGA_USART
#define UART0_RECEIVE_INTERRUPT   USART_RXC_vect
#define UART0_TRANSMIT_INTERRUPT  USART_UDRE_vect
#define UART0_STATUS   UCSRA
#define UART0_CONTROL  UCSRB
#define UART0_DATA     UDR
#define UART0_UDRIE    UDRIE


#else
#error "no UART definition for MCU available"
#endif
/*
 *  module global variables
 */
static volatile unsigned char UART_TxBuf[UART_TX_BUFFER_SIZE];
static volatile unsigned char UART_RxBuf[UART_RX_BUFFER_SIZE];
static volatile unsigned char UART_TxHead;
static volatile unsigned char UART_TxTail;
static volatile unsigned char UART_RxHead;
static volatile unsigned char UART_RxTail;
static volatile unsigned char UART_LastRxError;



SIGNAL(UART0_RECEIVE_INTERRUPT)
/*************************************************************************
Function: UART Receive Complete interrupt
Purpose:  called when the UART has received a character
**************************************************************************/
{
    unsigned char tmphead;
    unsigned char data;
    unsigned char lastRxError;


    /* read UART status register and UART data register */

    data = UART0_DATA;

    /* */
#if defined( ATMEGA_USART )
    lastRxError = (UART0_STATUS & (_BV(FE)|_BV(DOR)) );
```

```
#endif

    /* calculate buffer index */
    tmphead = ( UART_RxHead + 1) & UART_RX_BUFFER_MASK;

    if ( tmphead == UART_RxTail ) {
        /* error: receive buffer overflow */
        lastRxError = UART_BUFFER_OVERFLOW >> 8;
    }else{
        /* store new index */
        UART_RxHead = tmphead;
        /* store received data in buffer */
        UART_RxBuf[tmphead] = data;
    }
    UART_LastRxError = lastRxError;
}


SIGNAL(UART0_TRANSMIT_INTERRUPT)
/*************************************************************************
Function: UART Data Register Empty interrupt
Purpose:  called when the UART is ready to transmit the next byte
*************************************************************************/
{
    unsigned char tmptail;


    if ( UART_TxHead != UART_TxTail) {
        /* calculate and store new buffer index */
        tmptail = (UART_TxTail + 1) & UART_TX_BUFFER_MASK;
        UART_TxTail = tmptail;
        /* get one byte from buffer and write it to UART */
        UART0_DATA = UART_TxBuf[tmptail];  /* start transmission */
    }else{
        /* tx buffer empty, disable UDRE interrupt */
        UART0_CONTROL &= ~_BV(UART0_UDRIE);
    }
}


/*************************************************************************
Function: uart_init()
Purpose:  initialize UART and set baudrate
Input:    baudrate using macro UART_BAUD_SELECT()
Returns:  none
*************************************************************************/
void init_uart(unsigned int baudrate)
{
    UART_TxHead = 0;
    UART_TxTail = 0;
    UART_RxHead = 0;
    UART_RxTail = 0;

#if  defined (ATMEGA_USART)
    /* Set baud rate */
    if ( baudrate & 0x8000 ) UART0_STATUS = (1<<U2X);  //Enable 2x speed
    UBRRH = ((unsigned char)(baudrate>>8))&0x80;
    UBRRL = (unsigned char) baudrate;

    /* Enable USART receiver and transmitter and receive complete interrupt */
    UART0_CONTROL = _BV(RXCIE)|(1<<RXEN)|(1<<TXEN);

    /* Set frame format: asynchronous, 8data, no parity, 1stop bit */
    #ifdef URSEL
    UCSRC = (1<<URSEL)|(3<<UCSZ0);
    #else
    UCSRC = (3<<UCSZ0);
    #endif

#endif

}/* uart_init */
```

```
/*************************************************************************
Function: uart_getc()
Purpose:  return byte from ringbuffer
Returns:  lower byte:  received byte from ringbuffer
          higher byte: last receive error
*************************************************************************/
unsigned int uart_getc(void)
{
    unsigned char tmptail;
    unsigned char data;


    if ( UART_RxHead == UART_RxTail ) {
        return UART_NO_DATA;   /* no data available */
    }

    /* calculate /store buffer index */
    tmptail = (UART_RxTail + 1) & UART_RX_BUFFER_MASK;
    UART_RxTail = tmptail;

    /* get data from receive buffer */
    data = UART_RxBuf[tmptail];

    return (UART_LastRxError << 8) + data;

}/* uart_getc */


/*************************************************************************
Function: uart_putc()
Purpose:  write byte to ringbuffer for transmitting via UART
Input:    byte to be transmitted
Returns:  none
*************************************************************************/
void uart_putc(unsigned char data)
{
    unsigned char tmphead;


    tmphead  = (UART_TxHead + 1) & UART_TX_BUFFER_MASK;

    while ( tmphead == UART_TxTail ){
        ;/* wait for free space in buffer */
    }

    UART_TxBuf[tmphead] = data;
    UART_TxHead = tmphead;

    /* enable UDRE interrupt */
    UART0_CONTROL    |= _BV(UART0_UDRIE);

}/* uart_putc */


/*************************************************************************
Function: uart_puts()
Purpose:  transmit string to UART
Input:    string to be transmitted
Returns:  none
*************************************************************************/
void uart_puts(const char *s )
{
    while (*s)
      uart_putc(*s++);

}/* uart_puts */
/*************************************************************************
Function: uart_puts_p()
Purpose:  transmit string from program memory to UART
Input:    program memory string to be transmitted
Returns:  none
*************************************************************************/
void uart_puts_p(const char *progmem_s )
```

```
{
    register char c;

    while ( (c = pgm_read_byte(progmem_s++)) )
      uart_putc(c);

}/* uart_puts_p */
```

Izvorni kod lcd.h

```
#include<assert.h>

#define DATA_PORT       PORTA
#define DATA_DDRDDRA
#define DATA_PINPINA
#define CNTRL_PORT      PORTC
#define CNTRL_DDR       DDRC
#define CNTRL_PIN       PINC
#define RS_PIN          0
#define RW_PIN          2
#define ENABLE_PIN      1


void send_command(unsigned char cmnd)
{
        DATA_PORT = cmnd;
        CNTRL_PORT &= ~(1<<RW_PIN);
        CNTRL_PORT &= ~(1<<RS_PIN);
        CNTRL_PORT |= (1<<ENABLE_PIN);
        _delay_us(2);
        CNTRL_PORT &= ~(1<<ENABLE_PIN);
        _delay_us(100);
}

void send_data(unsigned char data)
{
        DATA_PORT = data;
        CNTRL_PORT &= ~(1<<RW_PIN);
        CNTRL_PORT |= (1<<RS_PIN);
        CNTRL_PORT |= (1<<ENABLE_PIN);
        _delay_us(2);
        CNTRL_PORT &= ~(1<<ENABLE_PIN);
        _delay_us(100);
}

void LCD_init()
{
        CNTRL_DDR = 0xFF;
        CNTRL_PORT = 0x00;
        CNTRL_PORT = 0x00;
        DATA_DDR = 0xFF;
        DATA_PORT = 0x00;

        _delay_ms(10);
        send_command(0x38);
        send_command(0x0C);
        send_command(0x01);
        _delay_ms(10);
        send_command(0x06);
}

void LCD_goto(unsigned char y, unsigned char x)
{
        unsigned char firstAddress[] = {0x80,0xC0,0x94,0xD4};

        send_command(firstAddress[y-1] + x-1);
        _delay_ms(10);
}

void print( char *string)
{
        unsigned char i;

        while(string[i]!=0)
```

```
        {
                send_data(string[i]);
                i++;
        }
}

void number_write(uint32_t number)
{

        const int n = snprintf(NULL, 0, "%lu", number);
        assert(n > 0);
        char buf[n+1];
        int c = snprintf(buf, n+1, "%lu", number);
        assert(buf[n] == '\0');
        assert(c == n);
        print(buf);
}
```

Rezultati mjerenja

mV_5 = [4857 5002 4864 5025 5037 4900 4895 4896 5015 5055 4875 4953 4940 5019 4973 5049 5061 4987 5031 4954 4959 5066 4957 4910 5005];
mV_20 = [19899 19902 20072 19878 20070 20027 20043 20025 19886 19959 19882 19974 20014 19907 19894 19980 19993 19921 20015 19940 20015 19916 19956 19944 19873];
mV_40 = [40011 39961 40089 40020 39973 39977 40130 40057 39998 39995 40087 39953 39971 39986 39912 39965 39962 40038 39982 40066 40049 40121 39988 40046 40090];
mV_60 = [60378 60393 60515 60278 60040 60033 60057 59914 59972 60053 59892 60029 59982 59980 59924 60081 59990 60028 59967 59947 59931 60022 60032 60076 60034];
mV_100 = [100170 100168 100080 100091 100124 100194 100170 100203 100108 100065 100117 99900 100012 99821 99951 99982 99919 99928 99957 100003 99939];
mV_200 = [199976 200100 200078 200025 199949 199934 199942 200003 199978 200025 199982 200012 199979 199872 200015 199949 199958 200003 199889 199988];
mV_300 = [299967 299850 299912 299891 299824 299884 299972 300082 299876 299943 299967 299934 300052 299911 299898 300003 299943 299955 299935 299934];
mV_400 = [399913 399979 399843 399892 399883 399894 399961 399905 399883 399949 399914 399929 400055 399964 399942 399963 399979 399908 399976 399840];
mV_500 = [499985 499924 499872 499951 499934 499891 499957 499938 499779 499894 499964 499903 499905 499907 500019 499986 499900 499971 499911 499915];
mV_600 = [599953 599924 599896 599889 599897 599923 599887 599847 599913 599972 599807 599896 599941 599947 599999 599919 599845 599837 599900 599880];
mV_700 = [699805 699810 699788 699723 699785 699801 699954 699861 699851 699860 699803 699895 699818 699851 699765 699817 699869 699817 699872 699807];
mV_800 = [799949 799805 799783 799692 799795 799791 799858 799891 799875 799932 799805 799924 799895 799881 799851 799830 799822 799915 799964 799814];
mV_900 = [899742 900072 899882 899969 899917 899996 899865 899798 899845 900014 899874 899952 899912 899935 899900 899942 899961 899831 899911 899922];
mV_1000 = [999871 999899 999849 999887 999775 999820 999856 999821 999901 999891 999791 999834 999837 999840 999836 999989 999828 999882 999837 999862];
mV_1100 = [1099954 1099876 1099842 1099835 1099812 1099798 1099748 1099793 1099780 1099788 1099893 1099821 1099860 1099942 1099744 1099887 1099805 1099871 1099811 1099858];
mV_1200 = [1199774 1199837 1199846 1199900 1199906 1199849 1199784 1199774 1199855 1199886 1199871 1199821 1199821 1199880 1199837 1199804 1199837 1199895 1199817 1199795];
mV_1300 = [1299860 1299890 1299831 1299806 1299856 1299800 1299718 1299832 1299828 1299783 1299818 1299841 1299813 1299856 1299912 1299921 1299956 1299860 1299946 1299943];
mV_1400 = [1399914 1399899 1399944 1399910 1399800 1399909 1399850 1399840 1399905 1399860 1399935 1399928 1399784 1399971 1399816 1399840 1399916 1399893 1399967 1399988];
mV_1500 = [1499860 1500022 1499929 1499930 1499886 1499937 1499902 1499951 1500012 1500017 1500033 1500051 1499946 1499971 1499932 1499984 1499851 1499965 1499880 1500113];
mV_1600 = [1599826 1599905 1599749 1599739 1599927 1599855 1599785 1599756 1599882 1599910 1599795 1599848 1599796 1599806 1599771 1599730 1599865 1599842 1599843 1599727];
mV_1700 = [1699825 1699884 1699871 1699886 1699839 1699743 1699872 1699828 1699810 1699855 1699868 1699768 1699878 1699887 1699816 1699817 1699833 1699947 1699805 1699924];
mV_1800 = [1799854 1799884 1799794 1799814 1799904 1799868 1799919 1799845 1799849 1799832 1799875 1799818 1799873 1799959 1799838 1799790 1799867 1799884 1799750 1799833];
mV_1900 = [1899862 1899783 1899844 1899805 1899733 1899831 1899903 1899830 1899835 1899751 1899817 1899832 1899792 1899713 1899855 1899858 1899849 1899873 1899842 1899889];
mV_2000 = [1999861 1999904 1999861 1999955 1999921 1999861 1999876 1999925 1999929 1999899 1999952 1999863 1999892 1999966 1999976 1999862 1999891 1999951 1999898 1999895];
mV_2100 = [2099761 2099776 2099865 2099843 2099834 2099930 2099775 2099865 2099887 2099914 2099825 2099951 2099974 2100054 2099988 2099983 2099953 2100001 2100069 2100043];
mV_2200 = [2198454 2192743 2194912 2195813 2196771 2202085 2199716 2199737 2199802 2199790 2199764 2199843 2199754 2199843 2199860 2199844 2199780 2199803 2199877 2199784];
mV_2300 = [2296550 2299492 2299936 2299767 2299787 2299804 2299780 2299696 2299793 2299709 2299685 2299844 2299706 2299807 2299724 2299756 2299752 2299786 2299829 2299830];
mV_2400 = [2399765 2399740 2399679 2399753 2399743 2399736 2399809 2399856 2399592 2399832 2399885 2399770 2400002 2399891 2399799 2399908 2399816 2399872 2399831 2399936];

mV_2500 = [2499706 2499653 2499581 2499567 2499604 2499572 2499600 2499657 2499624 2499555 2499594 2499658 2499684 2499414 2499615 2499588 2499593 2499657 2499616 2499615];
mV_2600 = [2599853 2599787 2599839 2599743 2599928 2600000 2599935 2599887 2599982 2599909 2599911 2600010 2600007 2600052 2599863 2599917 2599997 2599933 2599884 2599896];
mV_2700 = [2699738 2699738 2699713 2699707 2699717 2699709 2699608 2699686 2699790 2699697 2699774 2699732 2699841 2699769 2699805 2699725 2699842 2699707 2699884 2699821];
mV_2800 = [2799671 2799806 2799796 2799895 2799918 2799988 2799832 2799740 2799843 2799894 2800004 2797349 2802644 2800777 2799243 2799496 2799079 2800957 2799730 2799886];
mV_2900 = [2899119 2899292 2899890 2899832 2899823 2899858 2899837 2899861 2899893 2899920 2899875 2899817 2899876 2899870 2899929 2899944 2899883 2899840 2899932 2899890];
mV_3000 = [2996037 3001846 2998871 2998783 2996361 2999630 2999704 2999776 2999695 2999777 2999798 2999672 2999787 2999819 2999795 2999754 2999730 2999732 2999759 2999917];
mV_3100 = [3097638 3099682 3099754 3099785 3099731 3099778 3099807 3099801 3099714 3099744 3099787 3099742 3099725 3099750 3099668 3099794 3099732 3099726 3099631 3099728];
mV_3200 = [3199884 3199192 3199449 3199446 3199691 3199547 3199788 3199668 3199783 3199778 3199812 3199682 3199694 3199757 3199800 3199798 3199746 3199784 3199800 3199732];
mV_3300 = [3297920 3302047 3299241 3299196 3299469 3299376 3299579 3299619 3299545 3299664 3299570 3299474 3299422 3299371 3299450 3299357 3299444 3299380 3299448 3299517];
mV_3400 = [3399480 3399570 3399530 3399540 3399570 3399433 3399422 3399571 3399464 3399600 3399501 3399523 3399385 3399350 3399490 3399374 3399236 3399371 3399284 3399481];
mV_3500 = [3499613 3499526 3499592 3499570 3499613 3499562 3499622 3499552 3499527 3499383 3499482 3499454 3499479 3499451 3499482 3499448 3499396 3499407 3499509 3499464];
mV_3600 = [3599438 3599489 3599446 3599441 3599410 3599299 3599309 3599369 3599407 3599364 3599368 3599350 3599936 3599461 3599840 3599864 3599530 3599490 3599520 3599466];
mV_3700 = [3699584 3699483 3699462 3699544 3699565 3699421 3699508 3699562 3699239 3699484 3699476 3699530 3699495 3699593 3699503 3699575 3699508 3699490 3699538 3699541];
mV_3800 = [3799985 3799859 3799837 3799819 3799765 3799694 3799790 3799695 3799740 3799658 3799908 3799721 3799753 3799845 3799620 3799740 3799718 3799637 3799789 3799915];
mV_3900 = [3899221 3899208 3899216 3896271 3902083 3900261 3899335 3899293 3899309 3899151 3899220 3899149 3899296 3899351 3899263 3899255 3899428 3899387 3899440 3899395];
mV_4000 = [3999616 3999367 3999397 3999458 3999445 3999250 3999369 3999375 3999321 3999158 3999375 3999312 3999421 3999369 3999283 3999482 3999447 3999417 3999443 3999402];
mV_4100 = [4099648 4099372 4099382 4099309 4099534 4099482 4099422 4099342 4099190 4099366 4099386 4099317 4099356 4099266 4099251 4099384 4099306 4099344 4099415 4099331];
mV_4200 = [4199417 4199505 4199531 4199504 4199559 4199528 4199607 4199429 4199462 4199547 4199396 4199542 4199517 4199438 4199457 4199510 4199416 4199459 4199468 4199443];
mV_4300 = [4299536 4299408 4299429 4299454 4299434 4299361 4299428 4299430 4299404 4299281 4299249 4299316 4299460 4299428 4299561 4299310 4299338 4299349 4299430 4299579];
mV_4400 = [4399137 4399133 4398968 4398976 4399001 4399016 4398993 4399065 4398987 4398936 4399011 4398944 4398942 4398911 4398847 4398942 4398811 4398899 4398832 4399772];
mV_4500 = [4499240 4499758 4499268 4499885 4499237 4499158 4499422 4499358 4499422 4499471 4499459 4499367 4499481 4499202 4499408 4499402 4499382 4499379 4499406 4499409];
mV_4600 = [4599415 4599285 4599380 4599250 4599223 4599251 4599261 4599279 4599378 4599364 4599258 4599300 4599462 4599322 4599316 4599297 4599393 4599353 4599396 4599352];
mV_4700 = [4699530 4699189 4699430 4699363 4699397 4699283 4699547 4699442 4699269 4699580 4699397 4699492 4699138 4699300 4699347 4699138 4699134 4699167 4699264 4699369];
mV_4800 = [4799154 4799150 4799555 4799558 4799519 4799594 4799496 4799523 4799500 4799525 4799502 4799589 4799596 4799596 4799596 4799494 4799471 4799476 4799621 4799462];
mV_4900 = [4899558 4899356 4899566 4899629 4899623 4899629 4899581 4899513 4899567 4899585 4899556 4899595 4899733 4899620 4899652 4899692 4899736 4899649 4899622 4899654];
mV_5000 = [4999938 4999816 4999890 4999820 4999742 4999530 4999502 4999497 4999450 4999425 4999319 4999328 4999428 4999359 4999479 4999384 4999431 4999421 4999267 4999374];