UNIVERZA V LJUBLJANI

Fakulteta za elektrotehniko

5 MESTNI LED PRIKAZOVALNIK PREJETIH ŠTEVIL PREKO UART RS232 KOMUNIKACIJE

(Seminar)

Avtor: Vito Rome

Mentor: doc. dr. Marko Jankovec

Ljubljana, februar 2013

Kazalo vsebine

Ključne besede

LED prikazovalnik, RS232 komunikacija, UART, ASCII, ATxmega128-A3U, PWM, LM1117

Uvod

Cilj mojega seminarja je bilo prikazovanje teže izmerjene na tehtnici DIGI DS-781 na mnogo večjem 5 mestnem LED prikazovalniku. Pri tem sem moral spoznati družino mikroprocesorjev AVR XMEGA in programiranje z ASF v Atmel Studiu, logiko prikazovanja na LED prikazovalniku, osnovno komunikacijo UART preko RS232 izhoda ter načrtovanjem vezij v Alitum Designerju. Programiranje samega mikroprocesorja sem izvedel preko JTAG priključka in s pomočjo AVR Dragona. Sprva sem nameraval dodati USB priključek za kasnejšo lažjo nadgradnjo programske opreme, vendar sem že brez tega porabil vseh 64 pinov na mikroprocesorju. Vezje je dokaj veliko, saj mora biti dovolj prostora za fikso pritrditev LED prikazovalnikov. Uporabil sem 9V napajanje z nestandardnim 5pin DIN priključkom zaradi lažjega razločevanja z drugimi napajalniki. Možnost nastavljanja svetilnosti sem izvedel na eni strani s preprostim napetostnim delilnikom s potenciometrom na vhodu AD pretvornika. Na drugi strani pa z delovnim ciklom PWM, ki je odpiral glavni p-kanalni MOSFET tranzistor.

Specifikacije naprave

V bistvu čisto preproste zahteve:

- naprava mora prepoznati število poslano preko RS232 komunikacije iz tehtnice DIGI DS-781 in ga prikazati na 5 segmentnem LED prikazovalniku
- svetilnost LED prikazovalnika naj bo nastavljiva.
- napajanje naj bo zunanje z nestandardnim priključkom

Deluje pa tudi na katerikoli drugi napravi. Testna verzija je delovala v povezavi z osebnim računalnikom preko HyperTerminala oz. podobnega programa. Število je lahko največ 5 mestno, v primeru da je negativno pa 4 mestno, minus se mora pokazati. Decimalna pika ni potreba, čeprav sem jo skril v programsko kodo in se jo lahko ob nadgradnji aktivira. Prikazuje naj se torej teža kar v gramih in ne kilogramih. Tehtnica oddaja 7 mestno število oblike ''012.345''. Prvi znak v nizu je 0, saj je maksimalna dopustna teža 14999g, torej

''014.999''. V kolikor je število manj kot 5 mestno, naj se prednje ničle ne izpisujejo npr. ''000.045'' naj se prikaže kot '' 45'', oz. ''0-0.345'' kot ''- 345''.

Časovni in finančni plan

Z načrtovanjem tiskanih vezij sem se poglobljeno srečal šele drugič, s samim načrtovalskim orodjem Altium Designer pa prvič in zato mi je to vzelo največ časa, kar dobrih 14 dni dan in noč. Tiskanina je bila poslana in narejena pri zunanjem profi izvajalcu TIV. Spajkanje mi je vzelo 2 dni in potem še tretji dan za odpravljanje napak. Programiranje z ASF v Atmel Studiu mi je vzelo slab teden dni. Verjetno bi sprogramiral prej, v kolikor bi prej odkril napake na tiskanini, ki so izvirale še iz slabega načrtovanja. Skupno sem v seminar vložil preko 175h in se pri tem ogromno naučil. Ocenjujem, da bi zverziran elektronik za celotno izvedbo potreboval približno četrtino mojega časa, torej okoli 45h oz. 1 teden.

Še najdražji so bili LED prikazovalniki, kar 50% cene materiala oz. 90€. Nato je sledil poseben napajalnik za 33% cene oz. 60€. Vsi ostali elementi (uC, R, C, MOSFET, konektorji, potenciometer) so bili 17% cene oz. 30€. Izdelava TIV in ohišja ni bila v moji domeni, zato ne morem podati končnih skupnih stroškov, ocenjujem pa nekje na 300€.

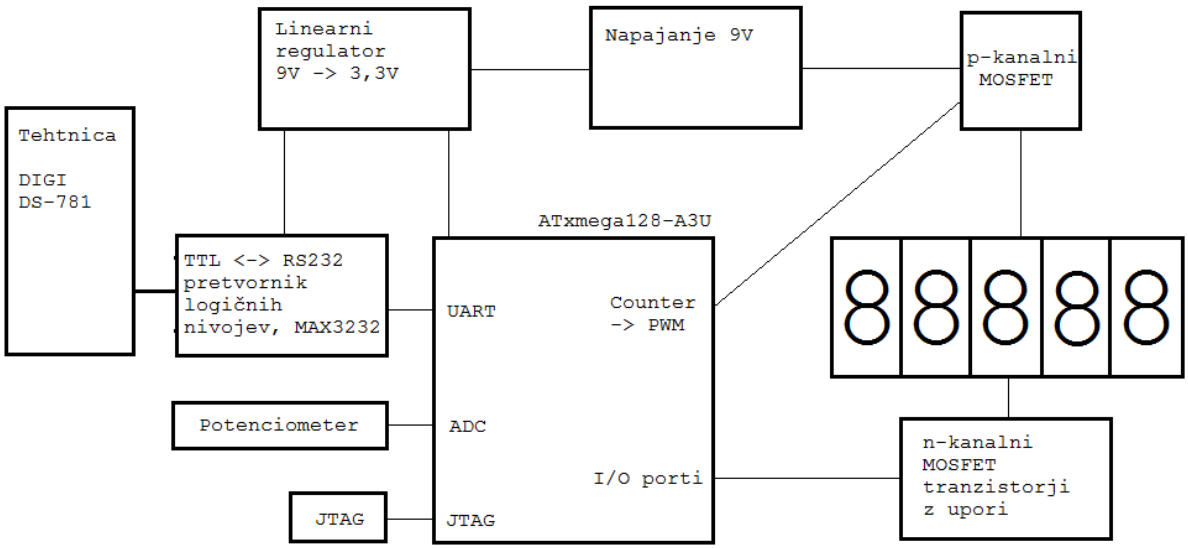
Tabela 1: kosovnica

komentar	opis	oznaka	naročniška št. farnell	kos	cena/kos	cena	%
Jumper			9733302	1	0,27	0,27	0,2
C+	Tantal	C1, C2	197518, 197520	2	0,79 + 1,4	2,19	1,2
C	Capacitor	C3-C14	1301790	12	0,29	3,48	1,9
5 pin DIN	Deltron 180	CON1	9175814	1	2,47	2,47	1,4
DB9M	AMP, D Sub	DB9M	8391327	1	2,02	2,02	1,1
Dpy Red-CA	SA40-19SRWA	DS1-DS5	1168647	5	17,85	89,25	49,8
Header 5X2	Header, 5-Pin, Dual row	JTAG	1248161	1	0,68	0,68	0,4
LM1117MP-3.3	LM1117-3V3	LM1117	9778195	1	0,92	0,92	0,5
FUSE	PTC Resettable Fuse	MF-USMF010	1294882	1	0,55	0,55	0,3
NX3008NBKW	N-Channel MOSFET	N	2069546	40	0,069	2,76	1,5
	N-Channel MOSFET	NPWM	2069546	1	0,069	0,069	0,0
P-MOSFET	P-Channel MOSFET	PMN48XP	1894623	1	0,26	0,26	0,1
1k	POTENTIOMETER	POT	9609512	1	8,19	8,19	4,6
R	Resistor	R1	1469860	1	0,025	0,025	0,0
590	Resistor	R2	1653021	1	0,079	0,079	0,0
1k	Resistor	R3	1652936	1	0,083	0,083	0,0
35R7	Resistor	R	2138792	35	0,008	0,28	0,2
255R	Resistor	R	2117098	5	0,26	1,3	0,7
ATxmega128A3-AU	8/16-bit AVR XMEGA	U1	2066298	1	3,54	3,54	2,0
MAX3232CSE+	True RS-232 Transceiver	U2	9725490	1	0,45	0,45	0,3
12V Zener	Zener Diode	Z	1798002	1	0,52	0,52	0,3
Napajalnik 9V	PCM50US09		1109890	1	59,8	59,8	33,4
					skupni znesekl:	**179,19**	**€**

Zasnova naprave

Blok shema

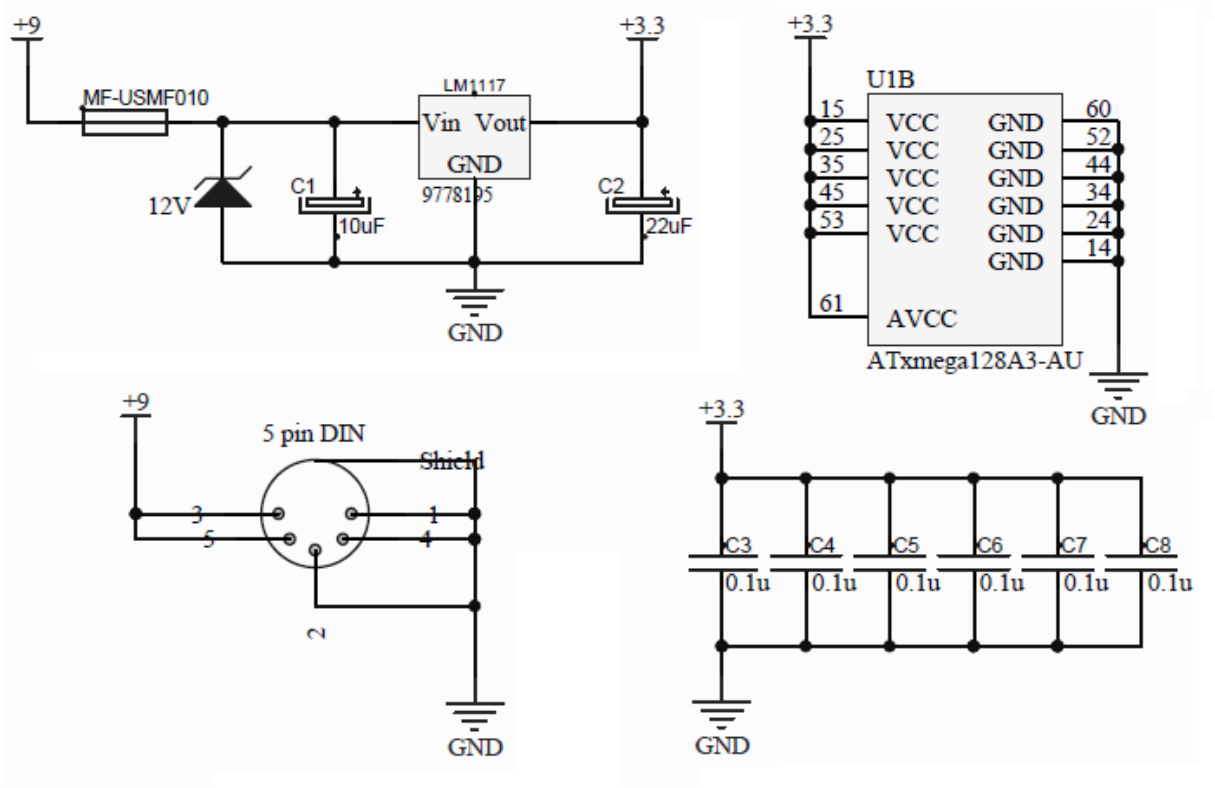
Slika 1: blok shema



Napajalni del

Zunanje napajanje 9V rabimo zaradi velikega padca napetosti na LED prikazovalnikih, in sicer kar 7,4V pri toku skozi ledice 40mA. Ostanek padca napetosti je na uporih, manj kot 0,1V pa na p-kanalnem in n-kanalnih MOSFET tranzistorjih. Mikrokrmilniški del deluje na napetosti 3,3V, ki jo zagotavlja linearni regulator LM1117 v prenapetostni vezavi z zener diodo in 100mA PTC varovalko. Uporabil sem poseben 5 pinski DIN konektor, ki se sicer uporablja v avdio tehniki, s pini 1,2 in 4 na masi in s pinoma 3 ter 5 na 9V. Na vseh napetostnih vhodih mikrokrmilnika ATxmega128-A3U so blokirni kondenzatorji 100nF, pravtako na pretvorniku napetostnih logičnih nivojev MAX3232.

Slika 2: napajalni del



LED prikazovalnik

Posamezni LED prikazovalnik s skupno anodo (skupen plus) je sestavljen iz 7 črt in pike. Zaradi kasnejšega lažjega programiranja, je v tabeli prikazana binarna razporeditev prižganih pinov za določeno števko in HEX zapis, ki potem pri programiranju lahko služi kar maski, da ne vklapljamo vsakega pina posebej. Vendar mi je to uspelo le pri treh digitih, da sem jih zvezal v celoti na posamezne porte. Dva digita pa sem moral sprogramirati pin za pinom, saj so komunikacija, ADC in JTAG zasedli vmesne pine na portu. Pike sem posebej sprogramiral.

Slika 3: en digit

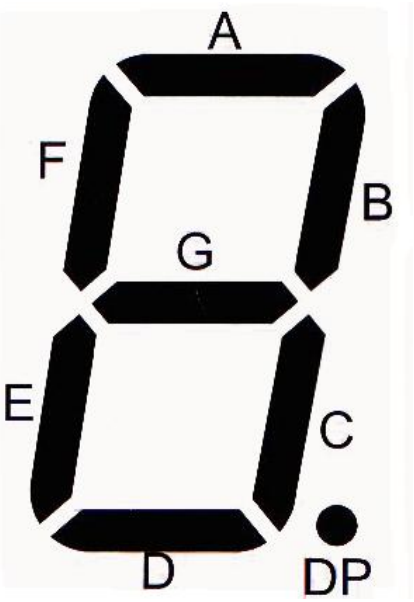
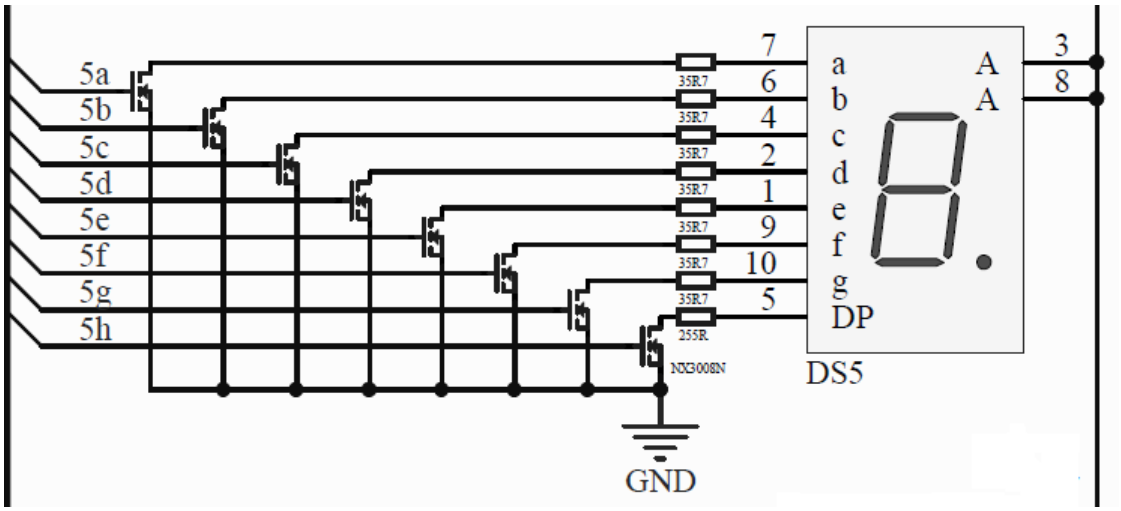


Tabela 2: binarna razporeditev pinov za prikaz določene števke in njen heksadecimalni zapis

	0	**G**	**F**	**E**	**D**	**C**	**B**	**A**	**HEX**
0	0	0	1	1	1	1	1	1	**3F**
1	0	0	0	0	0	1	1	0	**06**
2	0	1	0	1	1	0	1	1	**5B**
3	0	1	0	0	1	1	1	1	**4F**
4	0	1	1	0	0	1	1	0	**66**
5	0	1	1	0	1	1	0	1	**6D**
6	0	1	1	1	1	1	0	1	**7D**
7	0	0	0	0	0	1	1	1	**07**
8	0	1	1	1	1	1	1	1	**7F**
9	0	1	1	0	1	1	1	1	**6F**

Delovno točko 40mA sem določil glede na srednje vrednosti iz specifikacij, iz grafa pa razbral delovno napetost 7,4V. Nato preračunal dodatne upore 35Ω, v končni izvedbi 35,7Ω, za posamezno črto. Za piko pa je potreben upor 255Ω, v končni izvedbi 260Ω. Pika je namreč narejena iz 2 led diod vezanih zaporedno, črta pa iz 8 diod po 4 in 4 vzporedno.

Slika 4: shema vezave posameznega digita

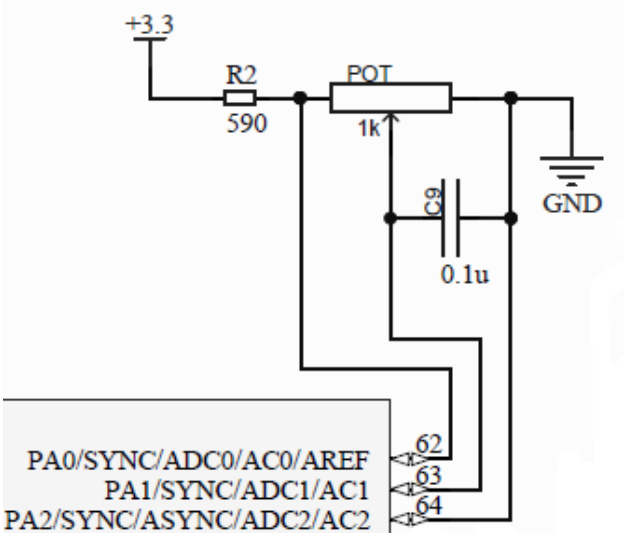


Kjer sem lahko uporabil cel port za nastavitev I/O pinov kot je prikazano na shemi, je bilo programiranje lažje, saj sem uporabil maske iz zgornje tabele.

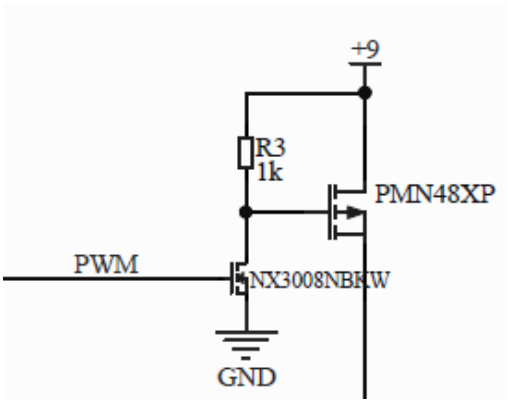
Uravnavanje svetlosti

Možnost nastavljanja svetlosti LED prikazovalnikov sem izvedel na eni strani s preprostim uporovnim delilnikom z 0-1kΩ potenciometrom na vhodu AD pretvornika. Referenčna napetost AREFA je 2,075V, kar ustreza optimalnemu delovanju analogno digitalnega pretvornika, ki naj bi bila okoli 3,3/1,6 = 2,06V. Za vsak slučaj, sem uporabil zunanjo maso in ne notranje mase mikrokrmilnika, vendar pa sem potem moral uporabiti diferencialno metodo zajema signala. Izbral sem 8-bitni diferencialni predznačeni ADCA, kar mi je po pretvorbi dalo vrednosti med 0 in 127. V kolikor bi uporabil notranjo maso in enojni nepredznačeni ADCA, bi bile vrednosti med 0 in 255. Te vrednosti sem potem malce skaliral, da sem dobil vrednosti med 1 in 100, ter jih na drugi strani uporabil za delovni cikel PWM, ki je posredno preko n-kanalnega odpiral glavni močni 4 amperski p-kanalni MOSFET tranzistor. Le-ta pa je skrbel za napajanje vseh petih LED prikazovalnikov in bi ob polni obremenitvi ''8.8.8.8.8.'' moral odpreti 1,5A toka. Med polnim delovanjem se zaradi predimenzioniranosti tako p-kanalni MOSFET sploh ni grel.

Slika 5: vezava potenciometra na ADCA



Slika 6: odpiranje glavnega p-kanalnega MOSFETa s PWM signalom



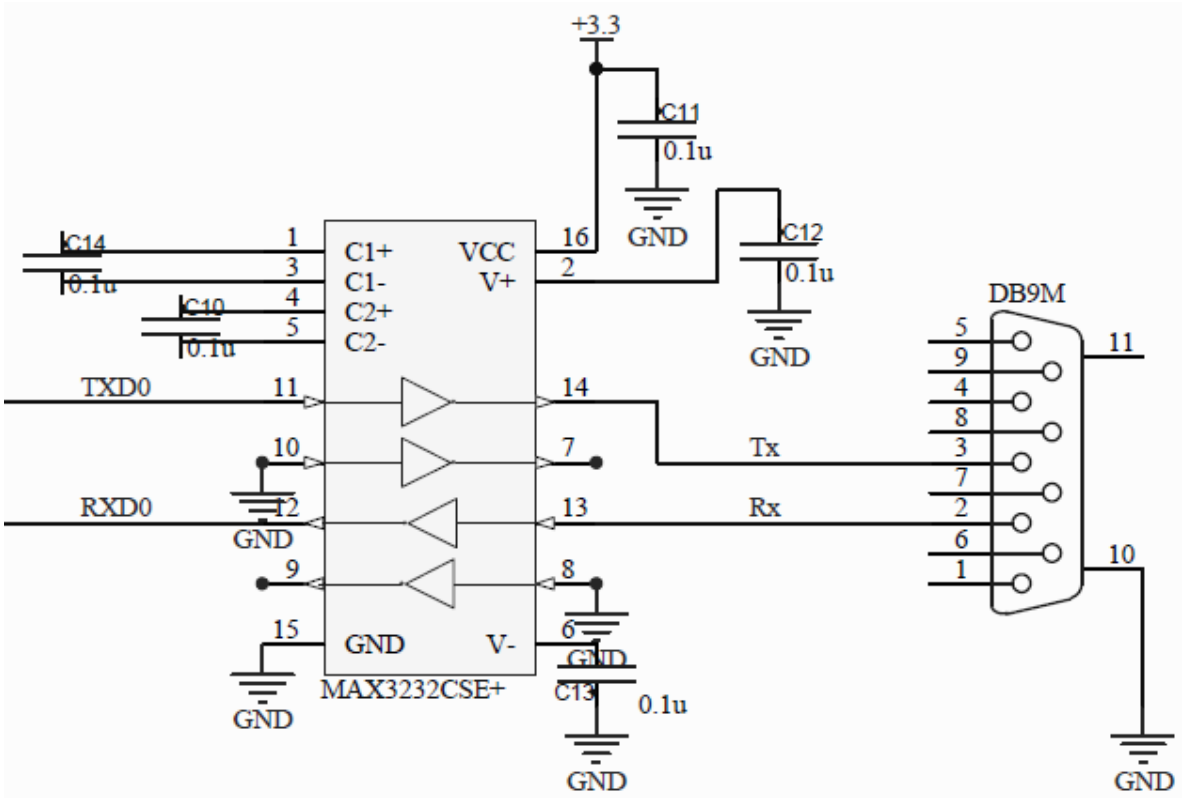
UART komunikacija

Za pretvorbo napetostnih nivojev podatkovnih linij TxD in RxD z obrnjeno polariteto, torej logične ničle na +15V in logične enice na -15V, ki so potrebni za UART prenos preko RS232 komunikacije, poskrbi MAX3232. Veljavni signali morajo biti med +3V in +15V oz. -3V in -15V, maksimalno do +-25V. Kontrolni signali CTS, RTS, ipd. pa imajo navadno polariteto. V osnovi protokol serijske komunikacije poteka tako, da start bit predstavlja začetek komunikacije, potem sledijo podatkovni biti in na koncu še stop bit ali več stop bitov. Potem pa lahko dodamo še sodi oz. lihi paritetni bit za detekcijo napake prenosa na podlagi lihih ali sodih seštevkov vseh podatkovnih bitov.

Slika 7: tipična ponazoritev UART komunikacije, start bit, data bajt, parity bit, 2 stop bita

Sam sem uporabil klasično hitrost 9600 Baud rate, 8 bitno dolžino, 1 stop bit ter brez paritete. Ker je v Atmel Studiu v ASF zbirki sama UART komunikacija že sprogramirana in jo je potrebno le pravilno inicializirati, se raje posvetimo pošiljanju znakov.

Slika 8: vezava MAX3232 na mikrokrmilnik in moški DB9



Prenosno kodo za pošiljanje znakov bom najlažje upodobil s heksadecimalnim zapisom ASCII kode, kjer števke od 0 do 9 predstavlja ASCII koda od 0x30 do 0x39. Samo logiko delovanja pa si poglejmo na starinskih pisalnih strojih. Ročica CR (carriage return) je na začetek papirja prestavila pisalno glavo. Tipka LF (line feed) pa je prestavila list za eno vrstico naprej. CR LF bi torej naredil oboje, prešel v novo vrstico in premaknil pisalno glavo čisto na levo. Tu je logika ista, CR LF oz. 0x0D0A pomeni konec podatkovnih bajtov. Za začetek podatkovnega niza pa se uporabi samo CR oz. 0x0D. Prepoznati oz. uloviti moramo torej začetni bajt 0x0D in nato vse podatkovne. Ker pa sta prvi in predzadnji bajt enaka, 0x0D, je potrebna previdnost pri lovljenju. Sprva sem malce nerodno sestavil program in mi je vsakič nehote uspelo uloviti le predzadnjega. Nato sem lovil vlak bajtov, ki se je začel z 0x0D in nadaljeval z 0x30 in je lepo delalo. V mojem primeru je prenosna koda, ki jo je pošiljala tehtnica izgledala nekako takole:

hex	0x40	0x40	0x0D	0x30	0x31	0x32	0x2E	0x33	0x34	0x35	0x0D	0x0A
znak	@	@	CR	0	1	2	.	3	4	5	CR	LF

@@ je bil odvisen od načina tehtanja, in se je spreminjal glede na nastavitve tehtanja (neto, bruto, tara, hitrost spreminjanja, negativno vrednost, ipd.), tako da se tema dvema bajtoma nisem posvečal.

Izločiti sem moral prvo ničlo in piko ter tako dobil število, ki me je zanimalo, npr. ''12345''. V kolikor je bilo število negativno, je bil drugi podatkovni bajt enak 2D, kar ustreza minusu v hex ASCII. Ker je 0x30 desetiško ravno 48, sem moral nato od ASCII vrednosti odšteti 48, da sem dobil posamezne števke, npr. 0x32 – 48 = 2.

Program

Glavni program -> inicializacija uC in vse periferije -> neskončna zanka

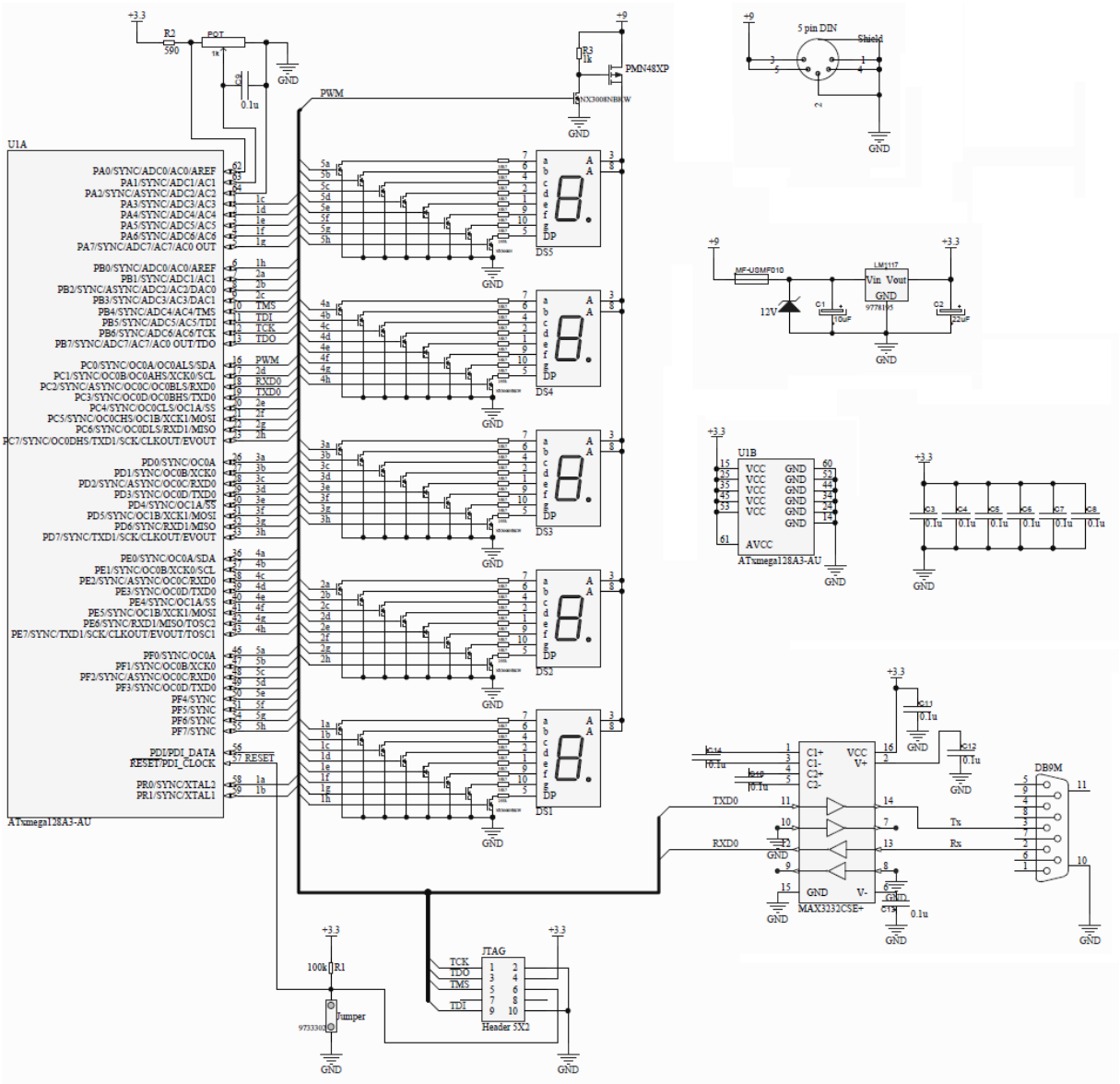
UART komunikacija deluje s prekinitvami in ima prednost pred ostalim izvajanjem. Med posameznimi vlaki podatkovnih bajtov se izvede preostala koda. ADC prožimo s frekvenco vmed 100Hz do 200Hz. Izmerjeno vrednost skaliramo in z njo nastavimo PWM. Nazadnje osvežimo še LED prikazovalnike.

Errata

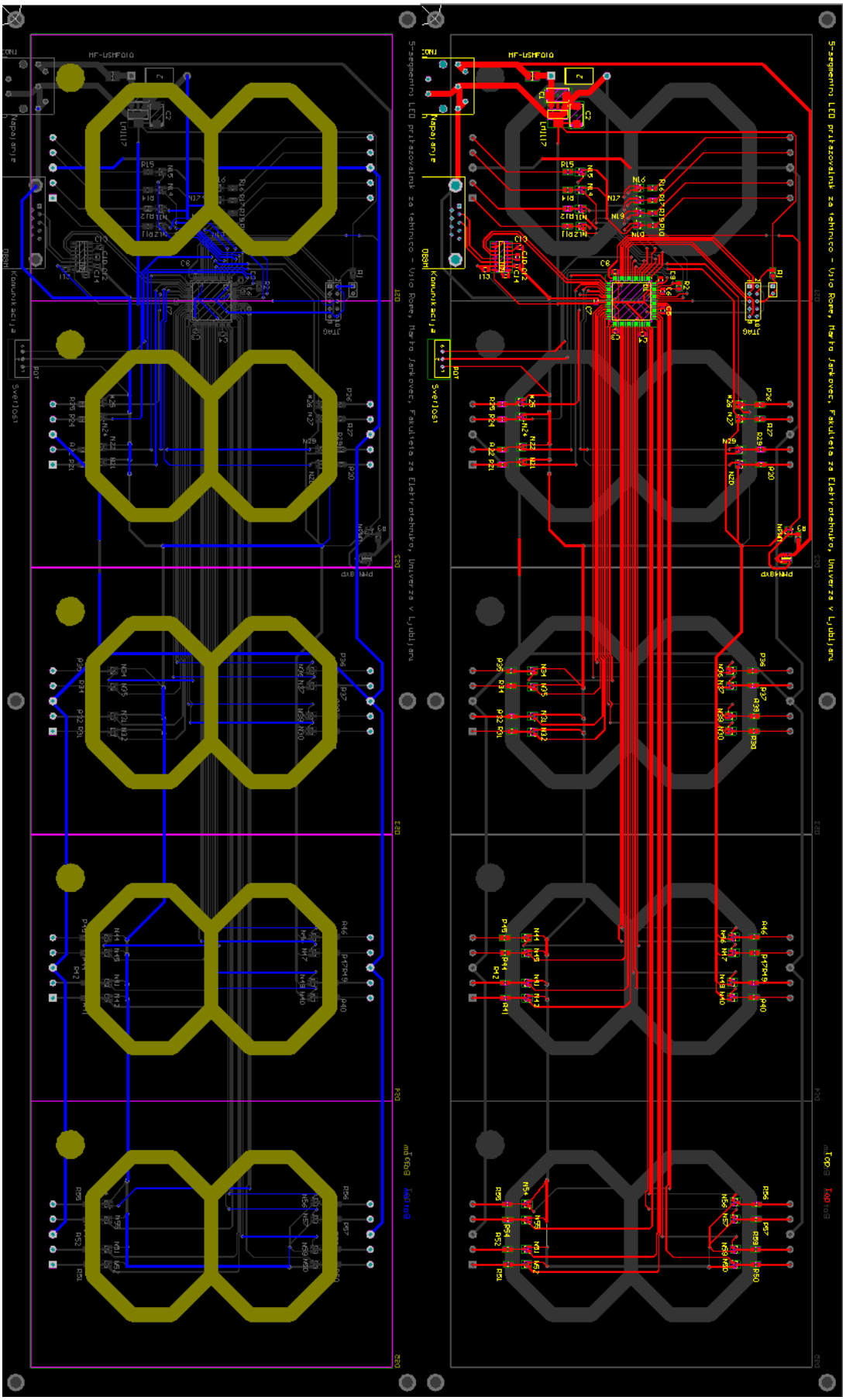
Zaradi nespretnega načrtovanja vezja sem naredil kar nekaj začetniških napak, k sreči se je vse dalo s spretnim spajkanjem rešiti. Za dobro se je izkazala zaščita na napajanju. Predno sem ugotovil, čemu mi LED prikazovalniki ne delajo in čemu se linearni regulator skoraj kuri, sem že skuril varovalko. Zamaknil sem namreč pinout tako n kot p MOSFETov in sem jih potem moral spajkati postrani, p-kanalnega pa celo malce premostiti. Na ploščo sem tudi nespretno postavil večje trough-hole elemente in so mi bili v napoto na drugi strani, kjer so bili prispajkani vsi LED prikazovalniki. Naročil sem tudi napačen DIN konektor in pravtako narobe načrtal footprint, vendar se je s novim konektorjem vseeno dalo postrani zadeti luknje in prispajkati. Na koncu se je vse rešilo in je delovalo kot mora. Po končani izvedbi sem v načrtu tiskanine in kosovnici vse te napake popravil, pravtako sem konektorje v načrtu prestavil bolj skupaj, tako tudi kontakti na spodnji strani niso v napoto.

Shema vezja in tiskanega vezja

Slika 8: shema vezja



Slika 9: zgornja in spodnja plast tiskanega vezja



Literatura

Nepogrešljivi vodnik skozi načrtovanje v Altium Designerju, uporabnik KonstruiranjeEnaprav
http://www.youtube.com/feed/UCPxkrupGOdz3Hy2zEZmUiLQ

Vse podrobnosti o delovanju ATxmega128A3U in njegove periferije
http://www.atmel.com/devices/ATXMEGA128A3U.aspx?tab=documents

manual: http://www.atmel.com/Images/Atmel-8331-8-and-16-bit-AVR-Microcontroller-XMEGA-AU_Manual.pdf
kratek manual: http://www.atmel.com/Images/doc8386.pdf

AVR USART: http://www.atmel.com/Images/doc8049.pdf
AVR ADC: http://www.atmel.com/Images/doc8032.pdf
AVR PWM: http://www.atmel.com/Images/doc8045.pdf
AVR USB: http://www.atmel.com/Images/doc8388.pdf
AVR Getting started (JTAG): http://www.atmel.com/Images/doc8169.pdf

RS-232 komunikacija
http://en.wikipedia.org/wiki/RS-232

ASCII
http://en.wikipedia.org/wiki/ASCII

C++ programski jezik
http://msdn.microsoft.com/en-us/library/dtefa218(v=vs.71).aspx

Atmel Software Framewrok (ASF) driverji, komponente, get started
http://asf.atmel.com/docs/latest/get_started.html