Univerza v Ljubljani

Fakulteta za elektrotehniko

Sašo Rudolf

Razširitvena plošča

za razvojni sistem *DE0-Nano*

Magistrsko delo

Mentor: izr. prof. dr. Andrej Trost

Ljubljana, 2015

Zahvala

Rad bi se zahvalil svojemu mentorju izr. prof. dr. Andreju Trostu za zaupanje pri dodelitvi teme, za vso izkazano pomoč, znanje, nasvete in potrpežljivost. Poleg tega se mu zahvaljujem za vse ure njegovega prostega časa, ki jih je posvetil mojim vprašanjem, za usmerjanje in podporo skozi celotno obdobje nastajanja magistrskega dela.

Zahvalil bi se dipl. inž. el. Danetu Seligerju za strokovne nasvete pri nastajanju magistrskega dela in za pomoč pri izdelavi tiskanih vezij.

Posebna zahvala gre mojemu očetu, ki mi je tekom šolanja in v zasebnem življenju vedno stal ob strani, me spodbujal in zagotavljal tako finančno kot tudi moralno podporo. Z njegovo pomočjo sem uspel doseči svoj pomemben življenjski cilj. Pot do tega bi bila brez njega zagotovo veliko težja. Hvala.

Babici Ivani posvečam zahvalo za vso vlito življenjsko voljo, skrbnost in modre nasvete.

Zahvala gre tudi dekletu Nini za potrpežljivost v času šolanja.

Vsebina

1 Uvod **5**

2 Razvojna plošča *DE0-Nano* **7**

3 Načrtovanje vezja **11**

3.1 Matrika LED (angl. *LED matrix*) 12

3.2 Drsnik (angl. *Slider*) 18

3.3 Pretvornik iz USB 2.0 v protokol UART (angl. *USB 2.0 to UART protocol converter*) 23

3.4 Gonilnik motorja (angl. *Motor driver*) 26

3.5 VGA vmesnik (angl. *Video Graphics Array*) 29

3.6 Rotacijski kodirnik (angl. *Rotary encoder*) 33

3.7 PWM izhod (angl. *Pulse-Width Modulation*) 36

3.8 Tipke (angl. *Keys*) 38

3.9 Analogni vhodi (angl. *Analog input*) 39

3.10 Razdeljevalnik (angl. *Multiplexer*) 41

3.11 Združitev komponent in kompenzacija motenj 45

4 Izdelava tiskanega vezja **49**

4.1 Pravila za risanje tiskanega vezja 49

4.2 Risanje tiskanine 51

4.3 Izdelava prototipne plošče (angl. *protoboard*) 55

4.4 Spajkanje (angl. *Soldering*) 60

5 Programiranje komponent **67**

5.1 Razdeljevalnik (angl. *Demultiplexer*) 67

5.1.1 Opis razdeljevalnika ... 69
5.1.2 Delovanje razdeljevalnika ... 72
5.1.3 Primer serijskega prenosa podatkov v pomikalna registra matrike LED ... 76
5.1.4 Nalaganje programske kode na CPLD ... 78
5.2 Izbiralnik (angl. *Multiplexer*) ... 81
5.2.1 Delovanje izbiralnika ... 82
5.2.2 Časovni potek signalov med izbiralnikom in razdeljevalnikom ... 84
5.2.3 Merjenje zakasnitve signalov z osciloskopom na glavnem priključku ... 86
5.3 Matrika LED (angl. *LED matrix*) ... 89
5.3.1 Pomnilnik znakov (*character_rom.vhd*) ... 89
5.3.2 Uporabniška logika matrike LED (*ledmatrix.vhd*) ... 91
5.3.3 Delovanje UL matrike LED ... 91
5.4 Drsnik (angl. *Slider*) ... 94
5.4.1 *I2C master* (*i2c_master.vhd*) ... 94
5.4.2 Delovanje *I2C master* ... 96
5.4.3 Simulacija kode *I2C master* – primer prenosa ... 99
5.4.4 Nastavitev hitrost serijske ure *SCL* (angl. *Serial Clock Line*) ... 100
5.4.5 Uporabniška logika drsnika (*slider.vhd*) ... 100
5.4.6 Opis UL drsnika ... 101
5.4.7 Delovanje UL drsnika ... 102
5.4.8 Vodilo ***I2C*** ... 106
5.4.9 Opis registrov integriranega vezja drsnika *MPR083* ... 107
5.4.10 Napake drsnika ... 112
5.5 Nastavitev UART integriranega vezja *MCP2200* ... 113

6 Sklepne ugotovitve **115**

Literatura **117**

Viri slik **121**

Seznam slik

Slika 2.1: Blok diagram razvojne plošče *DE0-Nano*..... 8
Slika 2.2: Razvojna plošča *DE0-Nano* z zgornje stran..... 9
Slika 2.3: Razvojna plošča *DE0-Nano* s spodnje strani..... 9
Slika 3.1: Primer matrike LED 5 x 7..... 12
Slika 3.2: Matrika LED 5 x 7: a) vrstice s skupno anodo b) vrstice s skupno katodo..... 12
Slika 3.3: Funkcijski diagram pomikalnega registra *74HC595*..... 13
Slika 3.4: Električna shema matrike LED; napetostno krmiljenje..... 14
Slika 3.5: Statična karakteristika za matriko LED *TC07-11SURKWA*; a) odvisnost toka od napetosti b) odvisnost svetilnosti od toka..... 15
Slika 3.6: Elementi matrike LED na razširitveni plošči..... 16
Slika 3.7: Električna shema matrike LED; tokovno krmiljenje..... 17
Slika 3.8: Delitev senzorjev dotikov; a) nič-dimenzijski, b) enodimenzijski vzdolžni, c) enodimenzijski krožni, in d) dvodimenzijski..... 18
Slika 3.9: Model kondenzatorja..... 19
Slika 3.10: Primer povezav elektrod; a) nepravilna b) pravilna..... 20
Slika 3.11: Funkcijski blok diagram za integrirano vezje *Freescale MPR083*... 21
Slika 3.12: Električna shema za drsnik..... 22
Slika 3.13: Območje drsnika na zgornji in območje elektrod na spodnji strani razširitvene plošče..... 22
Slika 3.14: Shema UART komunikacije..... 23
Slika 3.15: Blok diagram; integrirano vezje *MCP2200*..... 24
Slika 3.16: Električna shema pretvornika z USB v UART..... 25
Slika 3.17: Elementi pretvornika USB v UART na razširitveni plošči..... 25

Slika 3.18: Blok diagram integriranega vezja *LV8548MC*; a) vezava dveh DC motorjev, b) vezava koračnega motorja..... 26
Slika 3.19: Primera načina krmiljenja koračnega motorja; a) polni in b) polovični..... 27
Slika 3.20: Električna shema gonilnika motorja..... 28
Slika 3.21: Napajalna priključka gonilnika motorja in mere vtičnice..... 28
Slika 3.22: Napajalni priključek in izhodi gonilnika motorja na razširitveni plošči..... 28
Slika 3.23: Priključka VGA..... 29
Slika 3.24: Električna shema DA pretvornika; a) uporovni delilnik, b) nadomestno vezje..... 30
Slika 3.25: Priključek in signali VGA vmesnika na razširitveni plošči..... 32
Slika 3.26: Inkrementalni rotacijski kodirnik..... 33
Slika 3.27: Vezje inkrementalnega rotacijskega kodirnika..... 34
Slika 3.28: Časovni potek signala *A* in *B* ob vrtenju osi inkrementalnega kodirnika v smeri urinega kazalca..... 34
Slika 3.29: Plošča z inkrementalnim rotacijskim kodirnikom..... 35
Slika 3.30: Električna shema inkrementalnega rotacijskega kodirnika z negatorjem in svetlečima diodama..... 35
Slika 3.31: Priključki rotacijskega kodirnika na razširitveni plošči..... 36
Slika 3.32: Priključka za PWM izhod..... 36
Slika 3.33: Električna shema vezja PWM signala..... 36
Slika 3.34: Priključek PWM signala na razvojni plošči in plošči rotacijskega kodirnika..... 37
Slika 3.35: Plošča rotacijskega kodirnika; razporeditev signalov..... 37
Slika 3.36: Električna shema štirih tipk..... 38
Slika 3.37: Tipke na razširitveni plošči..... 38
Slika 3.38: Povezava analognih vhodov na glavnem priključku razširitvene plošče z AD pretvornikom *ADC128S022*..... 39
Slika 3.39: Priključki analognih vhodov na razširitveni plošči..... 40
Slika 3.40: Električna shema razdeljevalnika z integriranim vezjem..... 42
Slika 3.41: Električna shema za CPLD in vmesnik JTAG..... 43
Slika 3.42: CPLD in vmesnik JTAG na razširitveni plošči..... 44

Slika 3.43: Shematičen prikaz razvrstitve signalov komponent prek glavnega priključka..... 45
Slika 3.44: Razvrstitev signalov na glavnem priključku razvojne plošče *DE0-Nano*..... 46
Slika 3.45: Električna shema signalov in napajanja na glavnem priključku razširitvene plošče..... 46
Slika 3.46: Glavni priključek na razširitveni plošči..... 47
Slika 4.1: Razporeditev elementov na razširitveni plošči..... 52
Slika 4.2: Zgornja in spodnja maska bakrene plasti razširitvene plošče..... 54
Slika 4.3: Zgornja in spodnja maska za spajkanje razširitvene plošče..... 54
Slika 4.4: Zgornja in spodnja maska belega tiska razširitvene plošče..... 55
Slika 4.5: Maski za prototipno ploščo; levo zgornja in desno spodnja..... 56
Slika 4.6: Naprava za osvetljevanje..... 57
Slika 4.7: Spodnja stran prototipne plošče; levo pred in desno po postopku razvijanja..... 57
Slika 4.8: Cevosan, solna kislina in vodikov peroksid..... 58
Slika 4.9: Zgornja in spodnja stran prototipne plošča brez elementov..... 58
Slika 4.10: Zgornja in spodnja stran prototipne plošče..... 59
Slika 4.11: Spajkalna postaja *Weller*..... 60
Slika 4.12: Spajkalne konice *Weller*..... 60
Slika 4.13: Sestava tinol žice..... 61
Slika 4.14: Stojalo za spajkanje tiskanih vezij..... 61
Slika 4.15: Spajkanje klasičnih komponent; a) drža spajkala in b) pravilno zalita spajka..... 62
Slika 4.16: Primeri spajk v THT..... 62
Slika 4.17: Primeri spajk v SMD..... 63
Slika 4.18: Spajkanje razširitvenih plošč..... 64
Slika 5.1: Shema razdeljevalnika in komponent..... 68
Slika 5.2: Shema razporeditve signalov razdeljevalnika..... 70
Slika 5.3: Diagram poteka sekvenčnega procesa razdeljevalnika; 1.del..... 73
Slika 5.4: Diagram poteka sekvenčnega procesa razdeljevalnika; 2.del..... 74
Slika 5.5: Diagram poteka prireditvenega stavka razdeljevalnika..... 75
Slika 5.6: Rezultat simulacije na matriki LED..... 76

Slika 5.7: Časovni potek serijskega prenosa podatkov v pomikalna registra matrike LED pri simulaciji kode razdeljevalnika..... 77
Slika 5.8: Programator *Xilinx* in razširitvena plošča..... 78
Slika 5.9: Okno za nalaganje programa na *XC9572XL*..... 80
Slika 5.10: Shema izbiralnika in razdeljevalnika..... 81
Slika 5.11: Diagram poteka sekvenčnega procesa izbiralnika..... 83
Slika 5.12: Diagram poteka prireditvenega stavka izbiralnika..... 83
Slika 5.13: Časovni potek krmilnih signalov in aktivnost signalov komponent na vodilu *DATA*. Časovni potek krmilnih signalov in zajem signalov komponent razdeljevalnika..... 84
Slika 5.14: Časovni potek ob zakasnitvi signalov tipk *DATA(0..3)*..... 85
Slika 5.15: Časovni potek signala *DATA0* za zakasnitev aktivnega stanja tipke S1 v času naslova *ADDR*=01 in aktivnega stanja matrike LED v času *ADDR*=10..... 86
Slika 5.16: Časovni potek signala *DATA0* za zakasnitev neaktivnega stanja tipke S1 v času naslova *ADDR*=01 in aktivnega stanja matrike LED v času *ADDR*=10..... 87
Slika 5.17: Časovni potek signala *DATA0* za zakasnitev aktivnega stanja tipke S1 v času naslova *ADDR*=01 in neaktivnega stanja matrike LED v času *ADDR*=10..... 88
Slika 5.18: ASCII tabela znakov..... 89
Slika 5.19: Diagram poteka sekvenčnega procesa matrike LED..... 92
Slika 5.20: Diagram poteka prireditvenega stavka matrike LED..... 93
Slika 5.21: Vzdolžni drsnik..... 94
Slika 5.22: Shematski prikaz nadrejene enote *I2C master* z uporabniško logiko in podrejeno enoto..... 95
Slika 5.23: Diagram stanj komponente *I2C master*..... 96
Slika 5.24: Primer upravljanja in uporabe komponente *I2C master* a) vpis binarne vrednosti "00010001" na naslov 0x03, naprave *0x4D* in b) branje z naslova 0x02, naprave *0x4D*..... 98
Slika 5.25: Časovni potek branja podatka z registra na naslovu 0x02 po protokolu I^2C, s podrejene naprave na naslovu *0x4D* (simulacija kode *I2C master*)…..... 99

Slika 5.26: Diagram stanj uporabniške logike drsnika..... 101
Slika 5.27: Diagram poteka sekvenčnega procesa uporabniške logike drsnika..... 103
Slika 5.28: Diagram poteka uporabniške logike drsnika za blok; a) nastavitev naslovov registrov in b) nastavitev vsebine registrov.... 104
Slika 5.29: Diagram poteka sekvenčnega procesa; prepis rotacijskega statusnega registra na izhod..... 105
Slika 5.30: Vezava naprav na vodilu I^2C..... 106
Slika 5.31: Potek signalov *SCL* in *SDA*..... 107
Slika 5.32: Orodje za nastavitev integriranega vezja *Freescale MCP2200*..... 113
Slika 6.1: Električna shema priključka A in B..... 123
Slika 6.2: Električna shema vmesnika VGA..... 124
Slika 6.3: Zgornja in spodnja plast tiskanine plošče rotacijskega kodirnika..... 126
Slika 6.4: Razširitvena plošča, povezana s ploščo rotacijskega kodirnika..... 126
Slika 6.5: Razširitvena plošča, povezana z razvojno ploščo *DE0-Nano*..... 127
Slika 6.6: Razširitvena plošča skupaj z razvojno ter ploščo rotacijskega kodirnika..... 128

Seznam tabel

Tabela 3.1: Krmiljenje DC motorja..... 27
Tabela 3.2: Opis signalov vmesnika VGA..... 32
Tabela 3.3: Stanja inkrementalnega rotacijskega kodirnika..... 33
Tabela 3.4: Opis signalov AD pretvornika *ADC128S022*..... 40
Tabela 3.5: Seštevek signalov po komponentah..... 41
Tabela 5.1: Opis signalov razdeljevalnika..... 68
Tabela 5.2: Razporeditev podatkovnih signalov *DATA* vodila in signalov komponent glede na naslov..... 69
Tabela 5.3: Krmiljenje matrike LED..... 71
Tabela 5.4: Opis signalov JTAG..... 79
Tabela 5.5: Zasedenost virov na *XC9572XL*..... 79
Tabela 5.6: Opis signalov izbiralnika..... 82
Tabela 5.7: Opis signalov pomnilnika znakov..... 90
Tabela 5.8: Opis signalov UL matrike LED..... 91
Tabela 5.9: Opis signalov komponente *I2C master*..... 95
Tabela 5.10: Opis signalov uporabniške logike drsnika..... 100
Tabela 5.11: Register napak..... 107
Tabela 5.12: Opis polj registra napak..... 107
Tabela 5.13: Rotacijski statusni register..... 108
Tabela 5.14: Opis polj rotacijskega statusnega registra..... 108
Tabela 5.15: Rotacijski konfiguracijski register..... 108
Tabela 5.16: Opis polj rotacijskega konfiguracijskega registra..... 109
Tabela 5.17: Register meje občutljivosti..... 109
Tabela 5.18: Opis polj registra meje občutljivosti..... 110
Tabela 5.19: Register nadzornika diktirne ure..... 110

Tabela 5.20: Opis polj registra nadzornika diktirne ure..... 110
Tabela 5.21: Konfiguracijski register..... 111
Tabela 5.22: Opis polj konfiguracijskega registra..... 111
Tabela 6.1: Spisek elementov razširitvene plošče..... 125

Seznam uporabljenih simbolov

LNIV	laboratorij za načrtovanje integriranih vezij (angl. *Laboratory for Integrated Circuit Design*)
FPGA	programirljivo polje vrat (angl. *Field Programmable Gate Array*)
CPLD	kompleksno programirljivo digitalno vezje (angl. *Complex Programmable Logic Device*)
VHSIC	zelo hitro integrirano vezje (angl. *Very High Speed Integrated Circuit*)
VHDL	jezik za opis hitrih integriranih vezij (angl. *VHSIC Hardware Description Language*)
JTAG	protokol za programiranje (angl. *Joint Test Action Group*)
ISP	način programiranja že vgrajenih mikrokrmilnikov (angl. *In-System Programming*)
PC	osebni računalnik (angl. *Personal Computer*)
HID	vmesnik človek stroj (angl. *Human Interface Device*)
SDRAM	sinhroni dinamični bralno-pisalni pomnilnik (angl. *Synchronous Dynamic Random Access Memory*)
EEPROM	električno zbrisljiv in programirljiv bralni pomnilnik (angl. *Electrically Erasable Programmable Read-Only Memory*)
ASCII	ameriški standard kod za izmenjavo informacij (angl. *American Standard Code for Information Interchange*)
LE	Logični element (angl. *Logic Element*)
AND	logična vrata IN
OR	logična vrata ALI

MSB	najbolj utežen bit (angl. *Most Significant Bit*)
LSB	najmanj utežen bit (angl. *Least Significant Bit*)
USB	univerzalno serijsko vodilo (angl. *Universal Serial Bus*)
GPIO	splošno namenski vhodi/izhodi (angl. G*eneral Purpose Input/Output*)
FIFO	prvi noter, prvi ven (angl. *First In First Out*)
VGA	velikost grafičnega polja (angl. *Video Graphics Array*)
SVGA	velikost grafičnega polja (angl. *Super Video Graphics Array*)
XGA	velikost grafičnega polja (angl. *Extended Graphics Array*)
RGB	rdeča, zelena, modra (angl. *Red Green Blue*)
IBM	podjetje (angl. *International Business Machines Corporation*)
UART	univerzalni asinhroni sprejemnik/oddajnik (angl. *Universal Asynchronous Receiver/Transmitter*)
RS-232	standard za serijsko komunikacijo
RS-422	standard za serijsko komunikacijo
RS-485	standard za serijsko komunikacijo
I^2C	vodilo med integriranimi vezji (angl. *Inter Integrated Circuit*)
DSP	digitalni signalni procesor (angl. *Digital Signal Processor*)
SPI	serijski periferni vmesnik (angl. *Serial Peripheral Interface*)
QSPI	serijski periferni vmesnik s čakalno vrsto (angl. *Queued Serial Peripheral Interface*)
DA	digitalno-analogno (angl. *Digital to Analog*)
AD	analogno-digitalno (angl. *Analog to Digital*)
SIPO	zaporedni-vhod vzporedni-izhod (angl. *Serial-In Parallel-Out*)
PISO	vzporedni-vhod zaporedni-izhod (angl. *Parallel-In Serial-Out*)
PWM	pulzno širinska modulacija (angl. *Pulse Width Modulation*)
PLL	fazno sklenjena zanka (angl. *Phase Locked Loop*)
INL	integralna nelinearnost (angl. *Integral NonLinearity*)

DNL	diferencialna nelinearnost (angl. *Differential NonLinearity*)
PCB	tiskano vezje (angl. *Printed Circuit Board*)
FPCB	fleksibilno tiskano vezje (angl. *Flexible Printed Circuit Board*)
FR-2, 4	material za izdelavo tiskanih vezij (angl. *Flame Resistant*)
CEM-1	material za izdelavo tiskanih vezij (angl. *Composite Epoxy Materials*)
DIP	podnožje integriranega vezja z nožicami na obeh straneh (angl. *Dual In-line Package*)
THT	tehnologija pritrditve skozi luknje (angl. *Through Hole Type*)
SMD	tehnologija površinske pritrditve (angl. *Surface Mount Device*)
DFN	dvojno ploskovno podnožje brez nožic (angl. *Dual-Flat No-leads package*)
QFN	kvadratno ploskovno podnožje brez nožic (angl. *Quad-Flat No-leads package*)
LFCSP	vrsta lestvičnega podnožja (angl. *LeadFrame Chip Scale Package*)
DRC	preverjanje napak v načrtovanju (angl. *Design Rule Checking*)
EMC	elektromagnetna združljivost (angl. *Electromagnetic Compatibility*)
EMI	elektromagnetne motnje (angl. *Electromagnetic Interference*)
EIA	združenje elektronskih industrij (angl. *Electronic Industries Association*)
LED	polprevodniška svetleča dioda (angl. *Light Emitting Diode*)
NPN	tip bipolanega tranzistorja
TTL	tranzistor-tranzistor logika (angl. *Transistor-Transistor Logic*)
CMOS	komplementarni kovina-oksid-polprevodnik (angl. *Complementary Metal-Oxide-Semiconductor*)
RC	upor-kondenzator (angl. *Resistor-Capacitor*)
DC	enosmerni tok (angl. *Direct Current*)
VCC	pozitivna napajalna napetost (angl. *positive supply voltage*)
GND	masa (angl. *Ground*)

Seznam uporabljenih enot

V	Volt – enota za električno napetost
A	Amper – enota za električni tok
Ω	Ohm – enota za električno upornost
W	Watt – enota za električno moč
Hz	Hertz – enota za frekvenco
F	Farad – enota za kapacitivnost
°C	stopinja Celzija – temperaturna lestvica
m	meter – enota za dolžino
s	sekunda – enota za čas
b	bit – enota informacije
B	bajt, skupina bitov – enota za količino podatkov

Povzetek

V magistrskem delu je opisano načrtovanje razširitvene plošče za uporabo z razvojno ploščo *DE0-Nano*. Na začetku je na kratko predstavljena sama razvojna plošča, namen uporabe, tehnologija in kaj plošča vsebuje. V naslednjem poglavju so predstavljene komponente (matrika LED, vzdolžni drsnik, pretvornik iz USB v protokol UART, gonilnik motorja, VGA vmesnik, rotacijski kodirnik, PWM izhod, tipke, analogni vhodi, razdeljevalnik), ki jih vključuje razširitvena plošča. Za vsako komponento je opisan namen, delovanje, uporabljena integrirana vezja, prikazane so električne sheme in razporeditev signalov na priključkih. Na koncu poglavja je predstavljen razdeljevalnik signalov, ki je bil nujen za prenos signalov preko enega priključka.

V naslednjem poglavju so opisana osnovna pravila in postopek risanja tiskanine razširitvene plošče. Prikazane so osnovne maske za izdelavo tiskanine. Pred naročilom izdelave tiskanin smo izdelali prototipno ploščo za testiranje načrtanega vezja. Opisan je postopek domače izdelave tiskanine in potrebno orodje. Prav tako sta opisana postopka spajkanja klasičnih in SMD elementov, drža spajkala in pravilnost spajk.

Delo se nadaljuje s poglavjem programiranja komponent. Najprej je opisana koda razdeljevalnika, delovanje in nalaganje kode na CPLD. Za uporabo večine komponent, za katere potujejo signali prek razdeljevalnika, potrebujemo na razvojni plošči kodo izbiralnika. Opisano je delovanje kode, prikazan je časovni potek in zakasnitev signalov na vodilu, preko katerega komunicirata plošči. Za uporabo drsnika potrebujemo programsko kodo za komunikacijo po protokolu I^2C in uporabniško logiko, ki skrbi za pravilno nastavljanje delovanja in branje pozicij dotikov iz registrov integriranega vezja. Opisana je koda drsnika, vodilo I^2C, simulacija za primer prenosa podatkov na vodilu ter sestava registrov. V poglavju je predstavljeno tudi krmiljenje matrike LED in nastavitev integriranega vezja UART pretvornika.

Ključne besede: razvojna plošča *DE0-Nano*, razširitvena plošča, razdeljevalnik signalov, matrika LED, vzdolžni drsnik, pretvornik iz USB v protokol UART, gonilnik motorja, VGA vmesnik, rotacijski kodirnik, PWM izhod, tipke, analogni vhodi, izdelava tiskanine, spajkanje

Abstract

The thesis describes the configuration of the extension board for the use with the *DE0-Nano* development board. At the beginning, the development board itself is briefly presented, its purpose, technology and the contents of the board. The next chapter presents the components (LED matrix, linear slider, USB to UART protocol converter, motor driver, VGA interface, rotary encoder, PWM output, buttons, analogue inputs, demultiplexer) included in the expansion board. For each component, its purpose, function and used integrated circuits are described, wiring diagrams and the layout of signals at connections are shown. At the end of the chapter, the signal demultiplexer needed for the transmission of signals through a single connector is presented.

In the next chapter are described the basic rules and the procedure of drawing the expansion board stampings. Basic masks for the manufacturing of stampings are shown. Before ordering the manufacturing of stampings, we have developed a prototype board for testing the blueprinted circuit. The method and necessary tools for home manufacturing of stampings are described. Moreover, the process of soldering conventional and SMD components, soldering position and correctness of solders are described.

The thesis continues with the chapter on component programming. Firstly, the demultiplexer code, operation and loading of the code on the CPLD are described. To use the majority of the components for which signals travel through the demultiplexer, we need the selector code on the development board. The functioning of the code is described; the timing and signal delay on the bus through which the boards communicate is shown. To use the slider, we need the programme code for communication through protocol I^2C and user logic that is responsible for proper functioning adjustment and reading the positions of touches from the registers of the integrated circuit. The slider code, bus I^2C, a simulation of the case of data transfer on the bus and the composition of the registers are described. The chapter also

describes the control of the LED matrix and settings of the UART converter integrated circuit.

Keywords: *DE0-Nano* development board, extension board, signal demultiplexer, LED matrix, linear slider, USB to UART protocol converter, motor driver, VGA interface, rotary encoder, PWM output, buttons, analogue inputs, stampings manufacturing, soldering

1 Uvod

Digitalna vezja so velikokrat izvedena s standardnimi digitalnimi integriranimi vezji, vendar za zahtevnejše logične in aritmetične operacije ali procesiranje podatkov take izvedbe običajno niso dovolj. Za aplikacije, ko potrebujemo veliko računsko moč in vzporedno izvajanje procesov, se ne moremo izogniti uporabi hitrih digitalnih vezij. Z digitalnim programirljivim logičnim vezjem FPGA je mogoče doseči visoke frekvence vodenja, kar je lahko pogoj za vrhunske sisteme vodenja [1]. Vse več se porabljajo tam, kjer mikrokrmilnik ni dovolj zmogljiv in kjer ne obstajajo digitalna integrirana vezja za določno funkcijo. Namenski procesorji v tehnologiji FPGA so tako predmet številnih raziskav. Gre za električno programirljiva polja vrat FPGA, ki ne vsebujejo ravnin z AND in OR vrati, vendar se namesto njih v vezju nahaja logično polje elementov in programirljivih stikal za njihovo medsebojno povezavo. Ker tehnologija preprogramirljivih vezij omogoča spreminjanje delovanja že izdelanih elektronskih naprav, lahko vezju spremenimo funkcijo. Poleg fleksibilnosti in hitrosti zagotavljajo tudi zanesljivost, majhnost in cenenost [2] [3].

Zaradi pomanjkanja strojne opreme razvojne plošče FPGA *DE0-Nano* smo se v okviru laboratorija LNIV odločili za razširitev strojne opreme. Proizvajalec razvojne plošče ponuja le majhen del opreme, ki bi bil v okviru laboratorija in izvajanja študijskega programa zanimiv za uporabo. Poleg tega pa je cena enega zunanjega modula nekaj deset evrov. Namesto velikega števila zunanjih modulov smo izdelali razširitveno ploščo, ki ponuja popoln način za razširitev zmogljivosti vezja FPGA. Razširitvena plošča vključuje senzor vzdolžnih premikov prsta, prikazovalnik znakov na matriki LED, gonilnik koračnih in DC motorjev, povezljivost prek USB z UART protokolom, VGA vmesnik, pridobivanje podatkov kotnih premikov, možnost PWM modulacije ter priključitev analognih senzorjev. Plošči komunicirata prek enega priključka.

2 Razvojna plošča *DE0-Nano*

Plošča *DE0-Nano* predstavlja razvojno platformo FPGA, namenjeno izdelavi prototipnih aplikacij oz. projektov.

Ploščo *DE0-Nano* odlikuje zmogljiv *Alterin* FPGA *EP4CE22F17C6N* serije *Cyclone IV* z [4]:

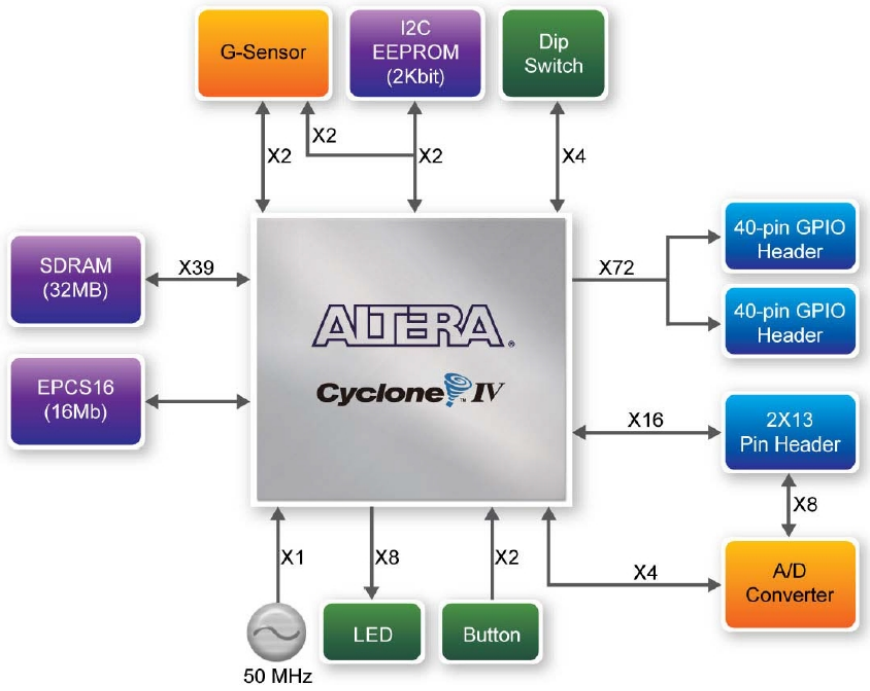
- 22.320 logičnih elementov (LE);
- 594 vgrajenega spomina (kbits);
- 66 vgrajenih množilnikov 18x18;
- 4 splošno namenske fazno sklenjene zanke (PLL);
- 153 vhodno-izhodnih linij.

Za povezovanje analognih senzorjev ima plošča *DE0-Nano* 8-kanalni AD pretvornik. Vsebuje tudi 3-osni merilnik pospeška z visoko 13-bitno resolucijo, 32 MB SDRAM, 2 kb EEPROM, 2 tipki, 4 DIP stikala in 8 LED. Značilnosti razvojne plošče so [4]:

- 50 MHz oscilator ure;
- 8-kanalni 12-bitni AD pretvornik;
- 3-osni merilnik pospeška s 13-bitno ločljivostjo;
- 32 MB SDRAM;
- 2 kb I^2C EEPROM;
- 2 tipki;
- 4 DIP stikala;
- 8 zelenih LED;
- USB programski vmesnik (vrata USB mini-AB);
- 2-pinski priključek za zunanje napajanje (3,6 V–5,7 V);
- 2 priključka s 40 nožicami zagotavljata: 72 vhodno-izhodnih 3,3 V linij, dva 5 V priključka, dva 3,3 V priključka in 4 mase;

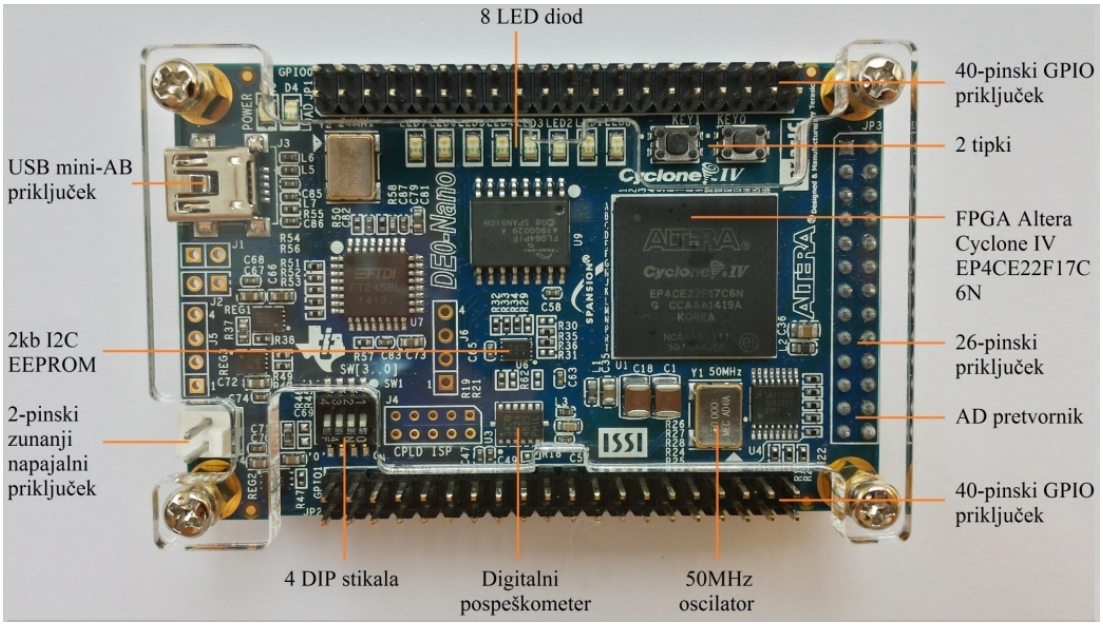
- priključek s 26 nožicami zagotavlja: 13 digitalnih vhodno-izhodnih 3,3 V linij, 3 digitalne vhodne 3,3 V linije, 8 analognih vhodnih linij, 3,3 V priključek in maso.

Slika 2.1 prikazuje blokovni diagram razvojne plošče. V sredini je FPGA *Altera Cyclone IV*, ki zagotavlja povezljivost ostalo periferijo.



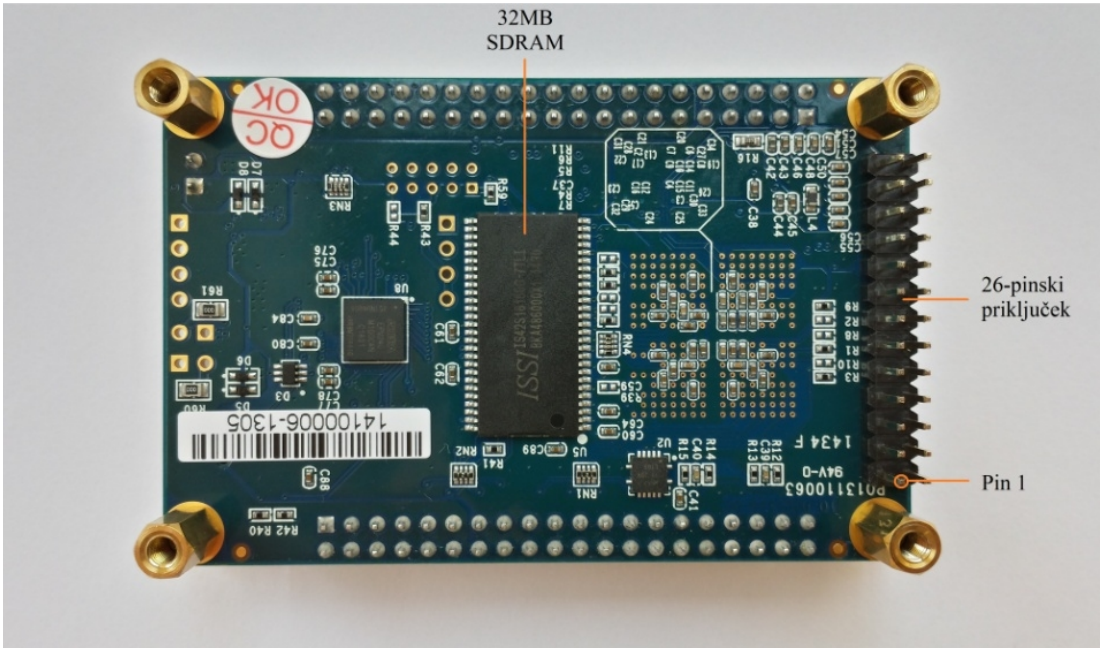
Slika 2.1: Blok diagram razvojne plošče *DE0-Nano* [33].

Slika 2.2 prikazuje ključne sestavne dele razvojne plošče z zgornje strani in postavitev dveh priključkov s 40 nožicami. Velikost plošče je 49 mm x 75,2 mm.



Slika 2.2: Razvojna plošča *DE0-Nano* z zgornje strani.

Slika 2.3 prikazuje sestavne dele s spodnje strani in postavitev priključka s 26 nožicami.



Slika 2.3: Razvojna plošča *DE0-Nano* s spodnje strani.

3 Načrtovanje vezja

Cilj je bil načrtovanje tiskanega vezja, ki bo enostavno za uporabo v povezavi z razvojno ploščo *DE0-Nano*. Proizvajalec razvojne plošče predstavlja izdelek, namenjen prenosnim aplikacijam. Plošča meri le 7,5 cm v dolžino in 4,9 cm v širino [4].

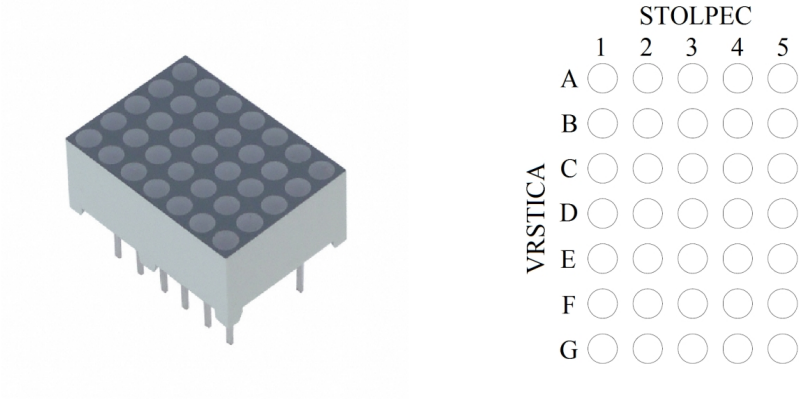
Da bi omogočili širšo paleto aplikacij na področju digitalnih vezij, smo na razširitveno ploščo vključili matriko LED velikosti 5 x 7, vzdolžni drsnik, pretvornik USB v UART protokol, gonilnik motorja, VGA vmesnik, vhod signalov za inkrementalni rotacijski kodirnik, PWM izhod, tipke in analogne vhode. Načrtovanje komponent je opisano v naslednjih podpoglavjih.

Kljub številu uporabljenih komponent smo želeli, da bi bila razširitvena plošča priročna in enostavna za uporabo, zato smo jo poskušali izdelati čim manjšo in brez dodatnih povezav. Porodila se je ideja, da bi načrtovali tiskanino, ki bi jo lahko pritrdili skupaj z razvojno ploščo in plošči istočasno povezali.

Tiskanino smo postavili pod razvojno ploščo, povezave pa speljali prek priključka na spodnji strani. Ob tem smo naleteli na težave, ki smo jih morali rešiti. Težava je bila predvsem prenos digitalnih signalov prek glavnega priključka s premalo digitalnimi linijami. Težavo smo rešili z razdeljevanjem in združevanjem signalov. Ime glavni priključek bom uporabljal za priključka, ki električno povezujeta obe plošči. Napajalna napetost *VCC* na priključku je +3,3 V, zato smo napajalne napetosti komponent prilagodili tej napetosti.

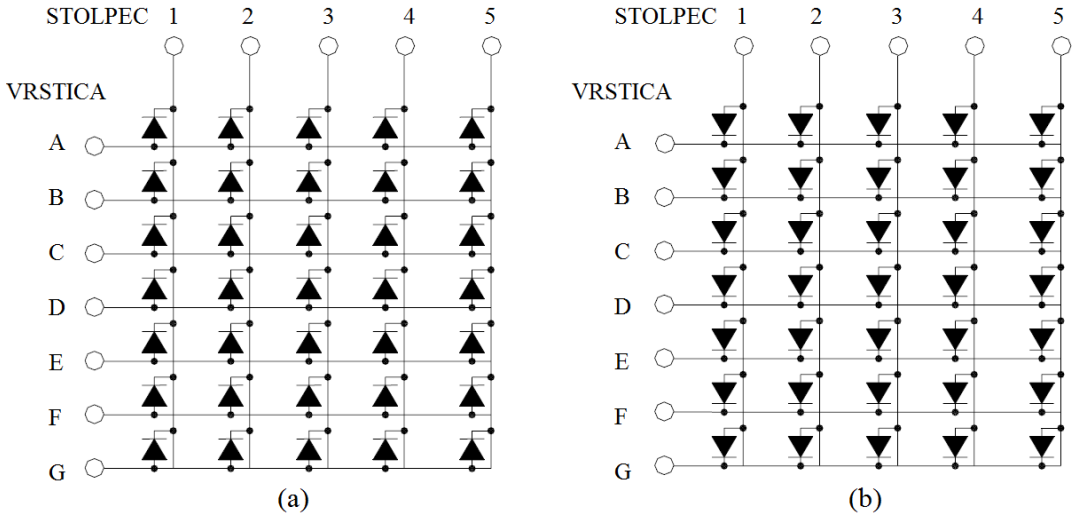
3.1 Matrika LED (angl. *LED matrix*)

Matrika LED je v matrični obliki (po stolpcih in vrsticah) razporejena skupina svetlečih diod (LED). Na sliki 3.1 so s črkami označene vrstice (*A..G*) in s številkami stolpci (*1..5*). Sedaj lahko vsako svetlečo diodo naslovimo po vrstici in stolpcu. Primer slika 3.1: zgornja leva LED (*A1*), spodnja desna LED (*G5*) [5]. Velikost naše matrike je 5 x 7.



Slika 3.1: Primer matrike LED 5 x 7.

Poznamo dve konfiguraciji matrik LED. Razlika je v postopku, ki se uporablja za krmiljenje svetlečih diod – ali gre za konfiguracijo vrstice s skupno anodo ali vrstice s skupno katodo (Slika 3.2). Pri konfiguraciji z vrsticami s skupno anodo vežemo tokovne vire na vrstice (*A..G*), ponore tokov pa na stolpce (*1..5*). Pri matrikah z vrsticami s skupno katodo vežemo tokovne vire in ponore obratno. Krmiljenje svetlečih diod upravljamo s tehniko izbiranja (angl. *multiplexing*) [5].

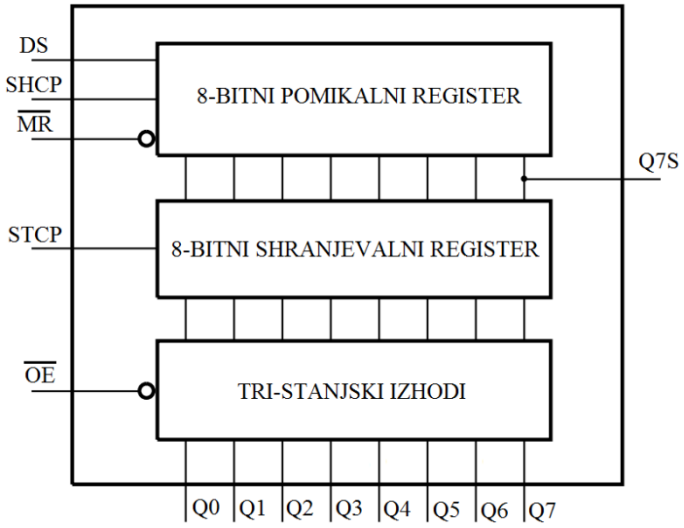


Slika 3.2: Matrika LED 5 x 7: a) vrstice s skupno anodo b) vrstice s skupno katodo.

Z izbiranjem krmilimo le po eno vrstico naenkrat, ostale pa so ta čas zatemnjene. Vidimo, da za krmiljenje matrike LED potrebujemo 12 linij, 7 za vrstice in 5 za stolpce. Podatke za krmiljenje vpisujemo vzporedno, s tem pa imamo slabo izkoriščenost linij. Časi vpisovanja niso zahtevni, zato smo razmislili, kako bi podatke za krmiljenje pošiljali serijsko in izpisovali paralelno na matriko LED. S tem bi privarčevali z linijami.

Serijsko-paralelni SIPO pretvornik je sestavljen iz serijsko-paralelnih pomikalnih registrov (angl. *shift register*). V registrih se podatki ob proženju pomaknejo za en bit. Serijsko-paralelni pomikalni registri so sestavljeni iz kaskade D celic z dinamičnim proženjem (angl. *D flip-flop*). Podatki se ob proženju »prelivajo« iz prejšnje celice v naslednjo [6]. Hitrost pomikanja določa ura serijskega pomika. Vpisani podatki v celicah se na matriko ne smejo prenašati ob vsakem proženju serijske ure, ampak šele, ko je v celicah celotna informacija. Zato na izhodu pomikalnega registra dodamo shranjevalni register (ang. *storage register*). Shranjevalni register sestavimo iz D celic z dinamičnim proženjem. Celice prožimo z uro paralelnega pomika. Ob vsakem proženju se stanje vhoda celic prenese na izhod.

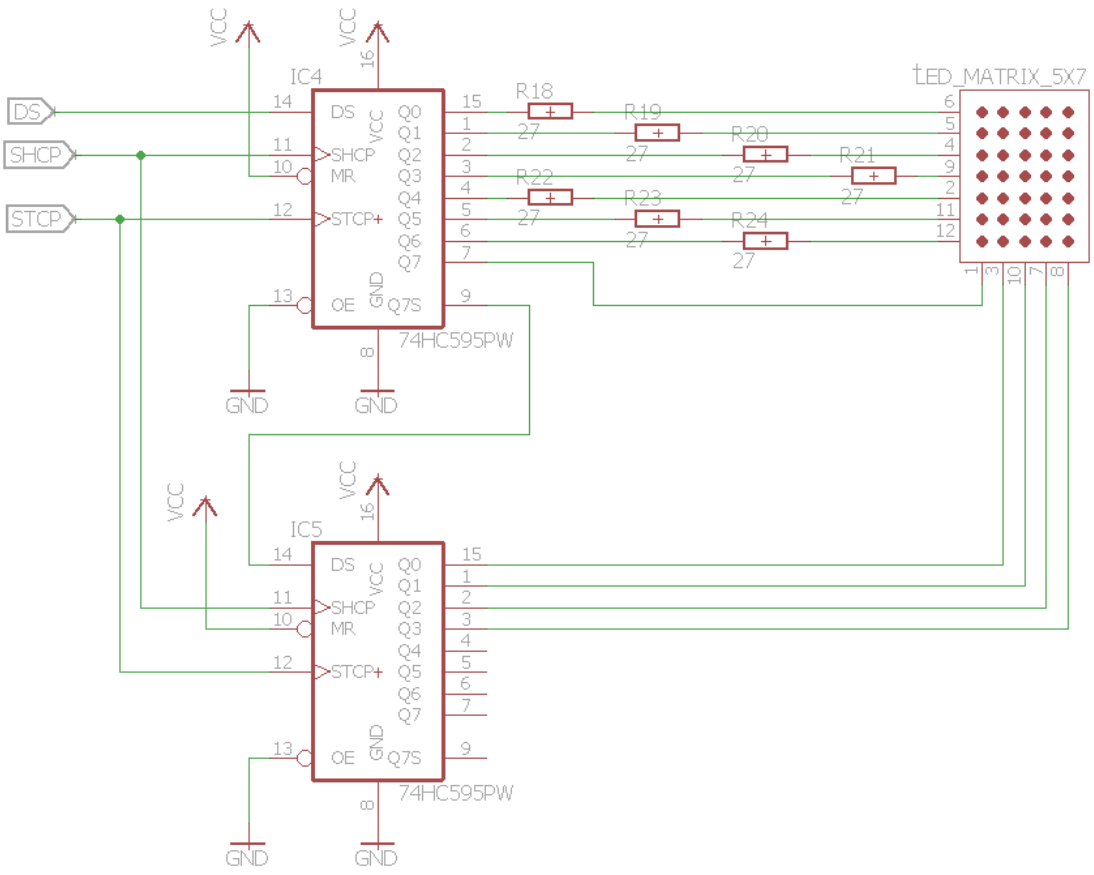
Poiskali smo integrirano vezje, s katerim bi opravljali serijsko-paralelno pretvorbo. Iz družine integriranih vezij *74HC...* smo vzeli 8-bitni pomikalni register s serijskimi vhodi in serijskimi ali paralelnimi izhodi ter izhodnimi zapahi. Integrirano vezje *74HC595* vsebuje poleg 8-bitnega serijskega pomikalnega registra shranjevalni register s tristanjskimi izhodi. Slika 3.3 prikazuje funkcijski diagram.



Slika 3.3: Funkcijski diagram pomikalnega registra *74HC595*.

Pomikalni register ima serijski podatkovni vhod *DS* in serijski izhod *Q7S* na koncu kaskade. Serijski podatki *DS* se pomikajo ob naraščajoči fronti ure pomika *SHCP* (angl. *Shift Register Clock Input*). Vezje ima tudi asinhrono ponastavitev $\overline{MR}$ za vseh 8 izhodov. Aktiven je v nizkem stanju. Podatki se prenesejo v register za shranjevanje ob naraščajoči fronti ure za shranjevanje *STCP* (angl. *Storage Register Clock Input*). Shranjevalni register ima na izhodu 8 vzporednih tristanjskih ojačevalnikov. Podatki na izhodu shranjevalnega registra *Q0..7* se pojavijo, kadar je omogočen vhod $\overline{OE}$. Vhod je aktiven v nizkem stanju [7].

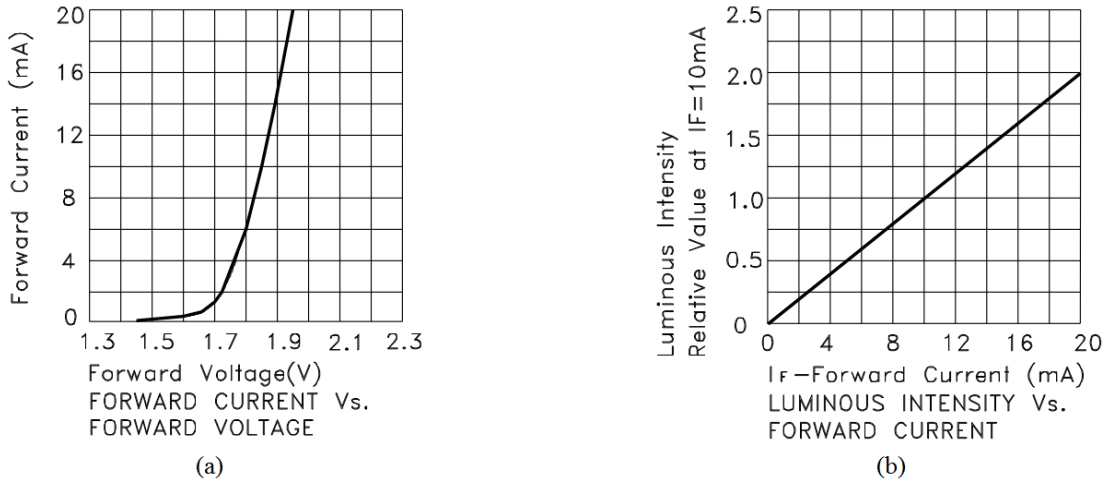
Izbrani pomikalni register ima le 8 izhodov, za izbrano matriko LED 5 x 7 pa jih potrebujemo 12. Uporabili bomo dva registra, zato serijski izhod *Q7S* s prvega registra peljemo na serijski podatkovni vhod drugega. Tako se bodo podatki s prvega prelivali v drug register. Ta ni izkoriščen v popolnosti. Električni načrt matrike LED prikazuje slika 3.4.



Slika 3.4: Električna shema matrike LED; napetostno krmiljenje.

Pri krmiljenju matrike LED *TC07-11SURKWA* s skupno anodo sem storil napako, ker sem na koncu načrtovanja pri optimizaciji komponent odstranil vezje (Slika 3.7), ki ima pomen pri krmiljenju svetlečih diod.

Za krmiljenje svetlečih diod praviloma uporabljamo tokovno krmiljenje. Predpostavil sem, da padec napetosti na uporih ne bo imel velikega vpliva na svetilnost in uporabil napetostno krmiljenje. Napetost na preduporih se spreminja s tokom oz. s številom svetlečih diod. Ker je napajalna napetost konstantna, se napetost na diodah zmanjša. S statične karakteristike svetleče diode vidimo, da tok z napetostjo naglo narašča (Slika 3.5a). Majhna razlika v napetosti na diodi povzroči veliko spremembo toka. Svetilnost svetlečih diod pa je premo sorazmerna električnemu toku skozi diodo (Slika 3.5b). Res je, da padec toka povzroči tudi padec napetosti na uporu, vendar je to razmerje manjše. Predupori služijo le za omejitev maksimalnega toka. Potrebno je tokovno krmiljenje svetlečih diod.



Slika 3.5: Statična karakteristika za matriko LED *TC07-11SURKWA*; a) odvisnost toka od napetosti b) odvisnost svetilnosti od toka [34].

Sklepal sem, da bo zaradi napetostnega krmiljenja prišlo do neenakomerne svetilnosti pri različnem številu aktivnih diod. Več aktivnih diod pomeni večji tok, s tem pa večji napetostni padec na uporih. Da bi bil vpliv manjši in ker ima vrstica manj diod, sem upore namestil na vrstice. Za matriko sem odčital maksimalni prevodni tok 185 mA. Ta vrednost se mi je zdela velika, zato sem vrednost delil s številom vseh diod matrike (35). Po delitvi se mi je za 2 mm majhno diodo, namenjeno indikaciji, tok 5,3 mA zdel primernejši. Dobimo vrednost uporov:

$$R = \frac{Vcc - V_{LED}}{I_{LED}} = \frac{3{,}3\ V - 1{,}95\ V}{5{,}3\ mA} \approx 255\ \Omega \tag{3.1}$$

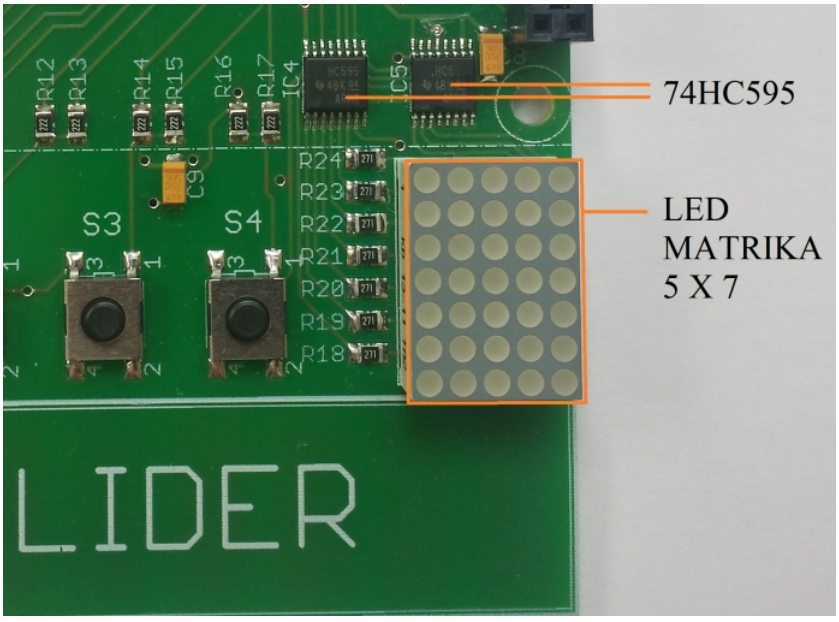
Padec napetosti na integriranem vezju *74HC595* je v mejah od 0 V do 0,1 V, upoštevan v najboljšem pogoju 0 V. Odčital sem napačen tok, zato upornost ni pravilna. Če pogledamo odvisnost svetilnosti od toka s slike 3.5b, vidimo, da bo pri

toku 5,3 mA svetilnost 0,5. Če gori le ena svetleča dioda, je svetilnost 4-krat manjša, kot pri toku 20 mA.

Popravimo vrednost preduporov. Pri izračunu upoštevajmo maksimalno enosmerno vrednost napetosti 2,5 V in toka 30 mA v prevodni smeri svetleče diode. S tem izhodnega toka 35 mA tristanjskega ojačevalnika ne presežemo. Dobimo novo upornost:

$$R = \frac{Vcc - V_{mLED}}{I_{mLED}} = \frac{3{,}3\ V - 2{,}5\ V}{30\ mA} \approx 27\ \Omega \tag{3.2}$$

Pri izračunu upoštevamo maksimalne vrednosti zato, da bi zagotovili boljšo svetilnost, ki pada s številom aktivnih diod. Neenakomerno svetilnost, ki nastane, smo kasneje odpravili programsko. Pri manj prižganih diodah večkrat izpustimo osveževanje vrstice in tako uravnovesimo svetilnost. Več je opisano na strani 73. Lahko bi se »poigrali« in če bi bil čas aktivnosti svetleče diode krajši od 10 ns, kolikor predpisuje proizvajalec za pulzno krmiljenje, bi lahko tok povečali. Ker je hitrost ure vpisa na matriko *SHCP* odvisna od sistemske ure razširitvene plošče (Stran 71), čas ne bi bil zagotovljen. V času priključitve razširitvene plošče na napajanje se lahko zgodijo napetostna nihanja, ki povzročijo izpis naključne vrednosti na izhodu pomikalnega registra. Za čas, ko je razširitvena plošča brez sistemske ure, je na izhodu registra enosmerna vrednost toka in napetosti. Slika 3.6 prikazuje elemente matrike LED na razširitveni plošči.



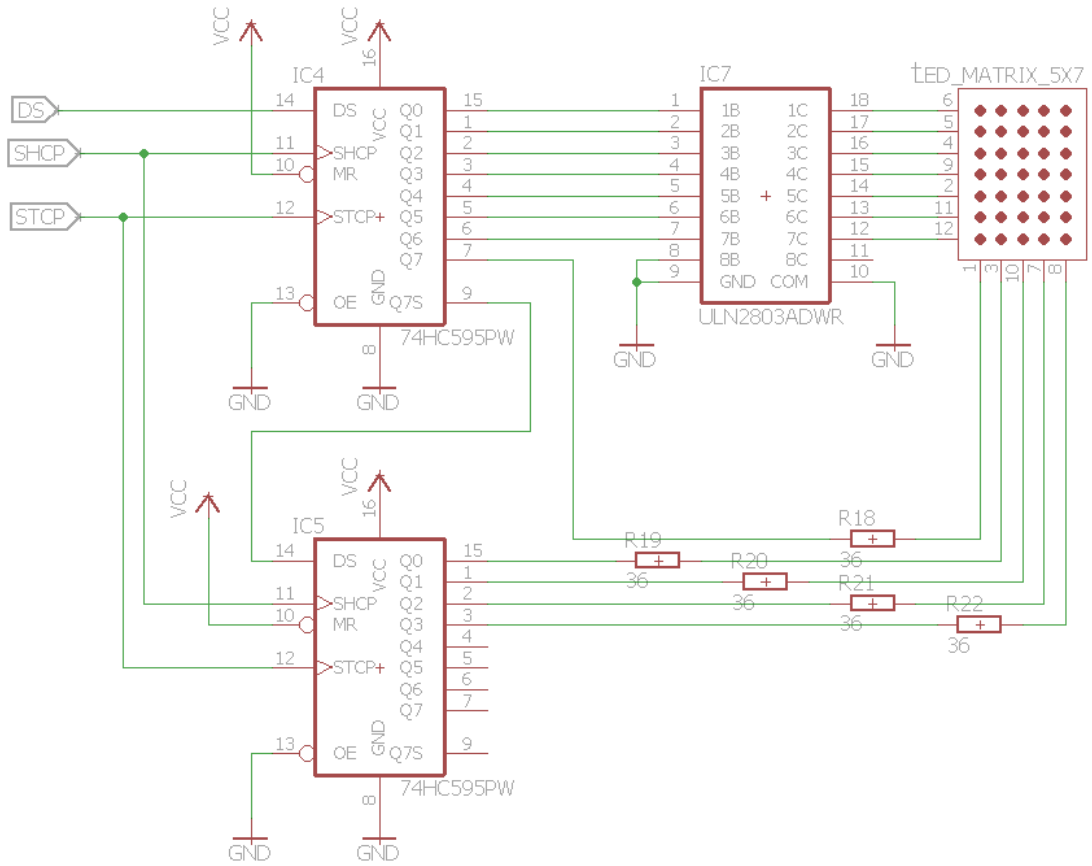
Slika 3.6: Elementi matrike LED na razširitveni plošči.

Za tokovno krmiljenje matrike LED skrbi integrirano vezje *ULN2803A*. Sestavljeno je z osmih NPN Darlingtonovih parov. Maksimalen kolektorski tok posameznega Darlington para je 500 mA. Naprava ima serijsko vezane bazne upore 2,7 kΩ na posamezen par za delovanje neposredno s TTL ali 5 V CMOS napravami.

Matrika LED v zgornjem sestavu (Slika 3.7) je *TA07-11SURKWA*. Karakteristika matrike je enaka prejšnji. Prevodna napetost je 1,75 V pri toku 2 mA. Pri izračunu uporov moramo sedaj upoštevati še padec na Darlington parih, ki znaša v najboljšem primeru 0,9 V. Maksimalni enosmerni tok v prevodni smeri diode je 30 mA. Upornost uporov določimo glede na svetilnost, ki jo želimo imeti. Če si izberemo prevodno napetost diode 1,9 V, dobimo s statične karakteristike (Slika 3.5a) tok 14 mA. Upornost preduporov je:

$$R = \frac{Vcc - V_{LED} - V_{ULN2803}}{I_{LED}} = \frac{3{,}3\ V - 1{,}9\ V - 0{,}9\ V}{14\ mA} \approx 36\ \Omega \qquad (3.3)$$

Slika 3.7 prikazuje električno shemo tokovnega krmiljenja matrike LED.



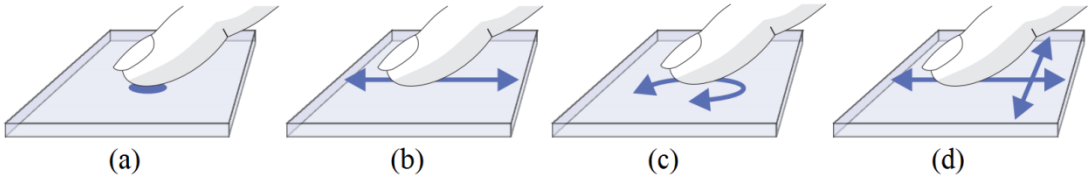
Slika 3.7: Električna shema matrike LED; tokovno krmiljenje.

3.2 Drsnik (angl. *Slider*)

Kapacitivno zaznavanje bližine je tehnologija, ki omogoča zaznavanje dotikov z merjenjem kapacitivnosti. Spremembe kapacitivnosti zaznava zaradi sprememb okoliških materialov. Senzorji ocenijo spremembo, ki jo ustvari električno polje. Za razliko od induktivnih senzorjev lahko kapacitivni senzor zazna vse, kar je prevodno ali ima drugačne dielektrične lastnosti v okolici elektrod. Senzorji dotikov so namenjeni odkrivanju dotikov in prisotnost predmetov brez fizičnih stikov. Postali so vse bolj priljubljeni, saj so v vse več vmesnikih človek-stroj namenjeni za uporabo pri nadzornih ploščah na dotik z zanesljivim odzivanjem na ukaze [9] [10].

Lastnosti:

- Prednosti: stroškovna učinkovitost, več svobode pri projektiranju, večja vzdržljivosti zaradi manjše mehanske obrabe, enostavno za čiščenje, svoboden videz.
- Uporaba: igralne konzole, gospodinjski aparati, mobilni telefoni, prenosne medijske naprave, nadzorniki dostopa, daljinski upravljalnik, mp3 predvajalniki [9] [10].
- Delimo jih lahko v skupine glede na število dimenzij, ki jih prepoznajo: nič-dimenzijski, enodimenzijski in dvodimenzijski (Slika 3.8) [11].



Slika 3.8: Delitev senzorjev dotikov; a) nič-dimenzijski, b) enodimenzijski vzdolžni, c) enodimenzijski krožni, in d) dvodimenzijski [35].

Razlaga pojmov:

- Senzor dotikov (angl. *touch sensor*) – povezava krmilnika senzorja dotikov z elektrodami.
- Krmilnik senzorja dotikov (angl. *touch sensor controller*) – inteligentni del senzorja dotikov, ki meri razlike kapacitivnosti.
- Plošča dotikov (angl. *touch pad*) – vrsta kapacitivnega senzorja, ki se uporablja kot neposredna zamenjava tipk. Območje, ki zaznava prisotnost prsta.
- Elektroda (angl. *electrode*) – prevodno fizično območje, ki se uporablja kot vmesnik.

- Drsnik (angl. *slider*) – skupina plošč dotikov, razporejenih v vrsto, kjer je stanje vsakega območja dotika uporabljeno za določanje položaja vzdolž območja. Sestavljen je iz treh ali več plošč (elektrod) [12].

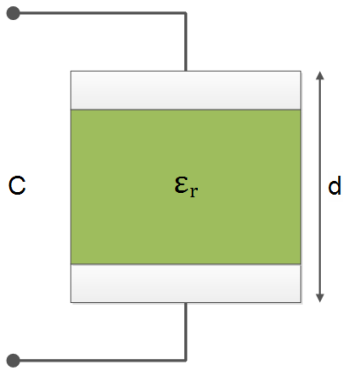
Obstajajo trije pomembni dejavniki pri razvoju kapacitivnih senzorjev dotikov:

- zasnova in oblikovanje blokov elektrod;
- različni dielektrični materiali za površine plošč;
- učinek na merjenje električnih polj v različnih pogojih.

Kapacitivni senzor dotikov lahko predstavimo z modelom kondenzatorja. Enačba (3.4) opisuje model kondenzatorja z dielektrikom med ploščama [9].

$$C = \mathcal{E}_0 \mathcal{E}_r \frac{A}{d} \tag{3.4}$$

C = kapacitivnost [F]
A = površina plošč [m^2]
d = razdalja med ploščama [m]
$\mathcal{E}_0$ = dielektričnost praznega prostora [$8{,}85 \cdot 10^{-12}\ \frac{F}{m}$]
$\mathcal{E}_r$ = relativna dielektričnost materiala med ploščama



Slika 3.9: Model kondenzatorja.

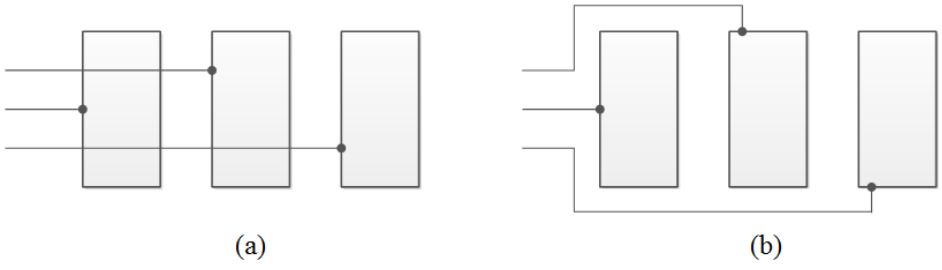
Pri tipični uporabi kapacitivnega senzorja predstavlja elektroda eno ploščo, drugi objekt (npr. prst) pa deluje kot druga plošča kondenzatorja. Substrat tiskanega vezja predstavlja dielektrik med ploščama. Enačba (3.4) ne upošteva zaščitne plasti na območju dotika. Primerni materiali za izdelavo kapacitivnih senzorjev dotika so:

- substrat: FR-4, FR-2, nizkocenovni CEM-1, poliamid PCB, poliester folije;
- materiali elektrod: baker, srebro, ogljik, kar koli električno prevodnega;
- območje dotika: plastika, steklo, kompoziti, lakirane površine (neprevodne barve) [11].

Substrat je osnovni material, ki nosi elektrode. Učinkovitost kapacitivnih senzorjev je odvisna od debeline in dielektrične konstante. Uporabimo lahko skoraj kateri koli izolacijski material. Na delovanje vpliva tudi velikost elektrod. Večje elektrode imajo večji razpon občutljivosti, vendar so bolj občutljive na motnje električnega šuma in statičnih polj v njihovi bližini. Izračuni dimenzij in oblik elektrod so zelo zahtevni. Pri načrtovanju je pomembno, da čim bolj zmanjšamo parazitne kapacitivnosti. Za učinkovito delovanje je pomembno razmerje med velikostjo elektrod in debelino materiala (dielektrika), ki ločuje elektrode in prste uporabnika. Za zelo majhne elektrode je potrebno uporabiti tanke plasti dielektrika, pri večjih pa lahko uporabimo debelejše materiale. Najmanjša veličina elektrod, ki jih predpisuje podjetje *Freescale,* je 3 mm x 3 mm v primeru, da uporabimo tanko akrilno plastiko debeline 0,5 mm. Največja še priporočena velikost meri 15 mm x 15 mm in uporaba plastike do 3 mm debeline. Za steklo, ki ima večjo dielektrično konstanto, lahko uporabimo plastiko do debeline 4 mm.

Pri oblikovanju je potrebno upoštevati tudi učinke okolja. Potenje rok lahko vpliva na sosednje bloke zaradi večjega toka med prstom in območjem druge elektrode. Širša ločitev elektrod lahko ublaži učinek. Priporočljiv je najmanjši razmik elektrod 0,5 mm.

Pri povezavi elektrod moramo poskrbeti, da so povezave kratke in ozke. Dolžine povezav so priporočene do največ 20 cm in ožje od 0,2 mm. Pomembno je, da povezave nimajo medsebojnega vpliva na sosednje elektrode, zato jih ne usmerjamo pod njimi. Slika 3.10a prikazuje primer nepravilne povezave elektrod [13].



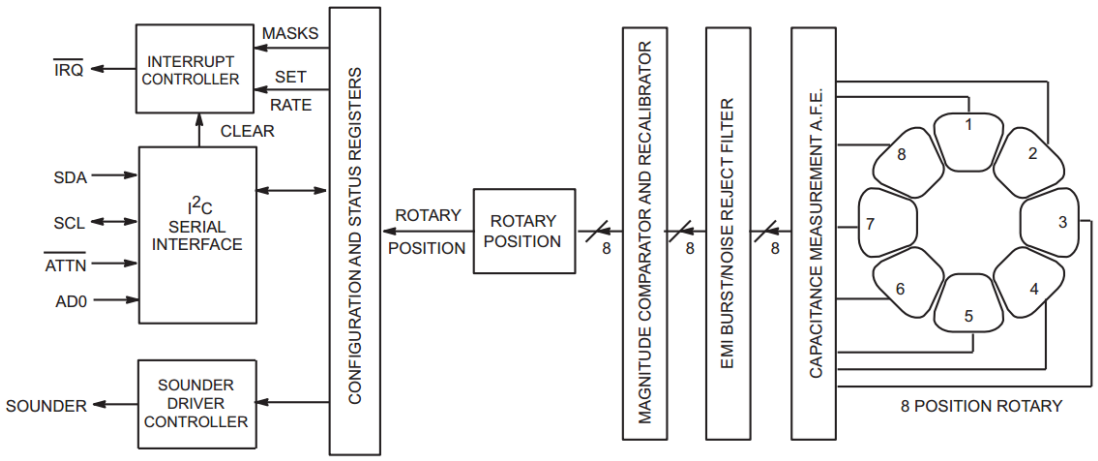
Slika 3.10: Primer povezav elektrod; a) nepravilna b) pravilna.

Vse pasivne komponente je potrebno namestiti čim bližje integriranega vezja, kot je mogoče s skladnostjo EMC. Če so ti deli postavljeni daleč od integriranega vezja, lahko nastanejo težave z nestabilnostjo zaradi motenj. Linije do elektrod naj bojo torej čim krajše in tanjše, da so vplivi motenj čim manjši. Upori na linijah, ki služijo zmanjševanju motenj, naj bodo čim bližje integriranemu vezju [11].

Obstaja veliko integriranih vezij, namenjenih krmiljenju kapacitivnih senzorjev dotikov. Pri izbiri primernega integriranega vezja je bila velika večina tistih, ki imajo pri površinski tehnologiji pritrditve SMD kontakte pod svojim ohišjem. Ta podnožja (večina je bila QFN in LFCSP) je težko spajkati, zato niso prišla v poštev. Nekaj jih ni bilo na seznamu prodajalca (*Farnell*), zato smo izbrali edino možnost *Freescale MPR083*.

Gre za krmilnik enodimenzijskega kapacitivnega senzorja dotikov, optimiziran za polje osmih elektrod. Vezje *MPR083* lahko nadzoruje krožni ali linearni drsnik. Mi smo zaradi stiske s prostorom izbrali vzdolžni (linearni) drsnik. Drsnik je podolgovata sledilna plošča, ki zaznava položaj prsta vzdolžno. Elektrode na tiskanem vezju so na nasprotni strani območja, kjer se zaznava dotik prstov. Velikost elektrod smo prilagajali velikosti razširitvene plošče. Dolžina elektrod meri 6 mm, širina 14 mm in razmik med 2 mm. Povezave so kratke, najdaljša meri 26 mm, in 0,2 mm široke. Debelina substrata je 1,6 mm.

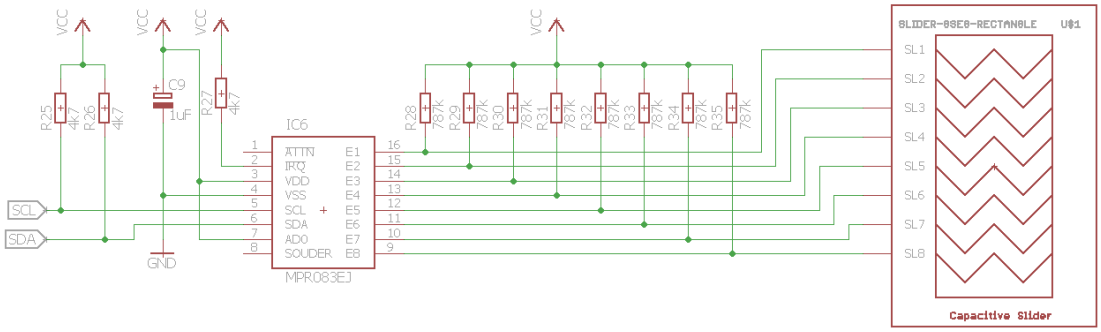
Ker služi prst kot druga plošča kondenzatorja (Slika 3.9), se kapacitivnost spreminja z dotiki. Ob dotiku kapacitivnost narašča. Integrirano vezje ob dotiku sproži števec in šteje. Ko napetost elektrode doseže 0,7 *VCC,* se števec ustavi in vrednost števca je izmerjena. Naprava sporoči dotik in položaj. Funkcijski blok diagram za integrirano vezje *MPR083* prikazuje slika 3.11 [14].



Slika 3.11: Funkcijski blok diagram za integrirano vezje *Freescale MPR083* [36].

Vezje omogoča umerjanje med dvema vhodnima referenčnima točkama, med minimalno in maksimalno pričakovano kapacitivnostjo. Te referenčne kapacitivnosti omogočajo odpravo napak pri merjenju elektrod zaradi vplivov okolja. Vezje *MPR083* uporablja I^2C vmesnik za komunikacijo z gostiteljem, ki nastavi delovanje.

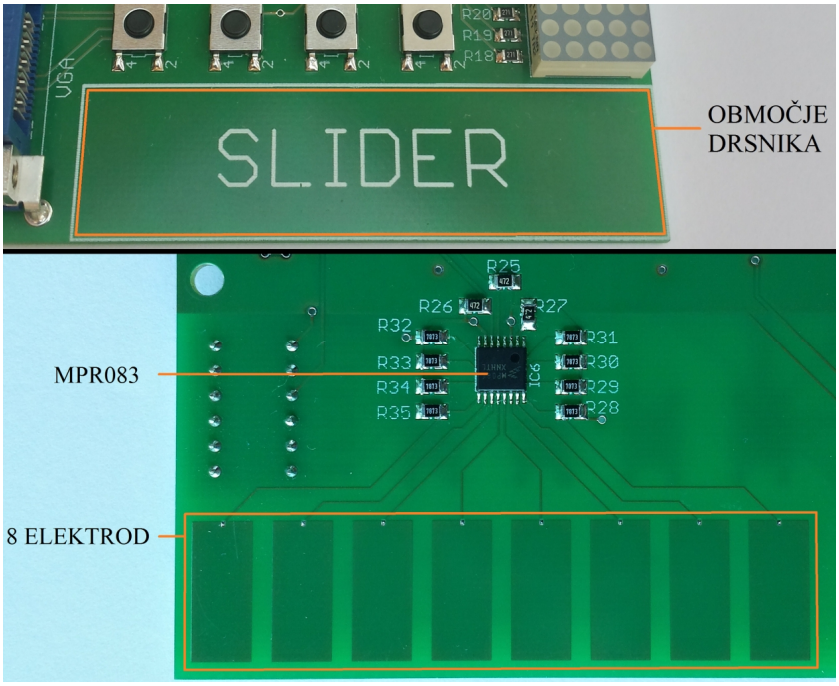
Naprava deluje v napetostnem območju od 1,8 V do 3,6 V, povprečni tok znaša 41 µA [12]. Električni načrt prikazuje slika 3.12.



Slika 3.12: Električna shema za drsnik.

Linije elektrod morajo imeti dvižne (angl. *pull-up*) upore. Priporočena vrednost uporov je 780 kΩ. Ker ta vrednost uporov ne obstaja, smo vzeli najbližjo 787 kΩ. Poleg osmih uporov je potrebno obvezno dodati sklopni (angl. *bypass*) kondenzator kapacitivnosti 1 µF med maso in *VCC*. Potrebujemo tudi dva dvižna upora 4,7 kΩ na liniji *SCL* in *SDA* vodila I^2C. Enako vrednost upora vstavimo med $\overline{IRQ}$ in *VCC*. Z vhodom *AD0*, vezanim na nizek ali visok nivo, določimo naslov naprave. V primeru, da vežemo v nizko stanje, je naslov *0x4C*. V našem primeru je v visokem stanju, zato je naslov podrejene naprave (*MPR083*) *0x4D* [12].

Območje drsnika na razširitveni plošči prikazuje slika 3.13.

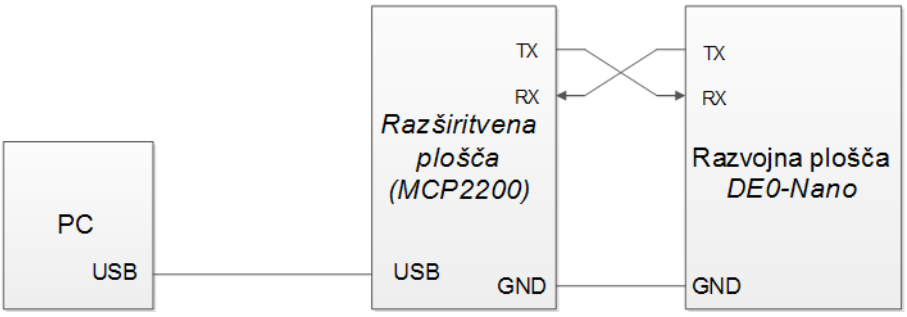


Slika 3.13: Območje drsnika na zgornji in območje elektrod na spodnji strani razširitvene plošče.

3.3 Pretvornik iz USB 2.0 v protokol UART (angl. *USB 2.0 to UART protocol converter*)

S pretvornikom USB v UART protokol smo želeli zagotoviti USB povezljivost z aplikacijami oz. napravami, ki imajo ali potrebujejo vmesnik UART. Uporaba programskega jezika je tu preprosta in učinkovita. Kratica UART pomeni univerzalni asinhronski sprejemnik-oddajnik. Gre za serijski prenos podatkov znane oblike in hitrosti prenosa. UART se pogosto uporablja v povezavi s komunikacijskimi standardi, kot so RS-232, RS-422 ali RS-485. Ne glede na to, kakšni so nivoji signalov, ki jih dobimo z namenskimi integriranimi vezji, gre na koncu za (nizke ali visoke) logične nivoje.

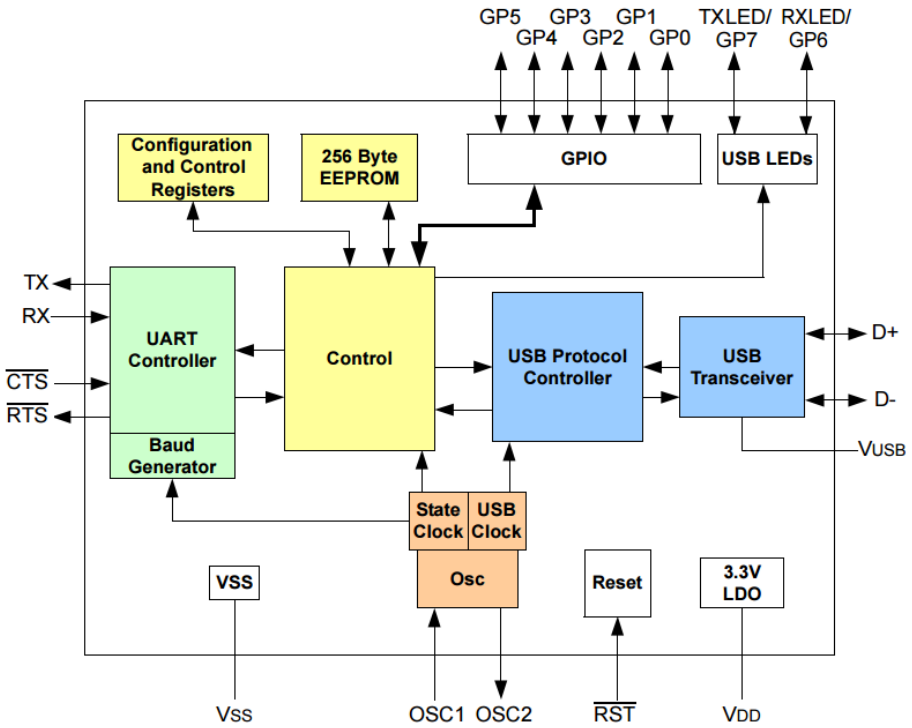
Običajno gre za strojno opremo (integrirano vezje), ki se uporablja za serijsko komunikacijo preko računalnika oz. univerzalnega serijskega vodila USB. Prenos podatkov po vodilu USB je na koncu za uporabnika popolnoma enak prenosu po protokolu UART (Slika 3.14) [15].



Slika 3.14: Shema UART komunikacije.

Obstaja veliko integriranih vezij, ki ponujajo take možnosti prenosa. Naše zahteve za uporabo integriranega vezja so bile predvsem majhnost in enostavnost vezja, ustrezna napetost logičnih nivojev in napajanja 3,3 V ali možnost napajanja prek USB vodila. Po pregledu podatkovnih listov (angl. *data sheet*) različnih proizvajalcev in izvedb, smo izbrali integrirano vezje *Microchip MCP2200*. V primerjavi z ostalimi vezji je ta cenejši, za delovanje pa potrebuje dodaten resonator.

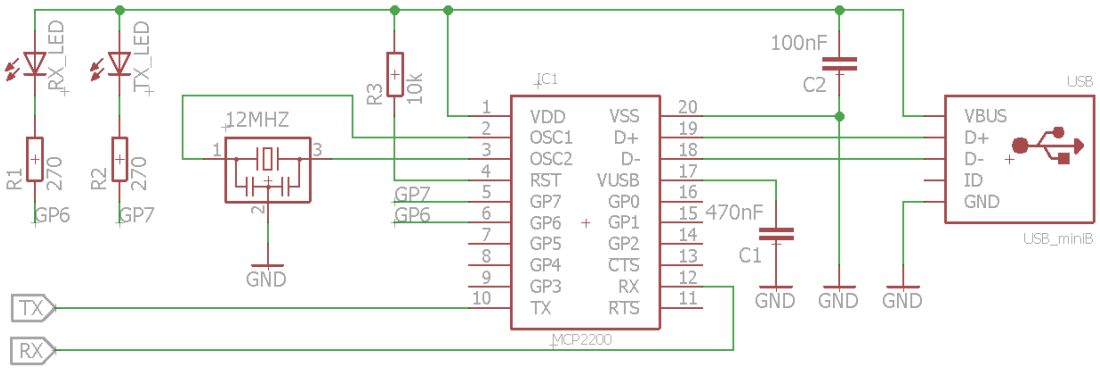
Gre za serijski pretvornik USB v UART s povezljivostjo z USB vodilom. Vezje *MCP2200* podpira USB 2.0 in hitrost UART prenosa od 300 do 1000k bitov na sekundo. Ker je obratovalna napetost med 3 V in 5,5 V, smo napajanje vezja izvedli prek vodila USB. Logični nivoji so v enakem napetostnem območju. Ima dva izhoda za prikaz dejavnosti *TX* in *RX* s svetlečima diodama. Slika 3.15 prikazuje blok diagram s podatkovnega lista integriranega vezja *MCP2200* [16].



Slika 3.15: Blok diagram; integrirano vezje *MCP2200* [37].

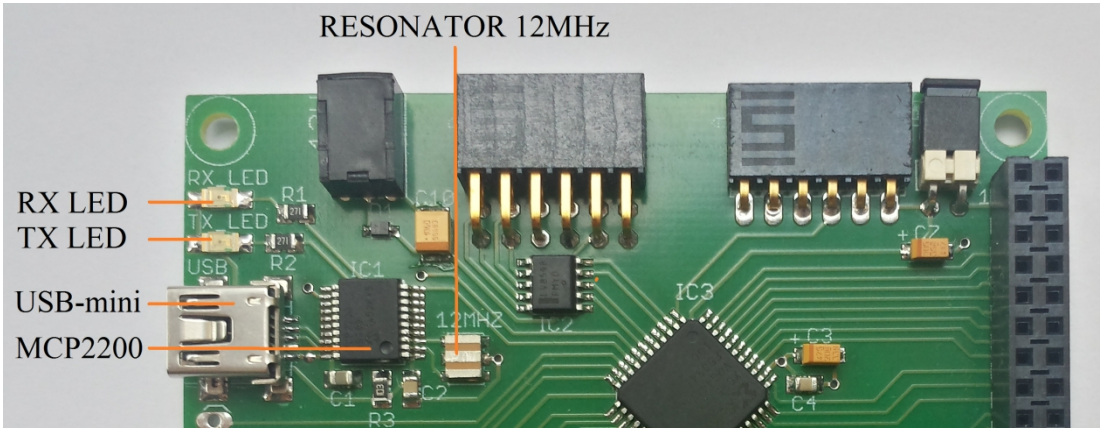
Glavni del naprave *MCP2200* je nadzorni blok (*Control*). Vsi ostali bloki so povezani skupaj preko nadzorne enote. Nadzorni blok skrbi za prenos podatkov med USB in UART, kot tudi za ukaze, ki krmilijo USB in ostale krmilne ukaze. UART je nastavljiv na več hitrosti prenosa. Naprave ni mogoče nadzorovati preko vmesnika UART. Nastavitev integriranega vezja je opisana v poglavju 5.5 z uporabo programa *Microchip MCP2200 Configuration Utility*.

Serijski prenos podatkov poteka preko dveh linij *TX* in *RX*. Vezava linij med UART napravama je križna, signal *TX* proti *RX* (Slika 3.14). Liniji *CTS* in *RTS* služita kot strojna mehanizma za regulacijo pretoka, ki ju nismo uporabili. Smo pa dodali dve svetleči diodi za status komunikacije. Električni načrt za vezje UART prikazuje slika 3.16.



Slika 3.16: Električna shema pretvornika z USB v UART.

Za uporabo napajanja prek vodila USB moramo po priporočilu proizvajalca med maso in nožico *VUSB* vključiti sklopni kondenzator velikosti med 0,22 µF in 0,47 µF. S tem naj bi po specifikacijah zagotovili zagonski tok USB vodila, skupne efektivne kapacitivnosti med maso in *VBUS*, ki pa ne sme preseči 10 µF. Blokirni kondenzator 0,1 µF vstavimo med maso in *VDD*. Da zagotovimo hitrost USB prenosa 12 Mb/s, vstavimo med sponki *OSC1* in *OSC2* keramični resonator s frekvenco 12 MHz. Vezje ima tudi signal za ponastavitev $\overline{RST}$. Ker signala nismo uporabili in je aktiven v nizkem stanju, ga povežemo preko 10 kΩ upora na linijo *VBUS* [16]. Maso USB vodila in maso razširitvene plošče povežemo skupaj. Slika 3.17 prikazuje priključek USB mini in elemente pretvornika.



Slika 3.17: Elementi pretvornika USB v UART na razširitveni plošči.

3.4 Gonilnik motorja (angl. *Motor driver*)

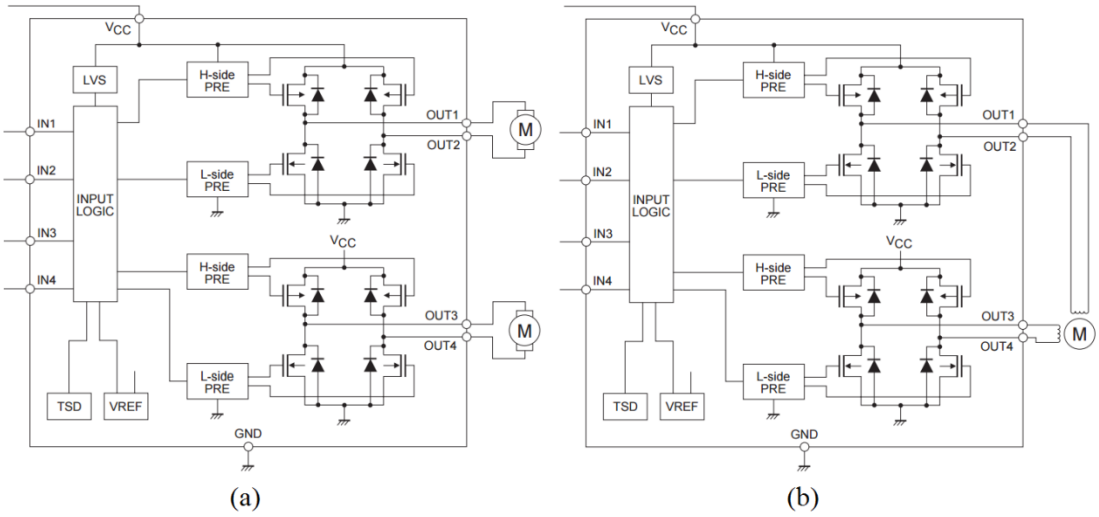
Zaradi uporabnosti in enostavnosti uporabe smo se odločili dodati tudi gonilnik koračnega motorja. Zahteve za gonilnik so bile majhnost integriranega vezja, napajalna napetost motorja vsaj 12 V, moč več kot 10 W in napetost krmilnih signalov 3,3 V.

Po pregledu integriranih vezij smo izbrali *LV8548MC*. Gre za 2-kanalni napetostni gonilnik, optimalen za poganjanje motorjev v 12 V sistemih. Vezje lahko poganja en koračni motor v polnem ali polovičnem korakanju, dva enosmerna motorja ali en enosmeren motor z vzporedno vezavo izhodov za večje tokove.

Lastnosti integriranega vezja *LV8548MC*:

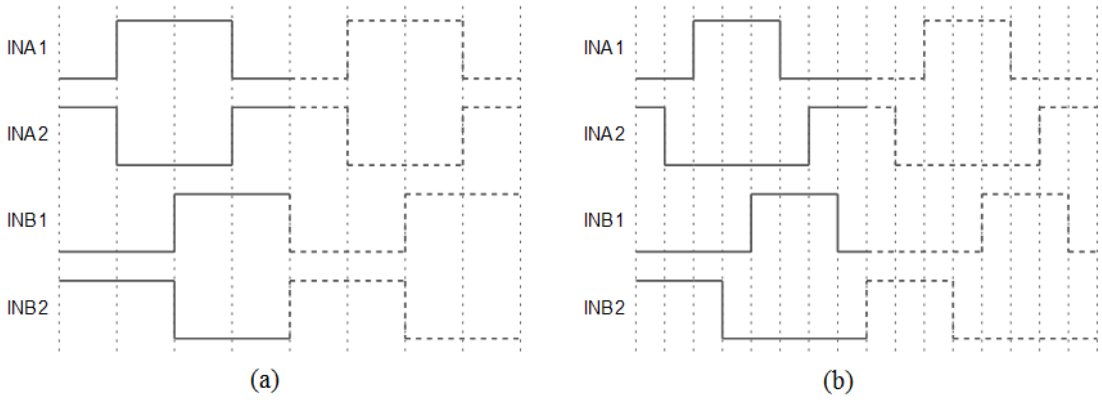
- Priporočena napajalna napetost: 4 V do 16 V (maksimalna napetost 20 V).
- Maksimalen tok: 1 A.
- Napetost logičnih nivojev: za visok nivo med 1,8 V in 5,5 V.
- Tok logičnih nivojev pri napetosti nivoja 5 V: 50 µA.
- Nizka izhodna upornost: tipična R_{ON} = 1 Ω.
- Eno napajanje (ne potrebuje dodatnega napajanja za krmilni sistem).
- Funkcija zavore [17].

Slika 3.18 prikazuje blok diagram integriranega vezja *LV8548MC* za primer a) vezave dveh DC motorjev in primer b) vezave koračnega motorja.



Slika 3.18: Blok diagram integriranega vezja *LV8548MC*; a) vezava dveh DC motorjev, b) vezava koračnega motorja [38].

Primer za polno koračno krmiljenje koračnega motorja prikazuje slika 3.19a, polovično krmiljenje pa slika 3.19b.



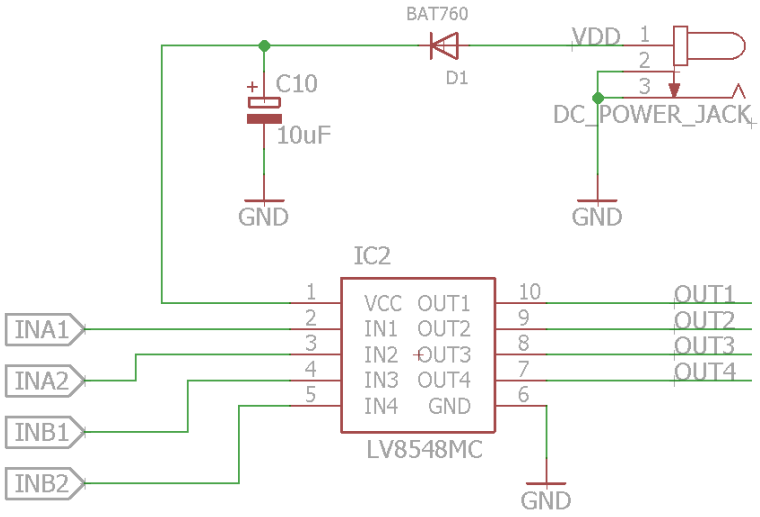
Slika 3.19: Primera načina krmiljenja koračnega motorja; a) polni in b) polovični.

Pri vzporedni vezavi izhodov za povečanje toka DC motorja kratko sklenemo izhoda *OUT1* in *OUT3* ter izhoda *OUT2* in *OUT4*. Krmiljenje za enosmerna motorja prikazuje tabela 3.1.

<table>
<tr><th colspan="4">Vhodi</th><th colspan="4">Izhodi</th><th colspan="2" rowspan="2">Opombe</th></tr>
<tr><td>INA1</td><td>INA2</td><td>INB1</td><td>INB2</td><td>OUT1</td><td>OUT2</td><td>OUT3</td><td>OUT4</td></tr>
<tr><td>0</td><td>0</td><td colspan="2" rowspan="4"></td><td>OFF</td><td>OFF</td><td colspan="2" rowspan="4"></td><td rowspan="4">Motor 1</td><td>V pripravljenosti</td></tr>
<tr><td>0</td><td>1</td><td>H</td><td>L</td><td>Vrtenje naprej</td></tr>
<tr><td>1</td><td>0</td><td>L</td><td>H</td><td>Vrtenje nazaj</td></tr>
<tr><td>1</td><td>1</td><td>L</td><td>L</td><td>Zavora</td></tr>
<tr><td colspan="2" rowspan="4"></td><td>0</td><td>0</td><td colspan="2" rowspan="4"></td><td>OFF</td><td>OFF</td><td rowspan="4">Motor 2</td><td>V pripravljenosti</td></tr>
<tr><td>0</td><td>1</td><td>H</td><td>L</td><td>Vrtenje naprej</td></tr>
<tr><td>1</td><td>0</td><td>L</td><td>H</td><td>Vrtenje nazaj</td></tr>
<tr><td>1</td><td>1</td><td>L</td><td>L</td><td>Zavora</td></tr>
</table>

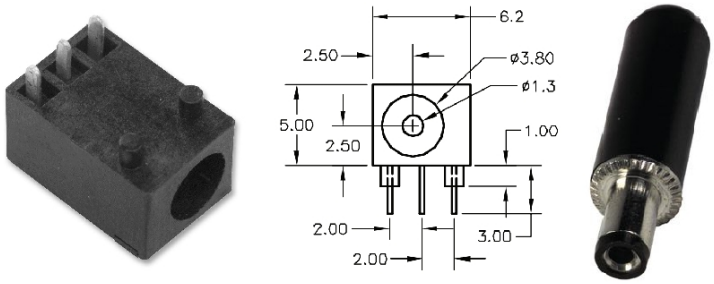
Tabela 3.1: Krmiljenje DC motorja [17].

Integrirano vezje za svoje delovanje ne zahteva skoraj nobenih dodatnih elementov. Za zmanjševanje visokofrekvenčnih motenj na napajalni liniji smo dodali blokirni kondenzator. Kondenzator je tantalski z največjo kapacitivnostjo 10 µF, kot jo priporoča proizvajalec. Napetostno območje kondenzatorja je do 20 V in je enako maksimalni napetosti integriranega vezja. Izbrali smo ga v ohišju 1210 (EIA koda). Na pozitivno napajalno linijo smo dodali tudi Schottky diodo (*BAT760*) za zaščito v primeru priključene napačne polaritete napajanja. S tem se napajalna napetost motorja zniža za 0,55 V. Električni načrt gonilnika motorja je prikazan na sliki 3.20.



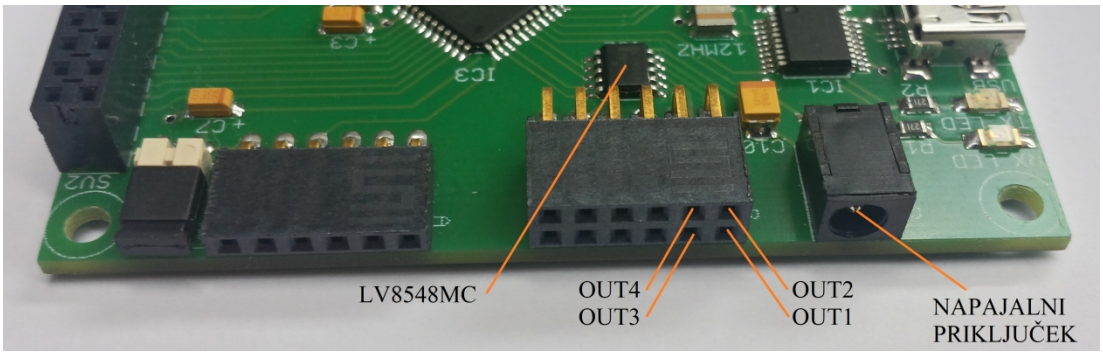
Slika 3.20: Električna shema gonilnika motorja.

Ohišje napajalnega priključka smo morali izbrati čim nižje zaradi stiske s prostorom nad razvojno ploščo. Izbrali smo priključek (*SPC21373*) z višino ohišja 11 mm. Velikost vtičnice je 3,8 mm x 1,3 mm (Slika 3.21).



Slika 3.21: Napajalna priključka gonilnika motorja in mere vtičnice.

Izhode *OUT1*, *OUT2*, *OUT3* in *OUT4* smo peljali na 90° letvico. Napajalni priključek in letvico izhodov gonilnika motorjev na razvojni plošči prikazuje slika 3.22.



Slika 3.22: Napajalni priključek in izhodi gonilnika motorja na razširitveni plošči.

3.5 VGA vmesnik (angl. *Video Graphics Array*)

VGA je video standard, ki ga je razvil IBM leta 1987. Prvi VGA standard je ponujal ločljivost 640 x 480 barvnega zaslona s hitrostjo osveževanja 60 Hz in s 16 barvami. Če se ločljivost zniža na 320 x 200, je lahko prikazanih 256 barv [18].

Standard VGA je bil nadomeščen s SVGA z ločljivostjo 800 x 600, čeprav so kabli in priključki ostali enaki. SVGA standard podpira 16 milijonov barv. IBM je kasneje uvedel XGA z ločljivostjo 1024 x 768, in čez leta so se dodajale izboljšave vse do danes, ko imamo večkratnik glede skupnega števila pik [19]. Analogni VGA vmesnik je na voljo pri številnih grafičnih karticah, televizijskih sprejemnikih, projektorjih, računalniških monitorjih... Priključek je tri vrstični, ima 15 nožic. Na sliki 3.23 je primer priključka VGA.

Slika 3.23: Priključka VGA.

Vmesnik VGA smo dodali tudi na razširitveno ploščico. Čeprav se vmesnik počasi poslavlja, bo na plošči služil za prenos video signalov nižjih ločljivosti.

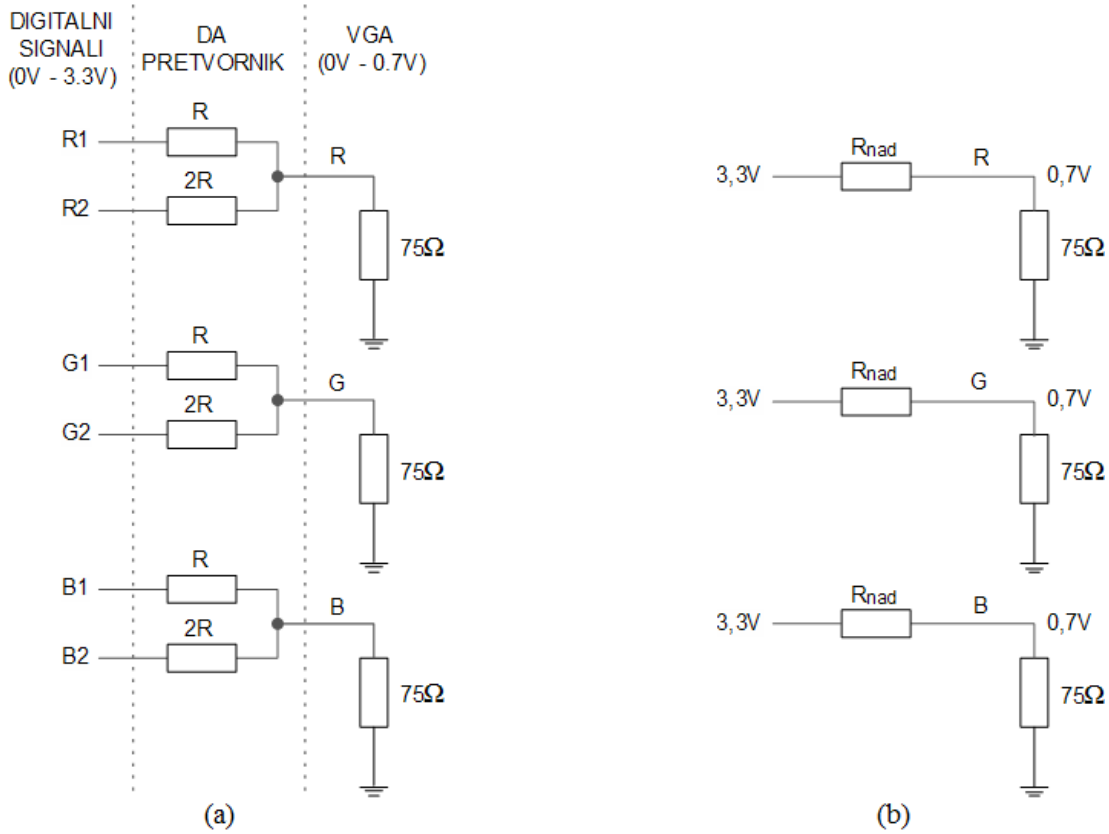
Fizični VGA vmesnik je 5 žični signal *RGBHV* (angl. *Red, Green, Blue, Horizontal sync, Vertical sync*), ki uporablja analogne nivoje za *RGB* signale in digitalne TTL nivoje za *HV* sinhronizacijska signala. Barva slikovne pike (angl. *pixel*) je določena z vrednostjo *RGB* signala. Barva je kombinacija treh osnovnih barv in je mešanica stopenj svetilnosti osnovnih barv. Vsak od treh signalov krmili svetlost osnovnih barv posamezne pike. Ena pika nosi malo informacije, okvir slikovnih pik pa tvori sliko. Slikovne pike izrisujemo na zaslon po stolpcih od leve proti desni in po vrsticah od zgoraj navzdol.

Kdaj pomakniti žarek v naslednji stolpec ali vrstico, določata signala *HV* (*HSYNC* in *VSYNC*), ki določata ločljivost zaslona (npr. 640 x 480). *HSYNC* in *VSYNC* signala sta vlak kvadratnih 5 V impulzov, medtem ko *RGB* signali sprejemajo vrednosti v neprekinjenem napetostnem območju od 0 V do 0,7 V. Najbolj svetla barva je predstavljena z napetostjo 0,7 V. Po standardu je logična enka sinhronizacijski signal (*HSYNC* in *VSYNC*) v napetostnem območju med 2,4 V in 5,5 V [20] [21]. Signala *HV* sta združljiva z napetostjo 3,3 V in ju peljemo neposredno na VGA priključek.

Ker so *RGB* signali analogni, lahko ustvarimo neskončno količino različnih barv, vendar je to odvisno od tega, koliko napetostnih nivojev lahko ustvarimo. Večina VGA sistemov omogočajo 256 (8 bitov) nivojev za vsako barvo, kar je 24 bitov barv ali 16 milijonov barv.

Iz digitalnega sveta dobimo neprekinjeno napetostno območje z digitalno analognim DA pretvornikom. V našem primeru imamo enostavni uporovni delilnik, ki digitalne signale spremeni v napetostne nivoje. Zaradi stiske z digitalnimi linijami smo za naše potrebe načrtali uporovni delilnik s štirimi napetostnimi nivoji na barvo. Imamo 2-bitno globino posamezne barve, skupno pa 6-bitno barvno globino ali 64 možnih barvnih kombinacij. Globina barv je dovolj dobra za prikaz besedila ali grafičnih simbolov.

Vsak vhod barve predstavlja 75 Ω obremenitev za uporovni delilnik [20]. Z upori kombiniramo obremenitev tako, da razdelimo napetost 3,3 V digitalnih logičnih nivojev na maksimalno napetost 0,7 V na izhodu VGA priključka. Za barvo imamo dve digitalni liniji, zato potrebujemo 2 upora na barvo (Slika 3.24a).



Slika 3.24: Električna shema DA pretvornika; a) uporovni delilnik, b) nadomestno vezje.

Upor *R* najbolj prispeva k končnemu rezultatu, zato z njim povežemo najbolj pomemben digitalni signal (*R1*, *G1*, *B1*). Drugi signal barve (*R2*, *G2*, *B2*) ima polovico pomembnosti prvega, zato je drugi upor dvakratnik prvega *2R*. Vsaka logična ničla ali enka signala prispeva h končnemu napetostnemu nivoju. Ob kombinaciji samih logičnih ničel dobimo napetost 0 V in ob enkah napetost 0,7 V.

Poglejmo, kako dobimo vrednost uporov. Najprej izračunamo napetostni padec na nadomestni upornosti R_{nad} za maksimalno vrednost digitalnih in analognih signalov (Slika 3.24b). Dobimo 2,6 V napetostnega padca. Izračunamo:

$$R_{nad} = \frac{V_{R_{nad}}}{\frac{V_{RGB}}{75\,\Omega}} = \frac{2{,}6\,V}{\frac{0{,}7\,V}{75\,\Omega}} \approx 279\,\Omega. \tag{3.5}$$

Iz enačbe za vzporedno vezavo uporov R in 2R

$$R_{nad} = \frac{R \cdot 2R}{R+2R} \tag{3.6}$$

dobimo upornost R za najbolj pomemben bit

$$R = \frac{3}{2}R_{nad} = \frac{3}{2}279\,\Omega \approx 418\,\Omega \tag{3.7}$$

in upornost $2R \approx 836\,\Omega$.

Ko smo naročali elemente, smo iskali najboljši približek vrednosti uporov. Upornost uporov smo izbrali 422 Ω in 845 Ω. Nadomestno upornost za izbrani vrednosti dobimo sedaj 281 Ω. Maksimalna napetost, ki jo dobimo na izhodu DA pretvornika, bo približno 0,694 V namesto 0,7 V. Odstopanje napetosti je minimalno.

Šum pri 2-bitni barvni globini nima takšnega vpliva, kot bi bil vpliv šuma pri večbitnih barvnih globinah. Večji problem kot šum predstavlja izrisovanje večjega števila slikovnih pik. Za več prikazanih slikovnih pik v okvirju se pojavi potreba po hitrejši sinhronizaciji. Nepravilno odkrivanje horizontalne in vertikalne sinhronizacije lahko pripelje do zmanjšanja prostorske opredelitve slik in v najslabšem primeru do nerazpoznavanja slik na zaslonu.

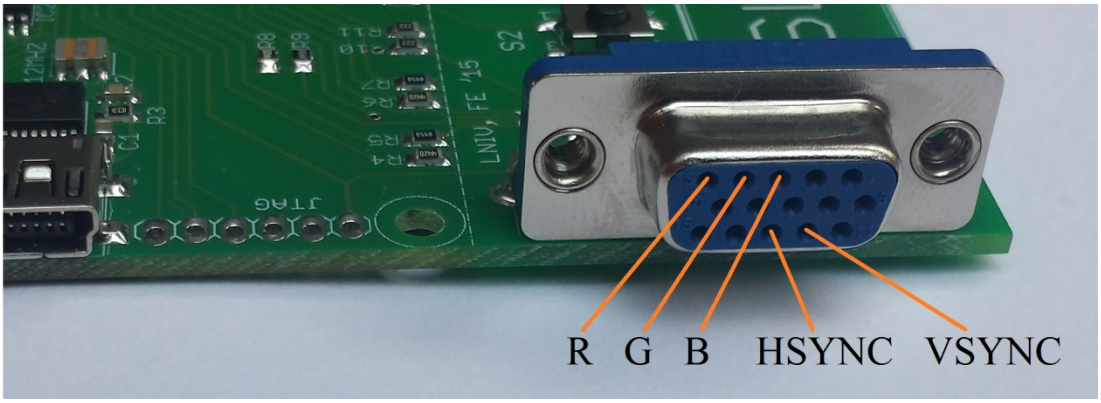
Pulzi *HSYNC* signala označujejo začetek in konec video linije (vrstice) in zagotovijo, da npr. monitor prikazuje slikovne pike med levim in desnim robom zaslona. Pulzi *VSYNC* signala označujejo začetek in konec okvirja, sestavljenega iz

video linij (vrstic) in zagotovijo, da so linije prikazane med zgornjim in spodnjim robom zaslona. Opis signalov VGA vmesnika prikazuje tabela 3.2.

Ime	Opis
R (R1..2)	Intenzivnost rdeče barve.
G (G1..2)	Intenzivnost zelene barve.
B (B1..2)	Intenzivnost modre barve.
HSYNC	Horizontalna sinhronizacija. Pisanje se prične v naslednji vrsti. Začne novo vrstico.
VSYNC	Vertikalna sinhronizacija. Pisanje se prične v prvi vrsti. Začne nov okvir.

Tabela 3.2: Opis signalov vmesnika VGA.

Slika 3.25 prikazuje VGA priključek na razvojni plošči in razpored signalov.



Slika 3.25: Priključek in signali VGA vmesnika na razširitveni plošči.

3.6 Rotacijski kodirnik (angl. *Rotary encoder*)

Med bolj pomembnimi elementi je bil inkrementalni rotacijski kodirnik (Slika 3.26). Je elektro-mehanski pretvornik, ki pretvarja vrtenje osi v električni signal. Inkrementalni rotacijski kodirniki dajejo z vrtenjem osi točno definirano število impulzov. Z njimi merimo kotne premike [22].

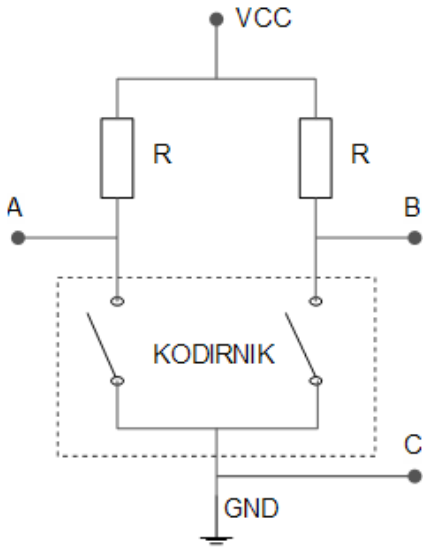
Slika 3.26: Inkrementalni rotacijski kodirnik [39].

Pri vrtenju osi proizvaja med seboj zamaknjene kvadratne impulze na izhodih *A* in *B*. Impulzi so med seboj zamaknjeni za kot 90°. Kateri signal bo prehiteval drugi signal, je odvisno od smeri vrtenja. Če vrtimo os v desno smer, bo *A* signal prehiteval *B* signal in obratno. Tabela 3.3 prikazuje potek stanj inkrementalnega kodirnika ob vrtenju osi.

V smeri urinega kazalca			**V nasprotni smeri urinega kazalca**		
POLOŽAJ	***A***	***B***	**POLOŽAJ**	***A***	***B***
1	0	0	**4**	1	0
2	0	1	**3**	1	1
3	1	1	**2**	0	1
4	1	0	**1**	0	0

Tabela 3.3: Stanja inkrementalnega rotacijskega kodirnika.

Inkrementalni kodirnik ima dva senzorja (stikala), ki s pomočjo dveh uporov, vezanih na *VCC*, generirata pravokotne impulze na izhodu *A* in *B*. Kodirnik ima še priključek *C,* kamor vežemo maso. Vezje rotacijskega kodirnika prikazuje slika 3.27.

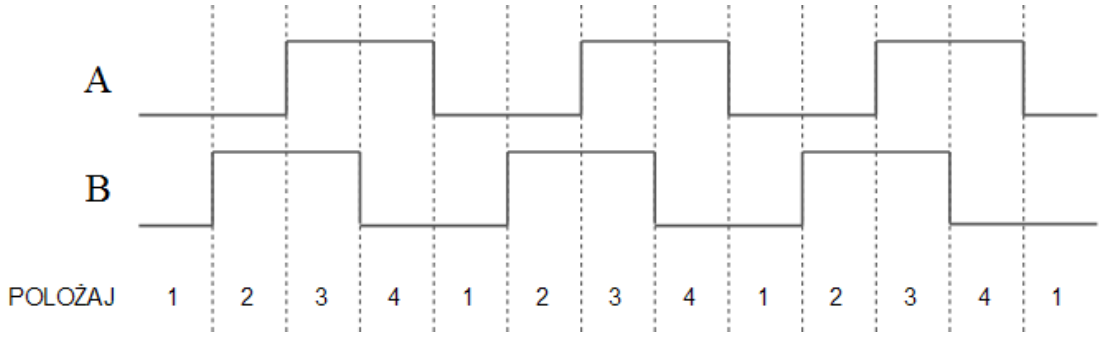


Slika 3.27: Vezje inkrementalnega rotacijskega kodirnika.

Primer prepoznavanja smeri vrtenja: če je bila zadnja vrednost signala *AB* npr. 00 (položaj 1) in je trenutna vrednost 01 (položaj 2), se je os obrnila za korak v smeri urinega signala.

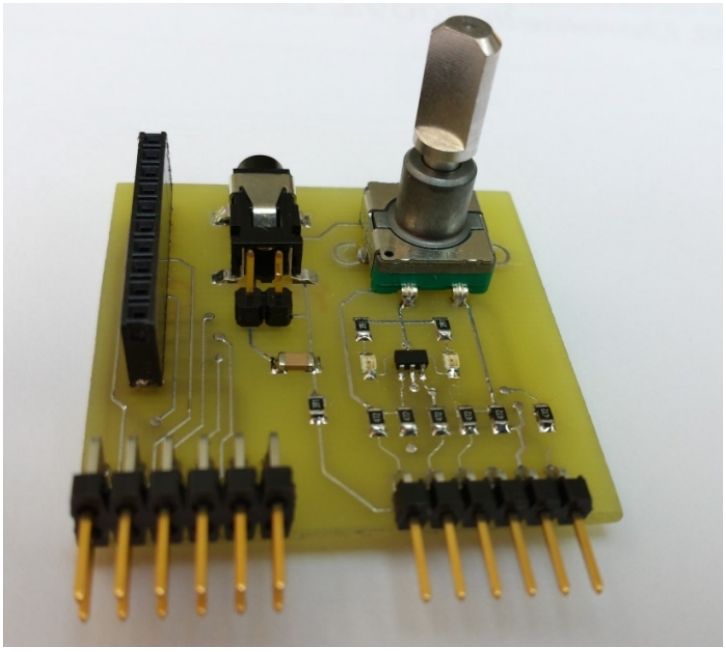
Za kodirnike z zobmi obstajajo različne poti za preklop stanj. V nekaterih primerih sta signala *A* in *B* vedno 00, medtem ko smo z osjo na zobu. Pojavi se cikel preklapljanja 00 → 00 za prehod iz enega zoba na drugega. V ostalih primerih je v položaju zobu izmenično preklapljanje vrednosti 00 → 11. Za vrtenje naprej dobimo prehod stanj 00 → 01 → 11 in za nazaj 00 → 10 → 11.

Branje vzorcev mora biti dovolj hitro, da ne zamudimo sprememb kode. Če se vrti os hitro in zamudimo branje, bomo videli le spremembo stanja 00 → 11. Za ta prehod ne vemo, v katero smer se je kodirnik obrnil. Tudi z branjem le enega signala rotacijskega kodirnika ni mogoče prebrati smeri vrtenja [22]. Časovni potek signala *A* in *B* ob vrtenju osi kodirnika v smeri urinega kazalca prikazuje slika 3.28.



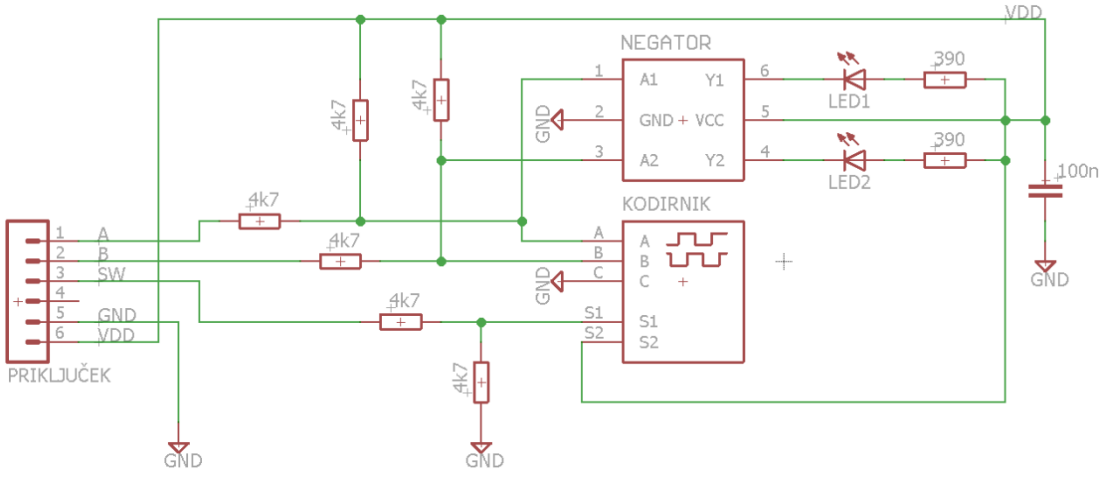
Slika 3.28: Časovni potek signala *A* in *B* ob vrtenju osi inkrementalnega kodirnika v smeri urinega kazalca.

Zaradi stiske s prostorom na razvojni plošči smo uporabili že prej načrtano ploščo z inkrementalnim rotacijskim kodirnikom (Slika 3.29).



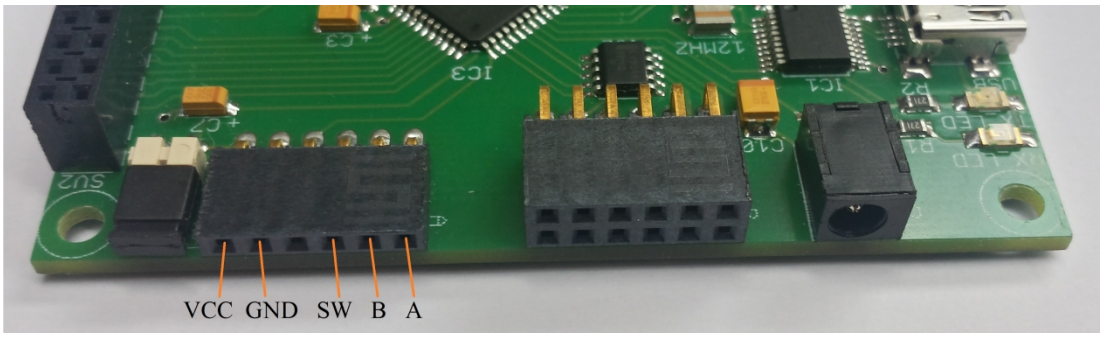
Slika 3.29: Plošča z inkrementalnim rotacijskim kodirnikom.

Plošča ima k rotacijskemu kodirniku dodani dve svetleči diodi skupaj z negatorjem signala *A* in *B*. Ob vrtenju osi kodirnika utripata svetleči diodi, ko je sklenjeno posamezno stikalo (A, B). Kodirnik ima vgrajeno tudi tipko (aktiviramo jo s pritiskom na os). Signal tipke *SW* je aktiven v visokem stanju (*VCC*). Na sliki 3.30 je električni načrt rotacijskega kodirnika.



Slika 3.30: Električna shema inkrementalnega rotacijskega kodirnika z negatorjem in svetlečima diodama.

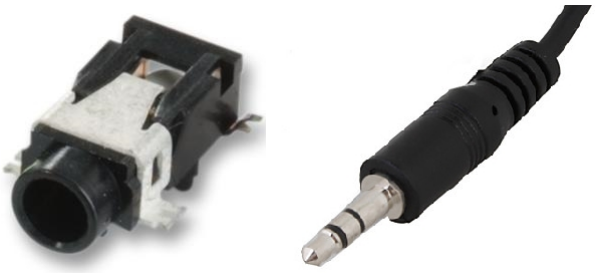
Priključek za ploščo kodirnika smo prilagodili z za 90° obrnjeno letvico na razširitveni plošči (Slika 3.31).



Slika 3.31: Priključki rotacijskega kodirnika na razširitveni plošči.

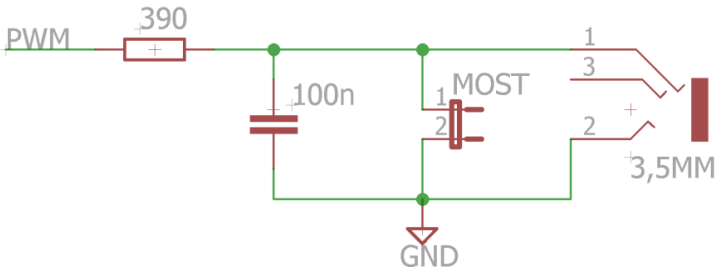
3.7 PWM izhod (angl. *Pulse-Width Modulation*)

Poleg rotacijskega kodirnika je na plošči rotacijskega kodirnika (Slika 3.29) izhod (3,5 mm avdio vtičnica (Slika 3.32)) za pulzno širinsko modulacijo PWM.



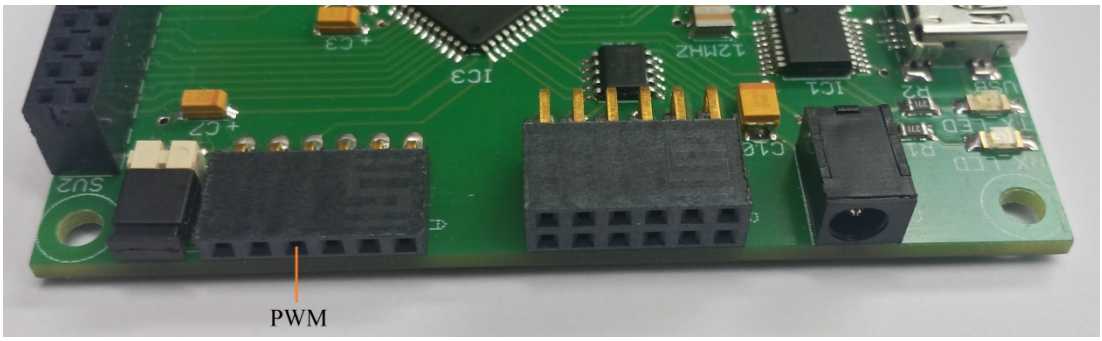
Slika 3.32: Priključka za PWM izhod.

Vezje vsebuje RC člen. Na vhod je pripeljan signal PWM, izhod RC člena gre poleg mase na 3,5 mm avdio vtičnico ali na mostiček (angl. *jumper*). Slika 3.33 prikazuje električni načrt vezja za PWM signal.



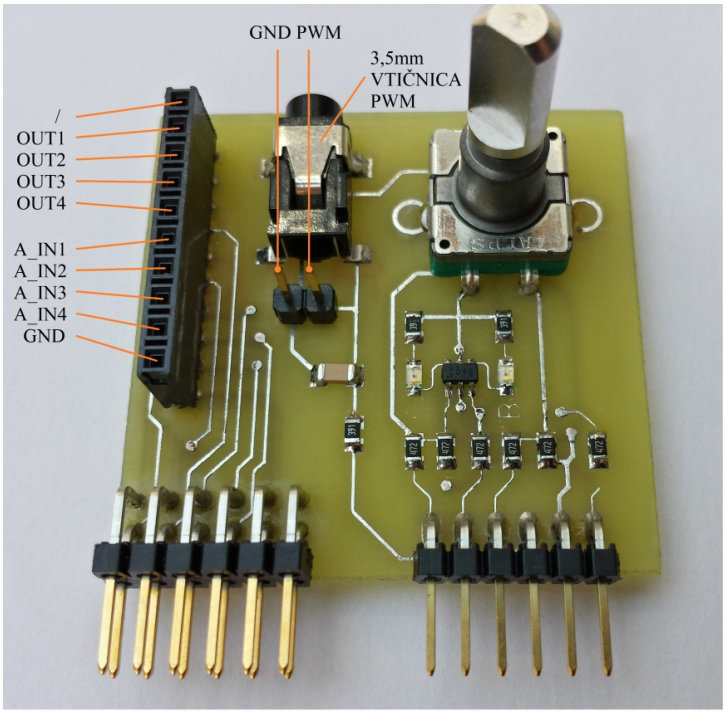
Slika 3.33: Električna shema vezja PWM signala.

Slika 3.34 prikazuje priključek PWM signala na razširitveni plošči, izhod PWM signala na plošči rotacijskega kodirnika pa prikazuje slika 3.35.



Slika 3.34: Priključek PWM signala na razvojni plošči in plošči rotacijskega kodirnika.

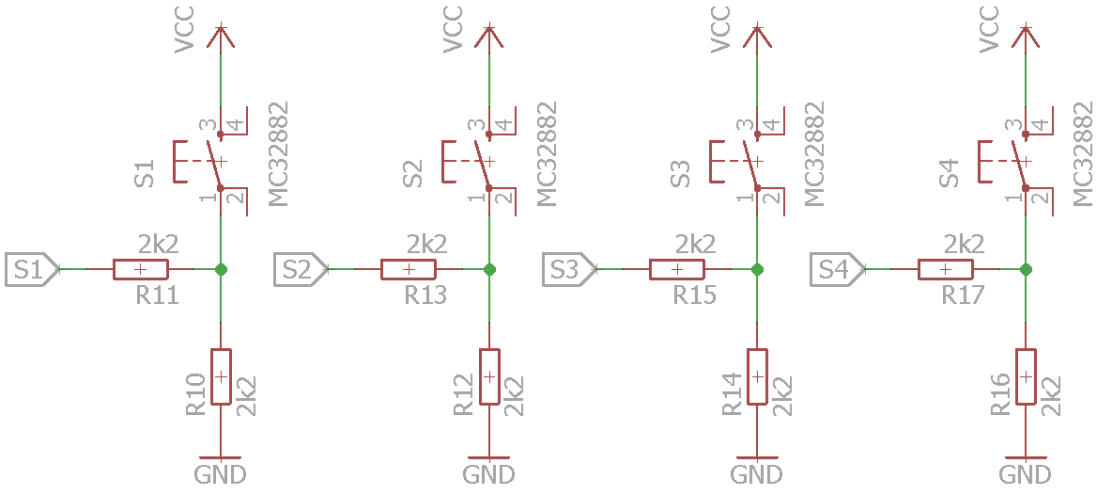
Plošča rotacijskega kodirnika ima poleg kodirnika in izhoda PWM tudi letvico za priklop izhodov za krmiljenje motorja *OUT(1..4)* in analognih vhodov *A_IN(1..4)*.



Slika 3.35: Plošča rotacijskega kodirnika; razporeditev signalov.

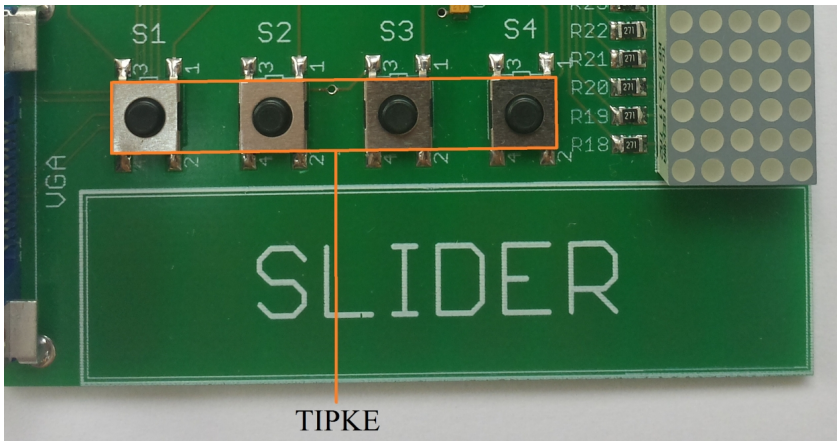
3.8 Tipke (angl. *Keys*)

Za uporabo tipk smo uporabili vezavo z dvema uporoma 2,2 kΩ na tipko proti masi (angl. *pull-down*). Pomeni, da ob pritisku tipke dobimo na izhodu za nadzor stanja tipk napetost *VCC* (+3,3 V). Imamo 4 tipke, zato imamo štiri izhode *S1*, *S2*, *S3* in *S4*. Električni načrt tipk prikazuje slika 3.36.



Slika 3.36: Električna shema štirih tipk.

Uporabili smo tipke *MC32882* s SMD podnožjem. Tipke na razvojni plošči prikazuje slika 3.37.

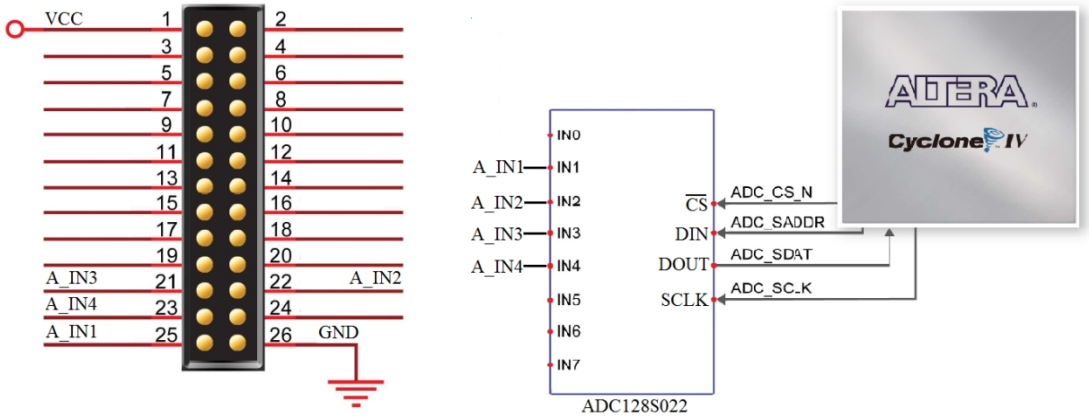


Slika 3.37: Tipke na razširitveni plošči.

3.9 Analogni vhodi (angl. *Analog input*)

Nismo pozabili na analogne vhode. Sprva smo želeli analogni vhod narediti s pomočjo AD pretvornika, vendar smo kasneje videli, da pretvornik vsebuje tudi plošča *DE0-Nano*.

Plošča vsebuje 12-bitni pretvornik *ADC128S022*. Pretvornik nudi prepustnost pretvorbe od 50 k do 200 k vzorcev na sekundo (angl. *samples per second*). Sprejme lahko do 8 vhodov signalov od *IN0* do *IN7* (Slika 3.38). Na razširitveno ploščo smo peljali le štiri analogne vhode (*A_IN1..4*). Slika 3.38 prikazuje razporeditev analognih vhodov na glavnem priključku razširitvene plošče in pretvorniku *ADC128S022* [4].



Slika 3.38: Povezava analognih vhodov na glavnem priključku razširitvene plošče z AD pretvornikom *ADC128S022* [33].

Lastnosti AD pretvornika:

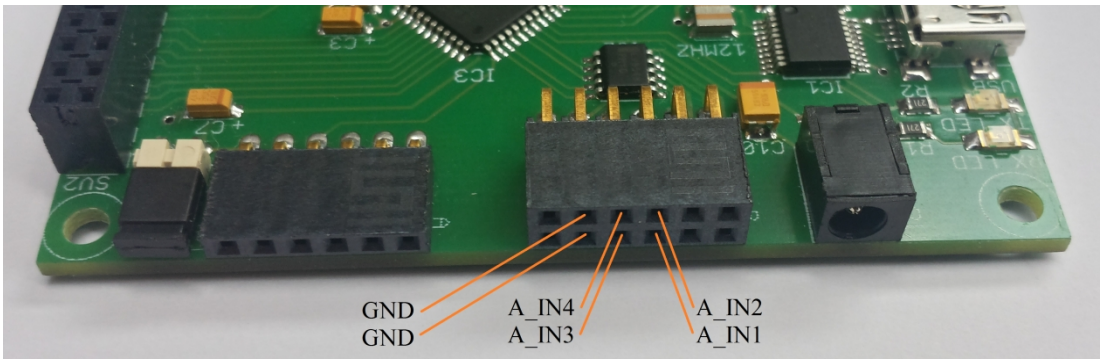
- 8 analognih vhodov.
- Območje napetosti analognih vhodov: od 0 V do 5,25 V.
- Hitrost pretvorbe: 50 ksps do 200 ksps.
- DNL (napajanje 5 V): +1.0 / -0.7 LSB (max).
- INL (napajanje 5 V): ±1.0 LSB (max).

Serijski podatkovni izhod *DOUT* je binaren in združljiv z več standardi, kot so: SPI, QSPI in mnoge druge skupine DSP serijskih vmesnikov. Opis signalov AD pretvornika prikazuje tabela 3.4.

Ime	Opis
IN0…7	Analogni vhodi.
SCLK	Digitalni vhod ure. Določeno območje frekvenc za vhod je od 0,8 MHz do 3,2 MHz. Ura nadzira postopek pretvorbe in izpis.
DOUT	Digitalni podatkovni izhod. Izhodni vzorci izstopajo s padajočo fronto ure *SCLK*.
DIN	Digitalni podatkovni vhod. Podatke v kontrolne registre pretvornika vpisujemo z naraščajočo fronto ure *SCLK*.
$\overline{CS}$	Izbira vezja. Ob padajočem robu $\overline{CS}$ se začne postopek pretvorbe. Pretvorba se nadaljuje dokler je $\overline{CS}$ v nizkem stanju.

Tabela 3.4: Opis signalov AD pretvornika *ADC128S022*.

Priključke analognih vhodov smo dodali poleg izhodov gonilnika motorja na 90° letvico (Slika 3.39). Analogne vhode lahko priključimo tudi na plošči rotacijskega kodirnika (Slika 3.35).



Slika 3.39: Priključki analognih vhodov na razširitveni plošči.

3.10 Razdeljevalnik (angl. *Multiplexer*)

Ob hitrem seštetju signalov vseh komponent smo prišli do problema, kako 26 vhodnih, 1 vhodno-izhodni in 8 vhodnih digitalnih signalov smiselno spraviti prek glavnega priključka. Priključek na plošči FPGA ponuja premalo digitalnih linij. Razmišljali smo, da bi signale razdelili v skupine. Skupine signalov bi prek glavnega priključka s ploščo FPGA komunicirale le ob naslovu, ki predstavlja skupino.

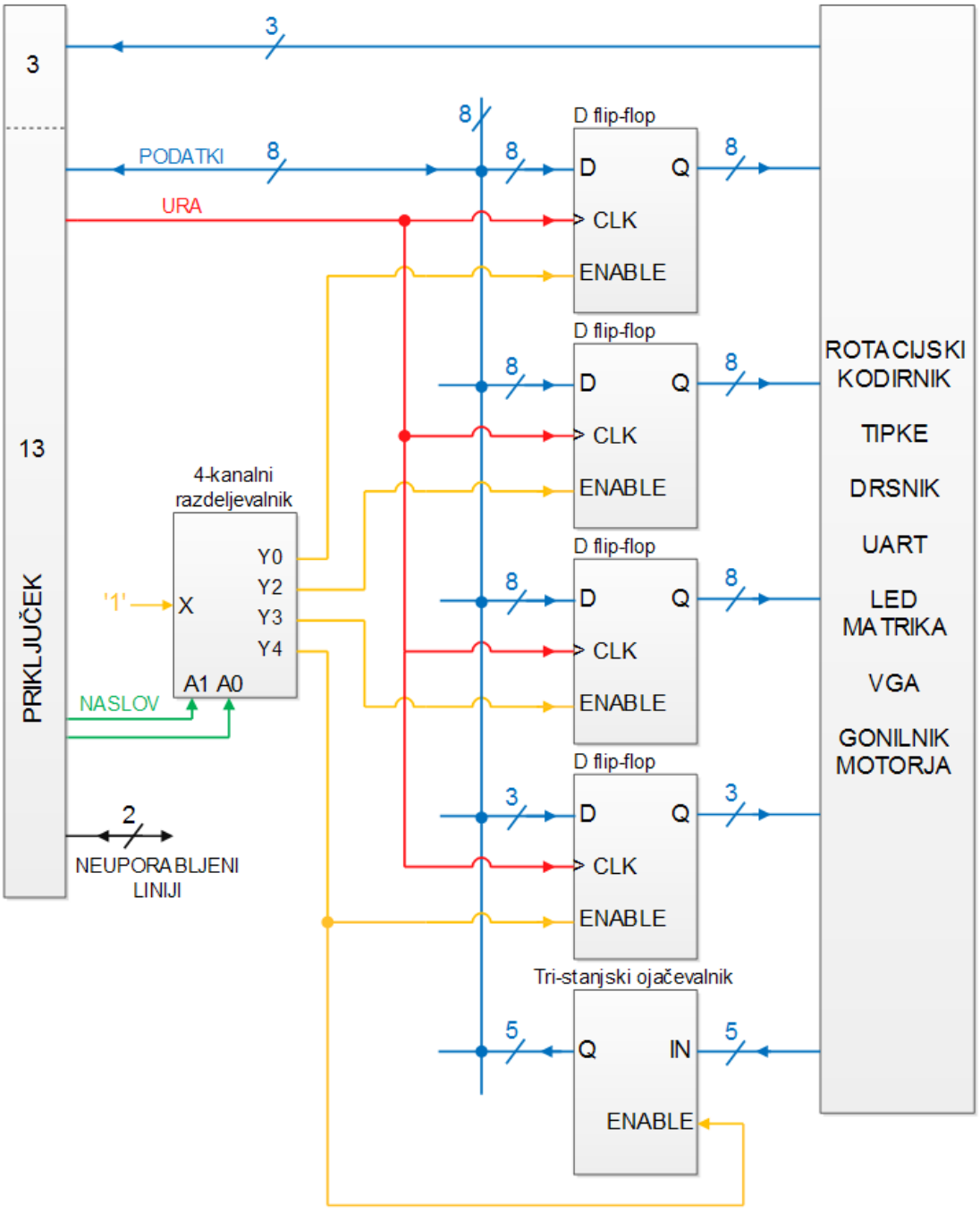
Skupine signalov smo morali prilagoditi glede na število in smer digitalnih linij. Priključek s 26 nožicami ima le 13 vhodno-izhodnih in 3 vhodne digitalne linije. Iz tabele 3.5 vidimo, da potrebujemo 35 linij (8 vhodnih in 27 vhodno-izhodnih) brez signalov, ki bi jih potrebovali dodatno za naslov in uro.

KOMPONENTA	SIGNALI KOMPONENT			GLEDE NA PRIKLJUČEK	
	VHODNI	VHODNO-IZHODNI	IZHODNI	VHODNI	VHODNO-IZHODNI
Rotacijski kodirnik	3			3	
Tipke	4			4	
Vzdolžni drsnik		1	1		2
UART vmesnik	1		1	1	1
Matrika LED			12 (kasneje 3)		12 (kasneje 3)
VGA vmesnik			8		8
Gonilnik motorja			4		4
SKUPAJ				**8**	**27 (kasneje 18)**

Tabela 3.5: Seštevek signalov po komponentah.

Če bi na skupino izbrali po 8 signalov, bi z dvema linijama za naslov (za štiri skupine) pridobili 32 linij, zasedli pa bi jih 11. Ostale tri signale bi speljali neposredno na priključek. Vezju, ki opravlja nalogo razdeljevanja, pravimo razdeljevalnik. Razdeljevalnik za takšno potrebo signalov na enem integriranem vezje ne obstaja. Poskušali smo ga realizirati iz več elementov.

Prva realizacija razdeljevalnika je bila iz D celic z dinamičnim proženjem (angl. *flip-flop*), tri-stanjskega ojačevalnika (angl. *three-state buffer*) in manjšega razdeljevalnika signalov (angl. *demultiplexer*). Električno shemo sestavljenega razdeljevalnika prikazuje slika 3.40.



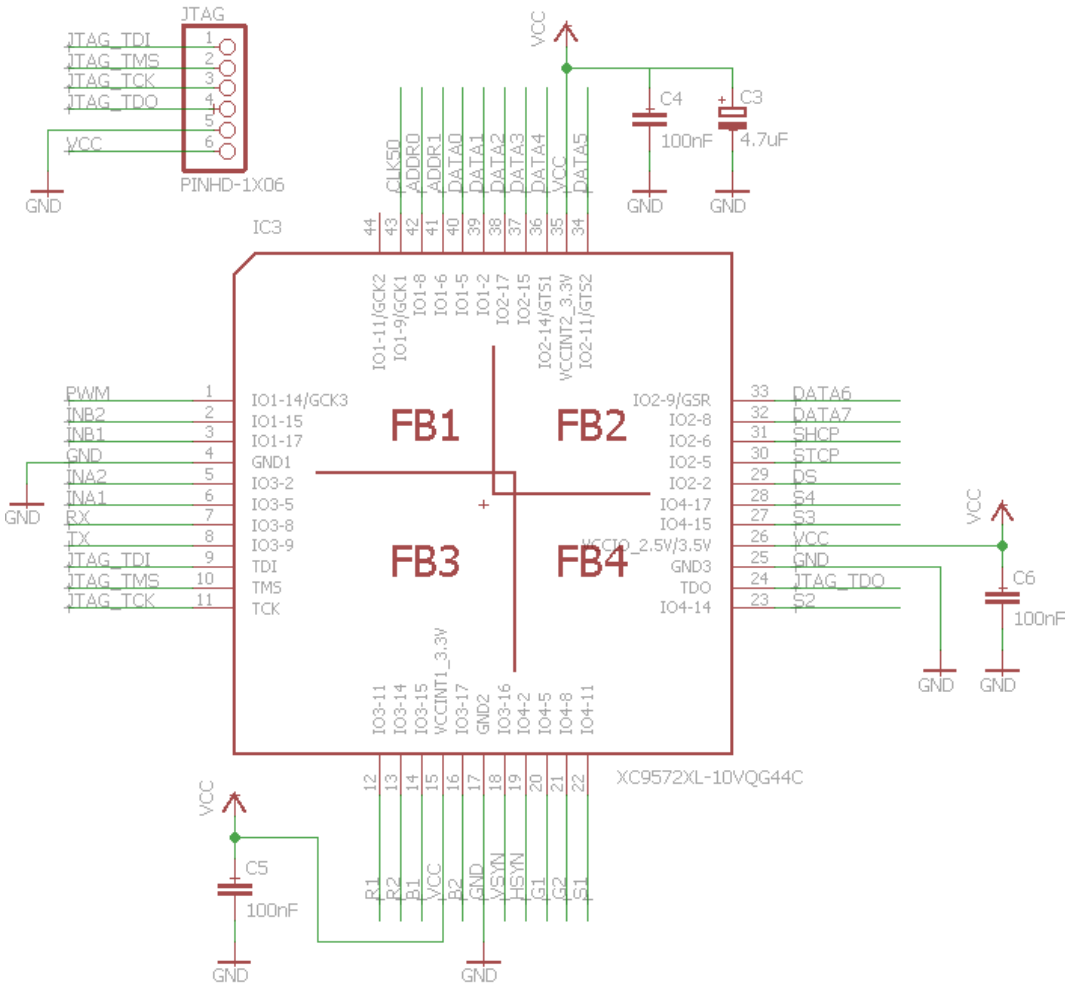
Slika 3.40: Električna shema razdeljevalnika z integriranim vezjem.

Za izdelavo primernega razdeljevalnika 8/32 bi za izhodne signale potrebovali tri 8-bitne D celice z dinamičnim proženjem in eno 4-bitno, en 8-bitni tri-stanjski ojačevalnik za vhodne signale in en 4-kanalni razdeljevalnik za izbiro. S kombinacijo naslova *A0* in *A1* peljemo logično enko na vhodu 4-kanalnega razdeljevalnika na vhod za omogočanje (angl. *enable*) D celic ali tri-stanjskega ojačevalnika. Ob logični enki in naraščajoči uri na vhodu D celice prepišemo stanja signalov podatkovnega

vodila h komponentam. Ob logični enki na vhodu tri-stanjskega ojačevalnika prepišemo 5 signalov na podatkovno vodilo.

Vidimo da ob taki konfiguraciji razdeljevalnika ostane pri 4-bitnem flip-flopu neizkoriščen en vhod, pri 8-bitnem tri-stanjskem ojačevalniku pa trije.

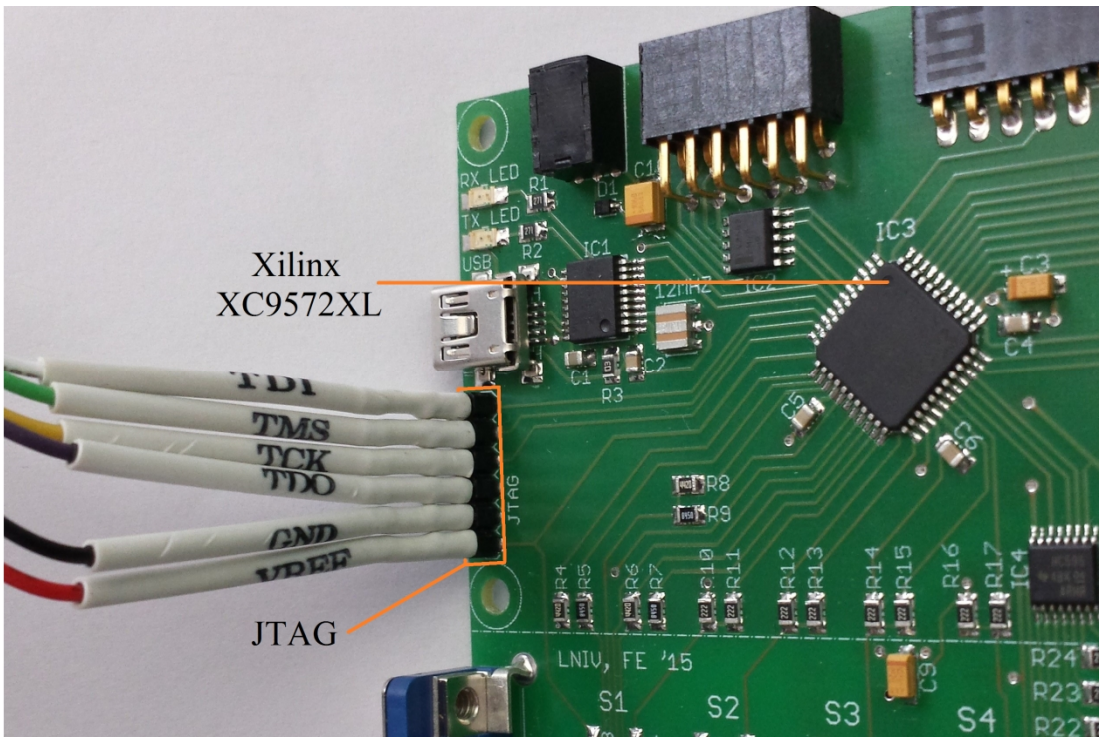
Ideja je bila, da bi se znebili 6 velikih integriranih vezij z veliko povezav (prevezav) in funkcijo razdeljevanja signalov naložili na prostorsko manj potraten CPLD. Začeli smo razmišljati o primerni izbiri. Odločali smo se med dvema glavnima proizvajalcema *Altera* in *Xilinx*. Ker programator za *Alterino* okolje ni bil na voljo, smo se odločili za CPLD družine *Xilinx*. Za nakup na spletni strani (*Farnell*) sta bila le dva primerna glede na potrebe in napetost napajanja 3,3 V. *Xilinx XC9572XL* in *Xilinx XC95144XL*. Prvi, manjši, s 44 nožicami omogoča 34 vhodno-izhodnih digitalnih linij. Drugi je mnogo večji od prvega s 100 nožicami in 81 vhodni-izhodnimi digitalnimi linijami. S sedanjim številom signalov prvi CPLD ni mogoče uporabiti.



Slika 3.41: Električna shema za CPLD in vmesnik JTAG.

Po želji krčenja signalov komponent smo videli, da lahko veliko linij prihranimo pri signalih matrike LED s pristopom serijskega vpisa podatkov. Več o tem je opisano na strani 13. Iz sedanjih dvanajstih linij smo prišli na 3 in jih tako prihranili 9.

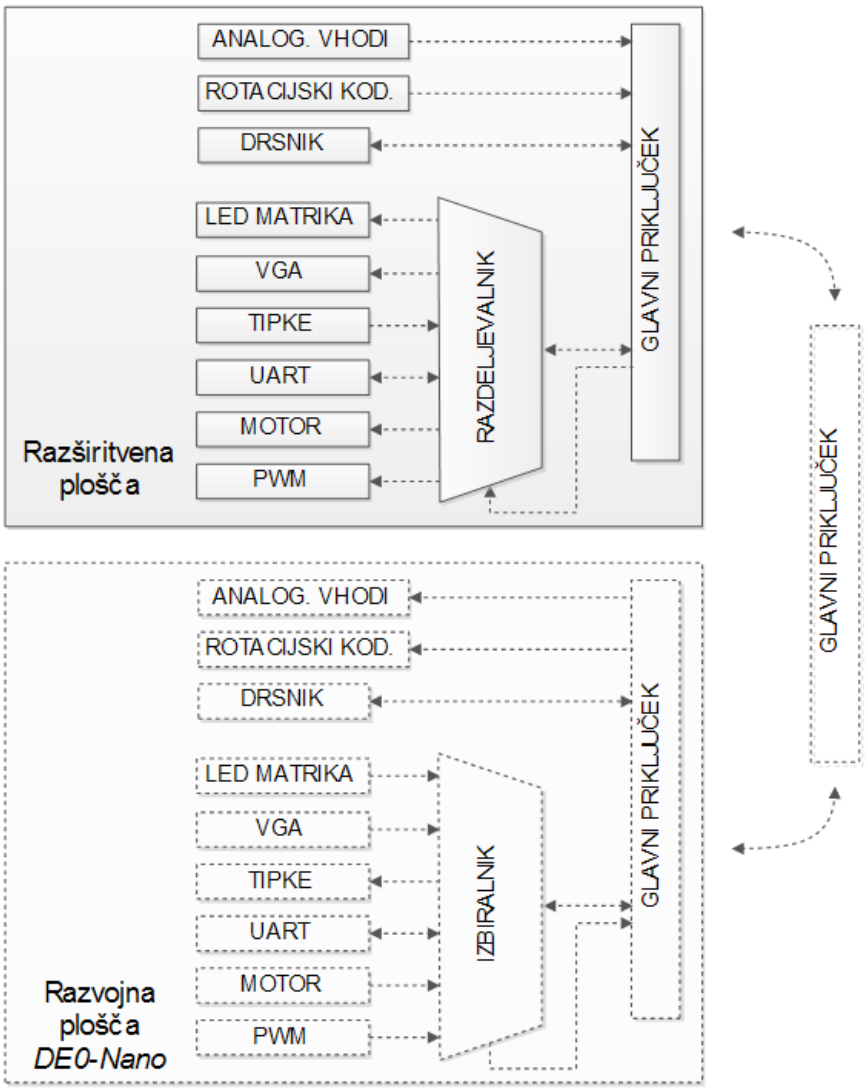
Če sedaj pogledamo tabelo 3.5, vidimo, da po krčenju signalov skupno potrebujemo le še 26 linij. Vendar ne smemo pozabiti, da za realizacijo razdeljevalnika s CPLD-jem potrebujemo še dodatnih 11 linij, 8 podatkovnih in 3 krmilne. Sedaj smo skupno na 37-ih linijah. Zmanjkale so nam 3 digitalne linije, vendar če signale pametno razporedimo tudi po ostalih linijah na glavnem priključku, je uporaba CPLD-ja *XC9572XL* v našem primeru izvedljiva. Programiranje razdeljevalnika je opisano v poglavju 5.1. Električni načrt in razporeditev signalov na priključke CPLD-ja prikazuje slika 3.41. Izbran CPLD in razporeditev signalov na priključku vmesnika JTAG na razširitveni plošči prikazuje slika 3.42.



Slika 3.42: CPLD in vmesnik JTAG na razširitveni plošči.

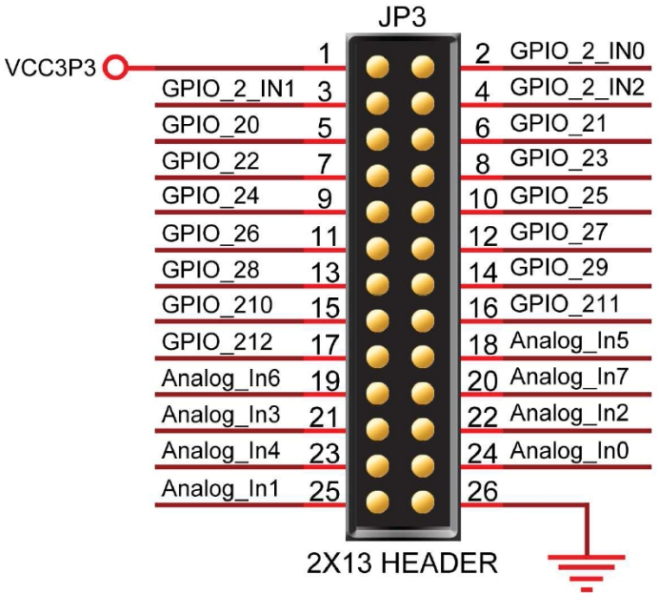
3.11 Združitev komponent in kompenzacija motenj

Sedaj lahko vseh 10 vezij združimo v eno vezje. Lažjo predstavo razvrstitve signalov komponent razširitvene plošče prek razdeljevalnika in glavnega priključka prikazuje slika 3.43.



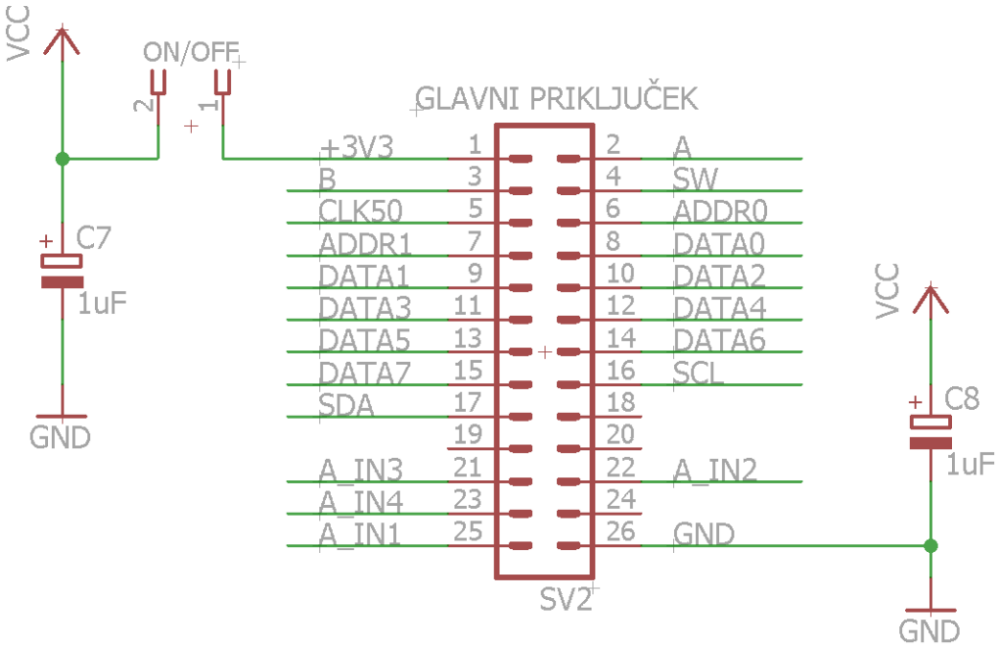
Slika 3.43: Shematičen prikaz razvrstitve signalov komponent prek glavnega priključka.

S sheme vidimo, da so digitalni signali drsnika in rotacijskega kodirnika na razširitveni plošči speljani neposredno na glavni priključek, medtem ko signali ostalih komponent (razen analognih vhodov) potujejo prek razdeljevalnika. S slike 3.43 vidimo, da za dostop do komponent prek razdeljevalnika potrebujemo na razvojni plošči programsko kodo izbiralnika. Več o izbiralniku bo opisano v poglavju 5. Razpored signalov na glavnem priključku razvojne plošče *DE0-Nano* prikazuje slika 3.44.



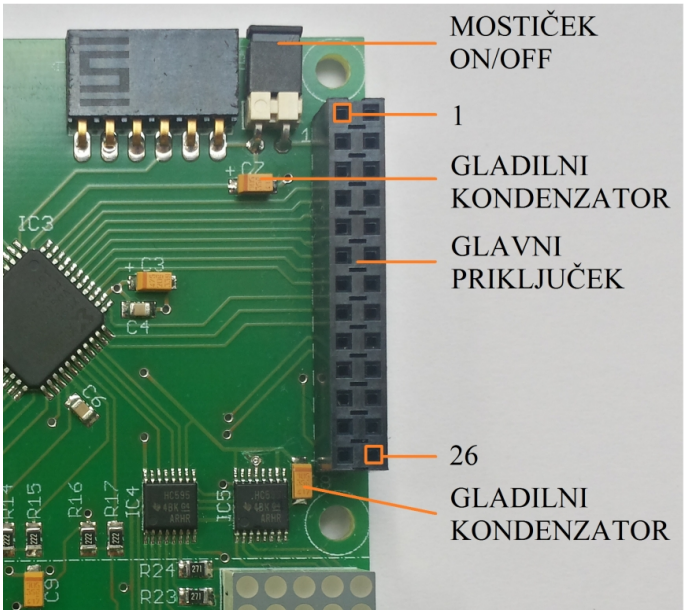
Slika 3.44: Razvrstitev signalov na glavnem priključku razvojne plošče *DE0-Nano* [33].

Na glavnem priključku smo z vezjem razširitvene plošče zapolnili vseh 16 digitalnih linij. Za signale razdeljevalnika *CLK50, ADDR(0..1)* in *DATA(0..7)* smo porabili 11 linij, 2 liniji *SCL* in *SDA* za integrirano vezje drsnika in 3 vhodne linije za signale *A*, *B* in *SW* rotacijskega kodirnika. Za analogne vhode *A_IN(0..3)* smo zapolnili 4 analogne linije. Za napajanje razširitvene plošče skrbi 3,3 V napajalna linija *+3V3* in masa *GND*. Razpored signalov na glavnem priključku razširitvene plošče prikazuje slika 3.45.



Slika 3.45: Električna shema signalov in napajanja na glavnem priključku razširitvene plošče.

Vezje razširitvena plošča vsebuje 6 integriranih vezij, ki lahko ob sočasnih preklopih povzročajo spremembe napajalne napetosti, z vklopi pa hitre spremembe toka. Za zagotavljanje konstantne napetosti in zadostnega toka za vse komponente pri hitrih digitalnih vezjih moramo blizu napajalnega priključka dodati gladilni (angl. *bulk*) kondenzator. Najpogosteje se ga postavi ob napajalni priključek. Kapacitivnosti gladilnih kondenzatorjev so od 1 µF do 100 µF. Običajno zadošča en gladilni kondenzator na plošči tiskanega vezja [23]. Mi smo v vezje vključili dva s kapacitivnostjo po 1 µF. Ker vezje matrike LED predstavlja veliko breme in imamo na razširitveni plošči tudi druge komponente, bomo kondenzatorja razporedili. Enega bomo namestili v bližino priključka mase *GND* in drugega v bližino napajalne linije *VCC*. K vezju razširitvene plošče smo dodali tudi mostiček *ON/OFF* za izklop napajanja, ko plošče ne uporabljamo. Slika 3.46 prikazuje glavni priključek in mostiček na razširitveni plošči.



Slika 3.46: Glavni priključek na razširitveni plošči.

4 Izdelava tiskanega vezja

Ko imamo narejene vse električne sheme, se lotimo načrtovanja tiskanine. Pri načrtovanju je potrebnega veliko znanja, da izdelamo tiskano vezje, ki bo ustrezalo vsem fizičnih in električnim zahtevam. Pri izdelavi vezij z večanjem števila plasti tiskanega vezja narašča tudi zahtevnost načrtovanja. Nekateri načrtovalci gledajo na načrtovanje tiskanih vezij kot na umetniško delo. Za razširitveno ploščo smo uporabili 2-plastno tiskano vezje.

Predvsem za tiskana vezja, ki delujejo pri visokih frekvencah, je pomembno pravilno načrtovanje. Z neustreznim načrtovanjem nastopijo neželeni učinki, ki so lahko razlog za nepravilno delovanja vezja. Glavna vzroka za nedelovanje sta odboj signala in presluh med dvema signalnima linijama. Učinke je potrebno v času načrtovanja predvideti in jih kar se da zmanjšati.

Pri načrtovanje tiskanin obstajajo standardi. Standarde nadzoruje inštitut IPC, glavni standard pri načrtovanju tiskanin je *IPC-2221* [24].

4.1 Pravila za risanje tiskanega vezja

Na začetku načrtovanja se moramo najprej zavedati kritičnih delov vezja. Največji viri sevanja so urini signali, hitre podatkovne linije ter napajalne linije, ki napajajo bodisi hitra digitalna vezja ali stikalne napajalnike. Od teh linij moramo kolikor se da umakniti občutljive dele vezja. Med občutljive dele spadajo kontrolne linije, prekinitvene linije in linije za ponastavitev (angl. *reset*). Da ugodimo tem zahtevam, moramo na začetku načrtovanja dele vezja, ki predstavljajo vire motenj, postaviti stran od delov vezja ali elementov, ki so nanje občutljiva. Tako se izognemo napakam ali celo onemogočenemu delovanja vezja. Viri motenj so lahko tudi zunanjega izvora [25].

Vedeti moramo, da imajo vse povezave (angl. *tracks*) na tiskanem vezju tudi svojo ohmsko upornost, kapacitivnost in induktivnost, zato naj bojo povezave čim krajše. Kritična dolžina bakrene povezave je, ko se časovni potek električnega

signala približa dolžini bakrene povezave. Pri visokofrekvenčnih vezjih z dolgimi povezavami lahko pride do odbojev, zato moramo linije pravilno zaključiti. Zagotoviti moramo dobro načrtovanje mas [26].

Širino povezav določimo glede na zahteve vezja in oddaljenosti med komponentami. Širša povezava na tiskanini pomeni manjšo ohmsko upornost in manjšo induktivnost. Širina povezav vpliva na tok, ki ga lahko prenašamo vzdolž povezave. Izračuni samih širin glede na tokovno zmogljivost so zelo zapleteni. Na tok ne vpliva samo širina povezave, ampak tudi debelina bakrenega nanosa [24]. V našem projektu smo uporabili različne širine od 0,2 mm do 0,4 mm za napajalne linije.

Poleg širine in dolžine povezav je pomemben tudi razmik med povezavami. Če so povezave ali spajkalna očesca (angl. *pads*) načrtovane preveč skupaj, lahko pri izdelavi tiskanega vezja pride do kratkega stika. Za običajno načrtovanje so minimalni razmiki pri SMD komponentah do 0,2 mm. Razmiki povezav so odvisni predvsem od tehnologije izdelave in napetosti na povezavah. Za napetosti do 15 V so razmiki povezav minimalno 0,1 mm [24].

Pri povezovanju je pomembna tudi oblika povezav. Povezave ne smejo biti pod pravim kotom ali kotom, večjim od 90°. Ostri koti povzročajo elektromagnetne motnje EMI, izgubimo lahko tudi prostor za druge povezave. Povezave naj bodo poševne pod kotom 45°. Ker je oblika povezovanja pomembna, povezave rišemo vzporedno in jih nikoli ne zaključujemo krožno. Izogibamo se povezavam med nožicami integriranih vezij, če so te narazen manj kot 2,54 mm. Lahko jih povezujemo pod ohišjem. Napajalne povezave rišemo širše. Paziti moramo na induktivne zanke (presluhi), zato obe napajalni liniji peljemo skupaj [24].

Za povezave med plastmi tiskanine uporabljamo medplastne povezave (angl. *vias*), ki med plastmi vezja povezujejo povezave ali plasti vezja. Imenujemo jih viasi ali skoznjiki. Podobne so očescem z izvrtinami. Za manjše parazitne induktivnosti naj bodo izvrtine čim manjše med 0,5 mm in 0,7 mm [26]. V našem primeru so velikosti 0,6 mm.

Pomembna so tudi spajkalna očesa, kamor se spajkajo oz. pritrdijo elementi. Vsak proizvajalec za element predpiše optimalno velikost. Za priključna očesca z izvrtinami velja pravilo, da naj bo očesce za faktor 1,8x večje od premera izvrtine oz. za najmanj 0,5 mm [26]. Za vse elemente smo SMD podnožja narisali sami, zato smo določili tudi optimalne velikosti očes, kot jih je predpisoval proizvajalec elementa. Večina SMD elementov uporablja pravokotne oblike spajkalnih očes.

Poligoni (angl. *polygons*) so polnila, ki zapolnjujejo določene dele tiskanine z bakrenim nanosom in so ločeni od povezav in spajkalnih očesc. Vseeno pa jih ne

rišemo brez, da ne bi bile povezane nikamor. Z njimi lahko poskrbimo za enakomerno napajanje komponent po celotni površini tiskanine. To lahko naredimo z večjimi napajalnimi površinami, saj bomo tako zmanjšali induktivnost in impedanco napajalnih priključkov ter tako vpliv motenj, ki jih povzročajo posamezni elementi. Dobro delovanje visokofrekvenčnih vezjih lahko dosežemo z dobro maso. Pomeni, da za ozemljitev porabimo bakrene površine ali poligone mase ob tem pa dodamo nekaj medplastnih povezav s povezavami mase na drugih plasteh [24].

Pri uporabi aktivnih elementov ne zadostuje samo dobra ozemljitev, zato moramo blokirati napajalne priključke. Blokiranje je omejitev razširjanja motenj v vezju, ki se onemogoči z energijo, shranjeno v blokirnem elementu. Velike preklopne tokove zgladimo z blokirnimi kondenzatorji. Postavimo jih čim bližje napajalnim priključkom aktivnih elementov, lahko tudi na drugo stran tiskanega vezja pod element. Kondenzatorje povežemo tako, da napajalno linijo povežemo najprej s kondenzatorjem, nato pa z integriranim vezjem. Tipične vrednosti kondenzatorjev so 100 nF [27].

Napajalne linije povežemo še z gladilnimi kondenzatorji, ki jih postavimo čim bližje napajalnega priključka. Ti skrbijo za stabilnost napajalne napetosti in toka celotnega vezja [23]. Na tiskano vezje smo postavili dva, pri priključku *VCC* in mase.

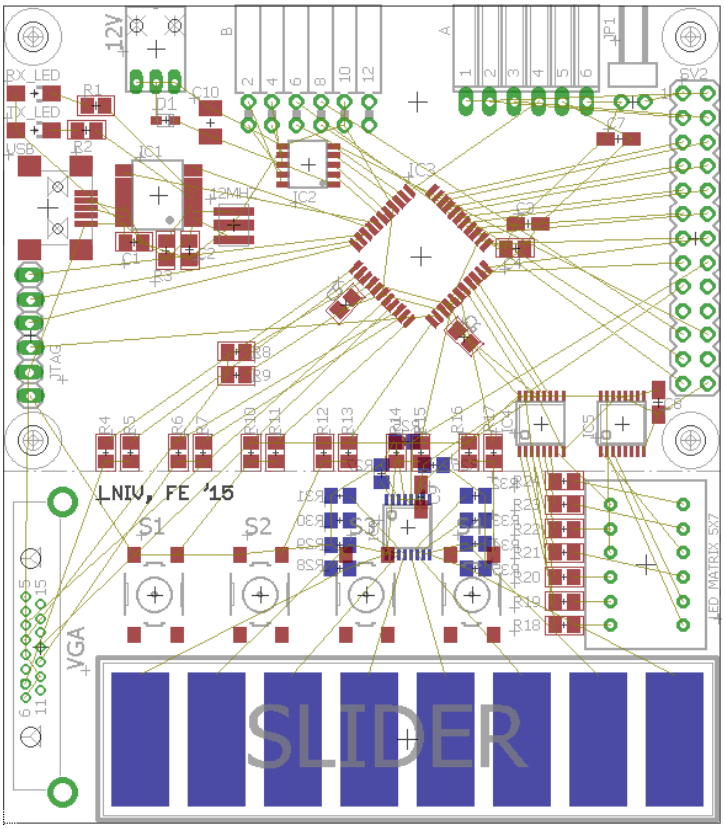
4.2 Risanje tiskanine

Pred začetkom risanja tiskanine je pomembna pravilna postavitev elementov. Skicirali smo približno razporeditev priključkov in pomembnih komponent. Velikost tiskanine smo omejili na dolžino razvojne plošče 75 mm. Dolžina naj bo fiksna, spreminjali smo le širino plošče glede na zmožnosti. Že pri načrtovanju komponent smo vedeli, da če želimo izdelati čim manjšo razširitveno ploščo, moramo paziti na velikost elementov. Prednost so imeli tisti elementi, ki imajo manjša podnožja. Integrirana vezja z DIP podnožji zato ne pridejo v poštev. Zaradi zahtevnosti spajkanja nismo uporabili tudi podnožij, ki imajo priključke pod ohišjem. To so predvsem DFN in QFN podnožja.

Glavni priključek postavimo na skrajno desno stran. Letvici z 90° zamaknjenimi priključki s šestimi in dvanajstimi nožicami imata fiksno medsebojno razdaljo (zaradi uporabe vezja z rotacijskim kodirnikom). Želeli smo, da je zgornji rob plošče poravnan z robom razvojne plošče, letvici postavimo na zgornji rob razširitvene plošče. Ploščo smo širili le na spodnji rob, toliko kot potrebujemo za postavitev matrike LED, območje drsnika, VGA priključka in štirih tipk. USB

priključek skupaj s svetlečima diodama UART komponente in JTAG vmesnik smo postavili skrajno levo pod razvojno ploščo. Zgoraj levo smo dodali napajalni priključek za gonilnik motorja. Večina komponent komunicira s CPLD-jem zato smo ga postavili na sredino.

Za izris tiskanine smo uporabili orodje *CadSoft Eagle*. Na začetku uvozimo vse elemente z izrisanih električnih shem v okno za izdelavo tiskanine PCB. Najprej se lotimo s postavitvijo štirih izvrtin za pritrditev tiskanega vezja in prostora okrog njih. Postavimo glavni priključek, pomembne elemente in ostale priključke. Elemente, ki jih ne moremo veliko prestavljati po tiskanini, postavimo najprej. Sledijo večja podnožja, razvrstimo jih glede na bloke z električnega načrta. Ob tem s predvidevanjem puščamo prostor za ostale elemente.



Slika 4.1: Razporeditev elementov na razširitveni plošči.

Iz izkušenj, kolikor jih imam pri načrtovanju, vem, da lahko že nepravilna rotacija čipa kasneje povzroči kup neuspešnih povezav ali kup prevezav. Po grobi postavitvi komponent sem si skiciral tudi položaj integriranih vezij in vseh pomembnih povezav. Menim, da je to za začetek ključnega pomena. »Veliko načrtovalcev pravi, da je pravilna postavitev elementov več kot pol narejenega dela«. Pri postavljanju ni napisanega pravila, kako iz načrta naredimo dobro tiskano vezje.

Pri postavljanju komponent se potrudimo poravnati komponente čim bolj v linije, priključke pa poravnamo z robovi. Če se da, naj bodo elementi obrnjeni v isto smer, vendar je to le stvar oblikovanja. Manjše število medplastnih povezav je pomembnejše, gledano tudi s stroškovnega vidika in ne le zaradi električnih lastnosti. Slika 4.1 prikazuje razporeditev elementov na razširitveni plošči.

Ko končamo s postavitvijo komponent, začnemo s povezovanjem. Pravila povezovanja so opisana v poglavju 4.1. Najprej se lotimo kritičnih povezav in povezav znotraj blokov. Bloke povežemo med seboj in dodamo napajanje.

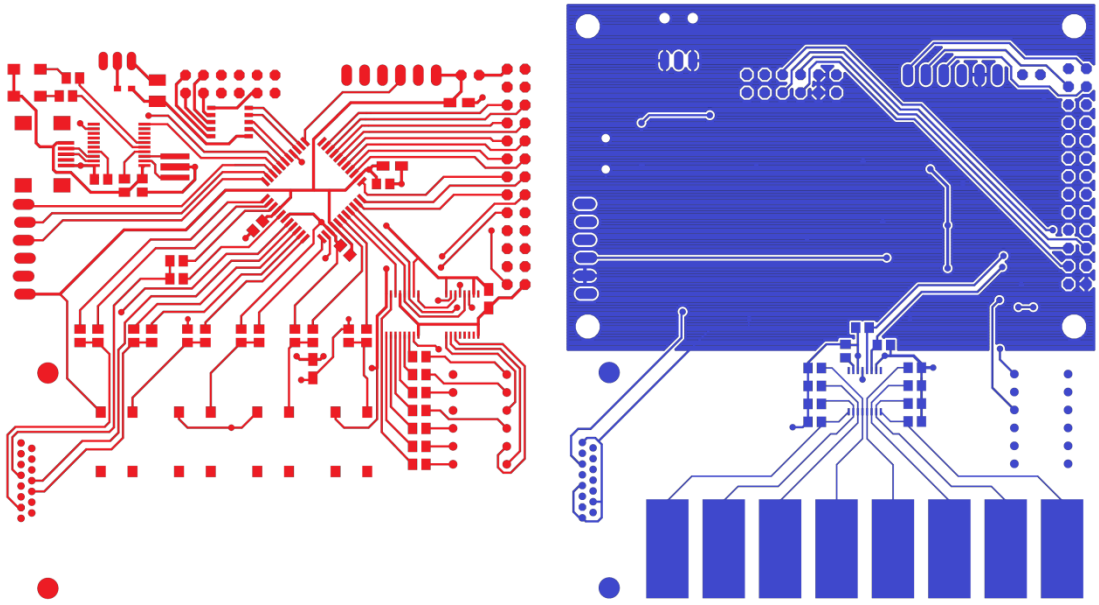
Kmalu se pokaže, kako uspešni smo bili pri postavitvi elementov. Če se nam izide večji del povezovanja, smo grobi del postavitve dobro opravili. Sedaj sledi optimizacija tiskanine. Cilj optimizacije je optimizacija pravil, natančnejša postavitev elementov, oblikovanje povezav, razmikov, števila medplastnih povezav itd. Elemente in povezave premikamo, da optimiziramo postavitve elementov in povezave. Če smo natančni pri načrtovanju, nam lahko tudi ta postopek vzame veliko časa. Sam sem se optimizacije lotil posamezno, od bloka do bloka. Najprej sem optimiziral povezave znotraj blokov, kasneje pa s premikanjem (delov) blokov optimiziral povezave med bloki.

Večina programov omogoča tudi preverjanje uspešnosti oz. pravilnost načrtovanja. To omogoča funkcija DRC. S funkcijo DRC preverimo pravilnost širine povezav, razmike med povezavami in spajkalnimi očesci, velikost spajkalnih očesc, nepravilnost povezav, ki nastanejo zaradi nepravilnih povezav v centru spajkalnih očesc in sestavljanja povezav, velikost izvrtin itd. Program nam prikaže napako glede na vstavljene pogoje.

Ker je naše vezje visokofrekvenčno, zapolnimo večje prazne površine na spodnji strani tiskanega vezja s poligoni mase. Na koncu popravimo tudi napise, oznake (imena komponent iz električne sheme) in označimo polaritete. Nepotrebne napise pobrišemo. Preverimo velikost izvrtin in montažnih lukenj, možnost prekrivanja ohišij elementov, dobrodošlo je preverjanje podnožij, še enkrat poženemo funkcijo DRC (če program ne omogoča sprotno preverjanje napak) itd.

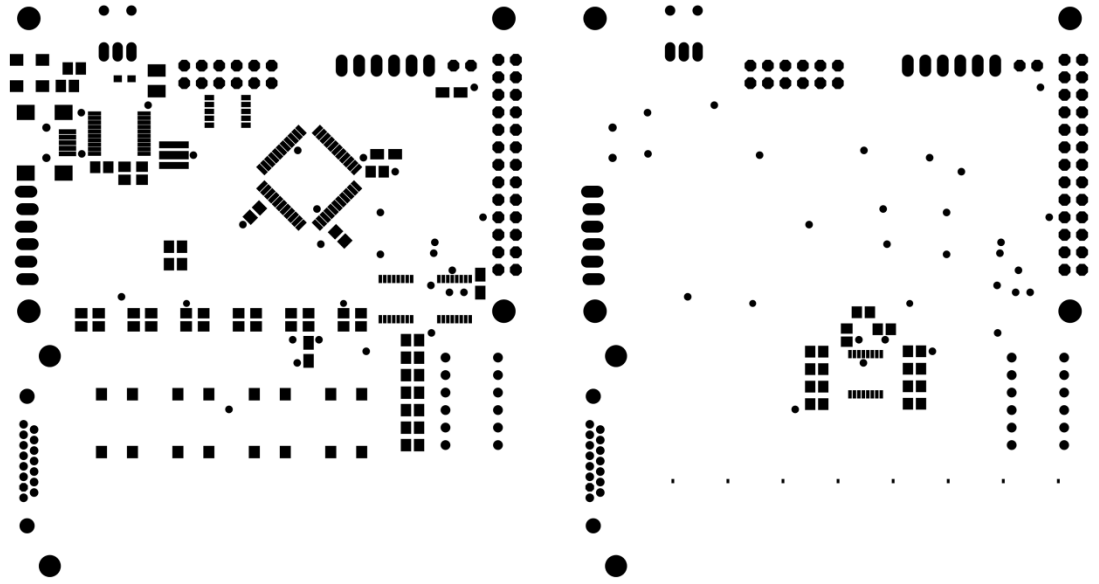
Ko je tiskanina izrisana, proizvajalcu tiskanine podamo datoteke za kreiranje mask (angl. *gerber files*) in za izdelavo izvrtin (angl. *NC drill files*).

Zgornjo in spodnjo masko bakrene plasti (angl. *layer mask*) razširitvene plošče prikazuje slika 4.2.



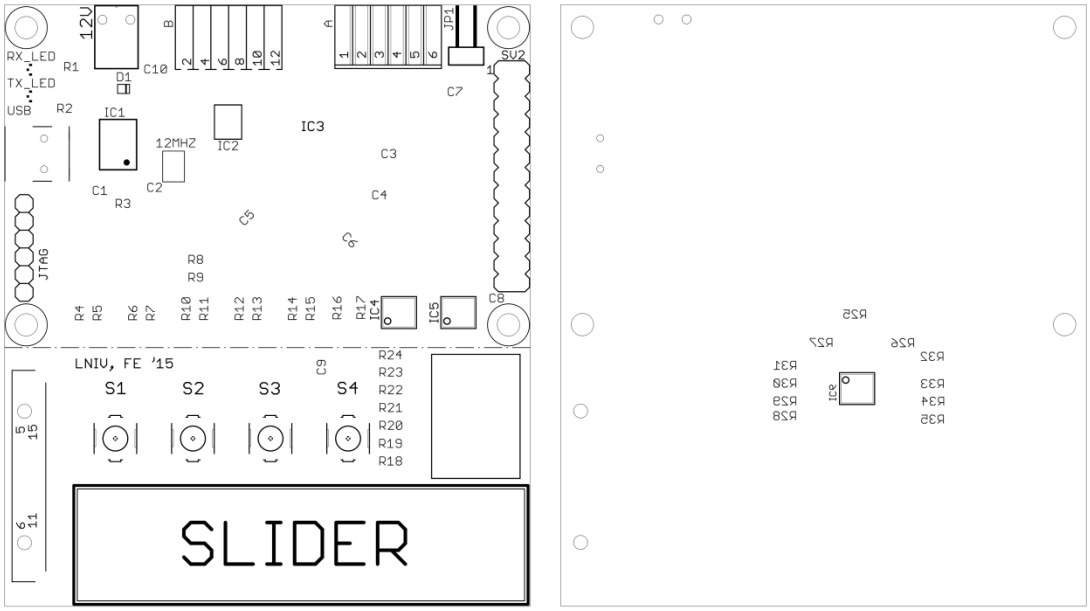
Slika 4.2: Zgornja in spodnja maska bakrene plasti razširitvene plošče.

Namen maske za spajkanje (angl. *solder mask*) je zaščita celotne tiskanine z zaščitnim lakom razen nad spajkalnimi očesci. Tu so odprtine, ki so namenjene spajkanju. Zaščitni lak je namenjen zaščiti tiskanine pred zunanjimi vplivi in kratkimi stiki, ki bi lahko nastali zaradi spajkanja. Barva zaščitnega laka je običajno zelena in se nanese neposredno na bakrene povezave [24]. Masko za spajkanje razširitvene plošče prikazuje slika 4.3.



Slika 4.3: Zgornja in spodnja maska za spajkanje razširitvene plošče.

Maska belega tiska (angl. *overlay mask*) označuje komponente na tiskani plošči. Beli tisk označuje obris komponent, oznake komponent brez vrednosti, polarizacijo itd. Običajno se za beli tisk uporablja belo barvo [24]. Masko belega tiska razširitvene plošče prikazuje slika 4.4.



Slika 4.4: Zgornja in spodnja maska belega tiska razširitvene plošče.

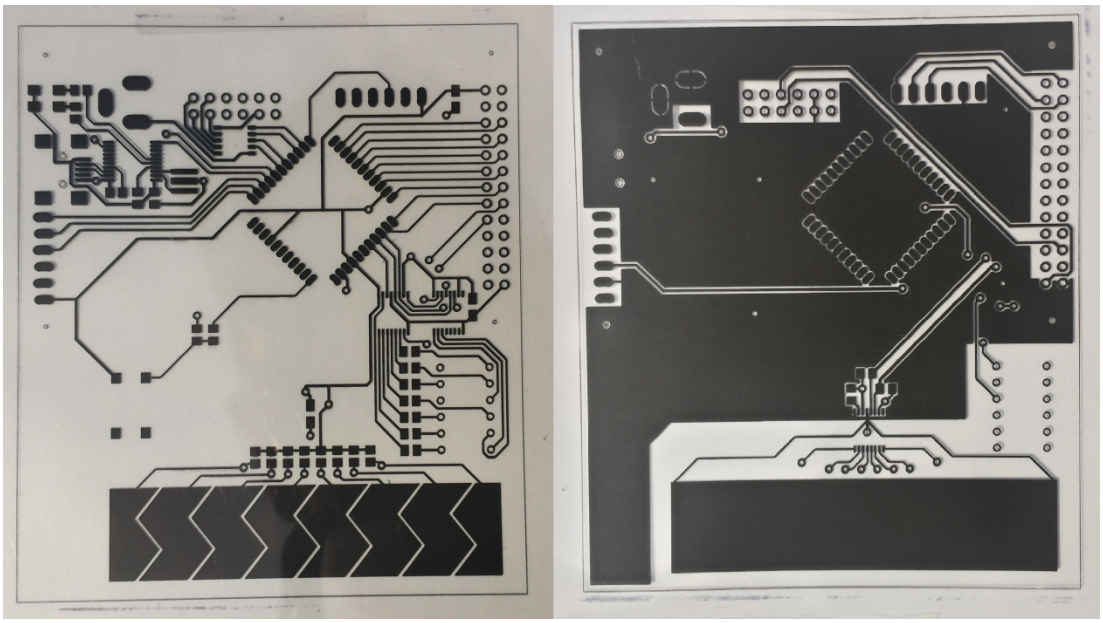
Da bi se prepričali v praktično delovanje vezij pred naročilom tiskanin, smo izdelali dvoplastno prototipno ploščo.

4.3 Izdelava prototipne plošče (angl. *protoboard*)

Zaradi časovne stiske smo želeli testirati čim več komponent, da bi v primeru napak popravili tiskanino. Prenos vezja na testno ploščo sem naredil s foto postopkom. Uporabil sem ploščice z dvoplastnim bakrenim nanosom brez foto občutljivega filma. Ploščo najprej razrežemo in na vsaki strani pustimo približno centimeter rezerve. Pred nanosom pozitivnega fotokopirnega laka je potrebno bakrene površine zbrusiti z vodobrusnim papirjem s čim manjšo granulacijo in ploščice dobro posušiti, da je čim manj nečistoč. Priporočljivo je, da površine tudi razmastimo z etilnim alkoholom. Ko nanesemo foto lak, ploščo pustimo nekaj ur v temi, da se lak posuši.

Za domačo izdelavo dvoplastne tiskanine potrebujemo le dve maski, za zgornjo in spodnjo plast bakrene površine (Slika 4.2). Izdelamo ju iz prosojnic, na katere z laserskim tiskalnikom natisnemo plasti (Slika 4.5). Pri izdelavi tiskanega vezja s foto

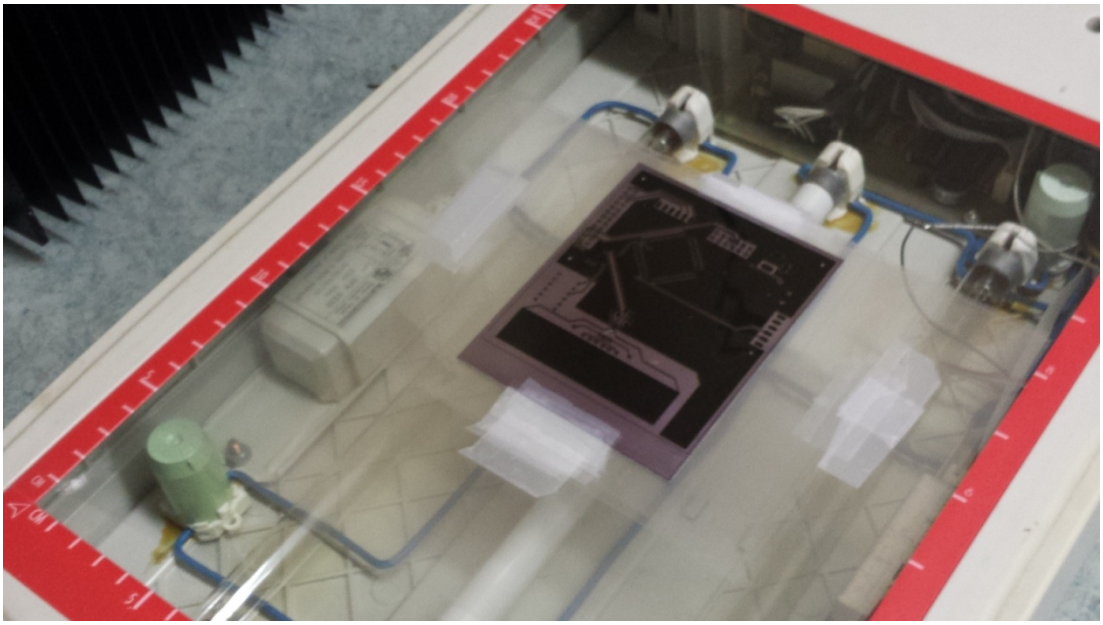
postopkom osvetljujemo pozitivni foto lak na mestih, kjer ne želimo imeti bakrenih površin. Da foto lak v postopku osvetljevanja ne osvetlimo skozi masko, potrebujemo za izdelavo vsake maske dve kopiji plasti. Kopiji moramo pred osvetlitvijo natančno poravnati in zlepiti med seboj. Pri izdelavi mask sem imel veliko težav, saj so uporabljene dimenzije in razmerja med povezavami zelo zahtevne za domačo izdelavo. Vsi laserski tiskalniki, ki sem jih uporabil, so imeli majhna odstopanja pri izrisu plasti tiskanine, razlike v dimenzijah bodisi po dolžini ali širini. Pri izdelavi ima maska velik pomen, saj mora biti za majhne dimenzije povezav zelo natančno izdelana. Da sem dobil čim bolj primerljivi kopiji, sem plasti natisnil na več prosojnic na isto območje.



Slika 4.5: Maski za prototipno ploščo; levo zgornja in desno spodnja.

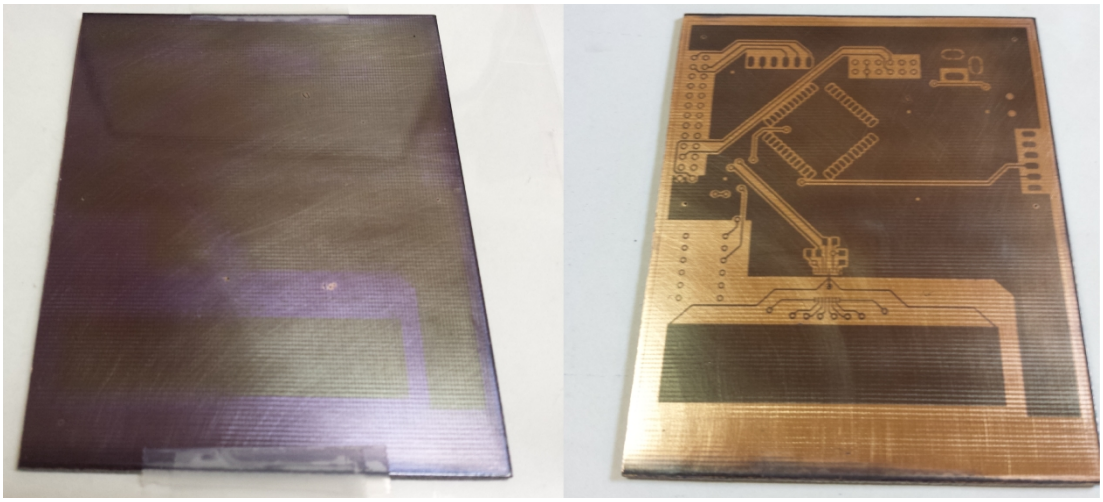
Ko so ploščica in maski pripravljeni, slednji namestimo vsako na svojo stran plošče. Pazimo, da sta pravilno obrnjeni in med seboj popolnoma poravnani, drugače nam izdelava tiskanine ne uspe. To lahko najlažje naredimo, da maski že pred namestitvijo zlepimo med seboj z lepilnim trakom in šele nato med njiju vstavimo ploščo. Poskrbimo tudi, da se plošča in maski med seboj ne premikajo.

Sedaj sledi postopek osvetljevanja. Najprej osvetlimo eno stran plošče in nato tudi drugo. Ploščo osvetljujemo z UV svetlobo, čas pa je odvisen od foto občutljivega filma in naprave, s katero osvetljujemo (Slika 4.6). V mojem primeru sem vsako stran osvetljeval 3 minute in pol. Foto občutljiv lak se osvetli na mestih, kjer ni zaščiten z masko (Slika 4.7a) in se na teh mestih (če uporabimo pozitivni foto lak) v postopku razvijanja raztopi (Slika 4.7b).



Slika 4.6: Naprava za osvetljevanje.

Razvijanje izvedemo v posebnih napravah za razvijanje filmov, kot razvijalec pa se uporabi 5-odstotna raztopina natrijevega hidroksida [28]. Jaz sem za razvijalec uporabil raztopino Cevosana (Slika 4.8). Po razvijanju dobimo na obeh bakrenih površinah plošče oblikovano plast foto laka, ki služi kot zaščita bakra pri jedkanju (Slika 4.7b).



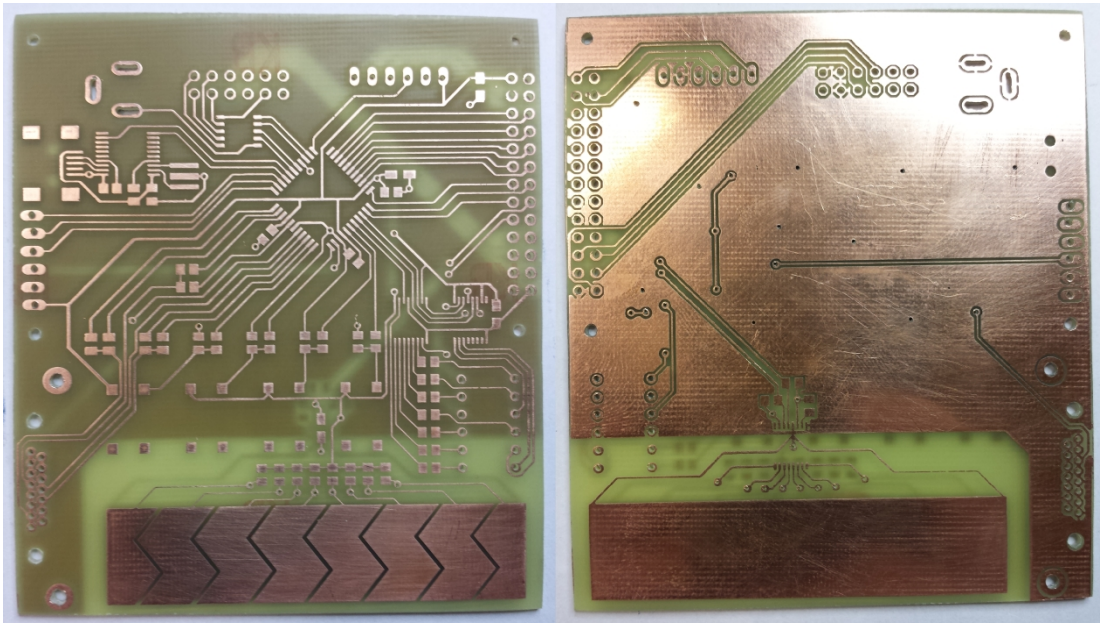
Slika 4.7: Spodnja stran prototipne plošče; levo pred in desno po postopku razvijanja.

Nezaščitene bakrene površine (Slika 4.7b) odstranimo s postopkom jedkanja. Raztopino pripravimo s 50 % vode, 25 % vodikovega peroksida (H_2O_2) in 25 % solne kisline (HCl) (Slika 4.8). Če smo ploščo dobro razvili, dobimo po vsej površini plošče lepo razporejene mehurčke, ki zagotavljajo enakomerno jedkanje [28].

Nezaščitene površine začnejo kmalu izginjati in ko izginejo vse, ploščo speremo pod vodo.

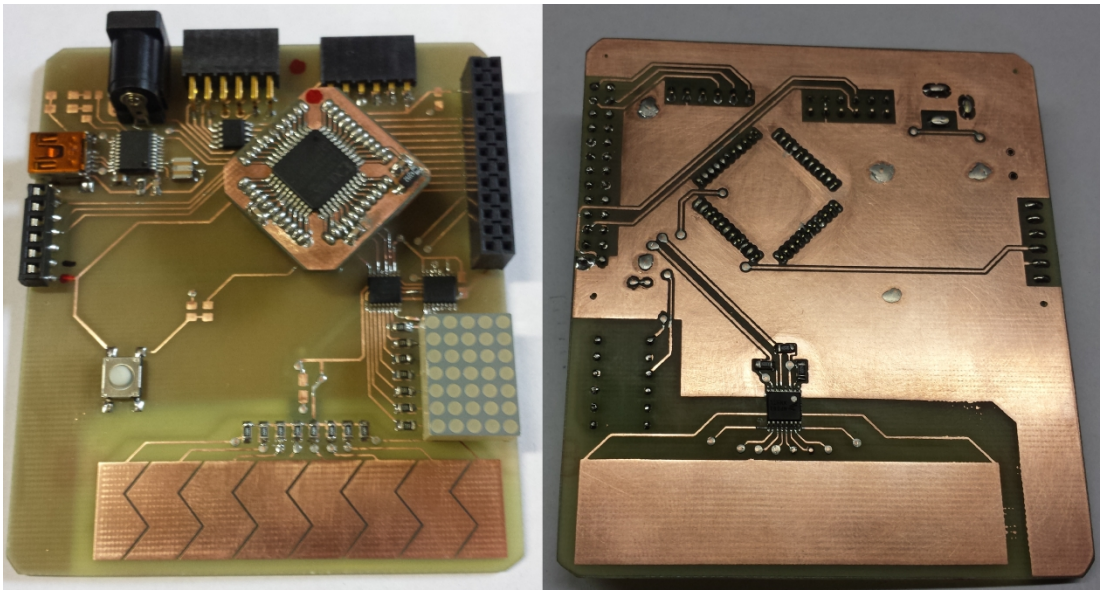
Slika 4.8: Cevosan, solna kislina in vodikov peroksid.

Po končanem postopku z razredčilom (lahko tudi vodobrusnim papirjem) odstranimo preostali del foto laka, preverimo povezave in možnost kratkih stikov, odrežemo odvečne površine ter naredimo izvrtine (Slika 4.9). Te naredimo tudi na mestih medplastnih povezav. Da povežemo zgornjo in spodnjo bakreno plast, povezavo najlažje izvedemo s tanko žico skozi izvrtino, ki jo naspajkamo med obe plasti. Ploščica je izdelana in pripravljena za spajkanje elementov.



Slika 4.9: Zgornja in spodnja stran prototipne plošča brez elementov.

Za prototipno ploščo (Slika 4.10) nismo uporabili vseh elementov, npr. za testiranje tipk je dovolj ena tipka, prav tako nismo dodali VGA vmesnika, ker smo bili prepričani v izračun uporovnega delilnika. CPLD smo namestili na dodatno podnožje v primeru, če bi se izkazalo nedelovanje komponent, da bi imeli možnost testiranja z istim CPLD-jem na popravljeni verziji prototipa.



Slika 4.10: Zgornja in spodnja stran prototipne plošče.

Pred priključitvijo plošče se še enkrat prepričamo, ali je prišlo pri spajkanju do neželenih povezav in jih odpravimo. Ko smo v to prepričani, lahko vezja preizkusimo. Da bi preverili delovanje, smo spisali osnovne kode za preverbo napak, ki niso predstavljene v tem delu. Predstavljene so programske kode za uporabo nekaj komponent in koda razdeljevalnika (Poglavje 5).

Pri testiranju smo na vezju odkrili napako v delovanju gonilnika motorjev. Povzročala je prekinjanje krmiljenja koračnega motorja zaradi pozabljenega blokirnega kondenzatorja na napajalni liniji. Zaradi napake v načrtovanju senzorja dotikov ni deloval drsnik, kar je opisano v poglavju 5.4.10. Napaka v krmiljenju matrike LED je opisana na strani 14.

Po testiranju vezij smo popravili vse odkrite napake. Datoteke popravljene tiskanine smo poslali preko foruma elektronikov, 25 kosov plošč pa je izdelalo podjetje *Lingva*. Zadnjo verzijo izrisane tiskanine za razširitveno ploščo prikazujejo slike iz poglavja 4.2.

4.4 Spajkanje (angl. *Soldering*)

Izdelava plošč je od časa naročila trajala približno 10 dni. V tem času smo se posvetili programiranju, po dostavi plošč pa spajkanju elementov.

Spajkanje je v elektroniki postopek povezovanja elementov, ki se običajno uporablja na ploščah tiskanih vezij. Za spajkanje uporabljamo spajkalnik. Od spajkalnih spojev je na koncu odvisno tudi delovanje in zanesljivost tiskanih vezij, zato je pomembna kakovost spajkanja [29].

Slika 4.11: Spajkalna postaja *Weller* [40].

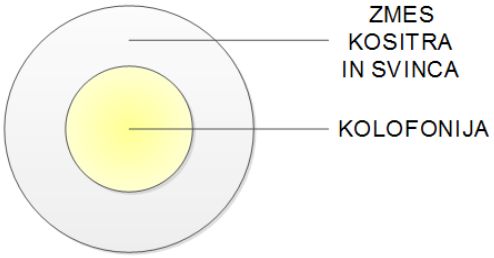
Za boljše spoje prilagajamo obliko in debelino spajkalnih konic, skladno z velikostmi podnožij elementov, ki jih spajkamo (Slika 4.12).

Slika 4.12: Spajkalne konice *Weller* [41].

Za večino elementov v tehnologiji SMD potrebujemo tanke in »špičaste« konice. V pomoč pridejo tudi konice, ki so ukrivljene za določen kot. Pomembno je,

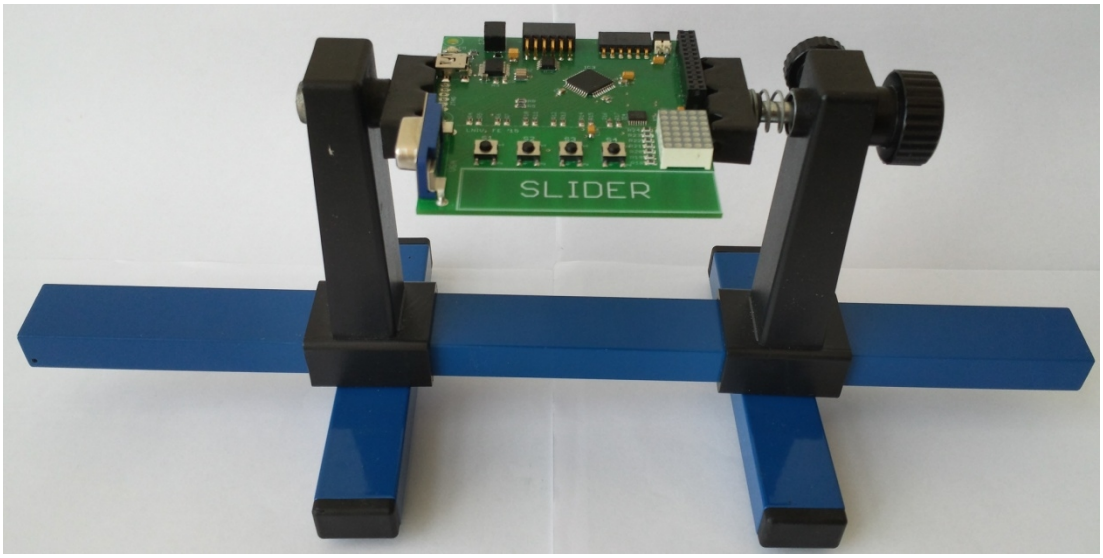
da je konica čista in segreta nad 200 °C [29]. Razširitvene plošče sem spajkal pri temperaturi konice 320 °C. Spajkalno temperaturo lahko običajno nastavljamo le pri spajkalnih postajah, zato je uporaba teh priporočljiva (Slika 4.11). Poleg tega je ročaj spajkala lažji in tanjši, pri dražjih je narejen bolj kakovostno, moč spajkal pa večja.

Za spajkanje potrebujemo tinol žico, ki služi za galvansko spajkanje elementov. Narejena je iz dveh delov; plašča iz zmesi kositra in svinca ter jedra iz kolofonije (Slika 4.13). Namen slednje je, da čisti mesto, kjer spajkamo. Temperatura taljenja tinol žice je odvisna od sestave, običajno znaša okoli 190 °C [29]. Obstajajo različne debeline žic, za spajkanje SMD vezij pa uporabimo debeline pod en mm.



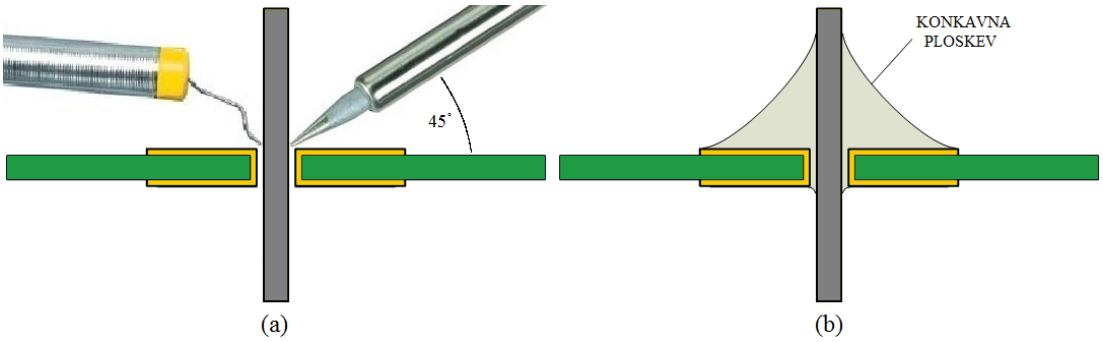
Slika 4.13: Sestava tinol žice.

Pri spajkanju klasičnih THT elementov potrebujemo klešče, pri površinski pritrditvi SMD elementov pa je nepogrešljiva nemagnetna pinceta pravilne oblike, igla s spajkalno pasto in odspajkovalna pletenica. V pomoč pride stojalo, kamor vpnemo ploščo tiskanega vezja (Slika 4.14), in povečevalno steklo za preverjanje spajkalnih spojev.

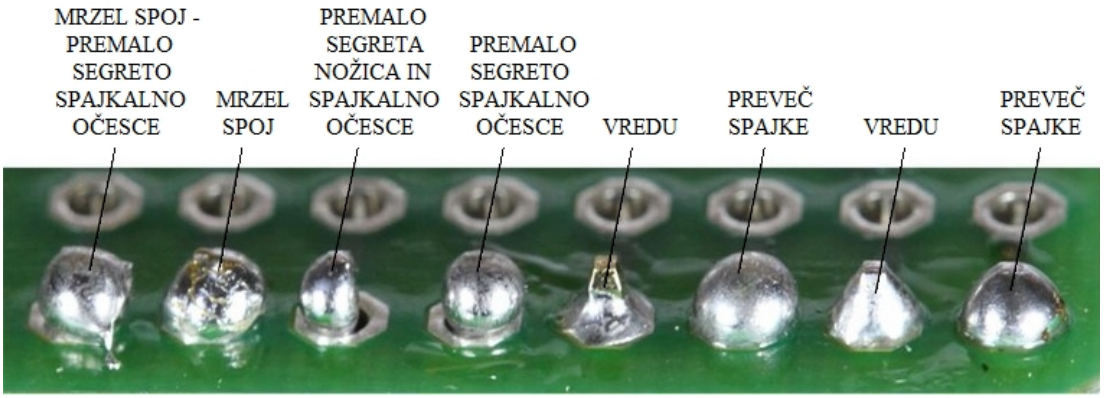


Slika 4.14: Stojalo za spajkanje tiskanih vezij.

Pri spajkanju klasičnih komponent prislonimo spajkalno konico na spajkalno mesto, tako da je kot med ploščo in spajkalom približno 45° (Slika 4.15a). Pri tem istočasno grejemo spajkalno površino in priključek elementa. Ko se oba dela segrejeta do določene temperature (segrevanje do nekaj sekund, odvisno od moči spajkala in od tega, kaj spajkamo), na mesto gretja prislonimo tinol žico. Najprej se stopi kolofonija, ki očisti spajkalno mesto. Ta izpari, zmes kositra in svinca pa se razlije po očiščenem mestu. Žica se pri segrevanju topi, zato jo dodajamo, da dobimo pravilno obliko spoja. Ko je spoj dovolj velik, žico odmaknemo pred spajkalom, da ta ne ostane prispajkana [29]. Ostanke žic odstranimo s kleščami.



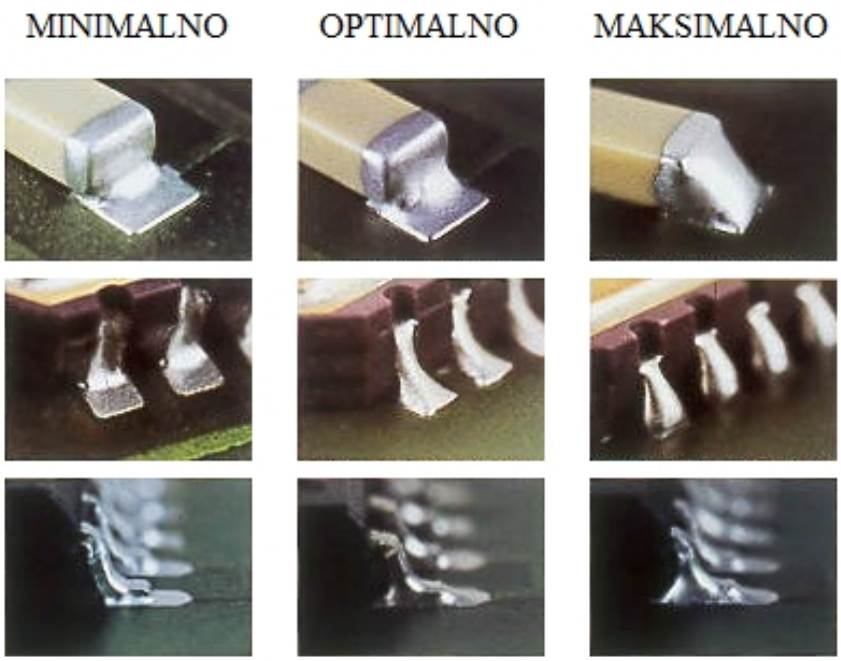
Slika 4.15: Spajkanje klasičnih komponent; a) drža spajkala in b) pravilno zalita spajka.



Slika 4.16: Primeri spajk v THT [42].

Pri spajkanju integriranih vezij s SMD podnožji je postopek spajkanja drugačen. Pred namestitvijo integriranega vezja s konico na spajkalna območja vezja nanesemo tanek sloj spajke. Na mesto podnožja integriranega vezja nanesemo spajkalno pasto in vezje položimo nanjo. Ko je nameščeno na pravilen položaj, ga rahlo pritisnemo k tiskanini ali ga držimo s pinceto na mestu. Po nožicah integriranega vezja preletimo s širšo spajkalno konico, da se sloj spajk stopi. Vezje je s tem pritrjeno, sledi zalivanje priključkov. Na koncu odstranimo odvečno spajko z

odspajkovalno pletenico [30]. Postopek spajkanja ostalih SMD komponent je podoben.

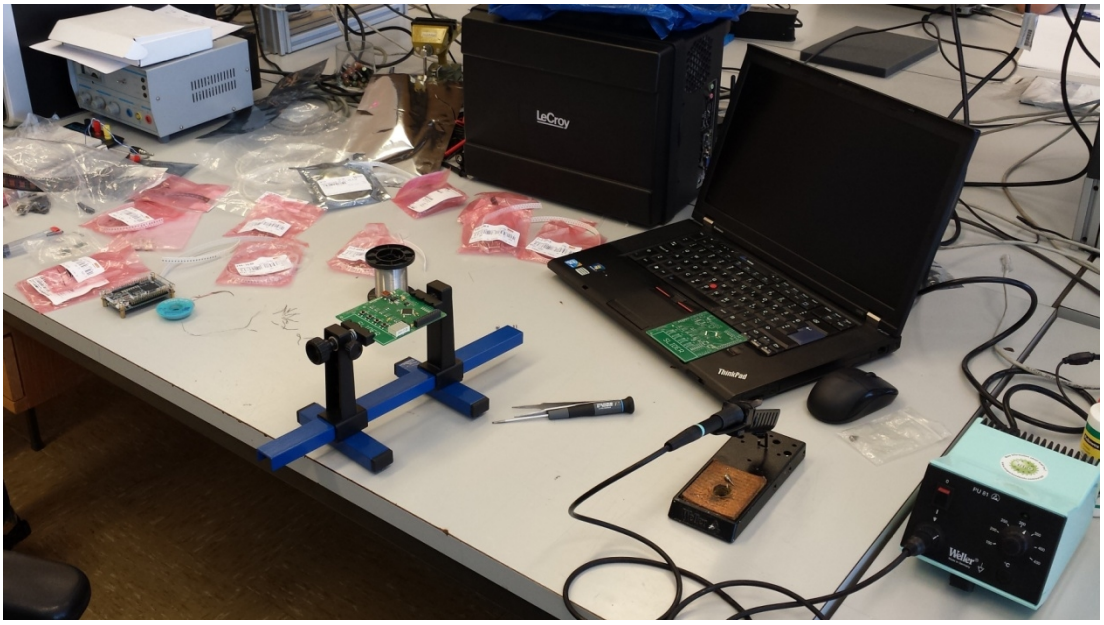


Slika 4.17: Primeri spajk v SMD [43] [44].

Za pravilni spoj mora biti spoj lepo razlit (konkavna ploskev), površina pa gladka in sijoča (Slika 4.16 in 4.17). Paziti moramo, da s predolgim gretjem ne pregrejemo elementov. Po končanem spajkanju tiskano vezje očistimo z etilnim alkoholom.

Pred spajkanjem vseh razširitvenih plošč smo najprej v celoti preverili delovanje komponent na prvi sestavljeni plošči. Za lažje preverjanje smo spisali kratek *DEMO* program, ki je povezoval kode, uporabljene pri testiranju prototipne plošče. Ko smo bili prepričani v delovanje, smo pričeli s spajkanjem 20 kosov plošč.

Razširitvena plošča ima 324 spajkalnih očesc, na katere smo naspajkali 55 pasivnih ter 6 aktivnih elementov in 7 priključkov. Za spajkanje ene razširitvene plošče sem ob vstavljanju elementov porabil približno 4 ure. Brez preverjanja delovanja je spajkanje trajalo slaba dva delovna tedna (Slika 4.18).



Slika 4.18: Spajkanje razširitvenih plošč.

Pred testiranjem vseh sestavljenih razširitvenih plošč smo na vse CPLD-je naložili programsko kodo razdeljevalnika, ki smo jo sprogramirali pred tem. Vse o delovanju kode je opisano v poglavju 5.1, nalaganje kode pa v podpoglavju 5.1.4. Pred prvo uporabo smo nastavili tudi integrirano vezje pretvornika UART, kot je opisano v poglavju 5.5. Že pri nastavljanju UART pretvornika smo naleteli na težave v neodzivanju vezja. Nedelujočim ploščam smo ponovno segreli spojke nožic vezja *MCP2200* in priključke USB priključka.

Le pri četrtini vseh preverjenih plošč so delovale vse komponente. Pri ostalih niso delovale zaradi težav matrik LED, drsnikov, UART vmesnikov in na nekaterih gonilniki motorjev. Na vseh nedelujočih smo ponovno segreli spojke priključkov kritičnih elementov. Za komponente matrik LED, UART vmesnika in gonilnike motorjev se je kmalu pokazalo uspešno opravljeno delo. Na nekaj ploščah so bili krivi spoji nožic glavnega priključka, za nedelovanje večine pa so bili vzrok mrzli spoji integriranih vezij in priključkov, nekaj pozabljenih zaspajkanih spojk in napačno obrnjenih elementov.

Še vedno pa večina plošč ni delovala v celoti zaradi težav v delovanju vzdolžnega drsnika. S segrevanjem smo popravili vse spojke priključkov vezja *MPR083*, vendar zaznavanje drsenja dotika ni bilo tekoče. Napako smo naredili, ker smo se takoj po spajkanju elementov posvetili testiranju plošč brez čiščenja tiskanega vezja z etilnim alkoholom, ki ga v tistem času tudi nismo imeli na razpolago. Ostale napake drsnika so opisane v poglavju 5.4.10. Pri spajkanju integriranega vezja na območje podnožja nanesemo spajkalno pasto. Te je nekaj najverjetneje ostalo pod

ohišjem in je med nožicami vezja povzročala napačno merjenje kapacitivnosti na elektrodah. Po čiščenju in še dodatni nastavitvi občutljivosti dotikov smo tako uspešno končali sestavljeno serijo razširitvenih plošč.

5 Programiranje komponent

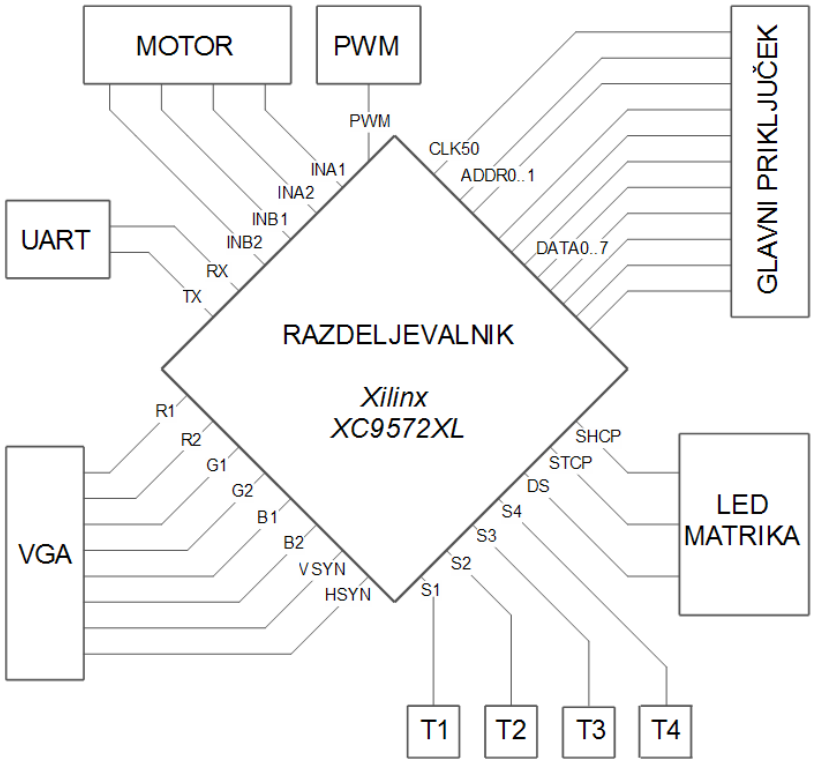
Delovanje razširitvene plošče ni mogoče brez sistema, ki bi krmilil digitalno vezje CPLD in opisane komponente. Krmiljenje razširitvene plošče izvajamo s programirljivo razvojno ploščo *DE0-Nano* v tehnologiji FPGA. Kode komponent so opisane v jeziku za opis hitrih integriranih vezij VHDL. Shematičen prikaz razporeditve blokov komponent prikazuje slika 3.43.

Pred uporabo razširitvene plošče smo spisali kodo za delovanje razdeljevalnika na kompleksnem programirljivem digitalnem vezju CPLD. Koda skrbi za prenos signalov prek glavnega priključka, od in k uporabljenim komponentam. V naslednjih podpoglavjih so opisane pomembnejše kode, ki skrbijo za delovanje komponent razširitvene plošče.

5.1 Razdeljevalnik (angl. *Demultiplexer*)

Zaradi pomanjkanja digitalnih linij na glavnem priključku razvojne plošče *DE0-Nano* smo v načrtovanju izbrali ustrezen CPLD (*Xilinx XC9572XL*) za razdeljevanje signalov. V okolju *Xilinx* smo spisali in naložili kodo razdeljevalnika. Razdeljevalnik razdeljuje signale iz 8-bitnega podatkovnega vhoda na 4-krat po 8-bitni izhod, glede na eno od štirih kombinacij naslova. Z izbranim naslovom preslikamo 8-bitno podatkovno vodilo *DATA* h komponentam, kot so matrika LED, VGA in UART vmesnik, gonilnik motorja, PWM izhod (Slika 5.1). Razdeljevalnik pošilja signale tudi v nasprotni smeri za uporabo tipk in UART vmesnika.

Poleg naloge razdeljevanja signalov opravlja PISO pretvorbo skupaj s krmiljenjem matrike LED in uravnavanjem osvetlitve.



Slika 5.1: Shema razdeljevalnika in komponent.

Oznake linij in nožic integriranih vezij so enake oznakam v električnih shemah v poglavju 3. Imena nožic in opis signalov kode razdeljevalnika prikazuje tabela 5.1.

Ime nožice	Ime	Dolžina	Tip	Opis
CLK50	*clk50*	1	vhodni	Sistemska ura
ADDR0..1	*addr(0..1)*	2	vhodni	Naslov
DATA0..7	*data(0..7)*	8	vhodni/izhodni	Podatki
R1, R2, G1, G2, B1, B2, VSYN, HSYN	*vga(0..7)*	8	izhodni	Komponenta VGA
INA1, INA2, INB1, INB2	*motor(0..3)*	4	izhodni	Koračni ali DC motor
S1, S2, S3, S4	*buttons(0..3)*	4	vhodni	Tipke
DS, SHCP, STCP	*matrix(0..2)*	3	izhodni	Matrika LED
TX	*uart_tx*	1	vhodni	Prejem podatkov UART
RX	*uart_rx*	1	izhodni	Oddaja podatkov UART
PWM	*pwm*	1	izhodni	PWM signal

Tabela 5.1: Opis signalov razdeljevalnika.

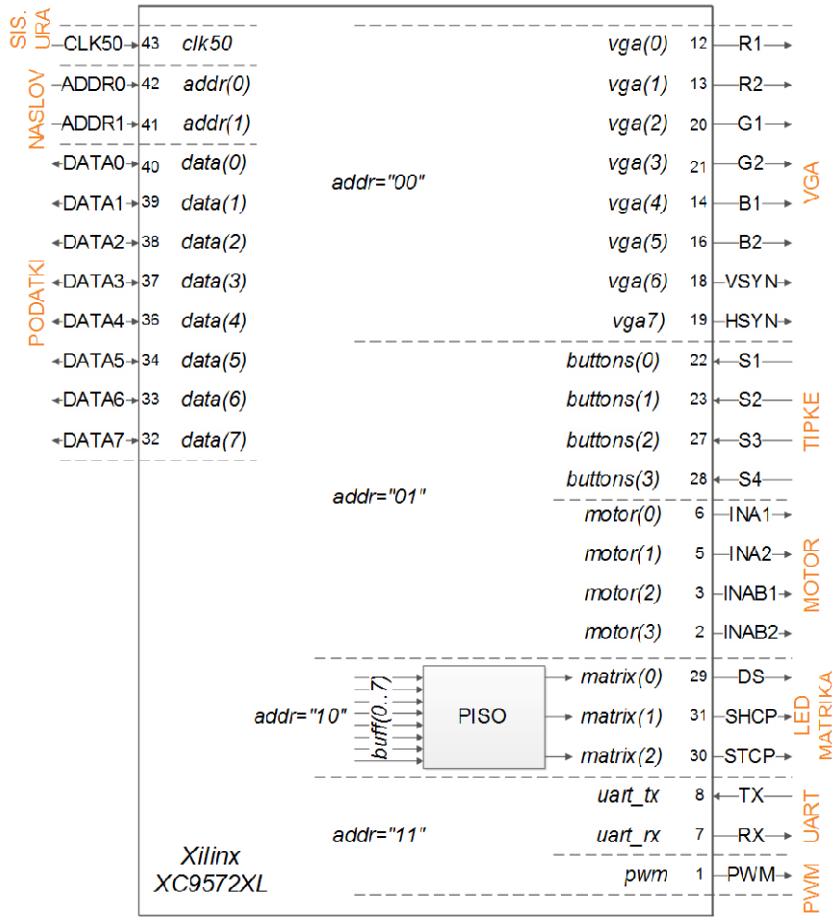
5.1.1 Opis razdeljevalnika

Razdeljevalnik preslika signale z *DATA* vodila h komponentam glede na naslov *ADDR* na vhodu razdeljevalnika, kot prikazuje slika 5.2. Vhodni podatki *DATA(0..7)* se ob naslovu *ADDR(0..1)* enakemu 00 preslikajo na vhod VGA vmesnika. Ob naslovu 01 se niz *DATA(4..7)* preslika na vhod gonilnika koračnega ali DC motorja, signali tipk pa na vodilo *DATA(0..3)*. Pri kombinaciji 10 se podatki *DATA(0..7)*, ki služijo za upravljanje matrike LED, zajamejo in shranijo v niz *buff(0..7)*. Zadnji naslov 11 je namenjen prenosu podatkov *DATA(6..7)* z UART vmesnikom in podatka *DATA(5)* na PWM izhod. Razporeditev signalov glede na naslov prikazuje tabela 5.2.

VODILO	**NASLOV *(ADDR1, ADDR0)***			
	»00«	***»01«***	***»10«***	***»11«***
DATA0	R1	S1	*buff(0)*	-
DATA1	R2	S2	*buff(1)*	-
DATA2	G1	S3	*buff(2)*	-
DATA3	G2	S4	*buff(3)*	-
DATA4	B1	INA1	*buff(4)*	-
DATA5	B2	INA2	*buff(5)*	PWM
DATA6	VSYN	INB1	*buff(6)*	RX
DATA7	HSYN	INB2	*buff(7)*	TX

Tabela 5.2: Razporeditev podatkovnih signalov *DATA* vodila in signalov komponent glede na naslov.

Kot vidimo na tabeli 5.2 in sliki 5.2 se pri treh naslovih podatki z *DATA* vodila preslikajo neposredno na nožice CPLD-ja. Izjema je le naslov 10, ki je namenjen matriki LED. Podatki se zajamejo v pomnilnik *buff*.



Slika 5.2: Shema razporeditve signalov razdeljevalnika.

Matriko LED upravljamo preko dveh zaporedno vezanih pomikalnih registrov (angl. *shift register*). Za prikazovanje uporabljamo tri signale: serijski podatki *DS* (angl. *Serial Data*), uro pomika podatkov *SHCP* (angl. *Shift Register Clock Input*) in uro vpisa podatkov *STCP* (angl. *Storage Register Clock Input*) na matriko LED. V kodi razdeljevalnika so signali označeni kot *matrix(0..2)*. Pomikalna registra krmilimo serijsko. Podatke vpisujemo v prvi register in se ob zapolnjenosti prvega prelivajo v drugi register. Za prikaz ene vrstice matrike LED potrebujemo 13 urinih ciklov. Šele v zadnjem ciklu sprožimo vpis podatkov na matriko s spremembo signala *STCP* z nizkega v visoko stanje (Slika 5.7).

Če bi istočasno uporabljali tudi druge kombinacije naslovov *addr*, bi potrebovali za eno osvežitev matrike 91 (13 ciklov po 7 vrstic) preklopov med naslovi za malo prenesene informacije, če predpostavimo, da enakomerno preklapljamo med naslovi. Upravljanje matrike na tak način bi bilo tudi zahtevno, saj moramo pomikalnima registroma zagotoviti ustrezno hitrost ure pomika podatkov *SHCP*. Da bi poenostavili upravljanje matrike LED smo se odločili, da bo krmiljenje opravljal CPLD na razvojni ploščici.

Želeli smo vzporeden vpis podatka, ki bo nosil informacijo aktivnih pik v vrstici matrike LED. Informacijo smo razdelili na stolpce in vrstice. Ker nam je ostajalo nekaj prostora pri naslavljanju, smo bili nekoliko svobodni pri izbiri, vendar smo želeli ostati znotraj enega naslova. Informacija o stolpcih zajema 5-bitni podatek. Ostali so trije biti. Ti predstavljajo binarno vrednost, ki določa aktivno vrstico matrike LED. Za primer, ko je vrednost 001, je aktivna vrstica 1, za 111 je aktivna vrstica 7. Za vrednost 000 je matrika neaktivna oz. zatemnjena. Opisano krmiljenje prikazuje tabela 5.3.

Ime	Dolžina	Tip	Opis
buff(0..4)	5	pomnilni	0: stolpec neaktiven 1: stolpec aktiven
buff(5..7)	3	pomnilni	000: matrika neaktivna 001: akt. vrstica 1 010: akt. vrstica 2 011: akt. vrstica 3 100: akt. vrstica 4 101: akt. vrstica 5 110: akt. vrstica 6 111: akt. vrstica 7

Tabela 5.3: Krmiljenje matrike LED.

Spisali smo PISO pretvorbo, vključno z urama za pomik in vpis podatkov v pomikalna registra. Izhod pretvornika je serijski, vendar s prilagojeno hitrostjo ure pomika *SHCP*. Hitrost ure pomika smo določili s podatkovnega lista registra *74HC595*. Tipična frekvenca ure pri sobni temperaturi in napajalni napetosti 2 V je 30 MHz. Za delilnik frekvence sistemske ure 50 MHz smo uporabili števec *sh_counter* s pogoji za delitev ure *shcp* s slike 5.3. Dobimo polovično frekvenco 25 MHz ure za pomik podatkov.

Maksimalna frekvenca vpisa podatkov za matriko v CPLD se je zaradi paralelnega vpisa zmanjšala. Hitrost je pogojena s frekvenco pomika f_{SHCP} in s številom ciklov, ki so potrebni za izpis vrstice. S frekvenco pomika 25 MHz in 13 ciklov potrebnih za izpis vrstice dobimo maksimalno frekvenco vzporednih vpisov 1,9 MHz. Koda je napisana tako, da spusti vpise nad to frekvenco. Dosežemo lahko 274 kHz osveževanje matrike, kar je mnogo več kot potrebujemo.

Zaradi napake v načrtovanju v krmiljenju matrike LED, opisane na strani 14, smo k temu delu dodali še kodo za enakomerno svetilnost matrike. Koda prešteje

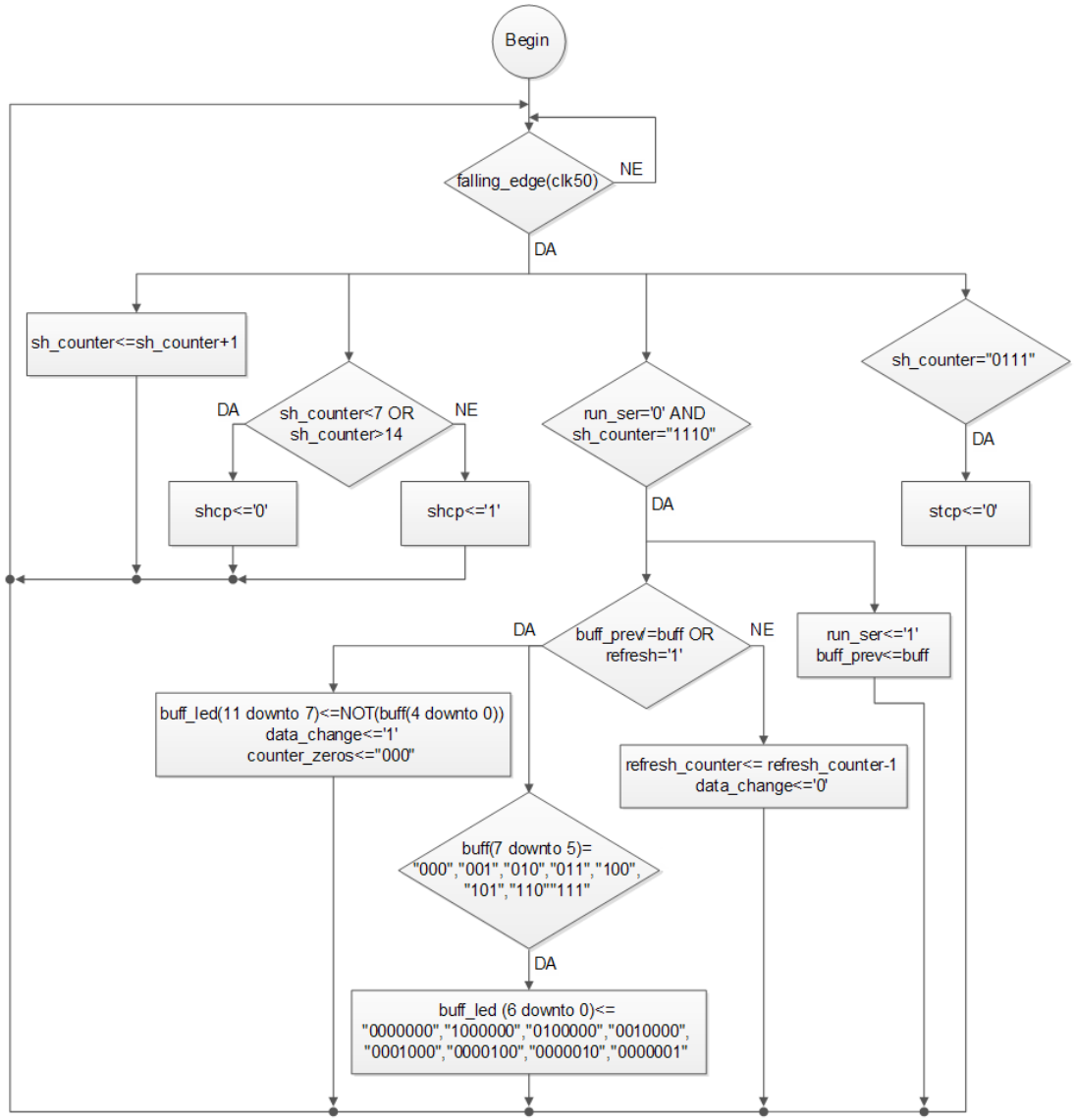
število aktivnih pik v vrstici. Več aktivnih pik v vrstici pomeni manjšo svetilnost pik in obratno. Glede na manj aktivnih pik, koda sorazmerno izpušča osveževanje določenih vrstic. Z izpuščanjem osveževanja vrstic do določene meje spreminjamo svetilnost in uravnovesimo svetilnost matrike LED na približno enako raven.

5.1.2 Delovanje razdeljevalnika

Ob kombinaciji 10 naslova *addr* se podatki *data* zajamejo v pomnilnik *buff* (Tabela 5.2). Niz *buff(0..4)* predstavlja aktivne pike (logična enka predstavlja aktivno stanje), vrednost *buff(5..7)* pa vrstico aktivnih pik. Pred začetkom nove PISO pretvorbe se nastavi 12-bitni niz *buff_led*. Ali se niz lahko nastavi, je odvisno od ukaza *run_ser*, ki z nizkim stanjem določa konec pretvorbe, in od stanja števca *sh_counter*. Če je vrednost števca za ena manjša od 1111, slednja določa začetek pretvorbe in se nastavi niz *buff_led*.

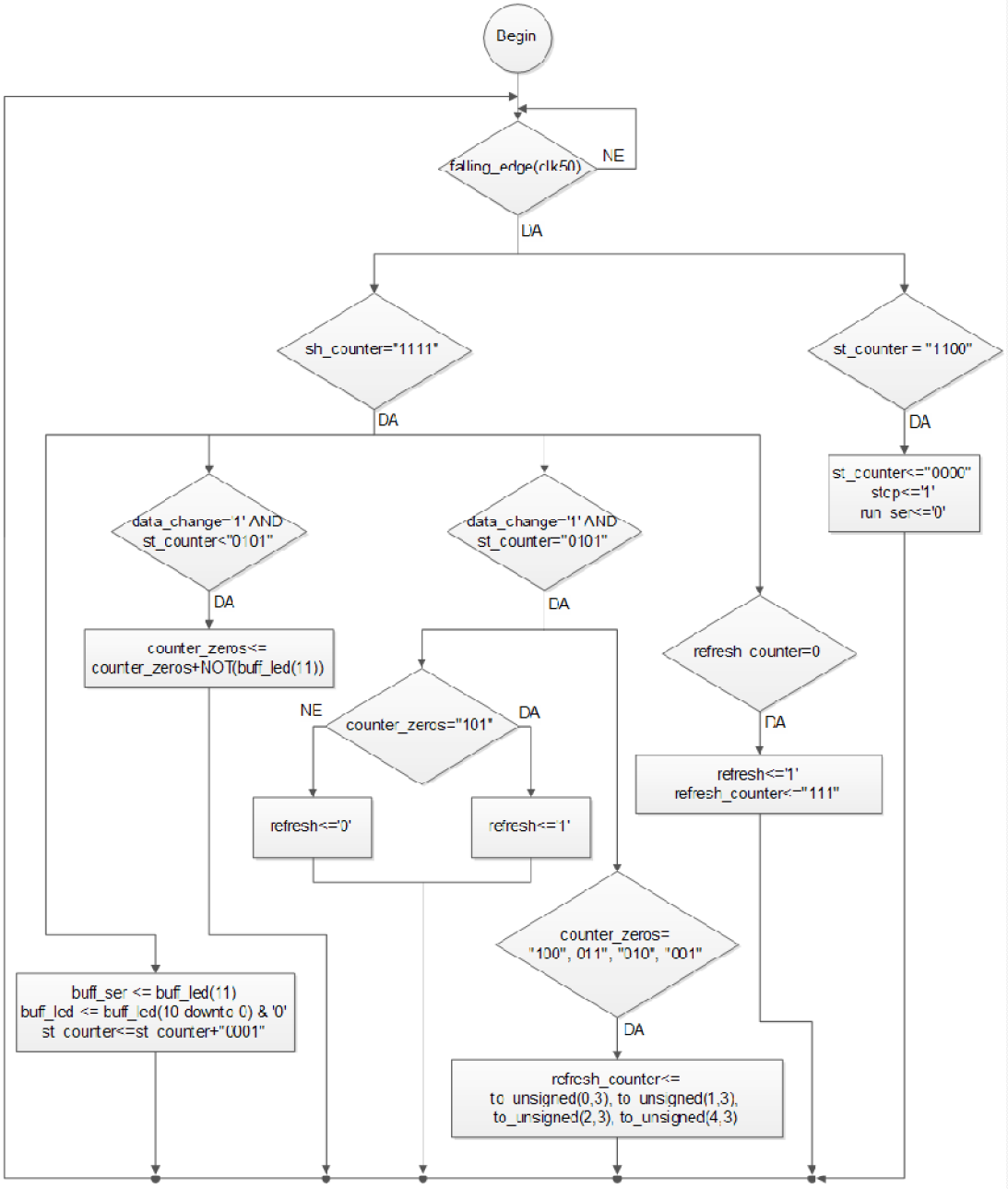
V prvi del *buff_led(7..11)* se shrani negacija pomnilnika *buff(0..4)*. Negacijo smo dodali, ker imamo konfiguracijo matrike LED s skupno anodo (Slika 3.2a). Pomeni, da je posamezen stolpec aktiven z logično ničlo. Ostali, 7-bitni del *buff_led(0..6)*, dobimo glede na binarno vrednost niza *buff(5..7)*. S tem delom določimo aktivno vrstico (Slika 5.3 spodnji blok). Sestavljen niz *buff_led* vsebuje vso informacijo za prikaz vrstice matrike LED. Služi kot pomnilnik za serijski prenos 12-bitne informacije prek signala *DS*. Ko je števec *sh_counter* enak 1111, zapišemo bit z največjo težo (MSB) pomnilnika *buff_led(11)* v enobitni pomnilnik *buff_ser* za serijski prenos informacij. Pomnilni signal *buff_ser* se v prireditvenem stavku preslika v *matrix(0)* (Slika 5.5) Nad ostalimi biti pomnilnika *buff_led* naredimo operacijo &'0' (Slika 5.4 blok spodaj levo). S tem z desne strani (LSB) pomnilnika *buff_led* dodamo ničlo oz. zamaknemo bite za eno mesto proti MSB. Ob tem izpade najbolj pomemben bit.

Kot prikazuje slika 5.7 se na linijo za serijski prenos podatkov *DS* vpiše najprej informacija za krmiljenje stolpcev nato za vrstice. S števcem *st_counter* štejemo število prenesenih bitov. Ko kaže 12 oz. binarno vrednost 1100, pomeni konec pretvorbe podatkov za eno vrstico. Signal vpisa podatkov na matriko *stcp* preide v visoko stanje, števec *st_counter* in ukaz za serijski prenos *run_ser* pa se ponastavita (Slika 5.4 desno zgoraj). Signal za vpis *stcp* se ponastavi, ko je števec *sh_counter* enak 0111 oz. ob prvem prehodu ure pomika *shcp* v visoko stanje (Slika 5.3 desno zgoraj). Po končanem prenosu se niz *buff_led* ponovno nastavi. Ali se je nastavil, spremljamo z ukazom *data_change*. Izhode PISO pretvorbe *buff_ser*, *shcp* in *stcp* prepišemo v *matrix(0..2)* v prireditvenem stavku (Slika 5.5). S PISO pretvorbo smo tako rešili problem zahtevne uporabe matrike LED.



Slika 5.3: Diagram poteka sekvenčnega procesa razdeljevalnika; 1.del.

K serijskemu krmiljenju matrike LED smo dodali izpuščanje osveževanja za enakomerno svetilnost matrike. Ko se v *buff* vpiše podatek, se ta informacija izpisuje na matriko toliko časa, dokler se vsebina *buff* ne spremeni. Število izpisov informacije v tem času je odvisno od tega, kako hitro spreminjamo vhodne podatke *data*, ki so namenjeni matriki LED. Kot smo izračunali zgoraj, je maksimalna frekvenca, potrebna za eno osvežitev vrstice, 1,9 MHz, za celotno matriko pa je 7-krat manjša. Ta frekvenca je še vedno mnogo večja, kot če vpisujemo podatke *DATA* za matriko tako hitro, da je hitrost osveževanja matrike 50 Hz. Izpuščanje osveževanja vrstic, zaradi velike razlike v hitrostih, zato ne bo problematično.

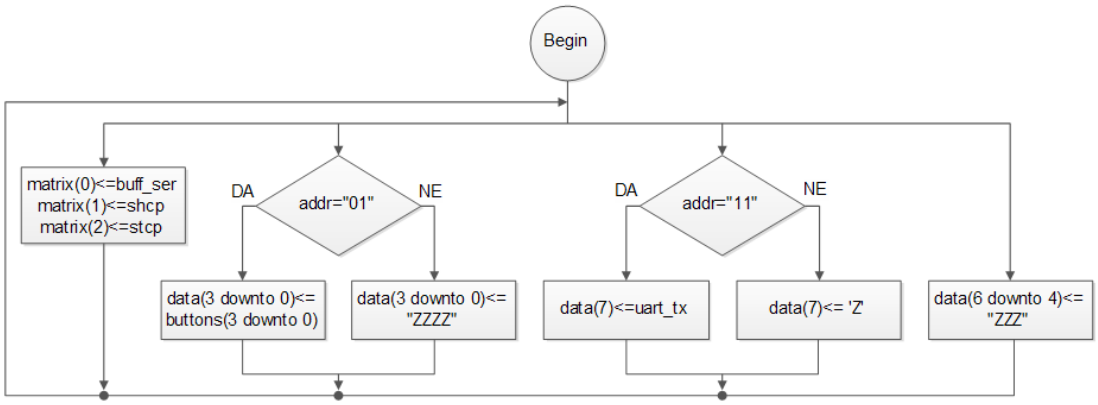


Slika 5.4: Diagram poteka sekvenčnega procesa razdeljevalnika; 2.del.

Kolikokrat bomo izpustili osveževanje, je odvisno od aktivnih pik v vrstici, ki jih beleži števec *counter_zeros* (Slika 5.4 blok na levi strani). Dokler je števec *st_counter* manjši od 0101 (manjši zato, da zajamemo informacijo o akt. stolpcih) štejemo logične enke. Štetje se začne istočasno, kot nastavimo pomnilnik *buff_led*. Nov vpis podatkov beleži ukaz *data_change* v visokem stanju. Kaj se bo zgodilo ob naslednjem ciklu osveževanja, je odvisno od stanja števca aktivnih pik *counter_zeros* in števca *st_counter* ob 0101 (Slika 5.4 sredina). V primeru, ko so aktivne vse pike v vrstici matrike, bo aktiven tudi interni signal *refresh*. Ta omogoči ponovni vpis *buff* v

buff_led in osvežitev iste vrstice. V nasprotnem primeru pustimo pomnilnik *buff_led* enak nič, ki je »prazen« zaradi dodajanja ničel z desne strani niza (Slika 5.4 spodaj levo). Ob osveževanju se na podatkovni liniji *DS* v pomikalna registra vpisujejo ničle, zato se zgodi zatemnitev. Kolikokrat se zatemnitev zgodi oz. koliko zaporednih ciklov je matrika zatemnjena, je odvisno od števca zatemnitev *refresh_counter*. Ta se nastavi glede na stanje števca *counter_zeros* (Slika 5.4) in se odšteva za ena ob vsaki zatemnitvi matrike (Slika 5.3). Ko prišteje do nič, sproži ponovno osvežitev s signalom *refresh* v visokem stanju. Izbira imena za števec zatemnitev *refresh_counter* ni najbolj smiselna.

Pomembno je povedati, da imamo skupno 5 podatkovnih signalov, ki potujejo z razširitvene plošče prek CPLD-ja na razvojno ploščo *DE0-Nano*. Štirje signali tipk *S(1..4)* in UART signal *TX* so z vidika razdeljevalnika vhodni signali (Tabela 5.1). Razdeljevalnik jih ob določenem naslovu *ADDR* preslika na podatkovno vodilo *DATA*. Če bi preslikavo izvedli v sekvenčnem procesu, bi se signali prepisali en urin cikel kasneje od zajema naslova *ADDR*. Pomenilo bi, da signale tipk in UART signal prepišemo na vodilo šele ob naslednji padajoči fronti ure *CLK50*. Problem nastane, če na vhodu razdeljevalnika spreminjamo naslove ob vsakem urinem ciklu. Sekvenčna logika ne more ob istem proženju ure sprejeti signal in glede vrednosti istočasno odločati o stanju drugega signala. Če bi na vhodu razdeljevalnika spreminjali naslove ob vsakem ciklu sistemske ure *CLK50*, bi se tako prepis signalov na vodilo zgodil ob napačnem naslovu. Na vodilu *DATA* bi povzročili kratek stik med izhodi CPLD in FPGA. Informacija bi bila izgubljena. Ker signali, ki potujejo v nasprotno smer *S(1..4)* in *TX* niso časovno zahtevni glede na našo sistemsko uro 50 MHz, smo prepis signalov izvedli v prireditvenem stavku (Slika 5.5).

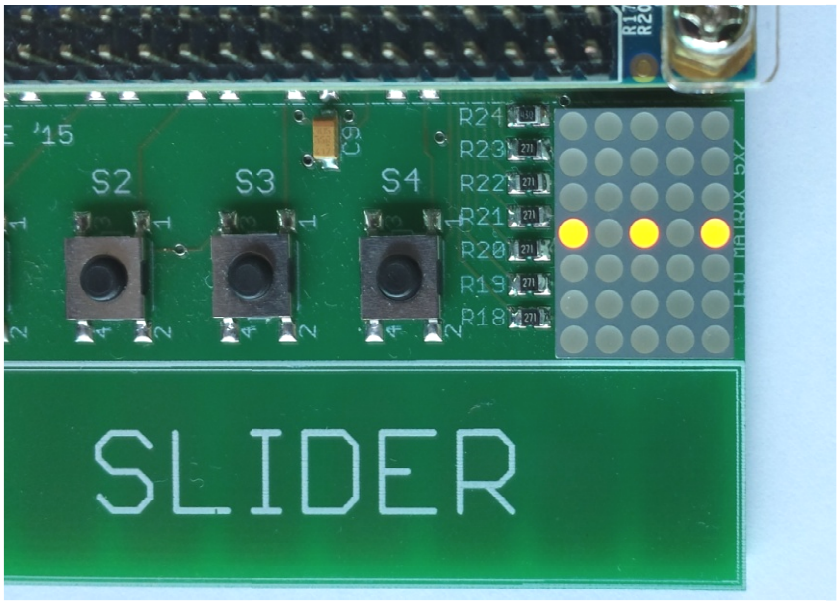


Slika 5.5: Diagram poteka prireditvenega stavka razdeljevalnika.

Če pogledamo tabelo 5.2, vidimo, da so signali *data* dvosmerni. Pomeni, da je *DATA(0..7)* dvosmerno vodilo, preko katerega lahko signali potujejo v obe smeri. Dvosmerni signali ne morejo biti istočasno vhodni in izhodni signali. Ko jih uporabljamo kot izhodne, jim priredimo vrednosti. Ko pa jih uporabljamo kot vhodne signale, torej ko z njih beremo, jih moramo postaviti v visoko impedančno stanje ali odprte sponke. Ob prepisu tipk *buttons(0..3)* in signala *uart_tx* na dvosmerno vodilo uporabimo interne signale *data* kot izhodne signale, v ostalih primerih pa so vhodni. Torej, notranje podatkovne signale *data(0..3)* (kamor preslikamo signale tipk) za čase, ko naslov *addr* ni nastavljen na naslov tipk, prestavimo v visoko impedančno stanje (ZZZZ). Enako naredimo s signalom *data(7)*. Čeprav signalov *data(4..6)* ne uporabljamo kot izhodne, smo jih kljub temu pustili dvosmerne, zato jim priredimo ZZZ (Slika 5.5).

5.1.3 Primer serijskega prenosa podatkov v pomikalna registra matrike LED

Slika 5.7 prikazuje simulacijo časovnih potekov izhodnih in internih signalov razdeljevalnika. V simulaciji so samo signali razdeljevalnika, ki se uporabljajo za delovanje matrike LED. Signali za ostale komponente se preslikajo po tabeli 5.2.

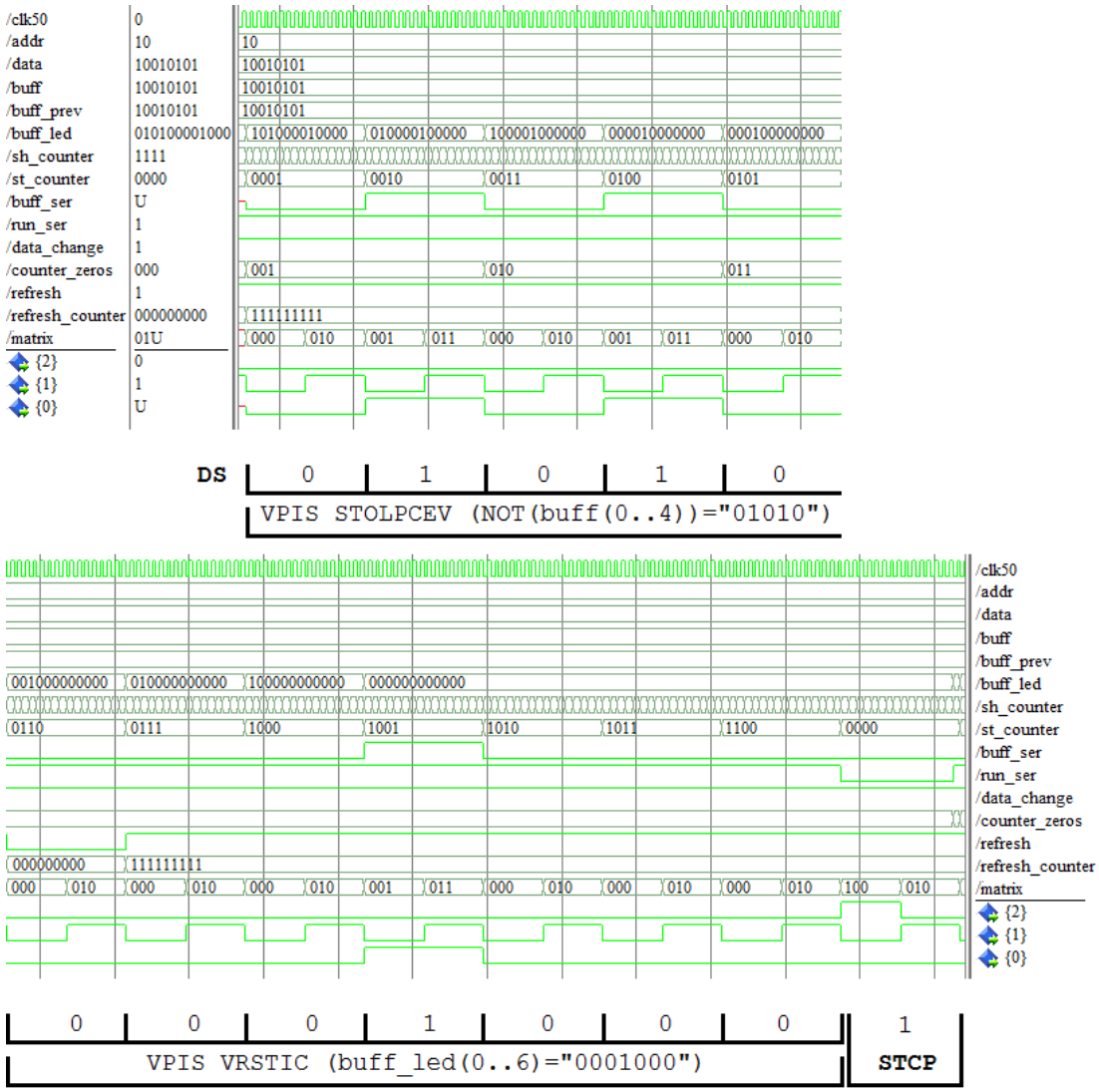


Slika 5.6: Rezultat simulacije na matriki LED.

V simulaciji sem kombinacijo naslova *addr* nastavil na 10 in vhodne podatke *data* na 10010101. Simulacijo sem »zagnal« za čas prenosa informacije ene vrstice. Izhodni signal PISO pretvorbe *matrix(0)* se prepiše na izhod *DS*. Prvih 5 bitov na izhodu *DS* (01010) predstavlja stanje stolpcev. Za konfiguracijo matrike s skupno

anodo (Slika 3.2a) so aktivni lihi stolpci matrike. Ostali del signala *DS* (0001000) predstavlja aktivnost vrstice 4 (vrstica je aktivna ob logični enki). Po 12 prenesenih bitih informacije sledi sprememba signala *matrix(2)* na visok potencial. Signal je preslikan na izhod *STCP*. Ob aktivnem stanju *STCP* registra prepišeta vpisane podatke na vhod matrike LED. Sprememba traja pol urinega cikla *SHCP*. Ura pomika podatkov *SHCP* v pomikalna registra je preslikava signala *matrix(1)*. Za izbran primer simulacije gorijo tri pike v četrti vrstici matrike LED (Slika 5.6).

Slika 5.7 prikazuje časovni potek signalov za eno osvežitev vrstice matrike LED. Ne prikazuje poteka signalov ob izpuščanju osveževanja ali drugačnega zapisa, saj se z izjemo *matrix(0)* bistveno ne razlikujejo.



Slika 5.7: Časovni potek serijskega prenosa podatkov v pomikalna registra matrike LED pri simulaciji kode razdeljevalnika.

5.1.4 Nalaganje programske kode na CPLD

V sistemih programiranja ISP imajo nekatere logične naprave sposobnost, da so sprogramirane po namestitvi v sistem. V sistemih tehnologije CPLD in FPGA je ključna uporaba vmesnika JTAG. Z vmesnikom lahko programiramo ali brišemo programe, ter preverjamo ali preizkušamo CPLD in FPGA naprave.

Za povezavo med programatorjem in ciljno napravo *Xilinx XC9572XL* uporabimo JTAG s šestimi povezavami. Razporeditev signalov JTAG na razširitveni plošči prikazuje slika 3.42. Opis signalov prikazuje tabela 5.4.

Slika 5.8: Programator *Xilinx* in razširitvena plošča.

Ker je uporabljen CPLD družine *Xilinx,* smo programsko kodo naložili s programatorjem iz iste družine (Slika 5.8). Platformo povežemo z računalnikom z USB kablom. Dodatnega napajanja ne potrebuje, ker se napaja prek USB.

Ime	Namen	Povezava
VCC	Napajanje 5V	Na VCC priključek
GND	Ozemljitev	Na GND priključek
TDO	Za branje podatkov iz ciljnega sistema.	Na TDO priključek
TDI	Za prenos serijskih navodil in podatkov v ciljni sistem.	Na TDI priključek
TCK	Ura poganja testno logiko vseh naprav.	Na TCK priključek
TMS	Za nadzor testne operacije.	Na TMS priključek

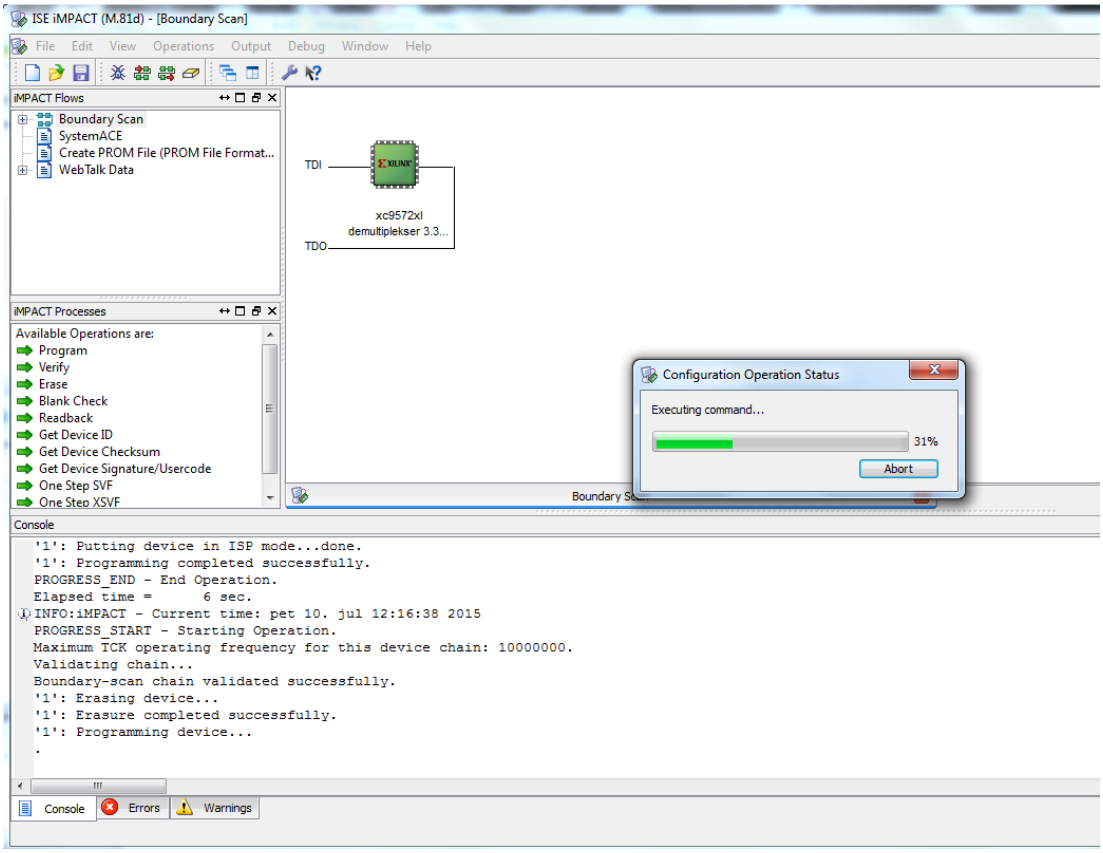
Tabela 5.4: Opis signalov JTAG.

Ko je koda spisana, generiramo programsko datoteko; dvoklik na »*Generate Programming File*«. Pred ustvarjanjem programske kode ne smemo pozabiti ustvariti datoteke *ucf*, kjer je opisan seznam fizičnih priključkov. Po prevajanju se nam izpiše zasedenost naprave. Izkoriščenost našega sistema prikazuje tabela 5.5.

Viri	Zasedenost
Makrocelice (angl. *makrocells*)	68/72 (95 %)
Produktni členi (angl. *product terms*)	235/360 (66 %)
Registri (angl. *registers*)	62/72 (87 %)
Pini (angl. *pins*)	33/34 (98 %)
Vhodni funkcijski bloki (angl. *function block inputs*)	115/216 (54 %)

Tabela 5.5: Zasedenost virov na *XC9572XL*.

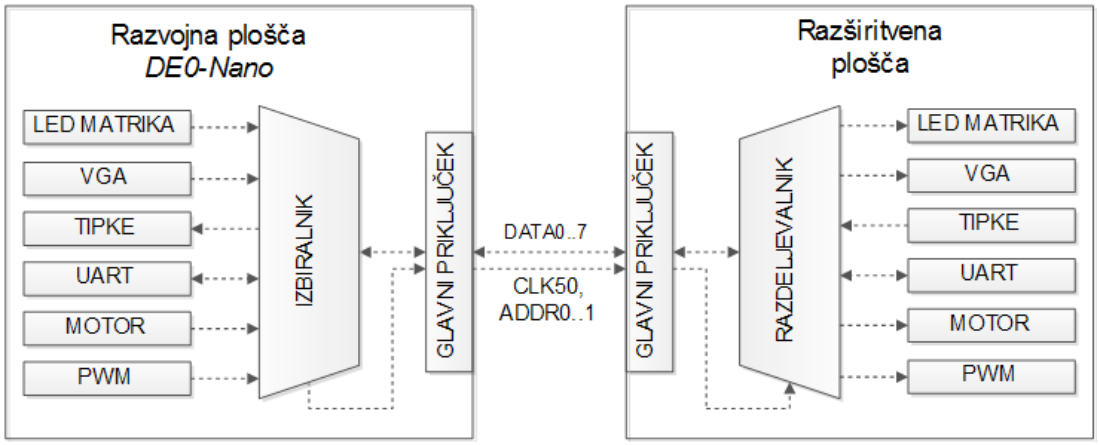
Ko imamo programsko kodo v formatu JEDEC (jed datoteka) za konfiguracijo, priključimo CPLD na programator in napajalno napetost 3,3 V. Dvokliknemo na upravljanje konfiguracij projekta »*Manage Configuration Project*« in odpre se nam okno, kjer dodamo napravo *XC9572XL*. Izberemo .jed datoteko in naložimo program (Slika 5.9).



Slika 5.9: Okno za nalaganje programa na *XC9572XL*.

5.2 Izbiralnik (angl. *Multiplexer*)

Naloga izbiralnika je nasprotna nalogi razdeljevalnika. Opravlja funkcijo pridruževanja večjega števila vhodnih signalov na manjše število kanalov. Vsebuje več podatkovnih vhodov in izhodov, naslove in sistemsko uro. Glede na kombinacijo naslova izbere skupino signalov in jih preslika na izhod. V našem primeru se mora sprogramiran izbiralnik ujemati s programiranim razdeljevalnikom s prejšnjega poglavja 5.1. Dvaindvajset vhodnih signalov blokov komponent posreduje po 8-bitnem podatkovnem vodilu *DATA* do razdeljevalnika (na razširitveni plošči). Ta jih razdeli k izbranim komponentam (Slika 5.10). Vsebuje 5 signalov, ki potujejo v nasprotni smeri.



Slika 5.10: Shema izbiralnika in razdeljevalnika.

Tipični izbiralniki (in razdeljevalniki) pridobijo krmilne signale s strani krmilne logike. V našem primeru vsebuje blok izbiralnika tudi krmilno logiko. Signale pošilja tudi razdeljevalniku, da lahko pravilno razdeli podatkovne signale. Za krmilne signale vzemimo sistemsko uro *CLK50* in naslova *ADDR(0..1)*. Opis signalov za blok izbiralnika prikazuje tabela 5.6.

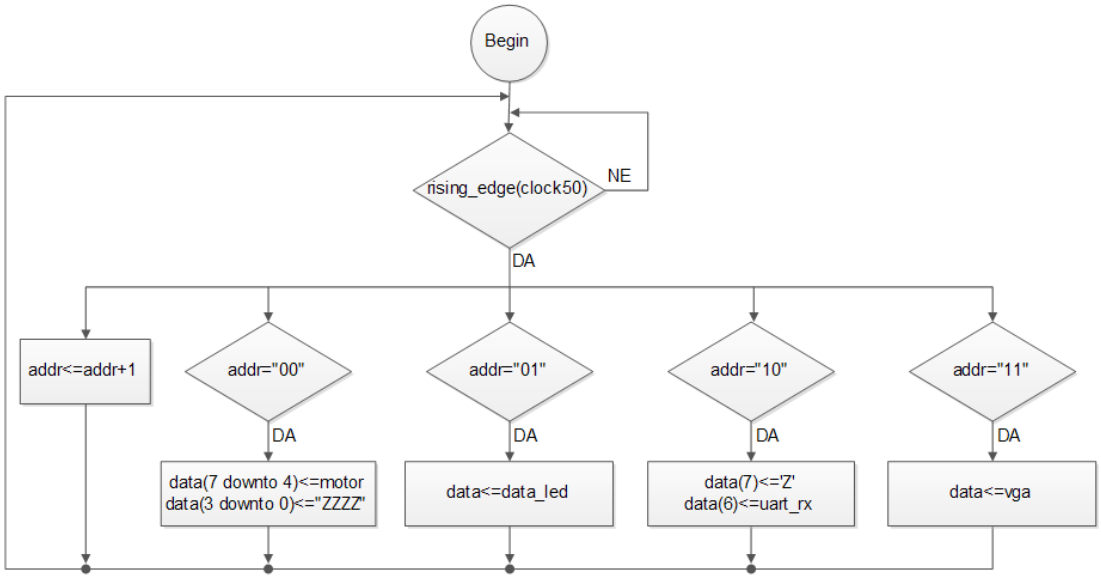
Ime nožice priključka	Ime	Dolžina	Tip	Opis
/	*clock50*	1	vhodni	Vhodna sistemska ura
CLK50	*clk50*	1	izhodni	Izhodna sistemska ura
ADDR0..1	*addr(0..1)*	2	medpomnilnik (angl. *buffer*)	Naslov
DATA0..7	*data(0..7)*	8	vhodni/izhodni	Podatki
/	*vga(0..7)*	8	vhodni	Komponenta VGA
/	*motor(0..3)*	4	vhodni	Koračni ali DC motor
/	*buttons(0..3)*	4	izhodni	Tipke
/	*data_led(0..7)*	8	vhodni	Matrika LED
/	*uart_tx*	1	izhodni	Prejem podatkov UART
/	*uart_rx*	1	vhodni	Oddaja podatkov UART
/	*pwm*	1	vhodni	PWM signal

Tabela 5.6: Opis signalov izbiralnika.

5.2.1 Delovanje izbiralnika

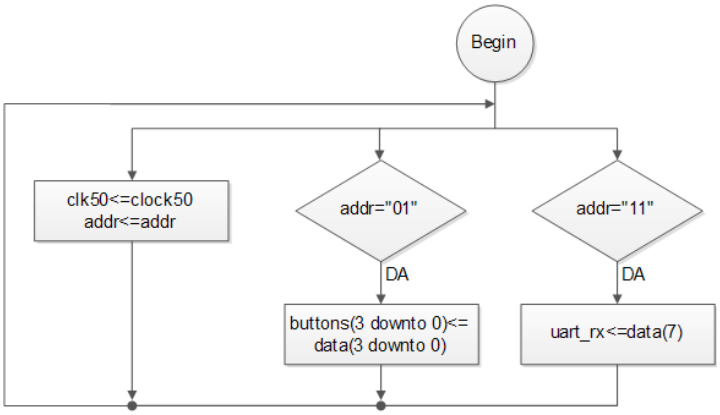
Koda izbiralnika je enostavna. Na sliki 5.11 vidimo diagram poteka sekvenčnega procesa izbiralnika. Naslov *addr* spreminjamo ob vsaki naraščajoči fronti ure *clock50*. Prepis naslovov *addr* in notranje sistemske ure *clock50* na izhod *ADDR* in *CLK50* je izveden v prireditvenem stavku (Slika 5.12). Kombinacijam naslovov *addr* prištevamo logično enko in jih vrtimo v krogu od 00 do 11 in ponovno od začetka (Slika 5.11). Izbiralnik s tem dodeljuje vsem komponentam enako dolge aktivne cikle na podatkovnem vodilu *DATA*. Ob določeni kombinaciji naslova priredimo nizu signalov *data* signale z blokov posameznih komponent. Vendar, če primerjamo diagram poteka s slike 5.11 s tabelo 5.2, vidimo, da so naslovi *addr* v odločitvenih blokih sekvenčnega procesa zamaknjeni za eno kombinacijo. Naslovi *addr* se v sekvenčnem procesu povečujejo istočasno s prepisom signalov na vodilo *DATA*, vendar se prepis podatkov zgodi za prejšnjo vrednost *addr*.

Če želimo prepis podatkov blokov komponent po naslovih po tabeli 5.2 za drugačen vrstni red naslovov *ADDR*, moramo v sekvenčnem stavku izbiralnika postaviti pogoje tako, da ustrezamo naslovom razdeljevalnika.



Slika 5.11: Diagram poteka sekvenčnega procesa izbiralnika.

Podobno, kot pri razdeljevalniku, imamo tudi pri izbiralniku 5 podatkovnih signalov *DATA*, ki jih poleg izhodnih uporabimo tudi kot vhodne signale. Štiri uporabljamo za stanja tipk *buttons(0..3)* in en za *uart_tx* signal UART komponente. Da se branje signalov zgodi brez zamika (ene periode sistemske ure), je branje izvedeno v prireditvenem stavku (Slika 5.12).

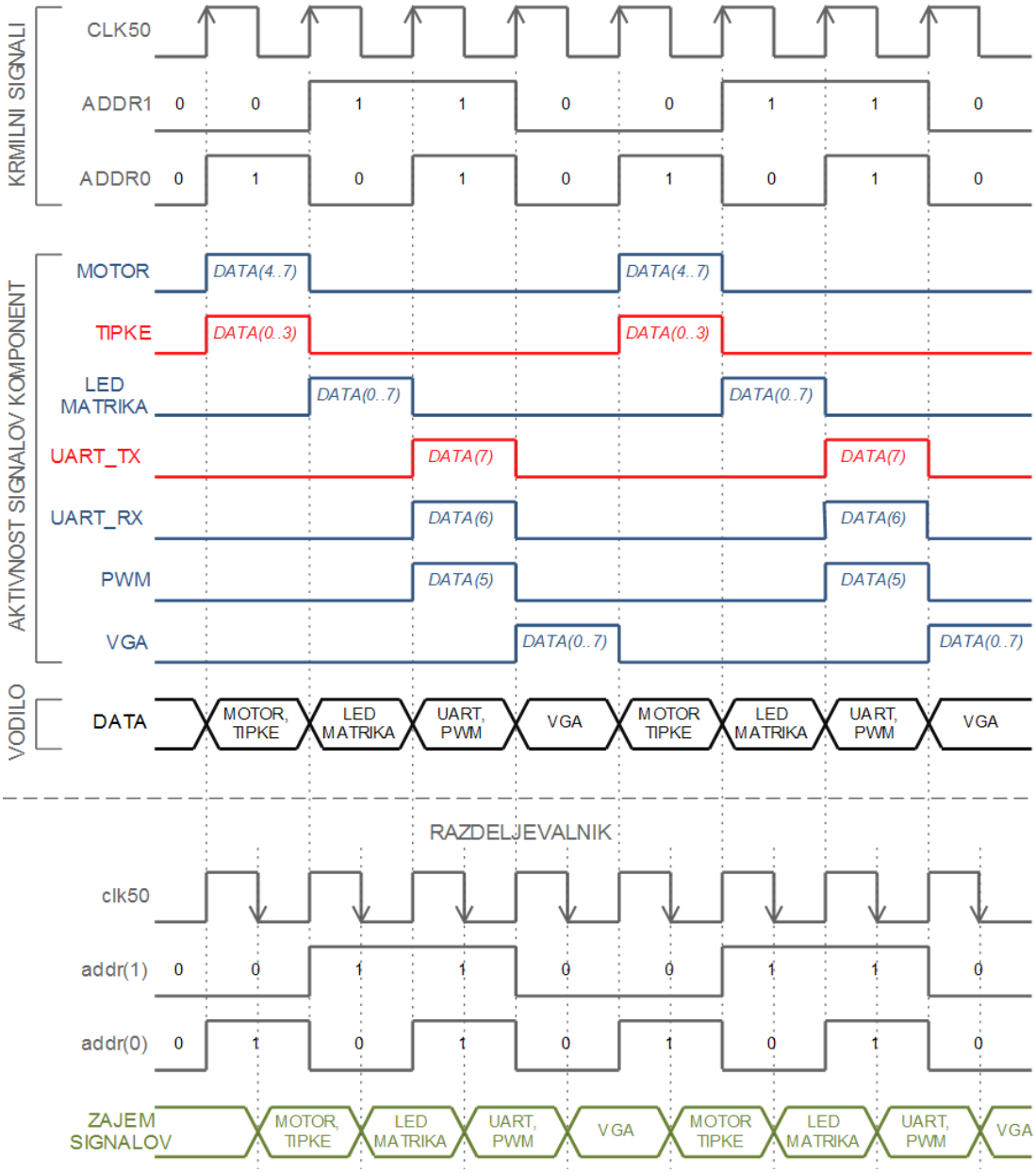


Slika 5.12: Diagram poteka prireditvenega stavka izbiralnika.

Pri uporabi tipk in UART pretvornika ne smemo pozabiti, da je vodilo *DATA* dvosmerno in s tem je dvosmeren tudi niz signalov *data*. Če želimo dvosmerni signal uporabiti kot vhodni, ga moramo postaviti v visoko impedančno stanje za čas branja. Podobno kot v kodi razdeljevalnika tudi tukaj ob določenih naslovih priredimo 'Z' signalu *data*, ko ga uporabljamo kot vhodni signal. To storimo za signale *data(0..3)* ob naslovu 01, ko beremo signale tipk, in ob naslovu 11 in za signal *data(7)*, ko beremo *TX* signal. V primeru, da tega ne storimo, nam izbiralnik za vhodne signale ne bo deloval.

5.2.2 Časovni potek signalov med izbiralnikom in razdeljevalnikom

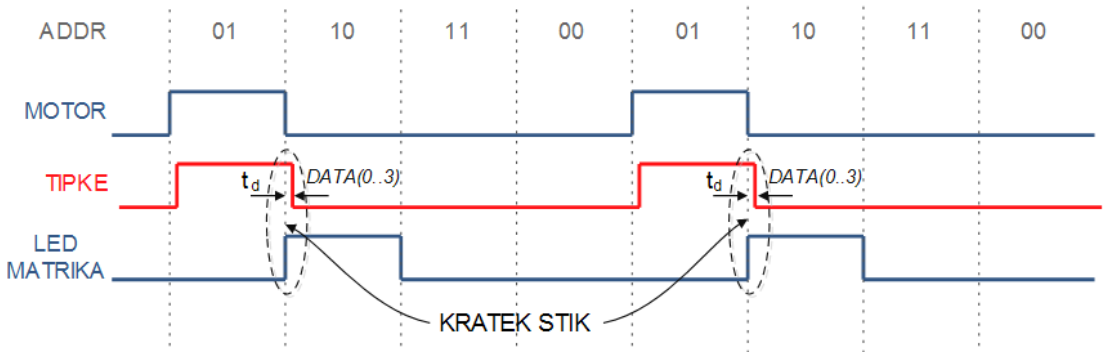
Slika 5.13 prikazuje časovni potek sistemske ure *CLK50* in naslovov *ADDR(0..1)*. Z modro barvo označeni časovni poteki predstavljajo aktivnost signalov komponent, ki potujejo v smeri izbiralnik–razdeljevalnik. Signali, ki potujejo v nasprotni smeri (razdeljevalnik–izbiralnik) po vodilu so označeni z rdečo barvo. To so signali tipk *data(0..3)* in UART signal *data(7)*. S črno barvo je prikazana skupna časovna razporeditev aktivnost signalov komponent na vodilu *DATA* ob spreminjanju naslovov od 00 do 11. Prenos signalov po vodilu se časovno izide.



Slika 5.13: Časovni potek krmilnih signalov in aktivnost signalov komponent na vodilu *DATA*. Časovni potek krmilnih signalov in zajem signalov komponent razdeljevalnika.

Spodnji del slike 5.13 prikazuje časovni potek krmilnih signalov na vhodih razdeljevalnika. Z zeleno barvo je ob padajoči fronti sistemske ure *CLK50* prikazan zajem podatkov komponent z vodila *DATA*. Zajem je zamaknjen za polovico cikla sistemske ure. Pri frekvenci ure 50 MHz je zakasnitev 10 ns.

V zgornjem časovnem poteku (Slika 5.13) so signali poravnani tako, da se časovno ne prekrivajo. V praksi so signali, ki potujejo v smeri CPLD–FPGA (razdeljevalnik–izbiralnik) nekoliko zakasnjeni glede na krmilne signale na izhodu FPGA. Primer: signali motorja se prepišejo na vodilo *DATA(4..7)* na izhodu FPGA v času naslova 01. Signali so zaradi sekvenčnega procesa časovno poravnani s krmilnimi signali. Signali tipk pa se na istem naslovu 01 prepišejo na vodilo *DATA(0..3)* na izhodu CPLD-ja nekaj kasneje po izpisu signalov motorja (Slika 5.14).



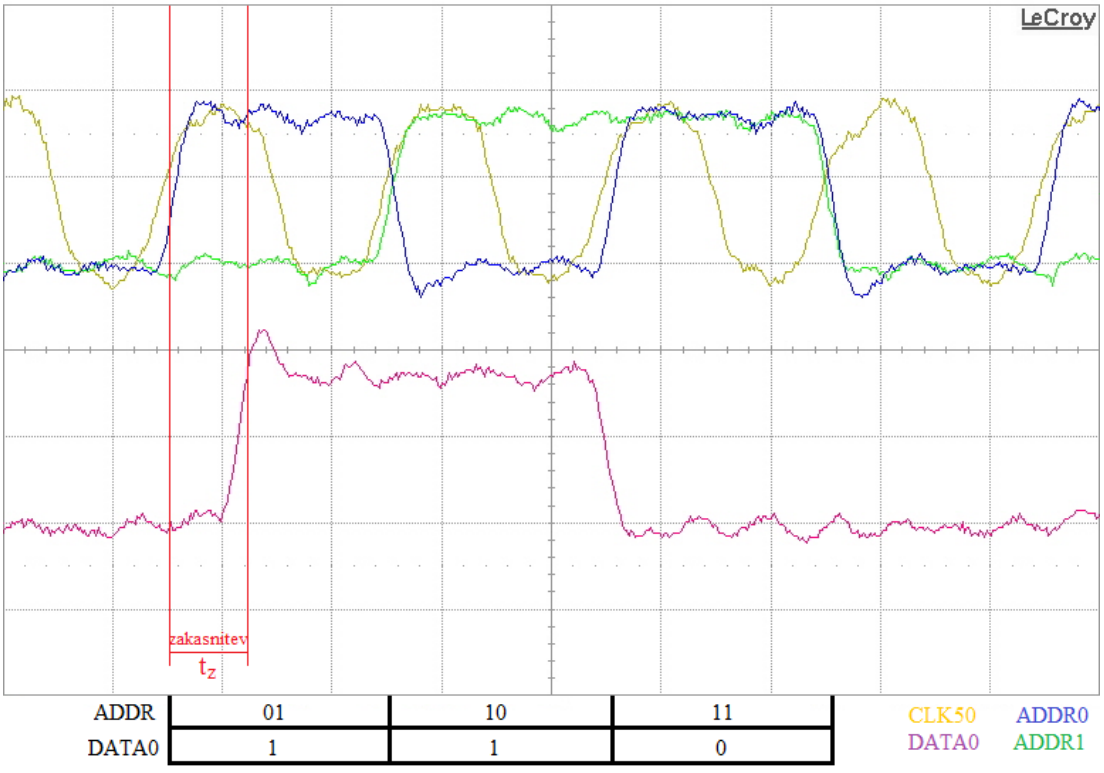
Slika 5.14: Časovni potek ob zakasnitvi signalov tipk *DATA(0..3)*.

Prepis je zakasnjen zaradi zakasnitve krmilnih signalov na povezavah in časa, ki ga porabi CPLD za prepis signalov tipk *DATA(0..3)* na izhod. Signali motorja in signali tipk niso časovno poravnani, čeprav so na istem naslovu *ADDR*. Zakasnitev ob vpisu signalov tipk na vodilo ni problematična kot je zakasnitev ob koncu naslova 01. Zaradi zakasnjenega odločanja CPLD-ja se aktivnost signalov tipk podaljša v naslednje obdobje naslova. V našem primeru zaporedja naslovov ima naslednji naslov rezervirana komponenta matrike LED. Med izhodi CPLD-ja in FPGA pride (če sta signala na različnih nivojih) do kratkega stika na vodilu DATA. Za drugačen vrstni red naslovov nastanejo kratki stiki med drugimi signali komponent. Predvidevamo, da je čas t_z kratkega stika majhen. Da bi se prepričali, koliko se signali zakasnijo in kako to vpliva na odločanje stanj signalov, smo zakasnitve pomerili z osciloskopom.

5.2.3 Merjenje zakasnitve signalov z osciloskopom na glavnem priključku

Z osciloskopom smo na glavnem priključku opazovali sistemsko uro *CLK50*, naslova *ADDR1* in *ADDR0* ter podatkovno linijo *DATA0*. Stanja na podatkovni liniji smo z razširitvene plošče spreminjali s tipko S1 in z razvojne plošče s stanji signala *data(0)* matrike LED. Ob časih drugih kombinacij naslovov 00 in 11 smo na vodilo vpisali logične ničle. Preizkusili smo tri primere glede na stanja signalov tipk in matrike LED.

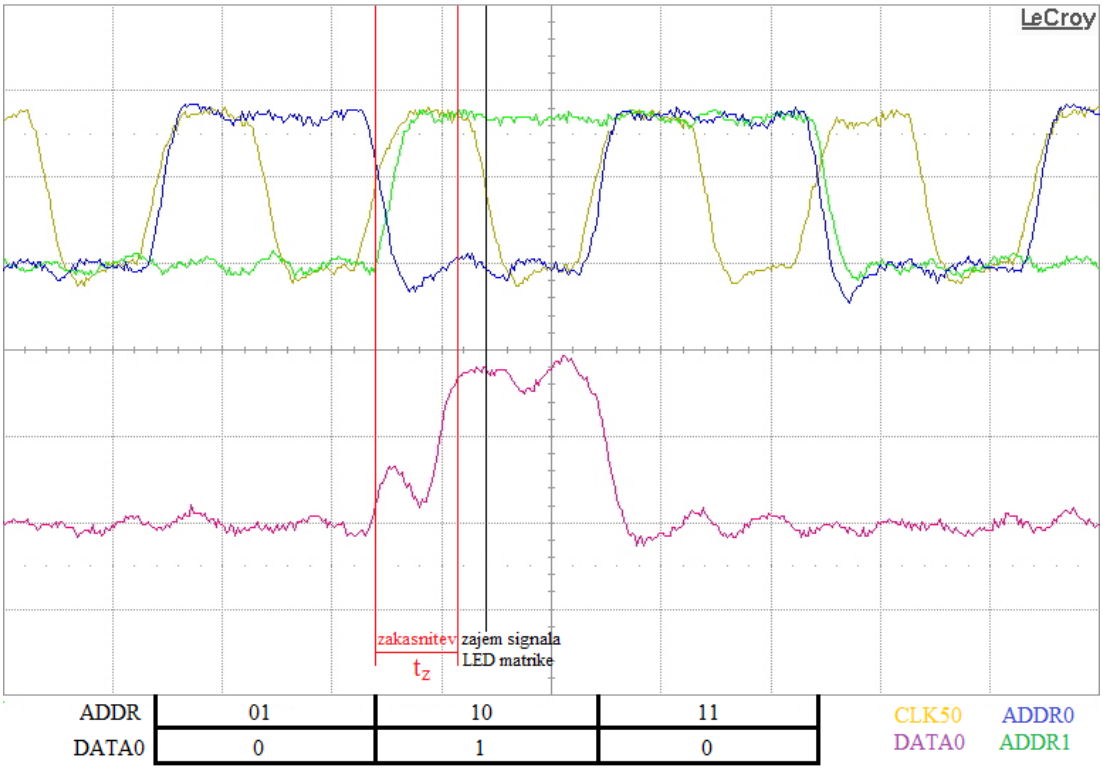
Slika 5.15 prikazuje časovni potek signala *DATA0* v primeru, ko pritisnemo tipko S1 na razširitveni plošči in ko je v času naslova matrike LED postavljeno visoko stanje. Na izhodu CPLD-ja dobimo ob začetku naslova 01 pričakovano zakasnitev signala tipke S1. Po času zakasnitve t_z se na linijo *DATA0* postavi logična enka, ki predstavlja pritisnjeno tipko. Čas zakasnitve, pomerjen z osciloskopom, je bil približno 7,5 ns. Logična enka zaradi aktivnega stanja matrike LED ostane do konca naslova 10, nato se *DATA0* postavi v nizko stanje.



Slika 5.15: Časovni potek signala *DATA0* za zakasnitev aktivnega stanja tipke S1 v času naslova *ADDR*=01 in aktivnega stanja matrike LED v času *ADDR*=10.

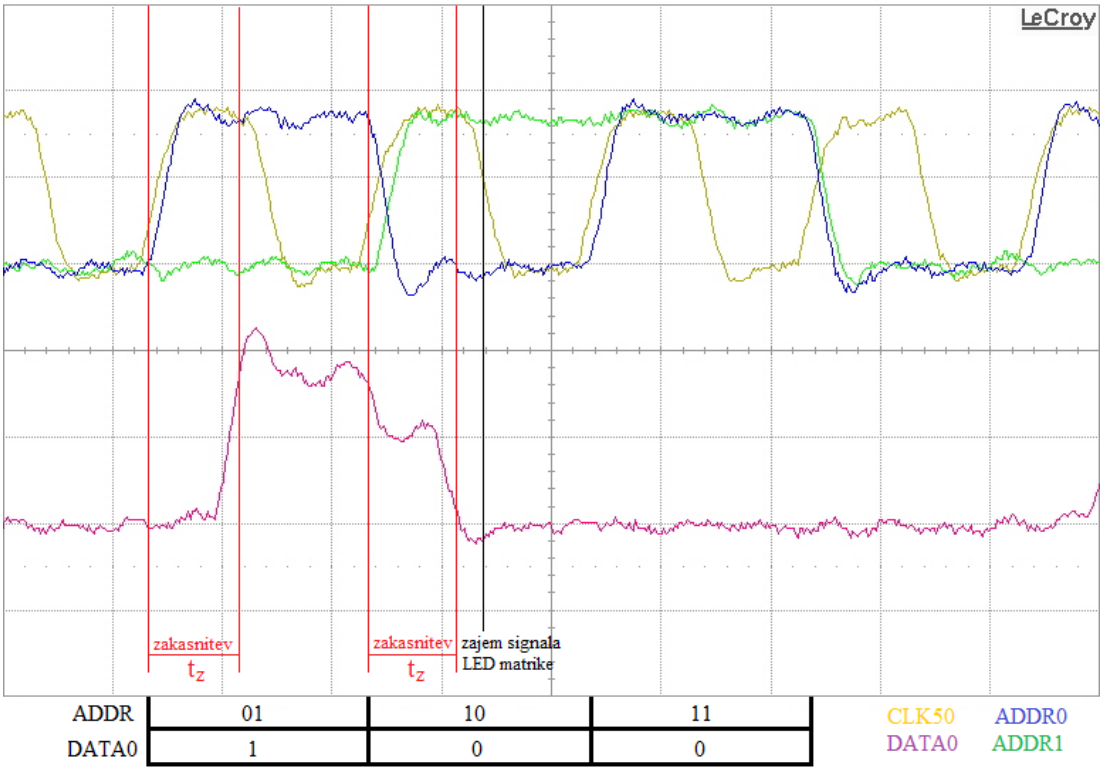
Drugi primer prikazuje časovni potek signala *DATA0* zakasnitve neaktivnega stanja tipke S1 in aktivnega matrike LED. Na sliki 5.16 vidimo, da zaradi zakasnitve t_z logične ničle na izhodu CPLD preide ničla v čas logične enke naslednje

kombinacije naslova *ADDR*. V času zakasnitve dobimo kratek stik med izhodom CPLD in izhodom razvojne plošče. Čas zakasnitve t_z je bil pri merjenju z osciloskopom maksimalno 8,2 ns. Linija *DATA0* ima v tem času prepovedano območje potenciala. Iz poglavja 5.1.2 vemo, da se zajem signalov v razdeljevalniku izvaja ob padajoči fronti sistemske ure *CLK50*. Zajem se zgodi po času kratkega stika na podatkovni liniji, zato kratek stik ne vpliva na zajem signala. Če se naslovi menjajo ob vsaki naraščajoči fronti sistemske ure 50 MHz, je čas trajanja naslova 20 ns. Zajem se izvede na sredini trajanja naslova oz. 1,8 ns po koncu kratkega stika in ne vpliva na interpretacijo potencialov.



Slika 5.16: Časovni potek signala *DATA0* za zakasnitev neaktivnega stanja tipke S1 v času naslova *ADDR*=01 in aktivnega stanja matrike LED v času *ADDR*=10.

Primer za časovni potek signala *DATA0* ob aktivnem stanju tipke S1 in neaktivnem matrike LED prikazuje slika 5.17. Prihod signala tipke na vodilo *DATA0* se zakasni kot pri primeru s slike 5.15. Zakasnitev signala je podobna in se podaljša v čas naslednjega naslova 10. Ker je med prehodom naslova prisotna sprememba logičnega nivoja, dobimo v času zakasnitve t_z ponovno prepovedano območje potenciala. Zajem logične ničle se zgodi kasneje, zato kratek stik ne vpliva na odločanje o stanju signala matrike LED.



Slika 5.17: Časovni potek signala *DATA0* za zakasnitev aktivnega stanja tipke S1 v času naslova *ADDR*=01 in neaktivnega stanja matrike LED v času *ADDR*=10.

V podatkovnem listu proizvajalec *Xilinx* za CPLD *XC9572XL* navaja čas zakasnitve logike med pini (angl. *pin-to-pin*) 5 ns. Pri merjenju časov zakasnitev z osciloskopom so bili izmerjeni časi približno od 6 ns do 8,2 ns. Nekaj zakasnitve se zgodi tudi na linijah sistemske ure in dveh naslovnih.

V kodi razdeljevalnika bi lahko nastavili proženje ob naraščajoči fronti sistemske ure. Proženje bi se zgodilo še pol periode (10 ns) kasneje, vendar bi problem lahko nastal pri odločanju stanja naslovov na razširitveni plošči. Linija sistemske ure in liniji za naslov niso enake dolžine, zato lahko pride do težav pri odločanju zaradi različnih zakasnitev. S proženjem ob padajoči fronti ure v razdeljevalniku ni bilo težav.

5.3 Matrika LED (angl. *LED matrix*)

Blok matrike LED skrbi za izpisovanje znakov na matriki 5 x 7 s pomočjo izbiranja. Izbiranje (angl. *multiplexing*) je tehnika, uporabljena za krmiljenje matrik LED. Z izbiranjem je prižgana le ena vrstica naenkrat. Ta pristop je potreben, ker je ena stran diod (anoda ali katoda) vezana na vrstico. Iz slike 3.2 vidimo, da če bi tok tekel čez dve vrstici istočasno, ni mogoče upravljati posameznih diod v teh dveh vrsticah. Z izbiranjem (skeniranjem po vrsticah) hitro utripnemo in nadaljujemo z naslednjo vrstico tako, da je v primeru, če smo v posamezni vrstici vklopili ali izklopili pravilen stolpec, mogoče dobiti prikaz slike ali znaka. Da naše oko ne zazna utripanja, mora biti hitrost osveževanja dovolj velika. S spreminjanjem hitrosti in časa utripa svetlečih diod lahko prilagajamo svetilnost. Za pravilen prikaz znakov s pomočjo tehnike izbiranja skrbi uporabniška logika matrike LED, ki podatke o znakih pridobi iz bloka pomnilnika znakov.

5.3.1 Pomnilnik znakov (*character_rom.vhd*)

V pomnilniku znakov so v obliki logičnih ničel in enk sestavljeni znaki v velikosti, prilagojeni matriki LED. Znaki so sestavljeni ročno. Nabor znakov je decimalno od 32 do 95 po ASCII tabeli (Slika 5.18). Pomnilnik ne vsebuje znakov malih črk zaradi slabe ločljivosti matrike (z velikostjo matrike 5 x 7 težko ločimo med znaki malih črk).

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Slika 5.18: ASCII tabela znakov [45].

Pomnilnik znakov je dokaj enostaven. Vsebuje polje (angl. *array*) 448 vrstic in 5 stolpcev. Vsaka vrstica je predstavljena z 9-bitnim naslovom *addr_rom*. Zajema 5-bitni podatek iz ničel in enk. Vsak znak je sestavljen iz sedmih naslovov, da dobimo velikost znakov 5 x 7. Ko v *addr_rom* vpišemo naslov vrstice, nam blok vrne 5-bitni podatek *data_rom* z vrstice vpisanega naslova. Tabela 5.7 prikazuje opis signalov.

Ime	Dolžina	Tip	Opis
addr_rom	9	vhodni	Naslov vrstice
data_rom	5	izhodni	Podatki v vrstici (logične ničle in enke)

Tabela 5.7: Opis signalov pomnilnika znakov.

Skupno imamo 64 znakov. Nabor vseh znakov prikazuje slika 5.18. Spodaj je primer sestavljene velike črke N:

```
type rom_type is array(0 to 447) of std_logic_vector(4
downto 0);  --64 znakov
    constant FONT: rom_type :=
    (
        -- 78  (4E hex)
        "10001", --#   #
        "10001", --#   #
        "11001", --##  #
        "10101", --# # #
        "10011", --#  ##
        "10001", --#   #
        "10001", --#   #
        ...
     );
    data_rom <= FONT(conv_integer(addr_rom));
```

5.3.2 Uporabniška logika matrike LED (*ledmatrix.vhd*)

Komponenta UL matrike LED skrbi za pravilen format izpisa podatka, ki ga zahteva razdeljevalnik na razvojni plošči za krmiljenje matrike LED (Poglavje 5.1.1). Poleg pravilnega formata zapisa skrbi tudi za pravilno časovno osveževanje vrstic.

Načinov operiranja z znaki je veliko. V našem primeru komponenta sprejme binarno vrednost *number* za en znak (po ASCII tabeli). Za primer znaka »N«, vpišemo binarno vrednost 01001110 (78). UL s pomnilnika znakov pridobi podatke aktivnih pik v vrstici posameznih znakov. Informacije vrstic se izpisujejo na izhodu *data_led*. Opis signalov UL matrike LED prikazuje tabela 5.8.

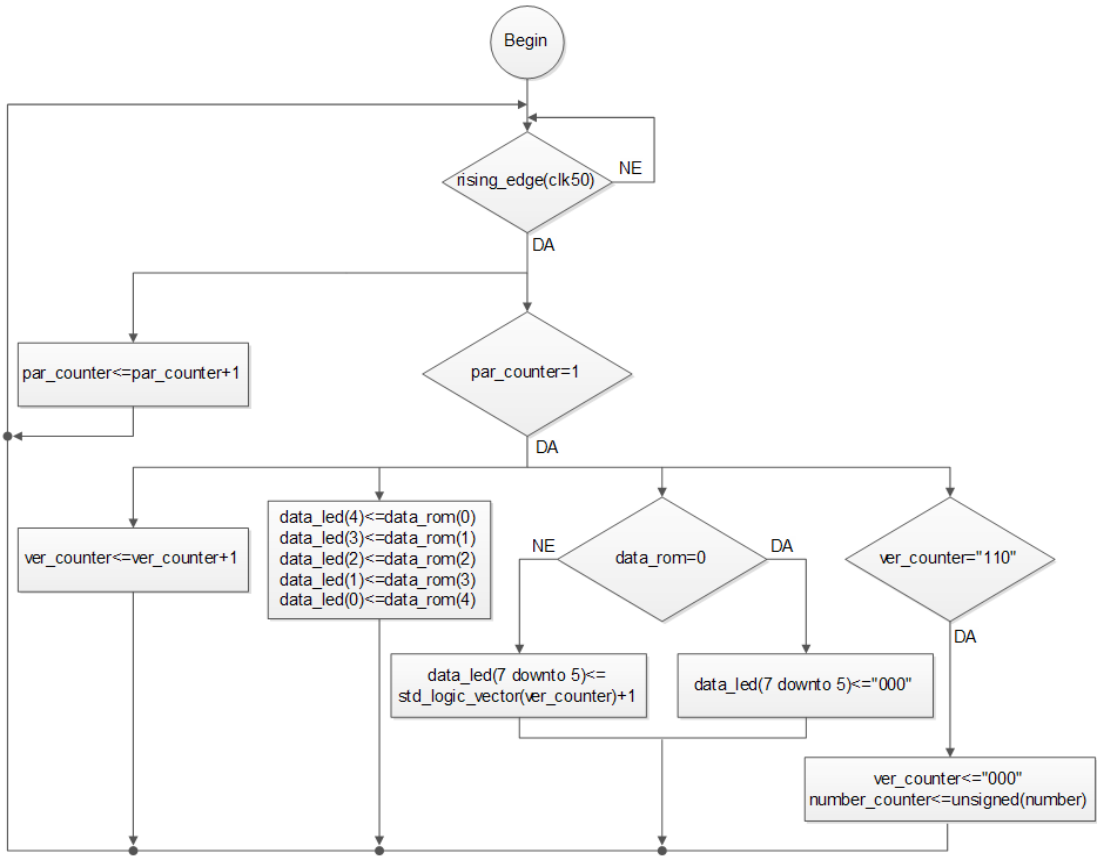
Ime	Dolžina	Tip	Opis
clock50	1	vhodni	Sistemska ura
number	8	vhodni	Vpis ASCII znaka
data_led	8	izhodni	Podatek za krmiljenje matrike LED

Tabela 5.8: Opis signalov UL matrike LED.

5.3.3 Delovanje UL matrike LED

Slika 5.19 prikazuje diagram poteka sekvenčnega procesa matrike LED. Ob vsaki naraščajoči fronti sistemske ure se za ena poveča 17-bitni števec *par_counter*, ki služi kot delilnik frekvence sistemske ure. Ob končni vrednosti sproži vpis podatka *data_rom* o vrstici aktualnega znaka v niz *data_led(0..4)* za krmiljenje stolpcev. UL pridobi informacijo s pomnilnika znakov. Ostali niz *data_led(5..7)* predstavlja aktivno vrstico matrike LED in se določi glede na stanje števca *ver_counter*. Števec beleži aktualno vrstico, ki jo izpisujemo. Ta se poveča za ena ob vsakem končnem stanju števca *par_counter*. Ko prešteje do 110 (šest), pomeni, da smo v zadnji vrstici znaka, nato se ponastavi na nič.

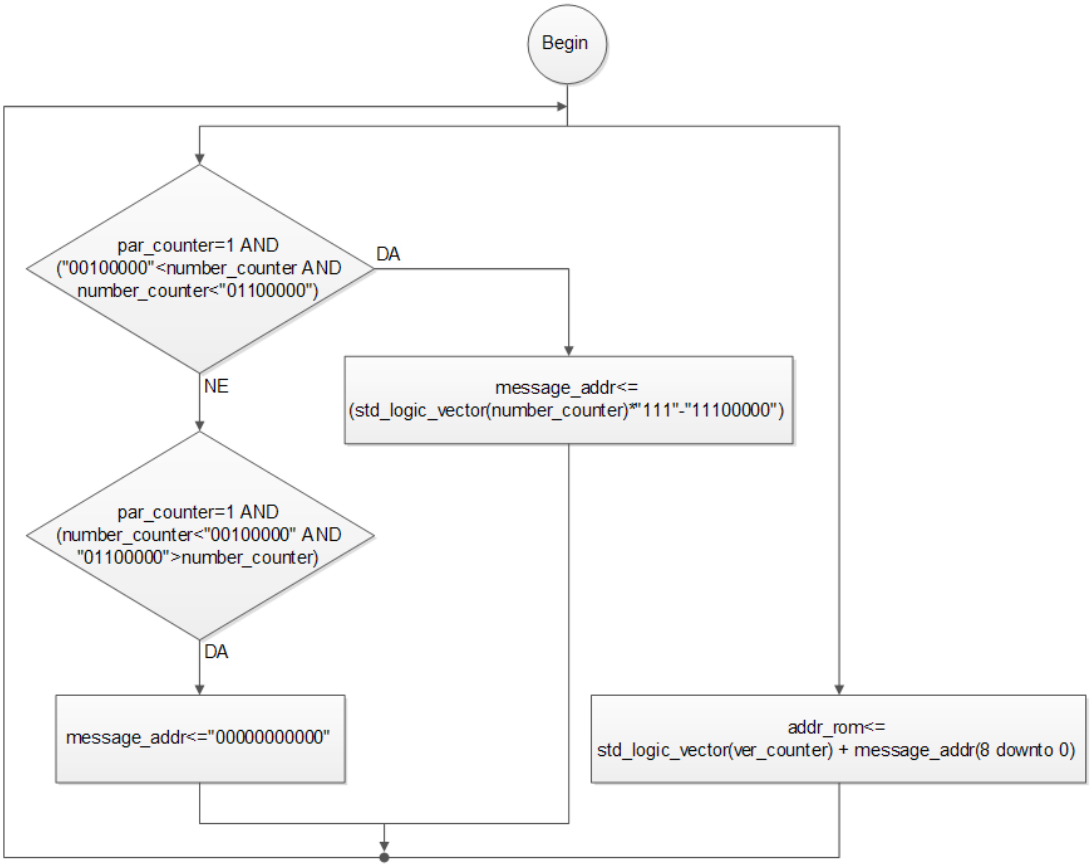
Glede na vhodno vrednost *number* se nastavi števec *number_counter*. Števec služi za nastavitev naslova posamezne vrstice znakov za prikaz, shranjenih v pomnilniku znakov. Vsaka vrstica v pomnilniku je predstavljena z 9-bitnim naslovom *addr_rom* (Poglavje 5.3.1). Naslov se nastavi skupaj z robnimi pogoji števcev *par_counter*, *number_counter* in *ver_counter* (Slika 5.20). Z robnimi pogoji določimo območje uporabljenih znakov iz ASCII tabele.



Slika 5.19: Diagram poteka sekvenčnega procesa matrike LED.

V primeru, da je vrednost *number_counter* znotraj območja znakov (Slika 5.20 blok zgoraj levo), lahko nastavimo naslov *addr_rom* za pridobitev podatka *data_rom*. Za pridobitev naslova moramo najprej izračunati vrstico v pomnilniku znakov, v katerem se nahaja podatek za izpis. V pomoč je 11-bitna vrednost *message_addr*, ki predstavlja zaporedno številko prve vrstice želenega znaka (v pomnilniku znakov). Izračuna se z množenjem števca *number_counter* s številom vrstic enega znaka 111 (sedem). Ker predstavlja števec *number_counter* znake od 32 naprej, moramo tej vrednosti odšteti 11100000 (32 znakov po 7 vrstic = 224 vrstic). S tem dobimo prvo vrstico znaka.

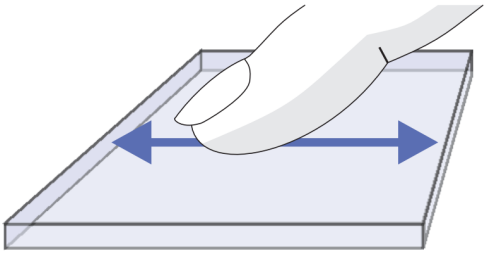
Naslov vrstice *addr_rom* v pomnilniku znakov sedaj dobimo, ko nizu *message_addr*, prištejemo stanje števca *ver_counter*. S povečevanjem števca od 0 do 6 se pomikamo po vrsticah. S tem pridobivamo podatke o vrsticah znakov *data_rom*, ki jih pošiljamo na razširitveno ploščo za izpis znaka na matriki LED. Hitrost osveževanja je fiksno nastavljena s števcem *par_counter*. Pri frekvenci sistemske ure 50 MHz je hitrost osveževanja znaka 54,5 Hz.



Slika 5.20: Diagram poteka prireditvenega stavka matrike LED.

5.4 Drsnik (angl. *Slider*)

Drsnik je enodimenzijski senzor, ki ob dotiku (ali v bližini) zaznava linearno gibanje prstov, premika po eni osi. Sestavljen je iz skupine blokov za prepoznavanje dotikov, nameščenih v vrsto. Stanje vsakega bloka se uporabi za določanje položaja vzdolž njih.



Slika 5.21: Vzdolžni drsnik [35].

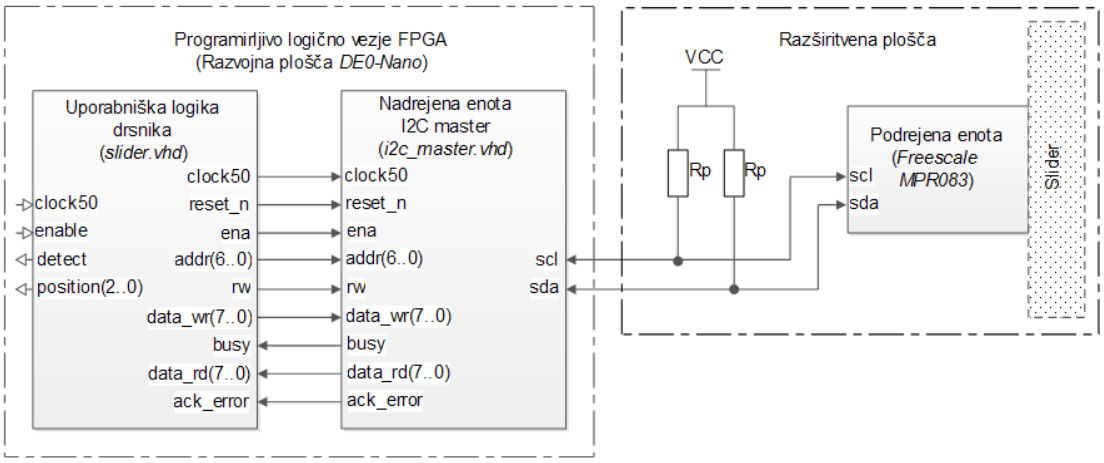
Za pravilno delovanje drsnika na razširitveni plošči moramo ob priklopu na napajanje najprej nastaviti registre integriranega vezja *Freescale MPR083*. Vezje ima skupno 12 registrov, preko katerih beremo ali vpisujemo podatke. Za nastavitev delovanja nismo uporabili vseh, ker smo z uporabljenimi dobili želeno delovanje. Opis uporabljenih je v poglavju 5.4.9. Za pravilno branje in nastavljanje registrov v pravilnem zaporedju skrbi uporabniška logika drsnika.

Dostop do registrov integriranega vezja *MPR083* zagotovimo s komunikacijo po protokolu I^2C. Kodo v jeziku VHDL (*i2c_master.vhd*), ki zagotavlja komunikacijo I^2C, smo poiskali na spletu [31]. Da bi jo s pridom uporabili, smo jo morali razumeti.

5.4.1 *I2C master* (*i2c_master.vhd*)

Komponenta komunicira z uporabniško logiko drsnika. Pridobljene podatke posreduje ali prejema preko serijskega vodila I^2C. Vodilo je opisano v poglavju 5.4.8. Komponenta *I2C master* je nadrejena enota (protokola I^2C), ki komunicira s podrejenim integriranim vezjem drsnika. Shematski prikaz prikazuje slika 5.22.

Slika 5.22 za lažje razumevanje prikazuje signala *SCL* in *SDA,* povezana neposredno na komponento *I2C master*. Signala v resnici potujeta preko uporabniške logike, ki na njiju ne vpliva. Tabela 5.9 prikazuje opis signalov komponente *I2C master*.



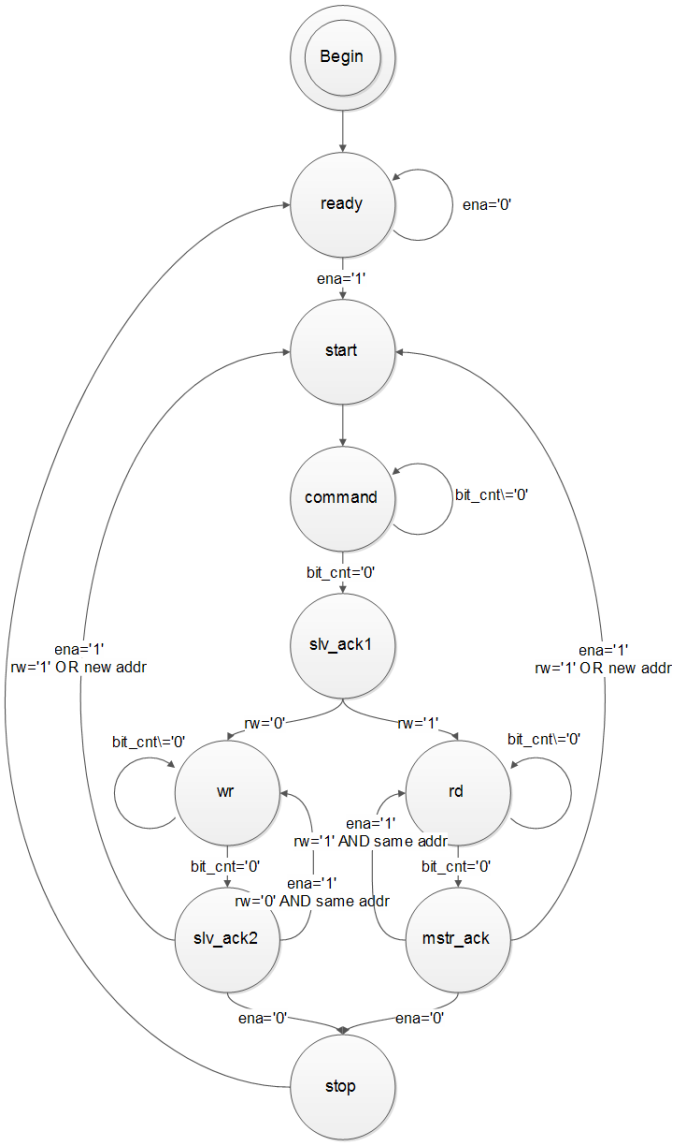
Slika 5.22: Shematski prikaz nadrejene enote *I2C master* z uporabniško logiko in podrejeno enoto.

Ime	Dolžina	Tip	Opis
clock50	1	vhodni	Sistemska ura
reset_n	1	vhodni	Asinhrona ponastavitev (aktivno nizko stanje)
ena	1	vhodni	0: prenos se ne začne 1: sproži prenos. Če je *ena* visok ob koncu prenosa (ko gre *busy* v nizko stanje), se prenos nadaljuje z novimi podatki.
addr	7	vhodni	Naslov podrejene enote
rw	1	vhodni	0: pišemo 1: beremo
data_wr	8	vhodni	Podatki za oddajo (če je *rw='0'*)
data_rd	8	izhodni	Prejeti podatki (če je *rw='1'*)
busy	1	izhodni	0: *I2C master* je nedejaven in zadnji podatki branja so v *data_rd*.
ack_error	1	medpomnilni (angl. *buffer*)	0: pri potrditvi ni napake 1: med zadnjim prenosom je prišlo do potrditvene napake. Napaka se izbriše na začetku vsakega prenosa.
sda	1	vhodni/izhodni	Serijska podatkovna linija vodila I^2C
scl	1	vhodni/ izhodni	Serijska linija ure vodila I^2C

Tabela 5.9: Opis signalov komponente *I2C master*.

5.4.2 Delovanje *I2C master*

Za lažje razumevanje delovanja poglejmo diagram stanj na sliki 5.23. Ob zagonu skočimo v stanje *ready* in čakamo v stanju, dokler ne preide vhodni signal *ena* v visoko stanje. V stanju *ready* je signal *busy* enak 0. Ob prehodu v naslednje stanje *start* se postavi na 1 (Slika 5.24). Zajamejo se parametri, kot so naslov ciljne naprave *addr*, 8-bitni podatek *data_wr* ter ukaz *rw*. Slednji določi, ali bomo podatke v prvem delu prenosa (Slika 5.25) protokola I^2C vpisovali ali brali. V stanju *start* se postavi tudi začetno stanje (*START* bit) na I^2C vodilu (Slika 5.25). V stanju smo le en urin cikel, nato skočimo v stanje *command*. V stanju ostanemo, dokler se preko vodila ne vpiše celoten naslov ciljne naprave (podrejene enote drsnika) in na koncu doda bit *rw*, ki določi smer podatkov.



Slika 5.23: Diagram stanj komponente *I2C master*.

Če želimo ciljni napravi sporočiti, do katerega registra želimo dostopati, mora *data_wr* v prvem delu prenosa predstavljati naslov registra, signal *rw* pa visok nivo. Ko se postopek izvede, se premaknemo v stanje *slv_ack1*. V stanju se preveri potrditev ciljne naprave. Koda v tem stanju glede na *rw* komando odloči, v katero stanje nadaljuje.

V primeru branja podatkov z naslovljene podrejene naprave preidemo v stanje *rd,* nasprotno v stanje *wr*. Stanji sta si podobna. V stanju *wr* smo, dokler se na vodilo ne vpiše celoten niz podatkov *data_wr*, nato skočimo v stanje *slv_ack2*, kjer se ponovno preveri odziv podrejene enote. V stanju *rd* smo, dokler ne preberemo vseh 8-bitov niza *data_rd*, ter skočimo v stanje *mstr_ack*, kjer se odda odziv naše naprave. Iz stanja *slv_ack2* lahko ponovimo vpisovanje (skočimo nazaj v stanje *wr*) tako, da ohranimo *ena* na 1, *rw* na 0 ter naslov ciljne naprave nespremenjen. Le v enem od prejšnjih stanj, kjer je bil *busy* enak 1, spremenimo podatek (*data_wr*), ki ga želimo vpisati (Slika 5.24a). V primeru, ko spremenimo signal *rw* v visok nivo ali spremenimo naslov, skočimo v stanje *start* (*busy* se ob tem postavi na 0) in se ponovi tudi začetno stanje protokola I^2C. Ta primer izvedemo v primeru, ko želimo spremeniti naslov podrejene naprave ali smer podatkov.

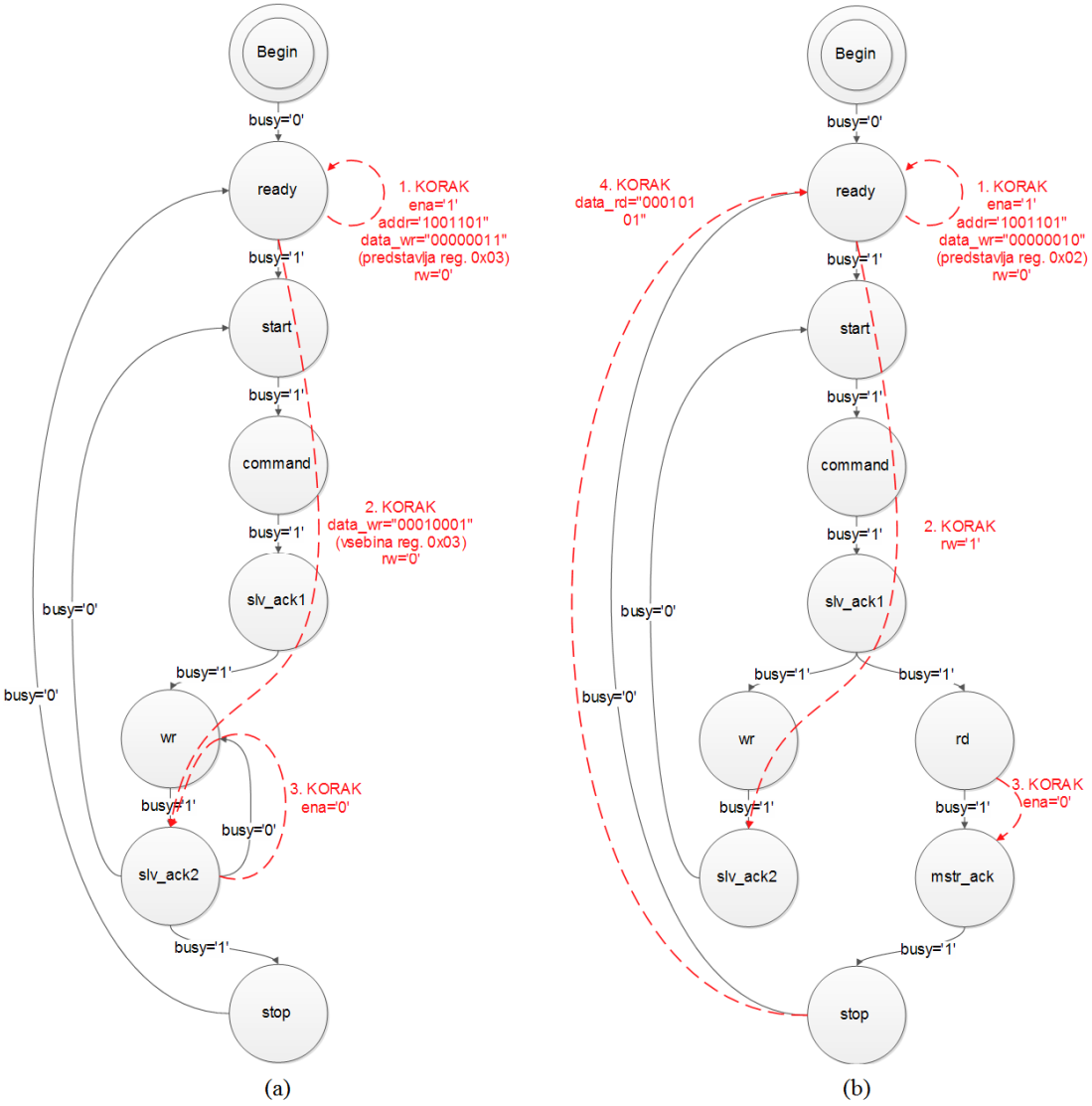
Ko želimo zaključiti prenos, postavimo *ena* na nizek nivo ter preidemo v stanje *stop* (*busy* se postavi na 0). Na vodilu se generirajo končni pogoji (*STOP* bit), vrnemo se v stanje *ready*. Stanje *mstr_ack* je podobno stanju *slv_ack2*; poleg oddaje odziva se razlikuje v tem, da iz stanja lahko ponavljamo branje podatkov.

Lažje predstavljanje upravljanja komponente prikazuje slika 5.24. Slika 5.24a prikazuje primer vpisa binarne vrednosti 00010001 v register na naslovu 0x03 v podrejeno napravo na naslovu *0x4D*. Slika 5.24b prikazuje primer branje podatkov z registra na naslovu 0x02 s podrejene naprave na naslovu *0x4D*.

Torej, če želimo v register na naslovu 0x03 vpisati vrednost 00010001, moramo v prvem koraku postaviti signale *ena*, *addr*, *data_wr* in *rw* kot prikazuje slika 5.24a. Podatek *data_wr* v prvem koraku predstavlja naslov registra. *Data_wr* bo v prvem koraku binarna vrednost naslova registra 0x03 (binarno 00000011). V drugem koraku bo *data_wr* enak vsebini registra (00010001). V tretjem koraku postavimo *ena* v nizko stanje, da zaključimo postopek.

Za drugi primer (Slika 5.24b) je prvi korak enak. V *data_wr* vpišemo binarno vrednost naslova registra 0x02 (00000010), s katerega bomo prebrali vrednost. V drugem koraku spremenimo ukaz *rw* v visoko stanje za branje podatkov z registra. V tretjem koraku spremenimo *ena* v nizko stanje. V zadnjem koraku odčitamo prebran podatek, zapisan v nizu *data_rd.*

Rdeče obarvane oznake prikazujejo korake, ki so potrebni za pravilno (želeno) delovanje komponente *I2C master*. Te korake izvaja uporabniška logika, opisana v poglavju 5.4.5.



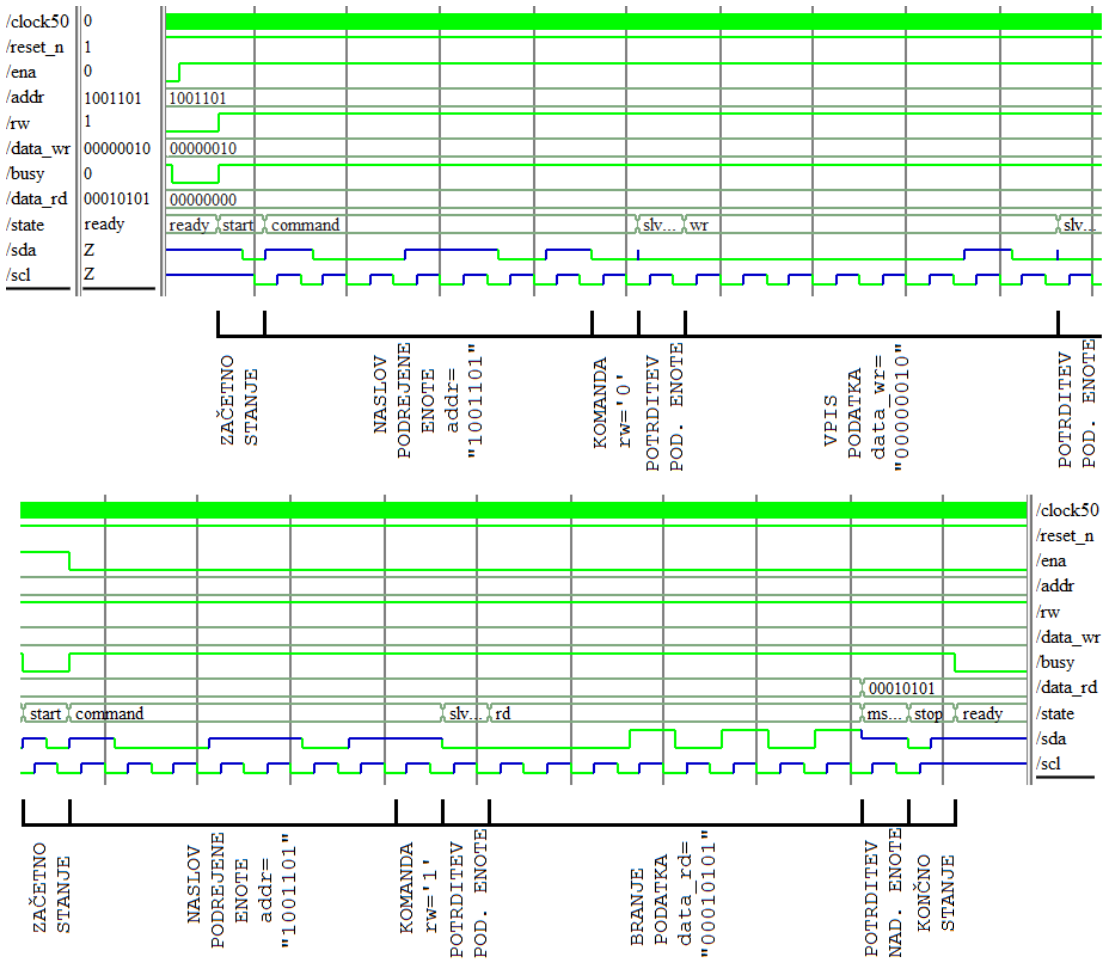
Slika 5.24: Primer upravljanja in uporabe komponente *I2C master* a) vpis binarne vrednosti "00010001" na naslov 0x03, naprave *0x4D* in b) branje z naslova 0x02, naprave *0x4D*.

Kako pa vemo, v katerem koraku smo in kdaj lahko spremenimo vhodne signale (*ena*, *addr*, *rw*, *data_wr*) komponente ne da bi prišlo do napak?

Kot lahko vidimo iz tabele 5.9, ki opisuje signale, vidimo, da imamo izhoden signal *busy*, ki označuje, kdaj poteka prenos. Prav s tem signalom se lahko orientiramo, kdaj je potrebno spremeniti stanja vhodov komponente *I2C master*, da zagotovimo pravilno delovanje. Uporabniška logika drsnika za opisano komponento je predstavljena v poglavju 5.4.5.

5.4.3 Simulacija kode *I2C master* – primer prenosa

Slika 5.25 prikazuje časovni potek branja podatkov za primer s slike 5.24b. Uporabniška logika poskrbi za sistemsko uro *clock50*, *reset_n*, naslov podrejene naprave *addr*, ukaz *rw* ter podatek za vpis *data_wr*, ki predstavlja register na naslovu 0x02. Ob prehodu signala *ena* v visoko stanje se vzpostavi začetno stanje (*START* bit) na vodilu I^2C. Za začetnim stanjem sledi naslov podrejene naprave *addr*. Parametri so se shranili, zato ukaz *rw* postavimo v visoko stanje, ki bo zagotovilo branje podatkov. Ko se vpis naslova na vodilo izvede, se doda bit *rw* enak nič. Sledi potrditev podrejene enote, vpis 8-bitnega niza *data_wr* ter ponovna potrditev prejema podatkov. Smo na polovici poteka. Sledi identičen drugi del, kjer se izvede branje podatkov z registra na naslovu 0x02, potrditev prejema nadrejene enote ter vzpostavitev končnega stanja (*STOP* bit).



Slika 5.25: Časovni potek branja podatka z registra na naslovu 0x02 po protokolu I^2C, s podrejene naprave na naslovu *0x4D* (simulacija kode *I2C master*).

5.4.4 Nastavitev hitrosti serijske ure *SCL* (angl. *Serial Clock Line*)

Komponenta *I2C master* omogoča nastavitev frekvence serijske ure na vodilu I^2C. Za zagotovitev zanesljive komunikacije smo frekvenco nastavili nekoliko nižje od maksimalne delovne frekvence integriranega vezja *MPR083*. Nastavili smo jo na 20 kHz, ker zadostuje našim potrebam drsnika.

Komponenta vsebuje dva splošna parametra za nastavitev serijske ure. To sta *input_clk* in *bus_clk*. Parameter *input_clk* mora biti nastavljen na vhodno sistemsko uro komponente *I2C master*, parameter *bus_clk* nastavimo kot želeno frekvenco serijske ure *SCL*. Koda je prikazana spodaj:

```
GENERIC(
  input_clk : INTEGER := 50_000_000; --vhodna hitrost ure
v Hz
  bus_clk : INTEGER := 20_000); --hitrost vodila i2c
(scl) v Hz
```

5.4.5 Uporabniška logika drsnika (*slider.vhd*)

Uporabniška logika drsnika upravlja s komponento *I2C master* (Slika 5.22). Logika skrbi za pridobivanje položajev dotikov ter pravilno delovanje integriranega vezja drsnika *MPR083*.

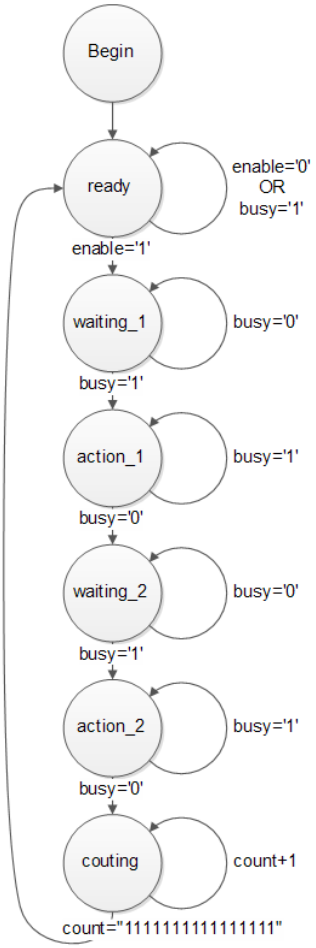
Tabela 5.10 prikazuje opis signalov uporabniške logike drsnika. Notranji signali, ki potekajo med UL in komponento *I2C master,* so prikazani na tabeli 5.9.

Ime	Dolžina	Tip	Opis
clock50	1	vhodni	Sistemska ura
sda	1	vhodni/izhodni	Serijska podatkovna linija vodila I^2C
scl	1	vhodni/ izhodni	Serijska linija ure vodila I^2C
enable	1	vhodni	0: onemogočimo delovanje 1: omogočimo delovanje
detect	1	izhodi	0: ni dotika 1: zaznan dotik
position	3	izhodi	Trenutni položaj dotika, ko je zaznan. 000: elektroda 1 ~ 111: elektroda 8

Tabela 5.10: Opis signalov uporabniške logike drsnika.

5.4.6 Opis UL drsnika

Ob prvem prehodu signala *enable* v visoko stanje (Slika 5.27) sprožimo nastavljanje registrov, potrebnih za delovanje drsnika, opisanih v poglavju 5.4.9. Ko so registri nastavljeni, logika prične z branjem položajev dotika. Informacijo vsebuje rotacijski statusni register (angl. *rotary status register*) na naslovu *0x02*. Ker se po vsaki izvedeni komunikaciji protokola I^2C (po končnem *STOP* bitu protokola) s podrejeno napravo (primer komunikacije prikazuje slika 5.25) sproži 16-bitni števec *count*, ki šteje do 65635 preden skoči v stanje *ready* (Slika 5.27), dobimo preračunano 1,3 ms zakasnitve. Zakasnitev je namerna, da v popolnosti ne zasedemo vodila ter da s tem tudi zmanjšamo frekvenco branja položajev iz 513 Hz na 308 Hz. Med stalno osveževanje smo vrinili tudi register napak (angl. *fault register*) z naslova *0x01*. Za konstantno osveževanje skrbi 8-bitni števec *error_count* (Slika 5.27). Ko prešteje do konca, se sproži preverjanje napak. Ker se napake, opisane v poglavju 5.4.10, ne dogajajo pogosto, je osveževanje napak več kot enkrat na sekundo zadostno.



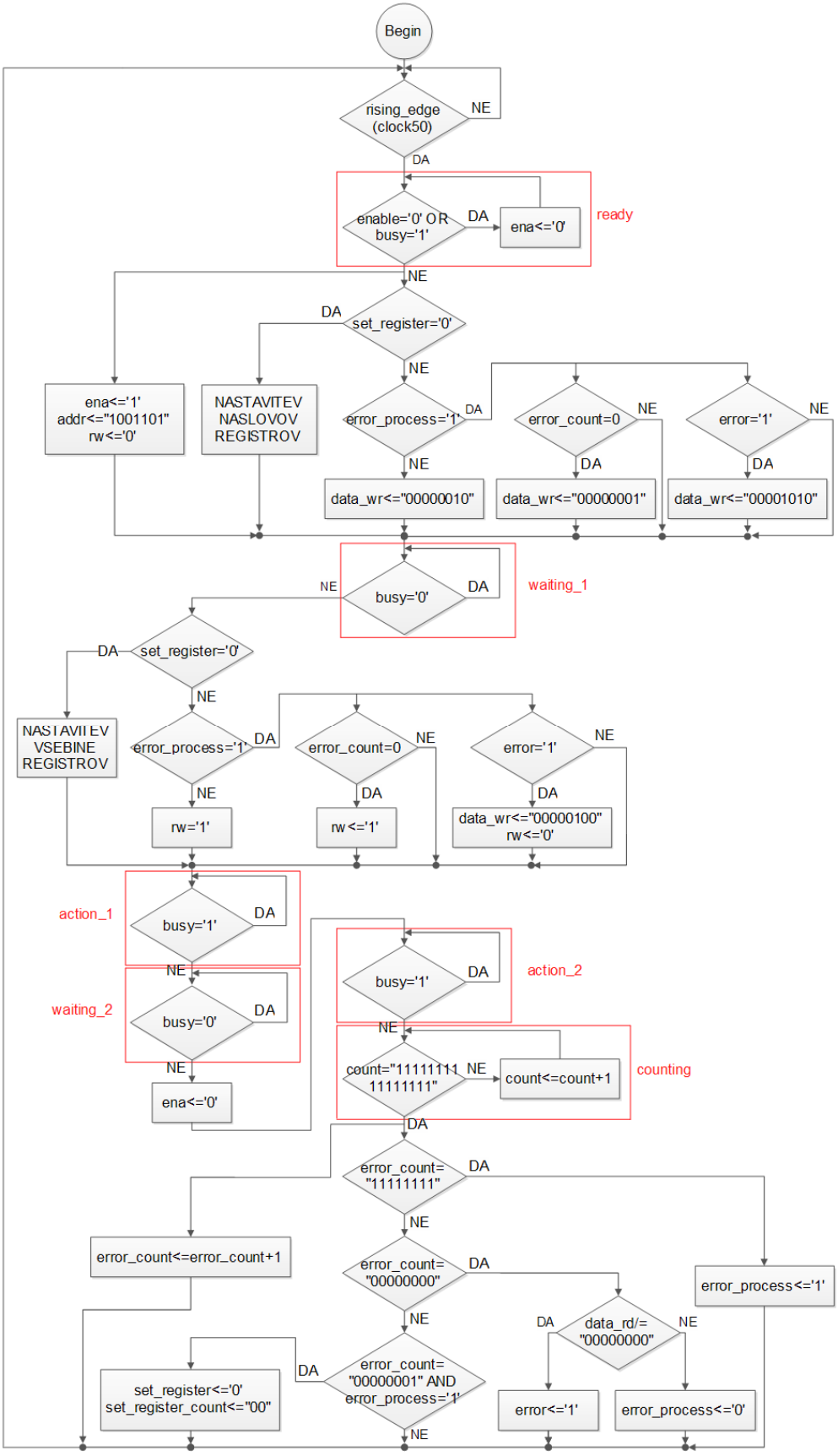
Slika 5.26: Diagram stanj uporabniške logike drsnika.

Če v pridobljenih podatkih registra napak ni bila zabeležena nobena od napak, se nadaljuje postopek pridobivanja podatkov položajev dotikov. V nasprotnem primeru se nastavi vsebina konfiguracijskega registra (angl. *configuration register*) na naslovu *0x0A* tako, da se nastavljeno stanje integriranega vezja drsnika ponastavi (resetira) ter preide v stanje, ki je enako stanju ob priklopu vezja na napajanje (*STOP* stanje). Ko se ta postopek izvede, sledi ponovno nastavljanje registrov.

Slika 5.26 prikazuje diagram prehajanje stanj uporabniške logike drsnika s pomočjo signala *busy*. Kot že povedano se s pomočjo slednjega upravlja komponenta *I2C master*.

5.4.7 Delovanje UL drsnika

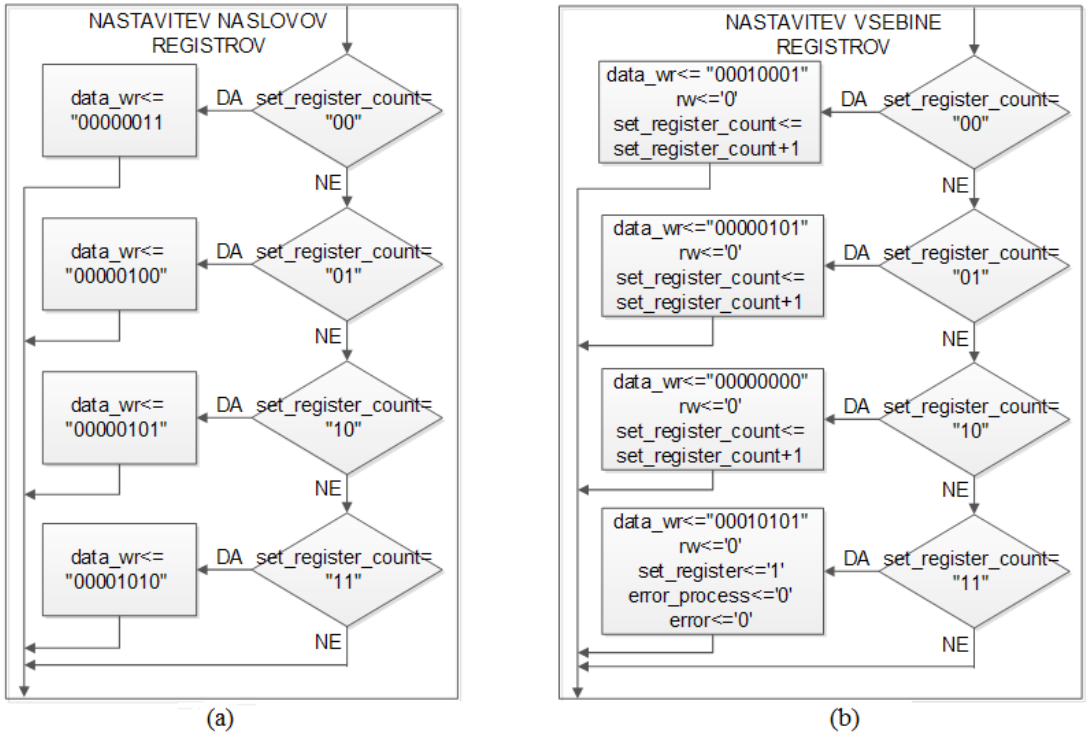
Celotno delovanje uporabniške logike drsnika prikazuje diagram poteka (Slika 5.27). Za uporabo drsnika upravljamo uporabniško logiko samo s signalom *enable* ter beremo podatke *detect* in *position*. Ko je signal *enable* v visokem stanju, se v primeru nenastavljenih registrov čipa *MPR083* začne postopek nastavljanja. Posamezno nastavimo štiri registre, prikazane na sliki 5.28a, tako da v niz *data_wr* vpišemo binarno vrednost naslova registra. Ko komponenta *I2C master* zajame niz *data_wr* in ostale parametre *ena*, *addr* ter *rw*, javi to s prehodom signala *busy* v visoko stanje, pred tem pa čaka v zanki *busy* je enak 0. Med vsako spremembo signala *busy* nastavimo določene parametre. Ko se zajem zgodi, poskrbimo za nastavitev vsebine prvega registra (Slika 5.28b) z binarnimi vrednostmi v niz *data_wr*. Za vpis poskrbimo z visokim stanjem signala *rw*. Kasneje s signalom *ena,* enakim 0, zaključimo nastavljanje prvega registra. Po končanem prenosu (Slika 5.27) vrinemo še zakasnitev s števcem *count*. Celoten postopek se ponovi tudi za ostale registre. Po vseh nastavljenih registrih se niz *data_wr* nastavi na vrednost, ki predstavlja rotacijski statusni register. To omogoči *set_register* v visokem stanju (Slika 5.28b). V registru se beležijo položaji dotikov.



Slika 5.27: Diagram poteka sekvenčnega procesa uporabniške logike drsnika.

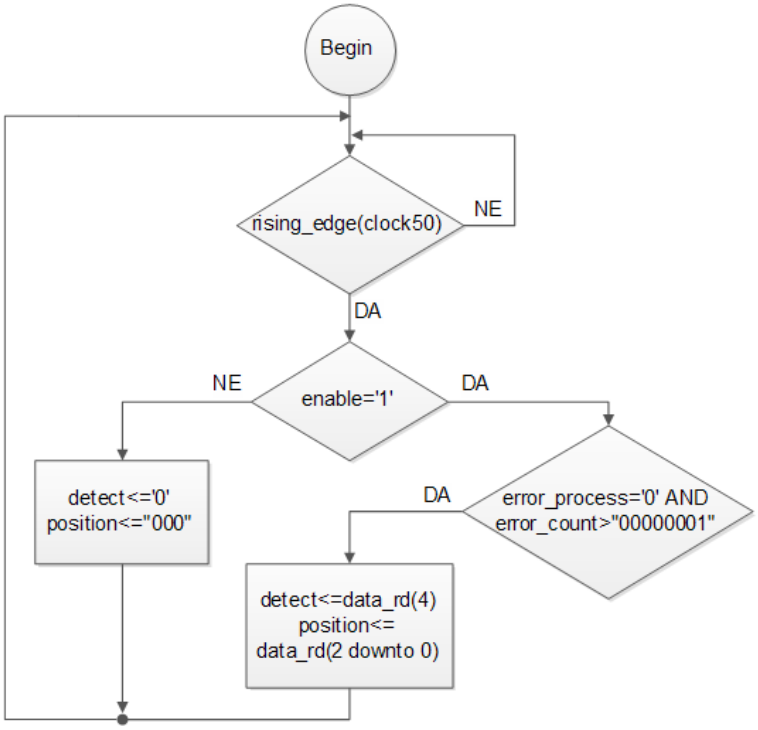
Da zagotovimo zanesljivost delovanja drsnika, moramo preverjati tudi register napak. Za to skrbi števec *error_count* in pogoj, ko je *error_count* enak 11111111 (prikazan na koncu diagrama poteka na sliki 5.27). Ob izpolnjenem pogoju se zapiše *error_process* enak 1. To zagotovi, da se na začetku prenosa (Slika 5.25) v *data_wr* nastavi vrednost, ki predstavlja register napak. Ali je v registru zapisana napaka, preverimo z vsebino registra napak, če je stanje različno od 0 (*data_rd* /= 00000000). V primeru napake postavimo signal *error* v visok stanje. Z njim zabeležimo, da je prišlo do napake. Ob naslednjem ciklu prenosa nastavimo konfiguracijski register tako, da zagotovimo ponastavitev integriranega vezja drsnika. V primeru, da ni napak, postavimo *error_process* na 0 ter nadaljujemo z branjem rotacijskega statusnega registra.

Na sliki 5.28 sta prikazana diagrama poteka za posamezna bloka. Primer a) prikazuje nastavitev naslovov registrov in primer b) nastavitev vsebine registrov. Poleg vpisa vsebine registra v niz *data_wr* (Slika 5.28b) nastavimo signal *rw* na nizko stanje, tako da se vsebina lahko vpiše v posamezne registre. Poveča se tudi števec *set_register_count*, ki omogoči nadaljevanje z naslednjim registrom. Razlikuje se le zadnji blok, kjer nastavimo *error_process* in *error* na 0 ter *set_register* na 1, s čimer povemo, da smo končali z nastavljanjem registrov.



Slika 5.28: Diagram poteka uporabniške logike drsnika za blok; a) nastavitev naslovov registrov in b) nastavitev vsebine registrov.

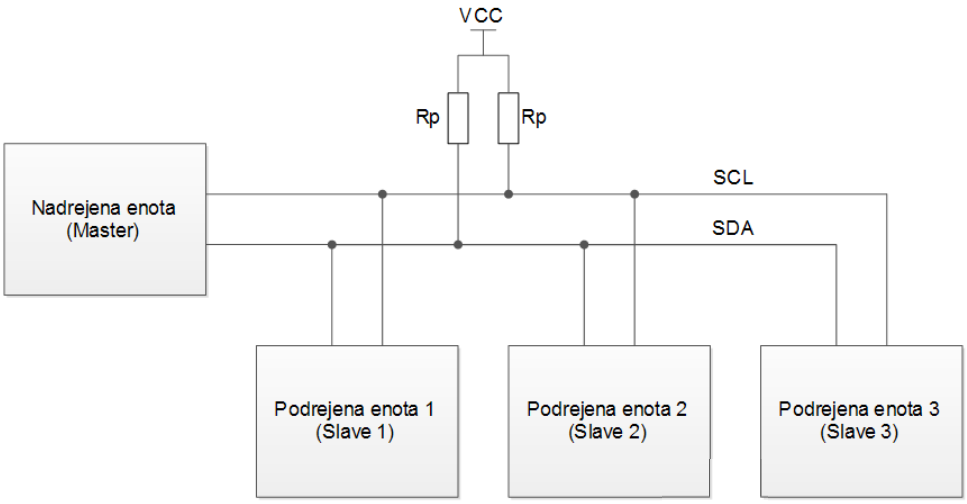
Prebrane vrednosti z rotacijskega statusnega registra prepišemo na izhoda *detect* in *position*. Zgodi se, ko se prebere vrednost registra in je vhod za omogočanje *enable* aktiven. V nasprotnem primeru se postavijo stanja izhodov na nič. Diagram poteka sekvenčnega procesa za prepis prikazuje slika 5.29.



Slika 5.29: Diagram poteka sekvenčnega procesa; prepis rotacijskega statusnega registra na izhod.

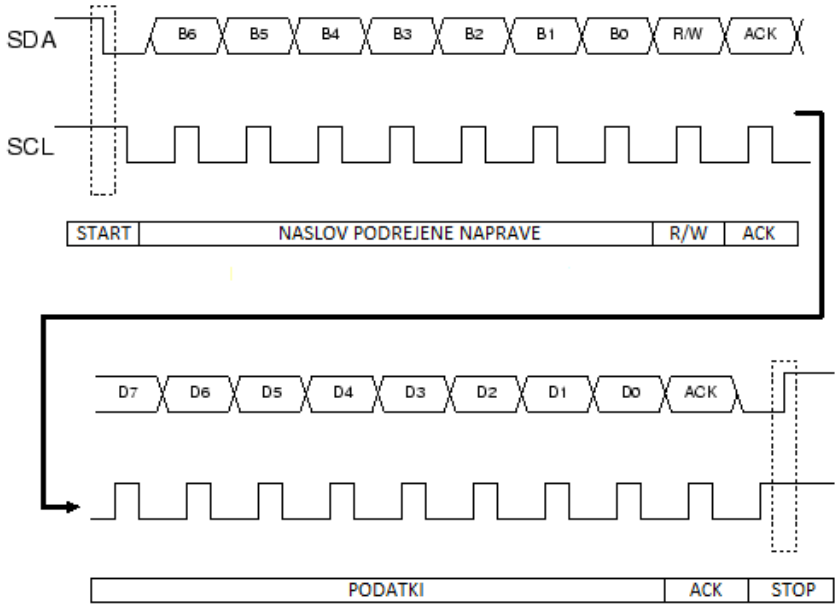
5.4.8 Vodilo I^2C

I^2C vodilo je dvožična (angl. *two wire*) izmenično enosmerna povezava (angl. *half duplex*), ki jo je izumil Philips. Vodilo ima dve aktivni dvosmerni liniji *SDA* in *SCL*, ki ju preko dvižnih uporov Rp (angl. *pull-up*) vežemo na napajanje (Slika 5.30). *SCL* je serijska linija ure, ki sinhronizira prenos ter določa hitrost komunikacije. *SDA* je serijska podatkovna linija, ki služi za prenos ukazov in podatkov. Naprave na vodilu komunicirajo tako, da sporočijo logično ničlo s sklenjeno podatkovno linijo proti masi, logično enko pa s sprostitvijo vodila. Vsaka naprava, priključena na vodilo in skupno maso, ima svoj naslov in lahko deluje kot sprejemnik in oddajnik. Komunikacijo upravlja nadrejena enota (angl. *master*), ki naslavlja podrejene enote (angl. *slave*) in jim pošilja ukaze in podatke. Podatke lahko pošilja tudi podrejena enota, vendar samo takrat, ko nadrejena enota to zahteva [32].



Slika 5.30: Vezava naprav na vodilu I^2C.

Komunikacija se začne z začetnim stanjem (*START* bit), ki ga generira nadrejena enota (Slika 5.31). Linijo *SDA* sklene proti masi, medtem ko je linija *SCL* na visokem nivoju. Začetno stanje je pogoj, da se komunikacija začne in da podrejene enote pričnejo poslušati vodilo. Sledi prenos naslova podrejene enote, ki prepozna svoj naslov in odgovori s potrditvijo (*ACK*). Ostale enote se deaktivirajo in čakajo na ponoven *START* bit. Podatki se pošiljajo bit za bitom. Po osmih prenesenih bitih sledi bit potrditve *ACK* (angl. *acknowledge*). Po 7-bitnim naslovu ciljne naprave je 8. zaporedni bit ukaz *R/W*, ki določa smer podatkov. Nadrejena enota konča komunikacijo s končnim stanjem (*STOP* bit).



Slika 5.31: Potek signalov *SCL* in *SDA* [46].

5.4.9 Opis registrov integriranega vezja drsnika *MPR083*

- **Register napak (angl. *Fault Register*)** je bralno pisalni register, ki prikazuje številko napake senzorja dotikov. Vsako pisanje v bralni register izbriše stanje registra, ko je naprava v ustavljenem načinu (angl. *stop mode*). Registra ni mogoče pobrisati, ko je v delovnem načinu (angl. *run mode*). Naslov registra je 0x01 [12].

BIT	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	FAULT	
W								
Ponastavitev	0	0	0	0	0	0	0	0

Tabela 5.11: Register napak.

Polje	Opis
FAULT	Koda napake predstavlja trenutno sprejeto stanje napake. 00: ni napak 01: zaznan kratek stik z *VSS* 10: zaznan kratek stik z *VCC*

Tabela 5.12: Opis polj registra napak.

- **Rotacijski statusni register (angl. *Rotary Status Register*)** je le bralni register, ki se uporablja za branje trenutnega stanja položaja dotika. Naslov registra je 0x02 [12].

BIT	7	6	5	4	3	2	1	0
R	0	0	0	SF	CP			
W								
Ponastavitev	0	0	0	0	0	0	0	0

Tabela 5.13: Rotacijski statusni register.

Polje	Opis
SF	Statusna zastavica prikazuje, kdaj je bil zaznan dotik. 0: ni dotika 1: zaznan dotik
CP	Trenutni položaj dotika, ko je zaznan. 000: elektroda 1 ~ 111: elektroda 8

Tabela 5.14: Opis polj rotacijskega statusnega registra.

- **Z rotacijskim konfiguracijskim registrom (angl. *Rotary Configuration Register*)** nastavimo različne funkcije integriranega vezja *MPR083*. Naslov registra je 0x03 [12].

BIT	7	6	5	4	3	2	1	0
R	RSE	0	0	ACE	RRBE	RTBE	0	RE
W								
Ponastavitev	1	0	0	0	0	0	0	1

Tabela 5.15: Rotacijski konfiguracijski register.

Polje	Opis
RSE	Bit za omogočanje piskača. 0: onemogoči 1: omogoči
ACE	Bit za omogočanje avtomatske funkcije kalibracije. 0: onemogoči 1: omogoči
RRBE	Bit določa, ali so podatki v FIFO pomnilniku, ko je prehod dotika iz dotaknjenega stanja v nedotaknjeno. 0: onemogoči 1: omogoči
RTBE	Bit določa, ali so podatki v FIFO pomnilniku, ko je gumb pritisnjen. 0: onemogoči 1: omogoči
RE	Bit omogoči ali onemogoči senzor dotikov. 0: onemogoči 1: omogoči

Tabela 5.16: Opis polj rotacijskega konfiguracijskega registra.

- **Z registrom meje občutljivost (angl. *Sensitivity Threshold Register*)** lahko prilagodimo občutljivost za potrebo različnih senzorjev dotikov. Register je na naslovu 0x04 [12].

BIT	7	6	5	4	3	2	1	0
R	SL							
W								
Ponastavitev	0	0	0	0	0	0	0	0

Tabela 5.17: Register meje občutljivosti.

Polje	Opis
ST	Nastavimo prag občutljivosti. 00000000: nastavimo občutljivost na raven 1 ~ 00111111: nastavimo občutljivost na raven 64 Če je vrednost izven območja, se nastavi privzeta vrednost 00111111.

Tabela 5.18: Opis polj registra meje občutljivosti.

- **Register nadzornika diktirne ure (angl. *Master Tick Period Register*)** se uporablja za nastavljanje glavne ure sistema. Vsi ostali deli vezja *MPR083* so sinhronizirani s tem števcem. Privzeta vrednost je 8 ms in se uporablja za primarno uro. Register je na naslovu 0x05 [12].

BIT	7	6	5	4	3	2	1	0
R	MTP							
W								
Ponastavitev	0	0	0	0	0	1	0	1

Tabela 5.19: Register nadzornika diktirne ure.

Polje	Opis
MTP	Nastavi ali sporoči trenutno vrednost multiplikatorja primarne ure. Razpon ure mora biti med 5ms in 31ms. 00000000: nastavi na 5 ~ 00011010: nastavi na 31 Če je vrednost izven tega območja, se nastavi 00011010.

Tabela 5.20: Opis polj registra nadzornika diktirne ure.

- **Konfiguracijski register (angl. *Configuration Register*)** omogoča uporabniku, da ponastavi napravo, nastavi prekinitvene nastavitve in spreminja načine delovanja. Naslov registra je 0x0A [12].

BIT	7	6	5	4	3	2	1	0
R	IRQR			$\overline{\text{RST}}$	0	$\overline{\text{DCE}}$	IQEN	RUNE
W								
Ponastavitev	0	0	0	1	0	1	0	0

Tabela 5.21: Konfiguracijski register.

Polje	Opis
IRQR	Stopnja prekinitev; izbrana vrednost se množi z MTP za določitev minimalne zakasnitve med zaporednimi prekinitvami. 000: množitelj 1 ~ 111: množitelj 8
$\overline{\text{RST}}$	Globalna ponastavitev krmilnika senzorja dotikov. 0: je prisotna 1: ni prisotna
$\overline{\text{DCE}}$	Bit omogoči ali onemogoči delovni cikel (angl. *duty cyle*). 0: onemogočen 1: omogočen
IRQEN	Bit omogoči ali onemogoči prekinitve. 0: prekinitve onemogočene 1: prekinitve omogočene
RUNE	Omogoči ali onemogoči zaznavanje dotikov. Aktiven v visokem stanju. 0: pregledovanje elektrod onemogočeno 1: pregledovanje elektrod omogočeno

Tabela 5.22: Opis polj konfiguracijskega registra.

5.4.10 Napake drsnika

Odkrivanje napak nedelovanja drsnika je vzelo kar nekaj časa. Glavne napake so:

- napaka v načrtovanju drsnika (prototipna plošča);
- napačen programski postopek;
- nepopolna nastavitev registrov.

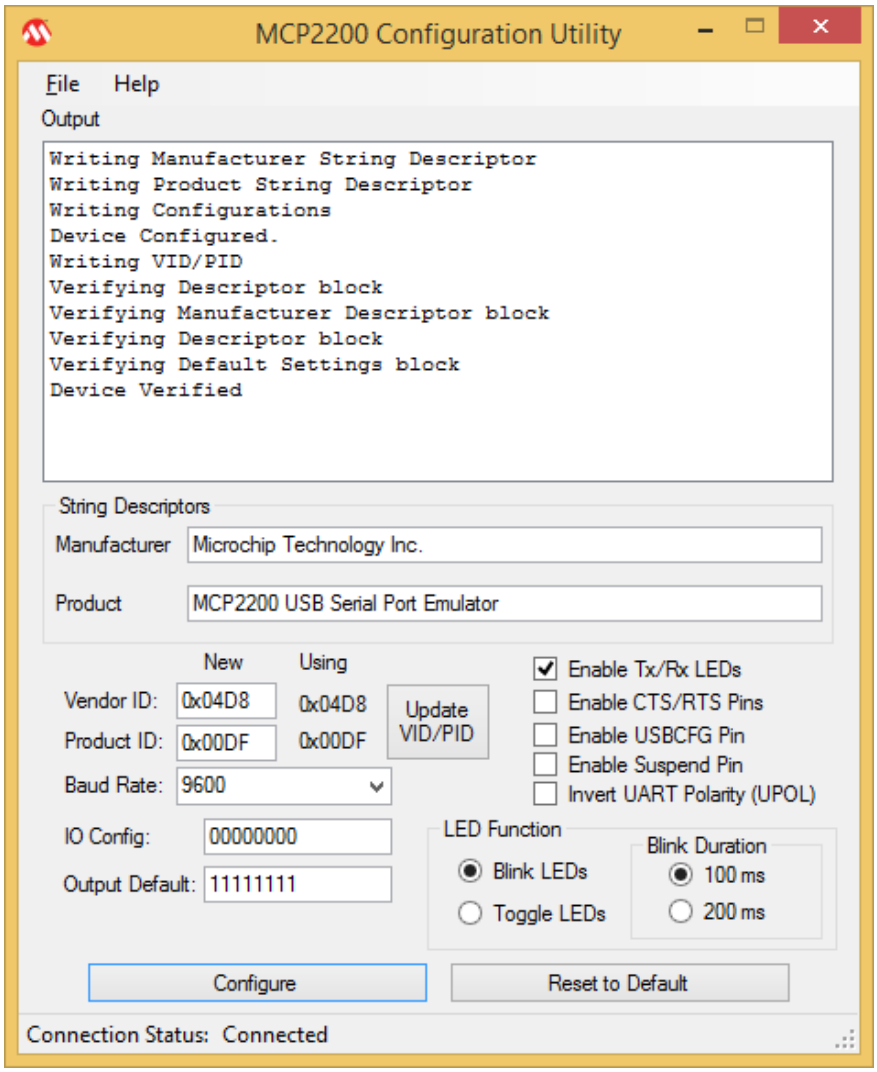
Napaka, ki se je pojavljala pri uporabi drsnika na prototipni ploščici, je onemogočala njegovo delovanje. Komunikacija po vodilu I^2C je delovala (lahko smo spreminjali stanja pisalnih registrov), odziva drsnika pa ni bilo (vrednost statusnega registra je bila 0). Pri branju registra napak smo dobili vrednost, ki predstavlja kodo napake 01 (Poglavje 5.4.9). Pojavila se je že ob prvem dotiku, pomeni pa zaznan kratek stik z nizkim nivojem. Najprej smo se osredotočali le na branje položajev dotika, na napako pa se nismo ozirali z mislijo, da bi moral drsnik kljub zapisani napaki vseeno delovati. Temu ni bilo tako, saj je integrirano vezje že ob prvi napaki onemogočilo pregledovanje elektrod. Odločili smo se, da najprej poskušamo razumeti, zakaj prihaja do tega, kar zaznava čip. Po razmisleku smo prišli do ugotovitve, da se napaka pojavlja zaradi povezave (neposrednega dotika) z elektrodo ali čipom preko nas z zemljo. Da bi preprečili neposredni dotik z elektrodami, smo se morali dotakniti območja drsnika z razdalje (z druge strani ploščice pod elektrodami). Tega nismo mogli narediti zaradi plasti bakra na nasprotni strani elektrod, ki je bil posledica napačnega načrtovanja drsnika. Bakreno plast smo morali odjedkali, elektrode pa so (na prototipni ploščici) ostale zgoraj, kar smo kasneje popravili z optimizacijo načrtovanja.

Po opravljenem delu je drsnik začel kazati znake delovanja, vendar daleč od želenih. Zaznavanje drsenja dotika ni bilo tekoče, ob pojavu iste napake pa drsnik ni več deloval. Pri registru napak gre za bralno/pisalni register, saj lahko z vpisom vrednosti v register pobrišemo zabeleženo napako. Ta postopek je uspešen, ko damo čip v stanje *STOP*, vpišemo vrednost nič v register napak ter spremenimo stanje v *START* (tretji vzrok napake). S tem postopkom pobrišemo napako. Nadaljujemo lahko z branjem položajev dotikov. Poleg odklopa napajanja obstaja še ena rešitev izbrisa napake, ta je ponastavitev (reset) čipa. Ob tem se pobrišejo tudi prednastavljeni registri. S pomočjo Arduina in knjižnice za vodilo I^2C nam je uspelo ugotoviti neodzivnost čipa.

Tretjo napako smo rešili na kasneje izdelanih ploščicah. Vzrok slabega odziva drsnika v nekaterih položajih je bil v nepopolno nastavljenih registrih.

5.5 Nastavitev UART integriranega vezja *MCP2200*

Pred (prvo) uporabo UART vmesnika moramo napravi *Freescale MCP2200* nastaviti posebne ukaze. Napravo nastavimo prek vmesnika HID s pomočjo orodja *Microchip Configuration Utility*. Orodje dobimo na uradni strani proizvajalca. Po namestitvi poženemo program in priključimo razvojno ploščo prek USB. Večina nastavitev je nastavljenih že ob zagonu. Hitrost prenosa (zavihek Baud Rate) nastavimo na vrednost 9600 ter omogočimo uporabo svetlečih diod za prikaz stanja *TX* in *RX* signala (v zavihku Enable Tx/Rx LEDs). Nastavimo tudi funkcije svetlečih diod (LED Function) in kliknemo »*Configure*«. Za uspešno konfiguracijo se v statusnem okvirčku izpiše »*Status Verified*« (Slika 5.32).



Slika 5.32: Orodje za nastavitev integriranega vezja *Freescale MCP2200*.

6 Sklepne ugotovitve

Med večjimi izzivi pri načrtovanju razširitvene plošče od začetka do konca je bil izdelati primeren razdeljevalnik signalov, ki skrbi za povezljivost obeh plošč. Ves čas načrtovanja smo bili na meji glede izida signalov komponent, z danim številom linij na glavnem priključku, enostavnostjo vezja in uporabo. Da bi naredili učinkovito razvrstitev signalov, smo večkrat spremenili sistem razvrščanja in signale prerazporedili za ohranitev posameznih komponent v isti skupini signalov. Poleg števila signalov so dodatno težavo predstavljale komponente z izhodnimi in dvosmernimi signali ter tri le enosmerne digitalne linije na priključku razvojne plošče. Menim, da smo dosegli zastavljen cilj, torej izdelati majhno vezje, ki bo enostavno za uporabo.

Veliko težav nam je povzročil kapacitivni senzor vzdolžnih dotikov, saj smo bili ves čas skeptični glede izbire integriranega vezja, vendar veliko izbire nismo imeli. Prve težave nedelovanja in kasneje netekoče zaznavanje dotikov z vezjem *MPR083* so pripeljale do spogledovanja po drugi izbiri integriranega vezja. Tik pred naročilom za izdelavo tiskanin smo našli zasilno rešitev z izbranim in verjeli, da bo učinkovitejše delovanje drsnika prinesla drugačna razvrstitev elektrod ter optimiziranje nastavitev drsnika (občutljivosti, hitrost zaznavanja itd). S popravki smo naredili učinkovit in dovolj občutljiv drsnik.

Za zmanjšanje vpliva napake pri krmiljenju matrike LED smo poskrbeli s programskim izpuščanjem osveževanj vrstic, kljub temu je svetilnost ostala dovoljšna in uporabnik z napako ni prikrajšan za enostavnost uporabe.

Poleg uporabljenih komponent bi lahko dodali še kakšno dodatno periferijo, vmesnik, napravo ali senzor, npr. za merjenje razdalje, svetilnosti itd., vendar je tu potreben kompromis med velikostjo plošče, ceno, namembnostjo komponent in zahtevami uporabnika. Pri izvedbi in vsebini plošče smo bili delno omejeni tudi s časom. Od začetka dogovora teme, vsebine, načrtovanja, testiranja do oddaje datotek za izdelavo razširitvene plošče smo v dobrem mesecu in pol načrtali ploščo, s katero smo nadomestili vsaj 8 modulov. S kakšnim tednom rezerve bi morda odpravili

kakšno pomanjkljivost in dodali kakšno komponento, ki bi bila zanimiva v sklopu laboratorija LNIV Fakultete za Elektrotehniko. S končnim izdelkom smo zadovoljni, saj smo dosegli zastavljen cilj.

Literatura

[1] M. Franc, Doktorska disertacija, »Robustno bilateralno teleoperiranje s programirljivim vezjem FPGA«, Univerza v Mariboru: Fakulteta za elektrotehniko, računalništvo in informatiko, str. XIII. Dosegljivo: https://dk.um.si/IzpisGradiva.php?id=42669. Dostopno: [september 2015].

[2] »Uporaba logičnih vezij«, str. 30. Dosegljivo: http://rso.tsckr.si/RSM3/01_Uporaba_logicnih_vezij.pdf. Dostopno: [september 2015].

[3] A. Trost, »Uvod v programirljive digitalne sisteme«, Univerza v Ljubljani: Fakulteta za elektrotehniko, str. 63. Dosegljivo: http://lniv.fe.uni-lj.si/courses/pds/PDS_2015.pdf. Dostopno: [september 2015].

[4] Terasic Technologies, »DE0-Nano User manual«, str. 5–10, 20, 21. Dosegljivo: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/de0_nano_user_manual_v1.9.pdf. Dostopno: [september 2015].

[5] Appelsiini, »How Does Led Matrix Work?«. Dosegljivo: http://www.appelsiini.net/2011/how-does-led-matrix-work. Dostopno: [september 2015].

[6] T. Kotnik, »Sekvenčna vezja in strukture«, Univerza v Ljubljani: Fakulteta za elektrotehniko, str. 25, 40, 50. Dosegljivo: http://lbk.fe.uni-lj.si/pdfs/PV-P07.pdf. Dostopno: [september 2015].

[7] NXP Semiconductors, »74HC595; 74HCT595 8-bit serial-in, serial or parallel-out shift register with output latches; 3-state«, str. 1, 2. Dosegljivo: http://www.nxp.com/documents/data_sheet/74HC_HCT595.pdf. Dostopno: [september 2015].

[8] Kingbrith, »TC07-11SURKWA«, str. 1–3. Dosegljivo: http://www.kingbrightusa.com/images/catalog/SPEC/TC07-11SURKWA.pdf. Dostopno: [september 2015].

[9] Freescale Semiconductor, »Proximity Capacitive Sensor Technology for Touch Sensing Applications«, str. 4–8. Dosegljivo: http://cache.freescale.com/files/sensors/doc/white_paper/PROXIMITYWP.pdf. Dostopno: [september 2015].

[10] Freescale Semiconductor, »Touch Sensors«, str. 1. Dosegljivo: http://cache.freescale.com/files/sensors/doc/fact_sheet/PROXFAMFS.pdf. Dostopno: [september 2015].

[11] Atmel corporation, »Sensors Design Guide«, str. 1–3, 9, 10. Dosegljivo: http://www.atmel.com/images/doc10752.pdf. Dostopno: [september 2015].

[12] Freescale Semiconductor, »Proximity Capacitive Touch Sensor Controller - Data Sheet«, str. 1–5. Dosegljivo: http://cache.freescale.com/files/sensors/doc/data_sheet/MPR083.pdf. Dostopno: [september 2015].

[13] Freescale Semiconductor, »Pad Layout Application Notes«, str. 1–3. Dosegljivo: http://cache.freescale.com/files/sensors/doc/app_note/AN3747.pdf. Dostopno: [september 2015].

[14] Freescale Semiconductor, »Medical Applications Guide – Brochure«, str. 22. Dosegljivo: http://cache.freescale.com/files/microcontrollers/doc/user_guide/MDAPPUSGDRM118.pdf. Dostopno: [september 2015].

[15] B. Kovač, »UART na RS232 in USB«, Svet elektronike. Dosegljivo: http://svet-el.si/o-reviji/samogradnje/1150-uart-na-rs232-in-usb. Dostopno: [september 2015].

[16] Microchip Technology, »MCP2200 - USB 2.0 to UART Protocol Converter with GPIO Data Sheet«, str. 1–9. Dosegljivo: http://ww1.microchip.com/downloads/en/DeviceDoc/22228B.pdf. Dostopno: [september 2015].

[17] ON Semiconductor, »LV8548MC Data Sheet«, str. 1–6. Dosegljivo: http://www.onsemi.com/pub_link/Collateral/LV8548MC-D.PDF. Dostopno: [september 2015].

[18] Computer Hope, »VGA«. Dosegljivo: http://www.computerhope.com/jargon/v/vga.htm. Dostopno: [september 2015].

[19] Wikipedia, »Super video graphics array«. Dosegljivo: https://en.wikipedia.org/wiki/Super_video_graphics_array. Dostopno: [september 2015].

[20] J. Hansen, Textronix, »VGA Interface Testing with the VM5000«, str. 1–3. Dosegljivo: http://www.tek.com/dl/25W_19395_0.pdf. Dostopno: [september 2015].

[21] Javier Valcarce's Homepage, »VGA Video Signal Format and Timing Specifications«. Dosegljivo: http://www.javiervalcarce.eu/html/vga-signal-format-timming-specs-en.html. Dostopno: [september 2015].

[22] Wikipedia, »Rotary encoder«. Dosegljivo: https://en.wikipedia.org/wiki/Rotary_encoder. Dostopno: [september 2015].

[23] M. Jankovec, »KONSTRUIRANJE ELEKTRONSKIH NAPRAV – Napajanje«, Univerza v Ljubljani: Fakulteta za elektrotehniko, str. 1–12. Dosegljivo: http://lpvo.fe.uni-lj.si/fileadmin/files/Izobrazevanje/KEN/Gradiva/11/11_KEN_Napajanje.pdf. Dostopno: [september 2015].

[24] E-rid, »Eagle navodila – Uvod«. Dosegljivo: http://erid.tsckr.si/12/eagle_navodila/uvod.html. Dostopno: [september 2015].

[25] M. Jankovec, »KONSTRUIRANJE ELEKTRONSKIH NAPRAV – EMC načrtovanje tiskanih vezij«, Univerza v Ljubljani: Fakulteta za elektrotehniko, str. 3–5. Dosegljivo: http://lpvo.fe.uni-lj.si/fileadmin/files/Izobrazevanje/KEN/Gradiva/07/07_KEN_EMC_dizajn_PCB.pdf. Dostopno: [september 2015].

[26] K. Steblovnik, »Osnove načrtovanje tiskanih vezij«, str. 7–22. Dosegljivo: http://studentski.net/get/vis_scv_ele_rvp_sno_osnove_nacrtovanja_tiskanih_vezij_01.pdf. Dostopno: [september 2015].

[27] »Tolerance: Posledica netočne izdelave je sipanje elementov«, str. 24–27. Dosegljivo: http://www.student-info.net/sis-mapa/skupina_doc/fe/knjiznica_datoteke/1214065997_realizacija_gradivo.doc. Dostopno: [september 2015].

[28] J. Mele, DIPLOMSKO DELO, »Elektronsko načrtovanje in izdelava tiskanih vezij«, Univerza v Ljubljani: Pedagoška fakulteta, Fakulteta za matematiko in fiziko, str. 68, 70. Dosegljivo: http://www.pef.uni-lj.si/slavkok/diplome/mele_j/Elektronsko_nacrtovanje_TIV.pdf. Dostopno: [september 2015].

[29] Svet-el, »Osnove spajkanja«, Svet elektronike. Dosegljivo: http://www.svet-el.si/literatura/377-osnove-spajkanja. Dostopno: [september 2015].

[30] Svet-el, »Spajkanje TQFP komponent«, Svet elektronike. Dosegljivo: http://www.svet-el.si/bazaznanja/vse-o-spajkanju/461-spajkanje-tqfp-komponent. Dostopno: [september 2015].

[31] S. Larson, »I2C Master (VHDL)«, eewiki. Dosegljivo: https://www.eewiki.net/display/LOGIC/I2C+Master+(VHDL). Dostopno: [september 2015].

[32] Best-Microcontoller-Project, »I2C tutorial«. Dosegljivo: http://www.best-microcontroller-projects.com/i2c-tutorial.html. Dostopno: [september 2015].

Viri slik

[33] https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/de0_nano_user_manual_v1.9.pdf. Dostopno: [september 2015].

[34] http://www.kingbrightusa.com/images/catalog/SPEC/TC07-11SURKWA.pdf. Dostopno: [september 2015].

[35] http://www.atmel.com/images/doc10752.pdf. Dostopno: [september 2015].

[36] http://cache.freescale.com/files/sensors/doc/data_sheet/MPR083.pdf. Dostopno: [september 2015].

[37] http://ww1.microchip.com/downloads/en/DeviceDoc/22228B.pdf. Dostopno: [september 2015].

[38] http://www.onsemi.com/pub_link/Collateral/LV8548MC-D.PDF. Dostopno: [september 2015].

[39] http://www.alps.com/prod/info/E/image/Switch/Encoder/EC11/EC11_D_81.GIF. Dostopno: [september 2015].

[40] http://www.myvolt.de/Media/Shop/p50023g-weller-ws-81-ws81.jpg. Dostopno: [september 2015].

[41] http://tomsidji.de/portfolio/wp-content/uploads/2013/04/weller.jpg. Dostopno: [september 2015].

[42] https://learn.adafruit.com/system/assets/assets/000/001/978/medium800/tools_Header_Joints.jpg?1396777967. Dostopno: [september 2015].

[43] http://www.psinter.com/images/practical/P_114.jpg. Dostopno: [september 2015].

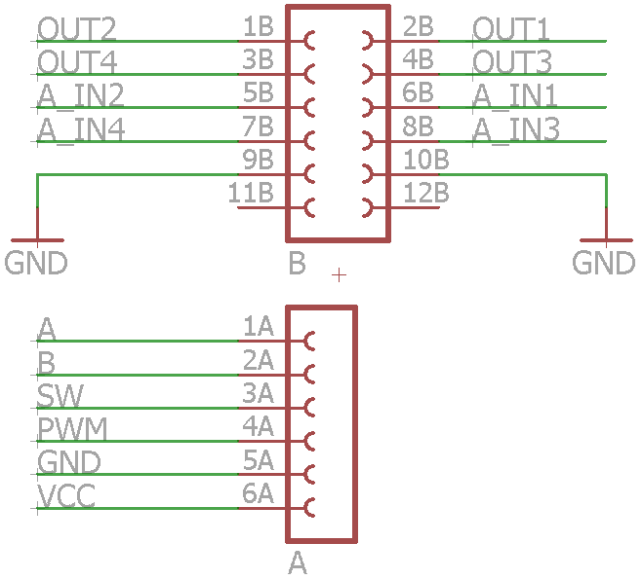
[44] http://www.psinter.com/images/practical/P_115.jpg. Dostopno: [september 2015].

[45] http://www.durofy.com/wp-content/uploads/2012/10/1000px-Ascii_Table-nocolor.svg_-940x654.png. Dostopno: [september 2015].

[46] http://www.best-microcontroller-projects.com/i2c-tutorial.html. Dostopno: [september 2015].

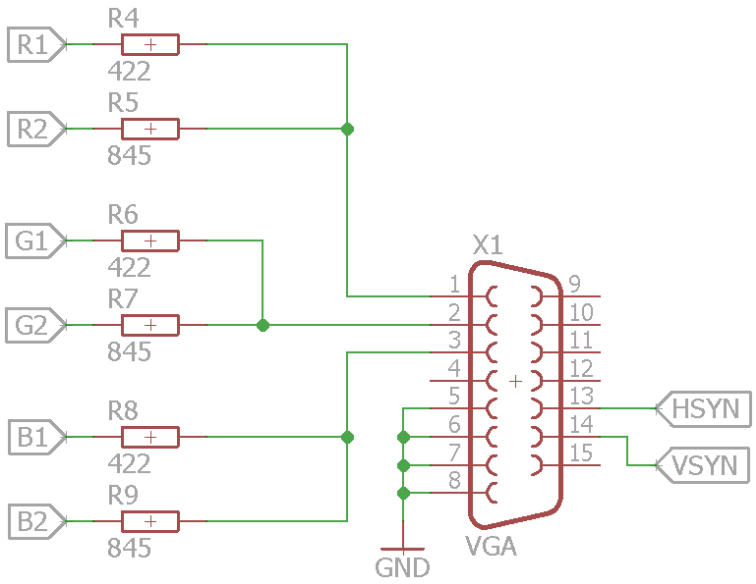
Dodatek

Slika 6.1 prikazuje električni načrt dveh priključkov pod pravim kotom A in B s šestimi in dvanajstimi nožicami za priklop plošče z rotacijskim kodirnikom.



Slika 6.1: Električna shema priključka A in B.

Slika 6.2 prikazuje električni načrt vmesnika VGA.



Slika 6.2: Električna shema vmesnika VGA.

Električne načrte ostalih komponent razširitvene plošče prikazujejo slike: 3.4 – matriko LED, 3.12 – drsnik, 3.16 – pretvornik z USB v UART, 3.20 – gonilnik motorja, 3.36 – štiri tipke, 3.41 – CPLD in vmesnik JTAG, 3.45 – signalov in napajanja na glavnem priključku razširitvene plošče.

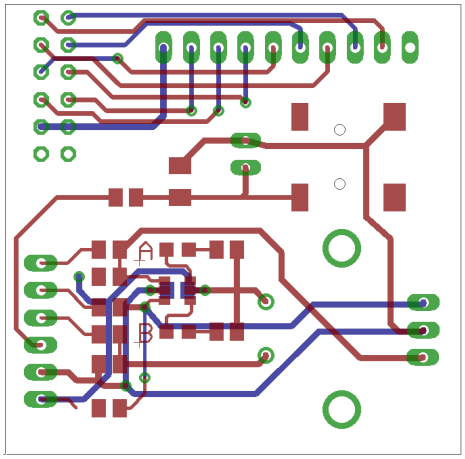
Tabela 6.1 prikazuje spisek elementov za razširitveno ploščo.

IME	KOLIČINA	VREDNOST	PODNOŽJE	OPIS
R1, R2	2	270Ω	2012 (metrično)	
R3	1	10kΩ	2012 (metrično)	
R4, R6, R8	3	422Ω	2012 (metrično)	
R5, R7, R9	3	845Ω	2012 (metrično)	
R10, R11, R12, R13, R14, R15, R16, R17	8	2,2kΩ	2012 (metrično)	
R18, R19, R20, R21, R22, R23, R24	7	27Ω	2012 (metrično)	
R25, R26, R27	3	4,7kΩ	2012 (metrično)	
R28, R29, R30, R31, R32, R33, R34, R35	8	787kΩ	2012 (metrično)	

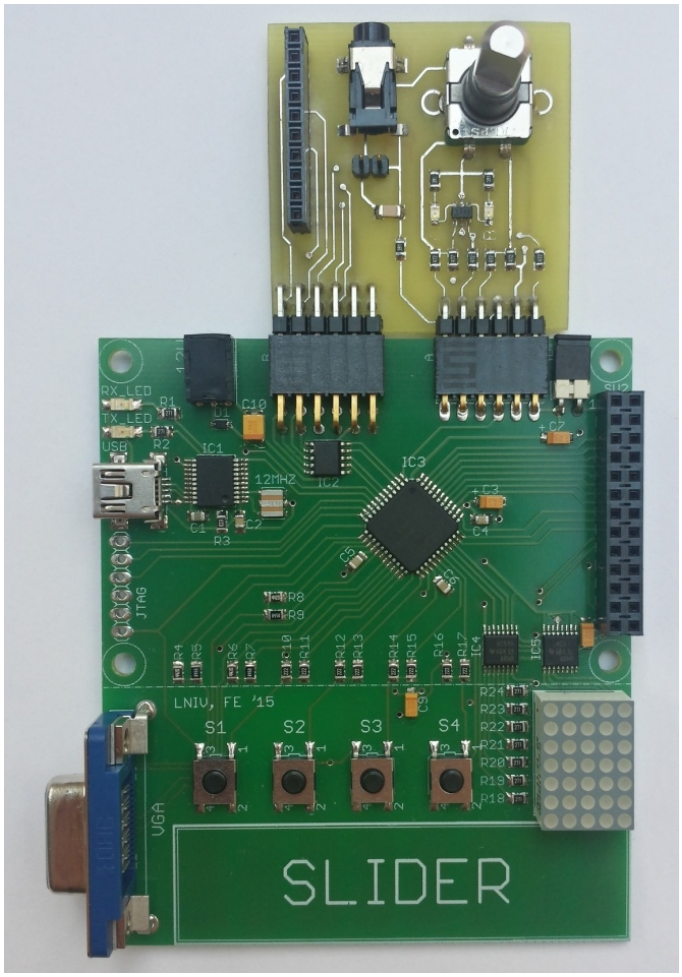
C1	1	470nF	2012 (metrično)	keramični
C2, C4, C5, C6	4	100nF	2012 (metrično)	keramični
C3	1	4,7µF	3216 (metrično)	tantalski
C7, C8, C9	3	1µF	3216 (metrično)	tantalski
C10	1	10µF	3216 (metrično)	tantalski
D1	1	BAT760	SOD-323	Schottky
IC1	1	MCP2200	SSOP-20	
IC2	1	LV8548MC	SOIC-10NB	
IC3	1	XC9572XL-10VQG44C	VQ44	
IC4, IC5	2	74HC595PW	TSSOP-16	
IC6	1	MPR083EJ	TSSOP-16	
12MHz	1	C12M000000S004	ZTTCV	keramični resonator
S1, S2, S3, S4	4	MC32882	MC32882	tipka
TX_LED, RX_LED	2	KP-3216CGCK	3216 (metrično)	zelena svetleča dioda
LED MATRIX 5 X 7	1	TC07-11SURKWA	TC07-11	matrika LED
USB	1	USB-mini tip B	10033526-N3212MLF	mini USB priključek
X1	1	D SUB	HD15SLIM	VGA priključek
12V	1	DC POWER JACK	SPC21373	priključek 20V, 1A
JP1	1	HTSW serija	HTSW-102-08-F-S-RA	priključek pod pravim kotom
SV2	1	BCS serija	BCS-113-L-D-TE	vertikalni priključek
A	1	SSW serija	SSW-106-02-G-D-RA	priključek pod pravim kotom
B	1	SSW serija	SSW-106-02-G-S-RA	priključek pod pravim kotom

Tabela 6.1: Spisek elementov razširitvene plošče.

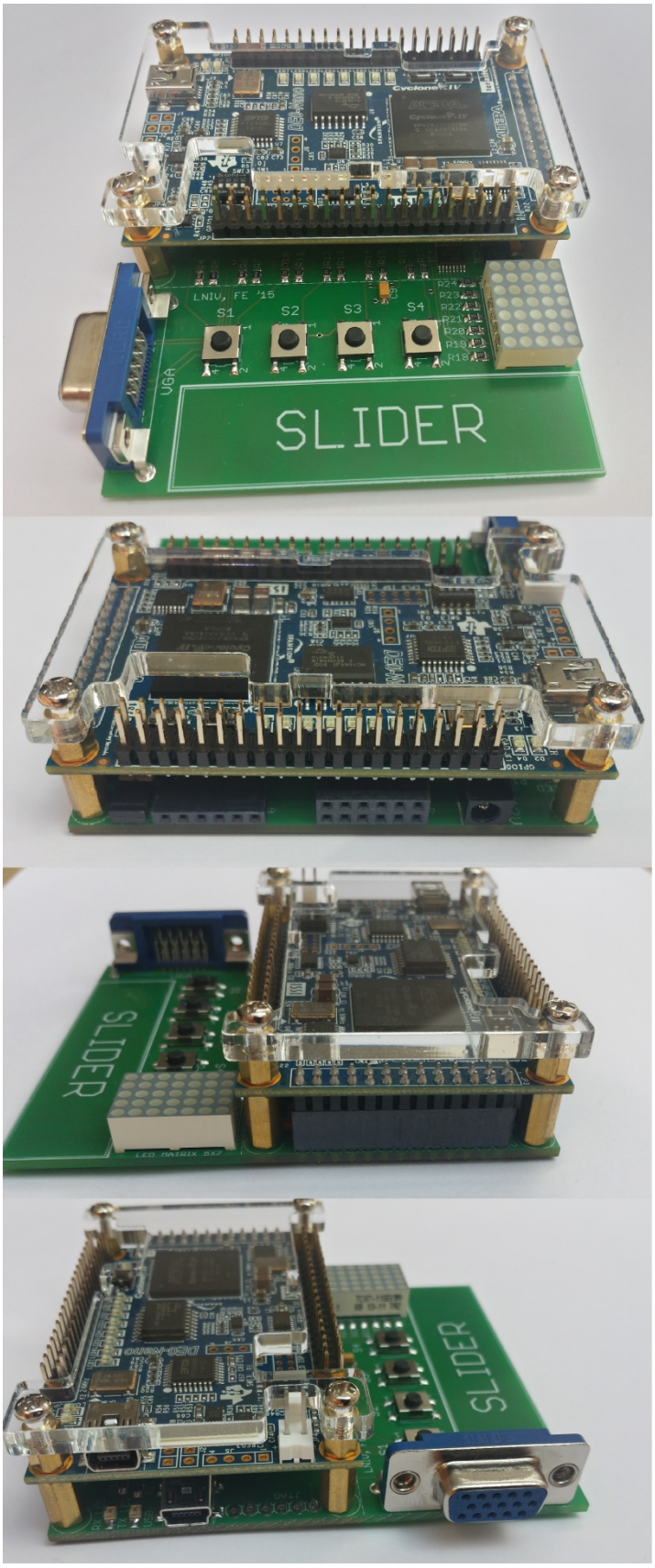
Električni načrt plošče inkrementalnega rotacijskega kodirnika prikazujeta sliki 3.30 in 3.33. Slika 6.3 prikazuje zgornjo in spodnjo plast tiskanine rotacijskega kodirnika.



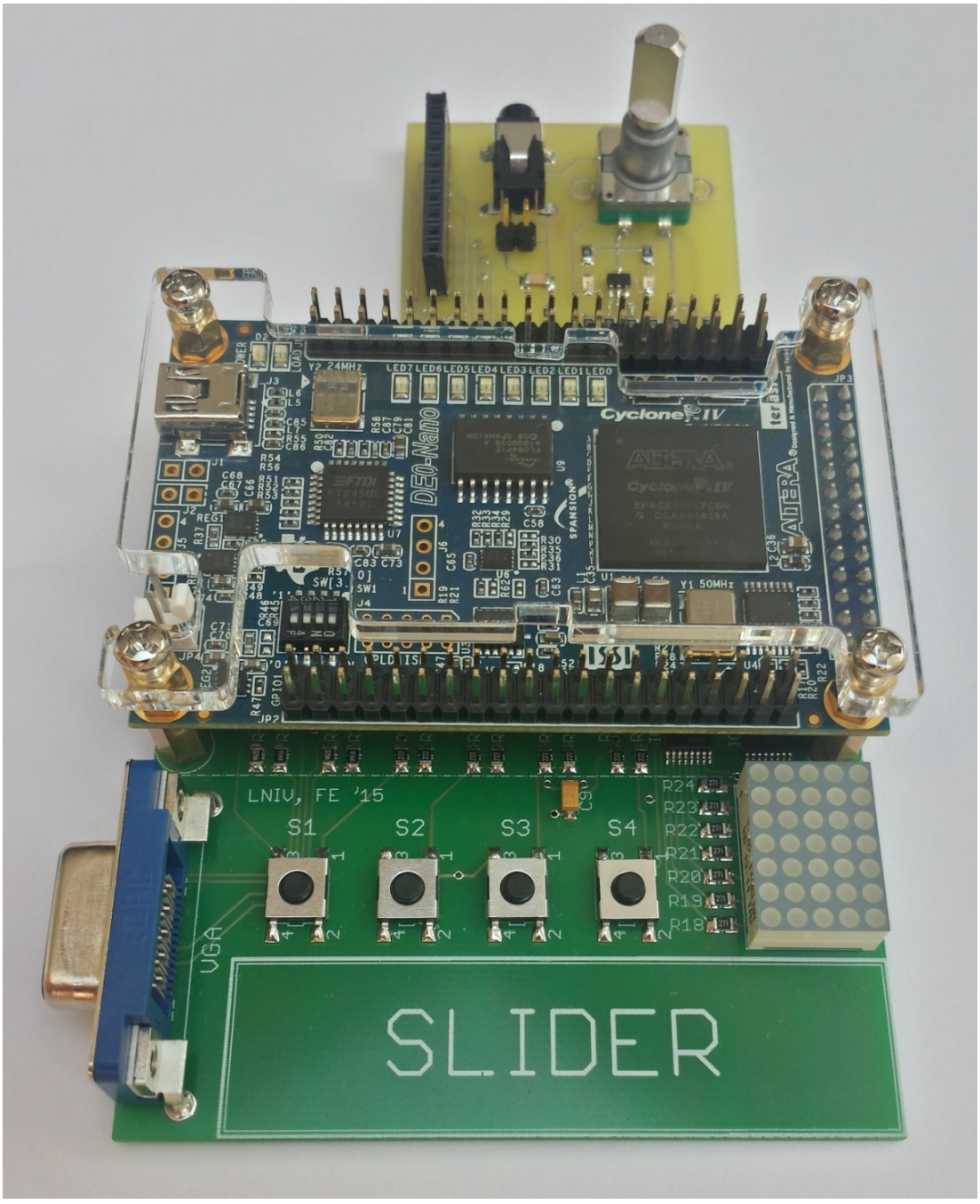
Slika 6.3: Zgornja in spodnja plast tiskanine plošče rotacijskega kodirnika.



Slika 6.4: Razširitvena plošča, povezana s ploščo rotacijskega kodirnika.



Slika 6.5: Razširitvena plošča, povezana z razvojno ploščo *DE0-Nano*.



Slika 6.6: Razširitvena plošča skupaj z razvojno ter ploščo rotacijskega kodirnika.