Mestna občina Celje
Komisija Mladi za Celje

IZDELAVA BIAS-LIGHTING SVETILA

Raziskovalna naloga

Področje: eletrotehnika, elektronika

Mentor:
Davor Zupanc, inž.

Avtor:
Benjamin Lipnik

Celje, januar 2021

Kazalo

Zahvala....4
Povzetek....5
1 Uvod....6
1.1 Predstavitev raziskovalnega problema....6
1.2 Hipoteze....6
1.3 Raziskovalne metode....6
2 Opis naprave....7
2.1 Sestavni deli naprave....8
2.2 Vhodni priključki naprave in njim pripadajoči signali....9
2.2.1 USB....9
2.2.2 Analogni vmesniki za prenos audio/video podatkov....10
2.2.3 Digitalni vmesniki za prenos audio/video podatkov....11
2.2.4 HDMI vmesnik....12
2.2.4.1 TMDS....13
2.2.4.2 EDID....14
2.2.4.3 HDCP....15
2.3 Dekodirnik signala....16
2.3.1 FPGA....17
2.4 Enota za upravljanje barv....18
2.5 Izhodna enota....18
3 Postopek izdelave naprave....19
3.1 Izbira komponent....19
3.1.1 Opis izbrane komponente IT6604....21
3.1.2 Opis izbrane komponente FPGA XC6SLX16....22
3.2 Izdelava tiskanega vezja (PCB)....23
3.3 Spajkanje vezja....24
3.4 Nastavljanje in načrtovanje vezja v FPGA s pomočjo jezika Verilog....25
3.4.1 Implementacija gonilnika za LED trak....26
3.4.2 Uporaba gonilnika z blokovnim RAM pomnilnikom....27
3.4.3 Implementacija modula za računanje x in y koordinate....29
3.4.4 Izbiranje slikovnih pik in barvna korekcija....30
3.5 Testiranje in odpravljanje napak....31
4 Razprava....32
5 Zaključek....33
6 Viri....34
6.1 Viri slik....36

Kazalo slik

Slika 1: Televizijski zaslon z nameščenim bias lighting svetilom.....7
Slika 2: Shema delovanja bias light svetila.....8
Slika 3: USB priključek.....9
Slika 4: Priključka analognega VGA vmesnika.....10
Slika 5: Priključka digitalnih vmesnikov Display Port in DVI.....11
Slika 6: Električne povezave znotraj HDMI priključka.....12
Slika 7: Priključek digitalnega vmesnika HDMI.....12
Slika 8: Časovni potek prenosa slikovnih podatkov po TMDS protokolu.....13
Slika 9: Struktura EDID sporočila.....14
Slika 10: Primer FPGA integriranega vezja.....17
Slika 11: Primeren tip LED traku za bias lighting svetilo.....18
Slika 12: Podnožje in povezave integriranega vezja IT6604.....21
Slika 13: Kupljena razvojna ploščica z FPGA integriranim vezjem XC6SLX16.....22
Slika 14: Izris tiskanega vezja v programu KiCad.....23
Slika 15: Izdelano tiskano vezje.....23
Slika 16: Delno sespajkano tiskano vezje.....24
Slika 17: Definicije logičnih stanj protokola LED traku.....26
Slika 18: Montaža LED traku na hrbtno stran televizijskega zaslona.....27
Slika 19: Implementacija dveh blokovnih RAM pomnilnikov, ki so povezani skupaj z gonilnikom LED traku.....28
Slika 20: Implementacija modula za računanje x in y koordinate.....29
Slika 21: Implementacija vrhnjega modula, ki izbira potrebne slikovne pike in jih shrani v pomnilnike drugih podmodulov.....30
Slika 22: Implementacija dvournega blokovnega RAM pomnilnika.....31
Slika 23: Delovanje mojega izdelka.....33

Zahvala

Zahvaljujem se profesorju in mentorju Davorju Zupancu, ki me je vspodbujal pri raziskovanju in mi omogočil dobre pogoje za raziskovanje v šoli, profesorju Mateju Kališku, ki mi je dal idejo za raziskovanje, lektorici Alji Polenek in seveda družini, ki me je vspodbujala med raziskovanjem.

Povzetek

Moje močno področje je študij literature in raziskovanje področja elektronike, zato sem si za svojo nalogo zadal izziv: doma izdelati uporabno elektronsko napravo - bias lighting svetilo. Raziskal sem, kako naprava deluje. Cilj moje naloge je bil, da naredim delujoč izdelek, ki ga lahko uporabim doma. Da bi dosegel cilj, je bilo potrebno branje literature, laboratorijsko raziskovanje, eksperimentiranje ter analiziranje pridobljenih podatkov. Ugotovil sem, da je bias lighting svetilo mogoče narediti doma in ga koristno uporabiti za prijetnejšo izkušnjo gledanja v različne zaslone. Domača izdelava je primerljiva s kupljenim izdelkom po delovanju, omogoča pa še možne nadaljnje nadgradnje. Z vidika cene je doma izdelana naprava, če upoštevamo le materialne stroške, občutno cenejša od kupljene. Uspešno raziskovanje mi je omogočilo srednješolsko znanje smeri elektrotehnika, a potrebno je bilo tudi samostojno delo in poglabljanje v številne, na spletu dostopne podatke. Ta način raziskovanja, s katerim naredimo uporaben izdelek, zelo spodbuja samostojno učenje in nadaljnje raziskovanje.

Ključne besede: Bias lighting, Ambilight, domača izdelava, FPGA, HDMI

1 Uvod

1.1 Predstavitev raziskovalnega problema

Tema raziskovalne naloge je domača izdelava bias-lighting oz. Ambilight svetila. To je naprava, ki okoli televizijskega ali računalniškega zaslona ustvari svetlobni sij v barvah, ki so na zaslonu.

Sij zaslon razširi. Barve zaslona se prelivajo tudi okoli njega in tako omilijo oster kontrast med svetlim zaslonom in temnim ozadjem. Bias lighting svetilo doprinese k prijetnejši izkušnji gledalca ter manjšemu naprezanju oči.

Na trgu takšna svetila obstajajo. Njihove cene se razlikujejo glede na njihovo delovanje, to pa je odvisno predvsem od načina priključitve svetila.

Za raziskovanje sem se odločil, ker me je zanimalo delovanje naprav, ki za svoj vhod uporabljajo video izhod drugih naprav. Menil sem, da bi takšno svetilo lahko izdelal sam, ob poglobljenem raziskovanju in eksperimentiranju. Svetilo, ki bi ga izdelal skozi raziskovanje, bi lahko, če bi bil uspešen, uporabil doma.

1.2 Hipoteze

Ob začetku raziskovanja sem si postavil različne hipoteze:

- Bias lighting svetilo je mogoče narediti doma.
- Domača izdelava svetila je cenejša od kupljenega.
- Svetilo, narejeno doma, je mogoče priključiti na več vrst naprav, kot so: televizijski sprejemniki, igralne konzole, osebni računalniki ...
- Izkušnja uporabe zaslona je s svetilom bias-lighting prijetnejša.

1.3 Raziskovalne metode

- Branje in študij literature
- Laboratorijsko raziskovanje
- Testiranje in analiza dobljenih podatkov

2 Opis naprave

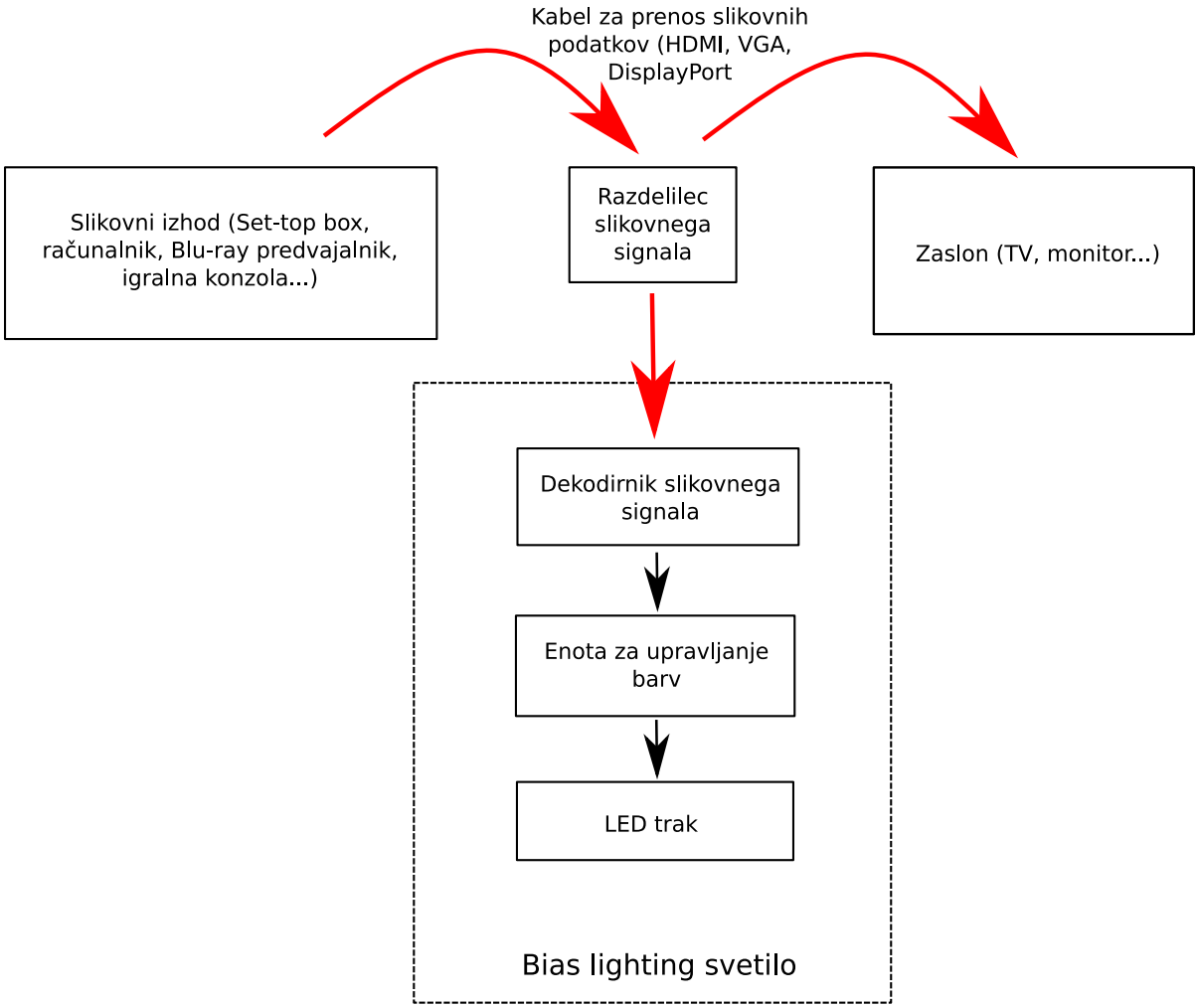
Svetilo deluje tako, da preko svoje vhodne enote pridobi podatke, ki so na sliki zaslona. Te podatke preračuna in pretvori, da so ustrezni za upravljanje izhodne enote naprave. Izhodna enota pa za zaslonom ustvari sij v barvah, ki so na zaslonu. Svetilo je prvo razvilo podjetje Philips. To podjetje prodaja linijo televizorjev, ki imajo takšno svetilo vgrajeno v televizor sam. Philips to svoje svetilo imenuje Ambilight.

Obstaja več različic delovanja naprave. Med seboj se naprave razlikujejo predvsem glede na to, kako pridobijo slikovne podatke iz vira. Naprava sicer deluje tako, da signal, ki ga sprejme od slikovnega vira, dekodira, izbere najbolj zunanje slikovne pike slike, in jih, primerno obdelane, pošlje izhodni enoti.

Slika 1: Televizijski zaslon z nameščenim bias lighting svetilom

2.1 Sestavni deli naprave

Napravo - svetilo - običajno priključimo med slikovnim virom in zaslonom, saj za delovanje potrebuje slikovne podatke, ki jih dobi zaslon. Da lahko podatke uporabita zaslon in svetilo hkrati, je potrebna uporaba razdelilca slikovnega signala. Razdelilec signal razdeli iz enega vhoda na dva enaka izhoda. V vhod se priključi naprava, ki je vir slike, izhod razdelilca pa priključimo na zaslon ter na vhodno enoto svetila – na vhodne priključke. Svetilo lahko ima razdelilec integriran, če pa ga nima, je potrebno uporabiti zunanji razdelilec. Signal tako ne potuje le v zaslon, ampak hkrati tudi v svetilo. Medtem, ko zaslon signal uporabi za prikaz slike, ga svetilo uporabi za preračunavanje in pretvorbo podatkov v takšne, da jih lahko uporabi za ustvarjanje svetlobnega sija, ki ga izžareva izhodna enota naprave. Da svetilo iz signala izlušči podatke, mora le-tega najprej dekodirati. Za dekodiranje signala je odgovorna dekodirna enota v napravi.



Slika 2: Shema delovanja bias light svetila

2.2 Vhodni priključki naprave in njim pripadajoči signali

2.2.1 USB

USB je digitalni protokol, ki je namenjen splošnemu prenosu podatkov med dvema napravama (gostiteljem in gostom). Da lahko uporabimo protokol USB kot vhod svetila, se mora na gostiteljski napravi izvajati program, ki zajame sliko z zaslona in preko protokola USB podatke pošlje elektroniki svetila. Tak način je omejen na gostitelje, na katere lahko naložimo svoj program in ga izvajamo. Tovrstni gostitelji so osebni računalniki, Raspberry pi ... Če te možnosti nimamo, moramo za svetilo izbrati drug način pridobivanja slikovnih podatkov. Pridobiti jih moramo direktno, iz povezave med zaslonom in napravo. Takšna povezava običajno poteka preko digitalnega ali analognega vmesnika (najpogostejši vmesniki so: HDMI, VGA, DisplayPort).

Delovanje svetila, ki za vhod slike uporablja USB, je najenostavnejše. Enostavno je, ker večino dela opravi program na gostiteljskem sistemu, zato je količina podatkov, ki jo moramo potem poslati elektroniki svetila, ne glede na resolucijo, minimalna. Poslati moramo le podatke, ki direktno vplivajo na izhodno enoto svetila. Za komunikacijo preko USB in upravljanje izhodne enote je potreben le enostaven mikrokrmilnik. Svetila, ki za vhod uporabljajo USB, so popularna zaradi dostopnosti. Seveda imajo omejeno možnost uporabe le na osebnih računalnikih.

Slika 3: USB priključek

2.2.2 Analogni vmesniki za prenos audio/video podatkov

Analogne protokole in signale uporabljajo za prenos slikovnih podatkov predvsem starejše naprave. Ti protokoli prenašajo slikovne podatke tako, da skozi čas - glede na barvo - spreminjajo napetost električnih signalov. Na tak način so podatki poslani sočasno za eno barvo naenkrat. Tako je poslanih več podatkov pri relativno nizkih frekvencah. Ampak prenos podatkov preko analognih signalov ima nekatere slabosti, kot je občutljivost na zunanji šum. Ob veliki količini zunanjega šuma, se kakovost slike, prenesene preko analognega protokola, lahko močno zmanjša. To je razlog, da se analogni protokoli uporabljajo vedno manj in so v sedanjih časih že precej zastarani. Frekvence analognih signalov, pri resoluciji 1920×1080 slikovnih pik in frekvenci osveževanja zaslona 60Hz, se gibljejo v okolici 125 Mhz.

Svetilo, ki bi ga priključili preko takega vmesnika, mora biti veliko naprednejše, v primerjavi s tistimi, ki jih priključimo preko USB. Če tam svetilo dobi podatke posredno iz gostitelja, na katerem se izvaja program, mora v tem primeru elektronika svetila podatke pridobiti in obdelati sama. Za dekodiranje signalov pri takšnih frekvencah, ne moremo več uporabljati mikrokrmilnikov, ampak potrebujemo diskretna vezja, ki običajno uporabljajo namenske čipe (integrirana vezja). Pri dekodiranju analognega signala direktno s povezave med napravo in zaslonom, se ukvarjamo z veliko količino podatkov, od katerih jih le del potrebujemo za ustvarjanje svetlobnega sija. Najpogosteje se v domačih napravah pojavljajo analogni vmesniki, kot so: VGA, Composite, SCART in S-VIDEO.

Slika 4: Priključka analognega VGA vmesnika

2.2.3 Digitalni vmesniki za prenos audio/video podatkov

Novejše naprave za slikovni izhod uporabljajo vmesnike, ki uporabljajo protokole, ti pa podatke pošljejo preko digitalnih signalov. Signali slikovne podatke prenašajo serijsko, bit za bitom. Ker so podatki preneseni digitalno, zunanji šum ne vpliva na kakovost slike. Običajno se pri takšnem prenosu slike uporabljajo po trije signali hkrati. Vsak signal je odgovoren za prenos ene izmed osnovnih barv na zaslonu. Ker so podatki preneseni serijsko, mora biti frekvenca prenosa višja kot pri ostalih načinih. Pri resoluciji 1920×1080 slikovnih pik in osvežitvi 60 Hz, dosežejo frekvenco 2,25 Ghz.

Tudi svetilo, ki bi ga priključili preko tovrstnega vmesnika, mora biti veliko naprednejše, v primerjavi s tistim, ki ga priključimo preko USB. V tem primeru mora elektronika svetila podatke prav tako pridobiti in obdelati sama. Za dekodiranje signalov s tako visokimi frekvencami, je uporaba integriranih vezij neizogibna. Pri dekodiranju digitalnega signala direktno iz povezave med napravo in zaslonom, se prav tako ukvarjamo z veliko količino podatkov, od katerih jih za ustvarjanje svetlobnega sija potrebujemo le del.

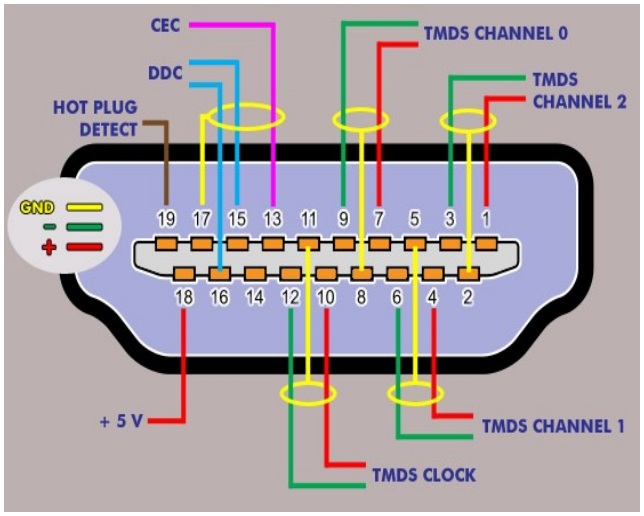
Najpogosteje v domačih napravah najdemo naslednje digitalne vmesnike: HDMI, DisplayPort, DVI. Ker sem v projektu uporabil vmesnik HDMI, ga spodaj podrobneje opisujem.

Slika 5: Priključka digitalnih vmesnikov Display Port in DVI

2.2.4 HDMI vmesnik

HDMI je kratica za High Definiton Multimedia Interface. To je vmesnik, ki določa digitalni prenos video in audio podatkov ter njihove električne priključke in povezave. HDMI implementira EIA/CEA-861 standarde, ki definirajo video formate, valovno obliko, transport audio podatkov, pomožnih podatkov in implementacijo EDID. Prenos slikovnih podatkov se pri HDMI protokolu prenese s pomočjo TMDS protokola. HDMI je vmesnik, ki podpira vroči priklop, to pomeni, da lahko priključimo naprave, ki delujejo s pomočjo HDMI vmesnika, med njihovim delovanjem. Obstaja več verzij HDMI vmesnika, razlikujejo se predvsem po resoluciji, ki jo vmesnik podpira. Vmesnik je narejen tako, da se preko njega lahko prenašajo podatki, zakodirani s HDCP. To otežuje piratstvo vsebine, ki bi se lahko prenašala.

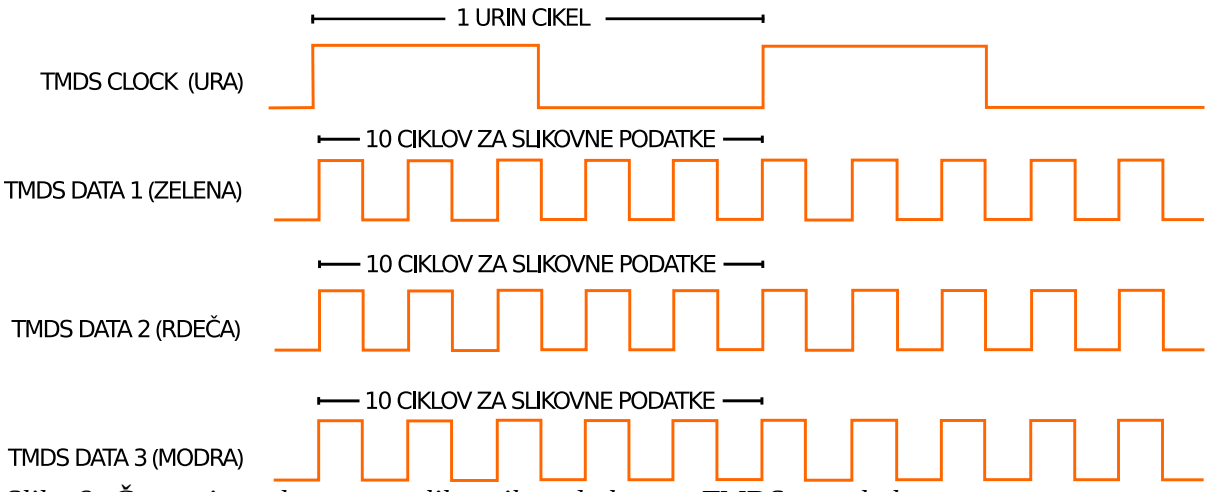
Slika 7: Priključek digitalnega vmesnika HDMI



Slika 6: Električne povezave znotraj HDMI priključka

2.2.4.1 TMDS

TMDS je kratica protokola (Transition-minimized differential signaling). Protokol je namenjen prenosu serijskih podatkov z zelo visoko hitrostjo. TMDS oddajnik vključuje napreden algoritem, podatke iz 8 bitov pretvori v 10 bitov tako, da se med prenosom podatka zmanjša elektromagnetna interferenca na žicah. Hkrati omogoča robustno obnovitev urinega takta pri sprejemniku in s tem v kombinaciji z diferencialnim signalom doseže visoko toleranco do popačenega signala, ki se pojavlja pri daljših kablih ali pri krajših kablih nižje kakovosti. Pri TMDS protokolu se uporabljajo štiri parice žic. Tri za barvo in ena za urin takt. Za vsak urin cikel se na vsaki izmed paric prenese vseh 10 bitov eden za drugim.



Slika 8: Časovni potek prenosa slikovnih podatkov po TMDS protokolu

2.2.4.2 EDID

EDID je kratica za Extended Display Identification Data. To je standardni format za naprave z zaslonom in je namenjen rokovanju med zaslonom in virom slike (gostiteljem). S pomočjo rokovanja si napravi izmenjata informaciji o resoluciji in frekvenci, ki jo podpirata. Izmenjata si tudi serijski številki, model, proizvajalca, zmožnost zaslona, da predvaja zvok ... EDID rokovanje med napravama pri HDMI vmesniku poteka preko I2C protokola. Ko naprava, ki je vir slike, zazna priključitev druge naprave (običajno zaslona), najprej preko I2C pošlje zahtevo na naslov 0x50. Druga naprava se mora na to odzvati z standardnim EDID sporočilom, ki vsebuje podatke o zaslonu in ima tipično dolžino 128 Bajtov.

Address (Decimal)	Data	General Description
0-7	Header	Constant fixed pattern
8-9	Manufacturer ID	Display product identification
10-11	Product ID Code	
12-15	Serial Number	
16-17	Manufacture Date	
18	EDID Version #	EDID version information
19	EDID Revision #	
20	Video Input Type	Basic display parameters. Video input type (analog or digital), display size, power management, sync, color space, and timing capabilities and preferences are reported here.
21	Horizontal Size (cm)	
22	Vertical Size (cm)	
23	Display Gamma	
24	Supported Features	
25-34	Color Characteristics	Color space definition
35-36	Established Supported Timings	Timing information for all resolutions supported by the display are reported here
37	Manufacturer's Reserved Timing	
38-53	EDID Standard Timings Supported	
54-71	Detailed Timing Descriptor Block 1	
72-89	Detailed Timing Descriptor Block 2	
90-107	Detailed Timing Descriptor Block 3	
108-125	Detailed Timing Descriptor Block 4	
126	Extension Flag	Number of (optional) 128-byte extension blocks to follow
127	Checksum	

Slika 9: Struktura EDID sporočila

2.2.4.3 HDCP

HDCP je kratica za High-bandwidth Digital Content Protection. To je oblika digitalne zaščite pred kopiranjem, ki jo je razvilo podjetje Intel. Namenjena je zaščiti podatkov, ko se prenašajo po povezavah (kablih) med napravami. Zaščito te vrste uporablja veliko audio/video vmesnikov, kot so HDMI, DisplayPort, DVI, GVIF ... Zaščita deluje tako, da se naprava, preden začne pošiljati podatke, rokuje s sprejemnikom in preveri, ali je sprejemnik pooblaščen, da sprejme podatke. Če je, jih oddajnik zakodira ter začne pošiljati. Da lahko naprava predvaja vsebino, ki je zakodirana s HDCP, si mora podjetje, ki napravo izdeluje, pridobiti HDCP ključ. Da ključ pridobi, mora naprava ustrezati določenim standardom (naprava ne sme imeti možnosti snemanja).

2.3 Dekodirnik signala

Dekodirnik signala je pomemben del svetila. Odgovoren je za sprejem slikovnih podatkov in shranjevanje teh podatkov v pomnilnik. Hkrati te podatke pretvori v takšno obliko, da jih lahko naslednja enota uporabi in z njimi računa.

Če za vhod svetila uporabimo USB vhod, te enote ne potrebujemo, saj je dekodirnik signala programsko implementiran na gostitelju.

Če pa za vhod uporabimo direkten slikovni signal, kjer se podatki pošiljajo pri zelo visokih frekvencah, mora ta enota svetila delovati na najvišji hitrosti oz. frekvenci v napravi. Takt te enote določa slikovni signal. Ker je frekvenca, s katero mora enota delovati, tako visoka, mora biti enota sestavljena iz integriranih vezij, ki so namensko narejena za zelo visoke hitrosti.

Dekodirnik signala za bias lighting svetilo je enota, ki ima zelo specifične zahteve, zato ga ne moremo kupiti kot eno integrirano vezje, ampak moramo tovrstno enoto sestaviti iz več takšnih vezij. Najprej lahko uporabimo integrirana vezja, ki signal pretvorijo iz serijskega v paralelnega ali vezja, ki zmanjšajo resolucijo slikovnega podatka. Na ta način zmanjšamo frekvenco, s katero moramo podatek obdelovati, da jo sploh lahko obvladujejo naslednje komponente. In sicer RAM pomnilnik in pomnilniški krmilnik, ki prejete podatke shranita.

Da se izognemo takšni kompleksnosti in hkrati močno povečamo fleksibilnost vezja, lahko namesto pomnilnika in pomnilniškega krmilnika uporabimo posebno nastavljivo integrirano vezje (FPGA). Tega nastavimo tako, da hkrati opravlja delo obeh. Ta element je opisan v nadaljevanju.

2.3.1 FPGA

FPGA je kratica za field-programmable gate array. To je integrirano vezje, narejeno tako, da ga nastavi uporabnik ali načrtovalec po proizvodnji. Konfiguracija je običajno specificirana z uporabo jezikov, ki opisujejo strojno opremo (hardware description language). Najpogostejša jezika sta Verilog in VHDL. Jeziki, ki opisujejo strojno opremo, se uporabljajo tudi pri načrtovanju navadnih integriranih vezij. FPGA vsebuje veliko število nastavljivih logičnih celic, ki jih lahko nastavimo, da izvajajo vrsto logičnih operacij. Te celice lahko potem vežemo skupaj v kompleksnejše strukture. Poleg logičnih celic, običajno FPGA vsebuje tudi pomnilniške elemente, kot so flip-flopi in blokovni ram. Ker se, glede na načrt, ki ga napišemo, logične celice v čipu direktno električno povežejo, lahko čip s tem doseže izjemno velike hitrosti (skoraj tako velike kot pri namenskih integriranih vezij), popolno paralelnost, kar pomeni, da lahko en del čipa izvaja nek proces, medtem ko drug del čipa počne nekaj popolnoma drugega.

FPGA torej odlikuje izjemna fleksibilnost in hitrost, ima pa tudi slabe lastnosti, kot so, visoka cena, manjša energetska učinkovitost ter kompleksnost delovanja in načrtovanja.

Slika 10: Primer FPGA integriranega vezja

2.4 Enota za upravljanje barv

Enota za upravljanje barv je namenjena izbiranju podatkov, ki so pomembni za izhod svetila. To so barve slikovnih pik ob robu zaslona. Ko enota izbere prave barve, na njih po potrebi opravi barvno korekcijo. Ta je potrebna, ker karakteristika izhodne enote ni linearna in se barve na izhodu ne ujemajo s tistimi na zaslonu. Poleg tega je barva, ki jo oddaja izhodna enota, odvisna od podlage, na katero sveti. Če je podlaga bele barve, je nemoteča in korekcija ni potrebna. Če pa je katere koli druge barve, je potrebno izvesti barvno korekcijo tudi glede na barvo podlage, da bo sij svetil v pravih barvah. Ko enota konča z barvno korekcijo, mora iz teh podatkov ustvariti signal, ki bo ustrezal izhodni enoti svetila.

Enota za upravljanje barv mora biti ravno tako kot dekodirnik signala, narejena iz več integriranih vezij. Če smo za dekodirnik signala uporabili posebno integrirano vezje FPGA, lahko tudi enoto za upravljanje barv, implementiramo vanj.

2.5 Izhodna enota

Izhodna enota bias lighting svetila, je komponenta, ki jo nadzoruje enota za upravljanje barv. Izhodna enota se montira za zaslon in ustvarja svetlobni sij okoli njega. Za izhodno enoto svetila je najprimernejša komponenta, ki jo lahko uporabimo, trak na katerem so pritrjene svetleče diode (Light Emmiting Diode, kratica LED). Barve LED so nastavljive. Trak ima na sebi tudi lepilo, ki pride prav pri montaži na zaslon. Obstaja več vrst LED trakov. Za izhodno enoto svetila so primerni trakovi takšnega tipa, ki imajo možnost nastavljanja barve vsake LED posebej, neodvisno druga od druge.

Slika 11: Primeren tip LED traku za bias lighting svetilo

3 Postopek izdelave naprave

3.1 Izbira komponent

Po teoretičnem raziskovanju, s katerim sem se seznanil z značilnostmi delovanja svetila, z njegovimi enotami, komponentami in njegovim delovanjem, sem lahko začel pripravljati načrt svojega svetila.

Najprej sem napisal približen seznam komponent, ki bi jih potreboval za izdelavo svetila. Ker sem želel narediti kompleksnejšo napravo in takšno, ki za svoje delovanje ne potrebuje računalnika, se nisem odločil za uporabo USB pri vhodni enoti svetila. HDMI vmesnik je trenutno najbolj razširjen vmesnik za povezavo med zaslonom in virom slike, zato sem izbral izdelavo svetila, ki bo za slikovni vhod uporabljalo HDMI vmesnik. Uporaba HDMI vmesnika pomeni, da bom imel delo z visokimi frekvencami.

Ob tej izbiri sem ugotovil, kakšne naloge morajo komponente opravljati, nisem pa še vedel, katere bi izbral.

Po temeljitem teoretičnem preučevanju projekta sem spoznal, da bi bila za moje raziskovanje najustreznejša uporaba posebnega FPGA integriranega vezja. Uporaba FPGA vezja omogoča spreminjanje implementiranih enot, kar je za raziskovanje ključnega pomena. Izbira najmanjšega FPGA vezja, ki še zadostuje potrebam nekega projekta, je zelo zahtevna, zato bi naj po oceni načrtovalca imela več kot le zadostne zmožnosti. Ko sem izbiral med različnimi FPGA vezji, sem ugotovil, da je najustreznejša rešitev za moj projekt kompromis, da uporabim kombinacijo splošno dostopnega in nizko cenovnega FPGA vezja s sprejemniškim HDMI integriranim vezjem. Sprejemniško integrirano vezje, ki pretvori signal s HDMI vmesnika iz serijskega v paralelnega, in s tem zmanjša frekvenco za 10-krat (iz 2,25 Ghz v 225 Mhz), omogoča uporabo FPGA vezja, ki deluje že pri frekvenci delovanja vsaj 225 Mhz.

Ker bom uporabil FPGA vezje, bom v njem implementiral vse ostale potrebne komponente za delovanje svetila. Tako bo naprava za delovanje potrebovala zelo majhno število integriranih vezij. Sprejemniško integrirano vezje ima tudi možnost računanja kontrolnih signalov, ki pridejo prav za lociranje trenutne slikovne pike, ki se prenaša po HDMI vmesniku.

Ko sem naredil izbor sprejemniških HDMI čipov, sta se mi ponudili dve možnosti. To sta bila čipa ADV7611 in IT6604. Čipa imata podobne lastnosti. Oba podpirata želeno resolucijo slike in izračunata potrebne kontrolne signale.

Prednost čipa ADV7611 je v enostavnosti. Ima manj pinov – nožic in ga je zaradi tega lažje prispajkati. Hkrati je zato potrebna manjša kompleksnost tiskanega vezja, ki je v tem primeru lahko že dvoplastno. Prednost čipa IT6604 je v tem, da podpira višji HDMI standard, HDCP dekripcijo in ima nižjo ceno. Njegova slabost je, da ga je zaradi veliko večjega števila pinov zelo težko prispajkati ročno, potrebuje kompleksno tiskano vezje, ki mora biti po navodilih proizvajalca vsaj

štiriplastno, ter slabše napisana navodila uporabe, proizvajalec čipa je kitajsko podjetje.

Po proučitvi obeh čipov, sem se odločil za čip IT6604. V tehnični dokumentaciji čipa in v dokumentaciji HDMI vmesnika sem ugotovil, da mora imeti vsaka HDMI naprava urejeno zaščito pred elektrostatičnim prebojem (ESD). V dokumentaciji čipa IT6604 je bila priporočena uporaba zaščitnih diod RClamp0524p. Ker je te diode zelo težko prispajkati ročno, sem namesto njih poiskal in uporabil čip, ki ga je relativno enostavno prispajkati in ki opravlja enako nalogo. To je čip TPD12S520DBTR. Napajanje za napravo sem že imel, v obliki starega polnilca za telefon. Njegova izhodna napetost je 5V. Ker čipi, s katerimi nameravam delati, potrebujejo 3,3 V in 1,8 V, sem moral v vezje vključiti regulacijo napetosti. Za regulacijo napetosti se običajno uporabljajo linearni regulatorji napetosti. Izbral sem dva takšna regulatorja, za vsak nivo napetosti enega (AMS1117 3,3 V in AMS1117 1,8 V). V tehnični dokumentaciji regulatorjev so predpisani elementi, ki so potrebni, da regulacija dobro deluje. Tako sem potreboval tudi več kondenzatorjev in nekaj feritnih jeder, ki pomagajo pri odpravi visokofrekvenčnega elektromagnetnega šuma.

Ker se doslej s FPGA vezjem še nisem ukvarjal, se mi je zdelo smiselno, da kupim FPGA, ki je že prispajkan na razvojno ploščico. Tako sem se izognil ročnemu spajkanju še enega čipa, ki ga je sicer zelo težko ročno prispajkati. Prednost razvojne ploščice je tudi v tem, da je pravilno delovanje FPGA vezja zagotovljeno. Izbral sem FPGA čip modela XC6SLX16, proizvajalca Xilinx.

Za izhodno enoto – LED trak sem izbral trak s svetlečimi diodami WS-2812. Te diode imajo vgrajen čip, ki omogoča, da preko serijske komunikacije z enim signalom nastavljamo barve vseh diod na traku posamično. Vso elektroniko, potrebno za krmiljenje traku, sem nameraval implementirati v FPGA.

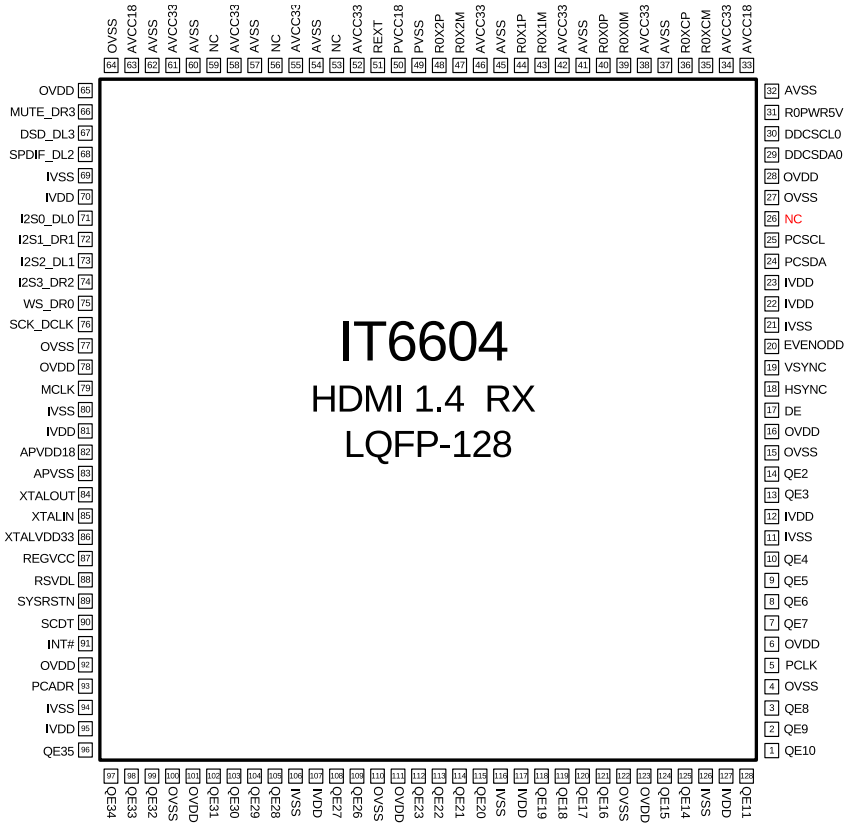
Vse komponente sem naročil iz Kitajske.

3.1.1 Opis izbrane komponente IT6604

Čip, ki sem ga uporabil za sprejemanje signala po HDMI vmesniku, je IT6604. Izbral sem ga zaradi cene ter dejstva, da vsebuje HDCP ključ in z njegovo pomočjo lahko dekodira tudi podatke, ki so kriptirani s HDCP enkripcijo. Čip je v mojem projektu uporabljen zato, da opravi TMDS dekodiranje iz 10 bitov v 8 bitov, zato da opravi HDCP dekripcijo in zato, da serijski TMDS signal pretvori v paralelnega in s tem efektivno zmanjša frekvenco, s katero je potrebno podatke obdelovati za 10-krat.

Pomembne tehnične lastnosti čipa za izdelavo mojega svetila:

- kompatibilnost s standardom HDMI 1.3 in 1.4a,
- skladnost s HDCP 1.4 in tovarniško vključen HDCP ključ,
- video izhod v različnih formatih, kot so: RGB/YCbCr 4:4:4 24/30-bit 16/20-bit YCbCr 4:2:2, 8/10-bit YCbCr 4:2:2,
- 128-pinsko LQFP (14 mm x 14 mm) ohišje,
- podpora za resolucijo do 1920x1080 slikovnih pik pri frekvenci 60 Hz.



Slika 12: Podnožje in povezave integriranega vezja IT6604

3.1.2 Opis izbrane komponente FPGA XC6SLX16

Ko sem se odločil, da bom za projekt uporabil FPGA, nisem vedel, kateri čip naj izberem. Nisem vedel niti, za katerega proizvajalca naj se odločim. Ker sem s FPGA vezji delal prvič, in ker pri načrtovanju zelo težko ocenimo, kako zmogljiv FPGA potrebujemo, sem se odločil, da izberem takšnega, ki bi po moji oceni moral zadostiti mojim potrebam vsaj 4-kratno. Hkrati pa nisem želel zapraviti preveč. Pri izbiri je bilo pomembno tudi, od katerega podjetja želim čip kupiti, saj je od tega odvisno, kako dobro podporo in pomoč pri načrtovanju lahko dobim na spletu. Na koncu sem se odločil za čip XC6SLX16, podjetja Xilinx. Čipa nisem kupil samostojno, ampak na razvojni ploščici, saj je čip v BGA ohišju in takšnega elementa ročno ni mogoče prispajkati na tiskano vezje. Hkrati pa sem tako imel zagotovilo, ta bo ta del projekta zagotovo deloval. Da sem lahko FPGA nastavljal, sem moral kupiti tudi JTAG vmesnik.

Pomembne specifikacije čipa XC6SLX16:

- 14.579 nastavljivih logičnih celic,
- 18.224 flip-flop pomnilnikov,
- 576 Kb blokovnega RAM pomnilnika,
- 4 vhodno-izhodne enote,
- 232 vhodno-izhodnih nastavljivih pinov,
- 2 pomnilniška krmilnika.

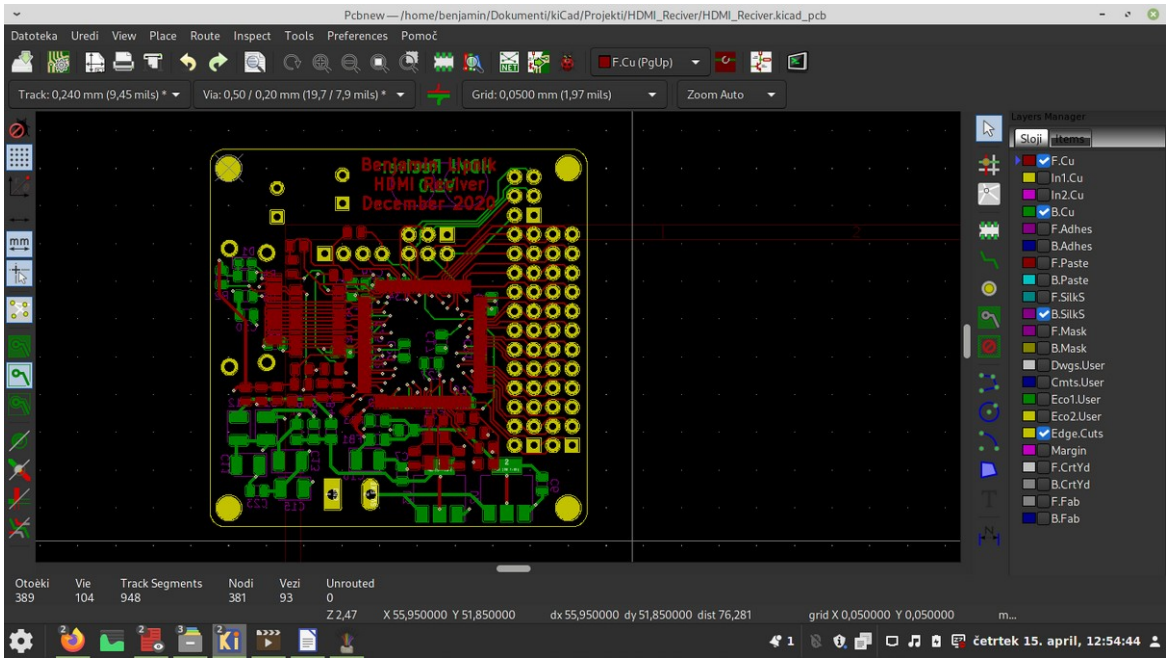
Slika 13: Kupljena razvojna ploščica z FPGA integriranim vezjem XC6SLX16

3.2 Izdelava tiskanega vezja (PCB)

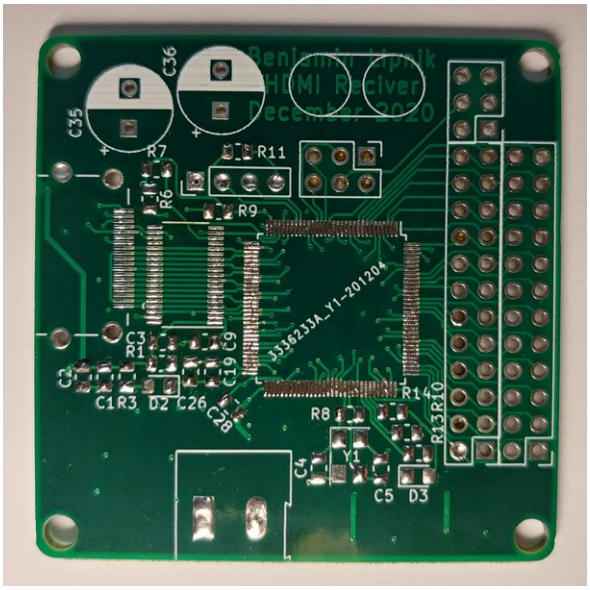
Vezje sem začel risati takoj po naročilu komponent. V dokumentaciji integriranih vezij je opisano, kakšne so zahteve, da integrirana vezja lahko dobro delujejo.

Tiskano vezje sem narisal v programu KiCAD, ter ga po risanju poslal podjetju JLCPB, ki mi je vezje izdelalo.

Ker sem FPGA kupil na razvojni ploščici, sem se odločil, da bom ploščice povezal s konektorjem. Na vezju sem pripravil položaj za čipe, HDMI konektor, in ostale pasivne komponente. Vezje sem poskušal pripraviti tako, da bi bil električni šum čim manjši. Tiskano vezje in razvojno ploščico sem povezal s konektorji.



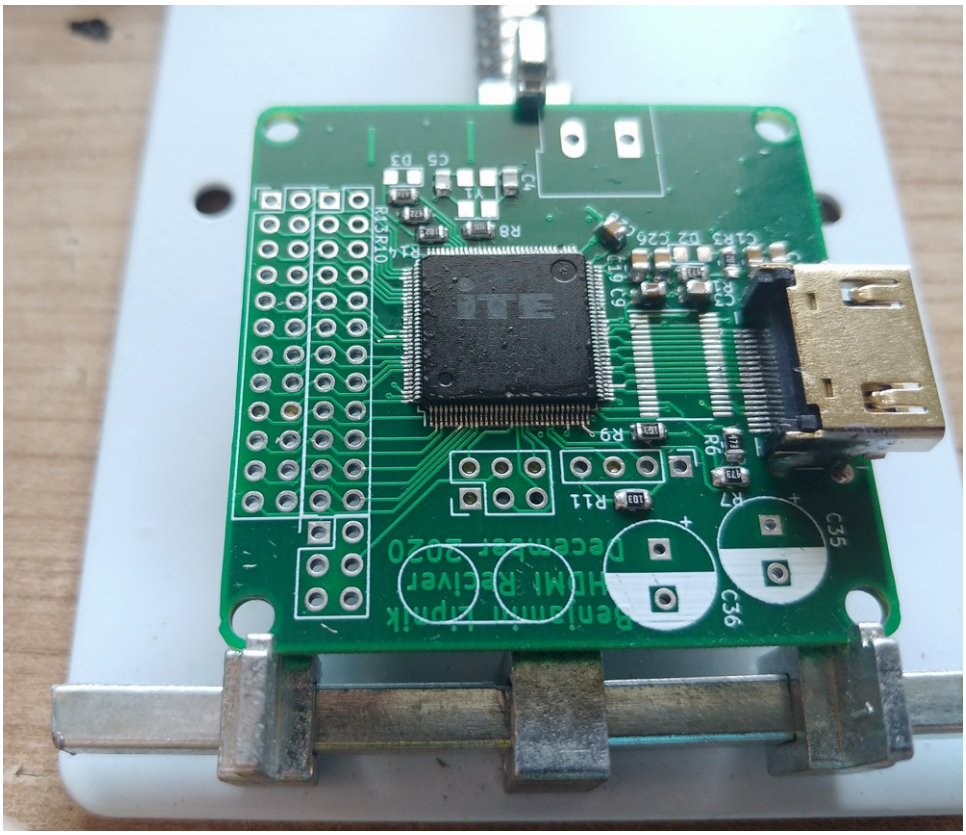
Slika 14: Izris tiskanega vezja v programu KiCad



Slika 15: Izdelano tiskano vezje

3.3 Spajkanje vezja

Ko sem dobil vse komponente, sem začel s spajkanjem elementov na tiskano vezje. Vse komponente sem prispajkal ročno. Začel sem z elementi, ki jih je najtežje prispajkati. To je bil čip IT6604, ker je tako majhen, da s prostim očesom ne moremo opaziti napak, ki se lahko pojavijo pri spajkanju. Po spajkanju sem vezje preveril še pod mikroskopom. Ker sem odkril več območij, kjer se je spajka dotikala več pinov hkrati, sem moral spajkanje ponoviti in ga pregledovati, dokler ni bilo popolno.



Slika 16: Delno sespajkano tiskano vezje

3.4 Nastavljanje in načrtovanje vezja v FPGA s pomočjo jezika Verilog

Verilog je eden izmed najpopularnejših jezikov za opisovanje strojne opreme. Ta jezik sem izbral za nastavljanje FPGA vezja zato, ker se mi je v primerjavi z VHDL (jezikom, ki je prav tako zelo popularen) zdel lažje razumljiv in berljiv. Ker sem se z načrtovanjem vezja v FPGA ukvarjal prvič, jezika Verilog še nisem uporabljal. Naučiti sem se ga moral na novo.

Enota, ki sem jo implementiral v Verilogu, je odgovorna za prevzem podatkov s IT6604, za izbiro, katere izmed teh podatkov potrebujemo, za povprečenje in shranjevanje teh podatkov, na koncu pa še za pripravljanje in pošiljanje teh podatkov na led trak. To so naloge dekodirnika signalov in enote za upravljanje barv.

V Verilogu se vezje načrtuje po modulih. Moduli se obnašajo kot integrirana vezja. Imajo določene vhode in izhode. V Verilogu module razporejamo hierarhično. Modul na vrhu (imenovan Top Level Module), predstavlja celotno vezje FPGA.

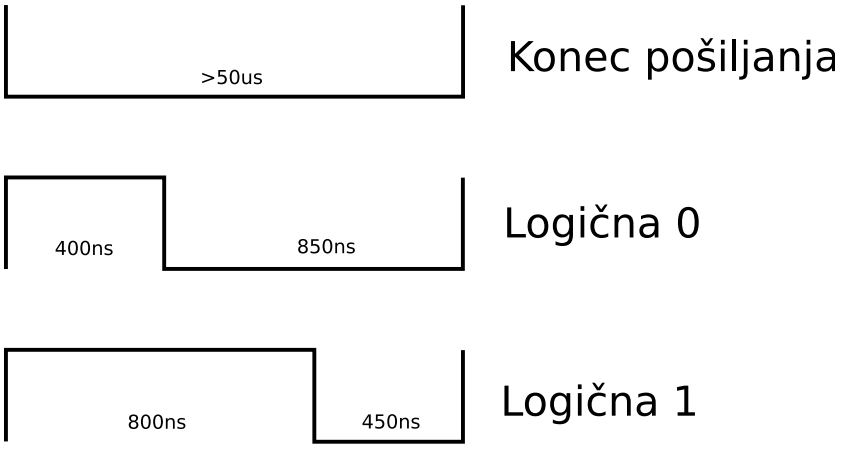
Najprej sem v Verilogu definiral Top Level Module. Definiral sem njegove vhode, ki so povezani na čip IT6604, in njegove izhode, ki so povezani na izhodno enoto – LED trak.

3.4.1 Implementacija gonilnika za LED trak

Prvi modul, ki sem se ga odločil implementirati, je bil gonilnik za LED trak, saj je gonilnik osnova višje nivojskih modulov. Poleg tega sem ga tako lahko dobro preizkusil, preden sem začel pisati ostale module. Naloga gonilnika je, da iz blokovnega ram pomnilnika pobira podatke ter iz njih naredi signal za krmiljenje LED traku. Signal sem povezal preko izhodnega pina FPGA vezja na LED trak.

Podatki se na LED trak pošiljajo serijsko. Za vsako LED, se binarno pošlje po 8 bitov podatka za zeleno barvo, 8 bitov za rdečo in 8 bitov za modro barvo. Naslednjih 24 bitov se nanaša na drugo LED traku, spet naslednjih 24 bitov se nanaša na tretjo LED in tako naprej. Podatki se prenašajo s pomočjo protokola, ki definira logično enico kot 800 ns dolgo visoko stanje in po prehodu 450 ns dolgo nizko stanje. Logična nič pa je definirana kot 400 ns dolgo visoko stanje in po prehodu 850 ns dolgo nizko stanje. Ko s pošiljanjem končamo, označimo konec z nizkim stanjem signala za 50 us. Časovna toleranca vseh prehodov je 150 ns.

Gonilnik sem implementiral kot modul. Za vhod modula sem določil urin signal, ki je povezan na 50 Mhz oscilator na razvojni ploščici, signal, ki je pogoj za začetek pošiljanja na trak in podatek, ki ga gonilnik mora poslati. Za izhod pa sem določil zaporedno številko LED, katerega podatek gonilnik trenutno pošilja in pa signal, ki bo povezan na LED trak. Začel sem tako, da sem urin takt oscilatorja na razvojni ploščici delil s 16 in tako dobil frekvenco 50 Mhz /16 = 3,125 Mhz. Čas periode te frekvence = 1/3,125 us = 320 ns. To mi je omogočilo, da sem lahko poslal en bit informacije v treh urinih ciklih. Vsakega prvega od treh sem postavil na visoko stanje (logična 1), drugi takt na stanje bita, ki ga želim poslati in tretji takt na nizko stanje (logična 0). Tako je podatek pretvorjen v signal in upošteva časovne tolerance. Po 24*3 urinih ciklih, povečam zaporedno število LED in začnem z novim pošiljanjem. Tako lahko gonilnik priključimo na blokovni RAM pomnilnik, v katerem so shranjene barve, ki jih želimo poslati.



Slika 17: Definicije logičnih stanj protokola LED traku

3.4.2 Uporaba gonilnika z blokovnim RAM pomnilnikom

Ko sem imel narejen gonilnik, ki lahko bere z blokovnega RAM pomnilnika, je bila za delovanje svetila potrebna le še izbira slikovnih pik, ki se prenašajo preko HDMI vmesnika in njihovo shranjevanje v RAM pomnilnik gonilnika za LED trak. Izbiro sem naredil glede na pozicijo slikovne pike, ki sovpada s pozicijo LED na traku, ki je montiran za televizorjem. V mojem primeru sem lahko na levo in desno stran televizijskega zaslona pritrdil dolžino traku, na katerem je bilo 25 LED. Na vrhu zaslona pa je na dolžino televizijskega zaslona prišlo 43 LED. Da lahko izračunamo, katera slikovna pika sovpada s katero LED, potrebujemo x in y koordinato slikovne pike na zaslonu.

Slika 18: Montaža LED traku na hrbtno stran televizijskega zaslona

```
module buffered_ws_driver #(parameter ST_LED = 500, ADDRESS_WIDTH = 9)
(
    input led_clk,
    input wclk,
    input we,
    input [23:0] color_into_buffer,
    input [ADDRESS_WIDTH-1:0] buffer_index,
    input led_send_en,
    input vsync,

    output ws_data

 );

//Preklaplanje stanje spremenljivke za vsako sliko
reg frame_even;
always@ (posedge vsync) begin
    frame_even <= ~frame_even;
end


wire we1 = we && !frame_even;
wire we2 = we && frame_even;

wire [23:0] podatek_iz_rama1;
wire [23:0] podatek_iz_rama2;

wire [ADDRESS_WIDTH-1:0] naslov_s_katerega_gonilnik_zeli_brati;
wire [23:0] podatek_iz_rama_v_trak = frame_even ? podatek_iz_rama1 : podatek_iz_rama2;


//Prvi blokovni RAM pomnilnik
dual_clock_ram #(ADDRESS_WIDTH,24,ST_LED) dvojni_ram1
(.rclk(led_clk), .wclk(wclk), .we(we1), .data(color_into_buffer), .raddr(naslov_s_katerega_gonilnik_zeli_brati),.
 .waddr(buffer_index), .q(podatek_iz_rama1));

//Drugi blokovni RAN pomnilnik
dual_clock_ram #(ADDRESS_WIDTH,24,ST_LED) dvojni_ram2
(.rclk(led_clk), .wclk(wclk), .we(we2), .data(color_into_buffer), .raddr(naslov_s_katerega_gonilnik_zeli_brati),.
 .waddr(buffer_index), .q(podatek_iz_rama2));

//gonilnik za led trak, ki bere podatke z enega ized blokovnih RAM pomnilnikov
ws_driver #(ST_LED, ADDRESS_WIDTH) led_driver
(.clk(led_clk), .start(led_send_en), .led_color_data(podatek_iz_rama_v_trak),
 .led_color_address(naslov_s_katerega_gonilnik_zeli_brati), .ws_data(ws_data));


endmodule
```

Slika 19: Implementacija dveh blokovnih RAM pomnilnikov, ki so povezani skupaj z gonilnikom LED traku

3.4.3 Implementacija modula za računanje x in y koordinate

Za računanje x in y koordinate slikovne pike, ki se prenaša po HDMI vmesniku, in je prikazana na zaslonu, sem naredil modul, ki za vhod vzame tri signale. To so urin takt slikovnega signala ter dva kontrolna signala, ki ju dobimo s čipa IT6604. Signala označujeta začetek nove vrste in začetek nove slike (horizontal sync in vertical sync). Izhod modula pa sta dva 11 bitna podatka, ki označujeta koordinati x in y. Koordinate v modulu izračunamo tako, da za vsak urin cikel povečamo koordinato x. Ob vsakem pozitivnem prehodu signala, ki označuje začetek nove vrste, pa koordinato y povečamo za 1 in koordinato x nastavimo nazaj na 0. Ob pozitivnem prehodu signala, ki označuje začetek nove slike pa nastavimo x in y koordinato na 0.

```
module position_extractor
(
    input pclk,
    input data_enable,
    input vsync,

    output reg [10:0] pos_x,
    output reg [10:0] pos_y
);

always@ (posedge pclk) begin
    if(vsync == 1'b1) begin
            pos_x <= 11'd0;
            pos_y <= 11'd0;
    end
    else if(data_enable == 1'b1) begin
        pos_x <= pos_x + 11'd1;
    end
    else begin
        pos_x <= 11'd0;
        pos_y <= (pos_x == 11'd0) ? pos_y : pos_y + 11'd1;
    end
end

endmodule
```

Slika 20: Implementacija modula za računanje x in y koordinate

3.4.4 Izbiranje slikovnih pik in barvna korekcija

Ko sem imel x in y koordinate slikovne pike, sem v Verilogu napisal logično enačbo, ki izračuna, ali je slikovna pika na katerem izmed robov. Nato s pomočjo druge enačbe izračunam, na katero pozicijo v RAM pomnilniku moramo to slikovno piko shraniti in jo tja tudi shranim.

Hkrati lahko naredimo barvno korekcijo slikovne pike. Eden izmed načinov barve korekcije je, da vsako izmed osnovnih barv (rdeča, modra in zelena) pomnožimo z nekim koeficientom. Tako lahko ojačamo ali omilimo nekatere vrste barvnih odtenkov ter kompenziramo podlago, ki ni bela. Eden izmed naprednejših načinov barvne korekcije je gama korekcija. Ta vrsta korekcije omogoča kompenzacije nelinearnosti izhodne enote.

```
//Pozicija slikovne pike, ki se trenutno prenasa po HDMI protokolu
wire [10:0] pos_x;
wire [10:0] pos_y;

//Pogoji za vpis slikovnih pik v seznam
wire pog_levo   = data_enable && (pos_x == 10)   && (pos_y > 40) && (pos_y < 1060) && (pos_y % 42 == 0);
wire pog_zgoraj = data_enable && (pos_y == 10)   && (pos_x > 40) && (pos_x < 1900) && (pos_x % 44 == 0);
wire pog_desno  = data_enable && (pos_x == 1910) && (pos_y > 40) && (pos_y < 1060) && (pos_y % 42 == 0);
wire vpisi_v_ram = pog_levo || pog_zgoraj || pog_desno;

//izracun lokacije na traku, ki ustreza trenutni slikovni piki
wire [10:0] naslov_za_vpis_v_ram = pog_levo ? (25 - (pos_y / 42)) : (pog_zgoraj ? ((pos_x / 44) + 24) :
( (pos_y / 42) + 67));

//Pogoj za zacet posiljanja podatkov na led trak
wire led_start_sending = pos_y > 100;

//podmodul, ki iz kontrolnih signal preracuna lokacijo trenutne slikovne pike, skupaj z integriranim vezjem
it6604 opravlja nalogo dekodirnika signala
position_extractor positioner
(.pclk(pclk), .data_enable(data_enable), .vsync(vsync), .pos_x(pos_x), .pos_y(pos_y));
//podmodul, ki vsebuje pomnilnik katerega napolnimo z zeljenimi barvami, opravlja nalogo enote za upravljanje
barv
buffered_ws_driver #(100, 7) color_writer
(.led_clk(clk), .wclk(pclk), .we(vpisi_v_ram), .color_into_buffer(grb), .buffer_index(naslov_za_vpis_v_ram),
 .led_send_en(led_start_sending), .ws_data(ws_data), .vsync(vsync));
```

Slika 21: Implementacija vrhnjega modula, ki izbira potrebne slikovne pike in jih shrani v pomnilnike drugih podmodulov

3.5 Testiranje in odpravljanje napak

Prvič sem vezje, ki sem ga sespajkal skupaj, priključil brez povezave med svojim tiskanim vezjem in vezjem FPGA razvojne ploščice. To sem storil zato, da morebitna napaka pri načrtovanju vezja ali pri spajkanju vezja ne bi uničila FPGA čipa. Ob prvi priključitvi je posvetila LED dioda, ki sem jo dodal v vezje za indikator napajanja. To je pomenilo, da del, odgovoren za napajanje, deluje. V vezje sem dodal tudi LED indikator, ki kaže na uspešno prejemanje podatkov preko HDMI vmesnika. Ta LED ni posvetila. Izkazalo se je, da sem zmotno pričakoval, da čip IT6604 samodejno opravi EDID rokovanje. Ker čip tega ni naredil, HDMI izvor ni začel pošiljati podatkov. Ko sem to ugotovil, sem moral rokovanje EDID opraviti drugače. Na pine, po katerih se opravi EDID rokovanje, sem priklopil mikrokrmilnik, na katerega sem naložil program, ki ob vklopu opravi rokovanje z EDID. Po tem popravku je moje vezje uspešno prejemalo podatke preko HDMI vmesnika. Nato sem vezje povezal s FPGA razvojno ploščico, na katero sem naložil moje nastavitve. Najprej sem testiral implementacijo gonilnika LED traku, ki sem ga naložil na FPGA. Po nekaj neuspelih poskusih, sem uspel odpraviti vse napake, ki sem jih naredil med implementacijo gonilnika. Takrat sem lahko testno prižgal svetleče diode na želeno barvo. V veliko pomoč mi je bil logični analizator. To je naprava, ki vizualno prikaže časovne poteke signalov. Naslednja stvar, ki sem jo želel preizkusiti, je branje prve slikovne pike, ki jo vezje sprejme po HDMI vmesniku. Tu sem se soočal s problemom dveh urinih ciklov. Podatki, ki jih prejmemo po HDMI vmesniku, so poslani glede na uro HDMI vmesnika, ki je odvisna od resolucije, frekvence ... Podatki, ki jih pošljemo na LED trak, pa morajo biti poslani z znano konstantno frekvenco, za ustvarjanje katere sem uporabil urin takt FPGA razvojne ploščice. Ker urina takta nimata enake frekvence, niti nista sofazna, se pojavlja problem pri shranjevanju podatkov v blokovni RAM pomnilnik, na katerega lahko priključimo le en urin signal. Ta problem sem rešil z uporabo dvournega blokovnega RAM pomnilnika. Tak pomnilnik uporablja en urin signal za branje podatkov in drug urin signal za zapisovanje podatkov.

```
module dual_clock_ram #(parameter ADDR_WIDTH = 8, DATA_WIDTH = 8, DEPTH = 32)
(
    input rclk, wclk, we,
    input [DATA_WIDTH-1:0] data, input [ADDR_WIDTH-1:0] raddr, waddr,
    output [DATA_WIDTH-1:0] q
);
   reg [ADDR_WIDTH-1:0] raddr_reg;
   reg [DATA_WIDTH-1:0] ram[0:DEPTH];
   always@(posedge wclk)
     begin
        if(we) ram[waddr] <= data;
     end
   always@(posedge rclk)
     begin
        raddr_reg <= raddr;
     end
   assign q = ram[raddr_reg];
endmodule
```

Slika 22: Implementacija dvournega blokovnega RAM pomnilnika

4 Razprava

Pred začetkom raziskovanja sem postavil hipoteze, ki sem jih skozi raziskovanje preveril:

- Takšno svetilo je mogoče narediti doma.

To hipoteze lahko povsem potrdim. Svetilo sem izdelal doma. Ob tem je potrebno poudariti, da je izdelava svetila doma možna pod pogojem, da ima izdelovalec dovolj znanja, časa in orodij, ki jih pri izdelavi potrebuje. Ob osnovnem znanju elektronike, ki sem si ga pridobil skozi srednješolsko izobrazbo, sem potreboval mnogo dodatnega znanja, ki sem ga pridobival tudi iz drugih virov. Imel sem dovolj znanja, prav tako dovolj časa za raziskovanje in precej dodatnih orodij, ki jih sicer uporabljam za svoja raziskovanja.

- Domača izdelava svetila je cenejša od kupljenega.

To hipotezo lahko delno potrdim. Strošek materialov doma izdelanega svetila se v mojem svetilu giblje v območju 50 €. To je v primerjavi s kupljenim svetilom (Philips Ambilight Hue trak + Philips Ambilight HDMI Sync box), ki stane nad 250€, malo. Kupljena naprava ima sicer tudi druge zmožnosti, kot so upravljanje barv in svetlobe z mobilnim telefonom. Moj izdelek tega nima, omogoča pa možnost takšne nadgradnje. Seveda pa bi za realnejšo oceno stroškov izdelave svetila moral všteti tudi ure izdelave. V primeru mojega raziskovanja, sem v izdelavo svetila vložil veliko ur dela. Ocenjujem, da bi za izdelavo svetila z znanjem, ki ga imam sedaj, potreboval med 20-30 ur dela. Pri tem bi največ časa porabil za spajkanje elementov. Če ceni prištejemo vrednost opravljenih delovnih ur, bi bila cena doma izdelanega in kupljenega svetila, primerljiva.

- Svetilo, narejeno doma, je mogoče priključiti na več vrst naprav, kot so: televizijski sprejemniki, igralne konzole, osebni računalnik ...

To hipotezo lahko povsem potrdim. Moje svetilo, ki uporablja HDMI, deluje na vseh naštetih napravah. To sem tudi preveril.

- Izkušnja uporabe zaslona je s svetilom bias-lighting, prijetnejša.

To hipotezo lahko potrdim s svojo subjektivno izkušnjo. Gledanje na zaslon je ob uporabi svetila meni osebno prijetnejše, kar sovpada z namenom svetila. Za objektivnejšo oceno, pa bi bilo potrebno nadaljnje raziskovanje.

5 Zaključek

Z rezultatom raziskave sem povsem zadovoljen. Izdelek, narejen med raziskovanjem, deluje tako dobro, kot sem si na začetku zamislil. Uporabljam ga že na domačem televizijskem zaslonu, s čimer sem poskrbel za prijetnejšo izkušnjo gledanja televizije vseh svojih domačih. Tri od štirih zastavljenih hipotez sem povsem potrdil, eno pa delno. Bias lighting je možno izdelati doma, ta izdelek je mogoče priključiti na različne naprave, kot so: TV sprejemniki, igralne konzole, osebni računalniki ..., subjektivna izkušnja gledanja v zaslon, opremljen s svetilom, je prijetnejša. Domača izdelava je lahko cenovno ugodna.

Med raziskovanjem sem spoznal delovanje različnih tehnologij, kot so FPGA integrirana vezja. Hkrati sem dobil vpogled v načrtovanje elektronike, ki deluje na zelo visokih frekvencah. Skozi raziskovanje sem si dokazal, da lahko ustvarim tehnično relativno zahtevno elektronsko napravo kar sam doma.

Svetilo je mogoče še nadgraditi. Napravi bi lahko dodal možnost Wi-Fi komunikacije, ter jo tako povezal z mobilnim telefonom, preko katerega bi lahko spreminjali nastavitve naprave. Če bi napravo še malo izboljšal ter jo uredil, menim, da bi se zanjo lahko našel prostor na tržišču.

V raziskavo sem vložil veliko število ur raziskovalnega dela. Ampak trdim, da uspešen rezultat mojega dela odtehta ves vložen trud.

Slika 23: Delovanje mojega izdelka

6 Viri

28c3: Implementation of MITM Attack on HDCP-Secured Links: https://www.youtube.com/watch?v=37SBMyGoCAU, dostop 8. 12. 2020.

AMS1117 1A LOW DROPOUT VOLTAGE REGULATOR: http://www.advanced-monolithic.com/pdf/ds1117.pdf, dostop 12. 1. 2021.

Ambilight: https://en.wikipedia.org/wiki/Ambilight, dostop 14. 11. 2020.

Bias lighting: https://en.wikipedia.org/wiki/Bias_lighting, dostop 1. 4. 2021.

Digital Visual Interface: https://en.wikipedia.org/wiki/Digital_Visual_Interface, dostop 12. 11. 2020.

Digital Visual Interface DVI: https://www.fpga4fun.com/files/dvi_spec-V1_0.pdf, dostop 28. 12. 2020.

Dissecting HDMI (33c3): https://www.youtube.com/watch?v=yqYBcZzMPGQ, dostop 31. 12. 2020.

Extended Display Identification Data: https://en.wikipedia.org/wiki/Extended_Display_Identification_Data, dostop 25. 3. 2021.

FPGAs 1 - What are they?: https://www.fpga4fun.com/FPGAinfo1.html, dostop 26. 3. 2021.

FPGAs 2 - How do they work?: https://www.fpga4fun.com/FPGAinfo2.html, dostop 19. 3. 2021.

Field-programmable gate array: https://en.wikipedia.org/wiki/Field-programmable_gate_array, dostop 5. 11. 2020.

HDMI: https://sl.wikipedia.org/wiki/HDMI, dostop 7. 1. 2021.

HDMI: https://www.fpga4fun.com/HDMI.html, dostop 2. 3. 2021.

HDMI Demystified: https://www.fpga4fun.com/files/HDMI_Demystified_rev_1_02.pdf, dostop 13. 3. 2021.

HDMI Specifications and Programs: https://www.hdmi.org/spec/index, dostop 6. 1. 2021.

HDMI – Hacking Displays Made Interesting: https://www.nccgroup.com/globalassets/our-research/uk/whitepapers/hdmi_-_hacking_displays_made_interesting.pdf, dostop 10. 11. 2020.

HDMI/DVI Receivers: https://www.analog.com/en/products/audio-video/hdmidvi-receivers.html, dostop 22. 12. 2020.

High Definition Multimedia Interface: https://en.wikipedia.org/wiki/HDMI, dostop 12. 11. 2020.

High-bandwidth Digital Content Protection: https://en.wikipedia.org/wiki/High-bandwidth_Digital_Content_Protection, dostop 30. 11. 2020.

I2C: https://en.wikipedia.org/wiki/I%C2%B2C, dostop 13. 3. 2021.

IT6604 Single-Link HDMI 1.4 Receiver with 3D Support: https://datasheetspdf.com/pdf-file/824214/ITETECH/IT6604/1, dostop 5. 12. 2020.

JTAG: https://www.fpga4fun.com/JTAG.html, dostop 4. 11. 2020.

Learn FPGA #1-21: Getting Started - Tutorial: https://www.youtube.com/watch?v=vjBsywUSKWk&list=PL2935W76vRNGRtB09yXBytO6F3zSZFZGr, dostop

15. 3. 2021.
Lecture #2: Verilog HDL: https://web.stanford.edu/class/ee183/handouts/lect2.pdf, dostop 12. 2. 2021.
Low Power, 165 MHz HDMI Receiver ADV7610: http://www.farnell.com/datasheets/2258665.pdf, dostop 19. 3. 2021.
Low Power, 165 MHz HDMI Receiver ADV7611: https://www.analog.com/en/products/adv7611.html, dostop 27. 12. 2020.
Modeline Database: https://www.mythtv.org/wiki/Modeline_Database, dostop 8. 12. 2020.
RClamp0524P: https://www.mouser.com/datasheet/2/761/rclamp0522p_0524p-1276864.pdf, dostop 5. 3. 2021.
SiI9135/SiI9135A HDMI Receiver with Enhanced Audio and Deep Ior Outputs: https://datasheet.lcsc.com/szlcsc/1903061002_Lattice-SII9135ACTU_C369570.pdf, dostop 28. 3. 2021.
Spartan-6 FPGA Data Sheet: https://www.xilinx.com/support/documentation/data_sheets/ds162.pdf, dostop 4. 2. 2021.
TPD12S520 Single-Chip HDMI Receiver, Port Protection and Interface Device: https://pdf1.alldatasheet.com/datasheet-pdf/view/211628/TI/TPD12S520/+Q1Q74-VRh/1TRG-tXCvN+/datasheet.pdf, dostop 7. 3. 2021.
Transition-minimized differential signaling: https://en.wikipedia.org/wiki/Transition-minimized_differential_signaling, dostop 13. 12. 2020.
Troubleshooting HDMI Systems - Diagnostic Principles & Techniques: https://www.quantumdata.com/pdf/HDMI_Presentation.pdf, dostop 27. 11. 2020.
USB: https://en.wikipedia.org/wiki/USB, dostop 27. 3. 2021.
Understanding DVI-D, HDMI And DisplayPort Signals: https://www.fpga4fun.com/files/Understanding%20DVI-D,%20HDMI%20and%20Display%20Port%20Signals.pdf, dostop 13. 12. 2020.
Univerzalno serijsko vodilo: https://sl.wikipedia.org/wiki/Univerzalno_serijsko_vodilo, dostop 5. 12. 2020.
Verilog HDL tips and tricks: https://www.fpga4fun.com/VerilogTips.html, dostop 22. 12. 2020.
Video Graphics Array: https://en.wikipedia.org/wiki/Video_Graphics_Array, dostop 8. 2. 2021.
Video/Image Processing on FPGA: https://web.wpi.edu/Pubs/ETD/Available/etd-042915-124951/unrestricted/jzhao.pdf, dostop 27. 11. 2020.
WS2812B: https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf, dostop 25. 12. 2020.

6.1 Viri slik

Slika 1: https://upload.wikimedia.org/wikipedia/commons/3/3c/Ambilight-1.jpg, dostop 6. 11. 2020.
Slika 3: https://cdn.sparkfun.com//assets/parts/2/7/7/00512-USB_Cable_A_to_B_-_6_Foot-02.jpg, dostop 1. 1. 2021.
Slika 4: https://res.cloudinary.com/rsc/image/upload/b_rgb:FFFFFF,c_pad,dpr_1.0,f_auto,h_843,q_auto,w_1500/c_pad,h_843,w_1500/F6659532-01?pgw=1&pgwact=1, dostop 8. 12. 2020.
Slika 5: https://www.techtrade.si/images/thumbs/0028839_displayport-dvi-kabel-2m-digitus_600.jpeg, dostop 14. 3. 2021.
Slika 6: https://static-cdn.imageservice.cloud/3046305/hdmi-connector-pinout-audio-wiring-diagram.jpg, dostop 12. 11. 2020.
Slika 7: https://www.computercablestore.com/content/images/thumbs/0013970_15-meter-4921-ft-high-speed-hdmi-cable-with-ethernet.jpeg, dostop 11. 1. 2021.
Slika 9: https://media.extron.com/public/company/img/table2.jpg, dostop 3. 2. 2021.
Slika 10: https://upload.wikimedia.org/wikipedia/commons/f/fa/Altera_StratixIVGX_FPGA.jpg, dostop 7. 1. 2021.
Slika 11: https://www.sdiplight.com/wp-content/uploads/2019/11/programmable-led-strip-IC-ws2812b.jpg, dostop 19. 1. 2021.
Slika 12: https://datasheetspdf.com/pdf-file/824214/ITETECH/IT6604/1, dostop 10. 12. 2020.

Slike 2,8,13,14,15,16,17,18,19,20,21,22,23: lasten arhiv

IZJAVA*

Mentor **Zupanc Davor** v skladu z 20. členom Pravilnika o organizaciji mladinske raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi z naslovom **Izdelava bias-lighting svetila,** katere avtor je **Lipnik Benjamin**:

- besedilo v tiskani in elektronski obliki istovetno,
- pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,
- da je za objavo fotografij v nalogi pridobljeno avtorjevo dovoljenje in je hranjeno v šolskem arhivu,
- da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na knjižničnih portalih z navedbo, da je raziskovalna naloga nastala v okviru projekta Mladi za Celje,
- da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,
- da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

Celje, 12. maj 2021

žig šole

ŠOLSKI CENTER CELJE
ŠOLA ZA KEMIJO, ELEKTROTEHNIKO IN RAČUNALNIŠTVO
SREDNJA 1
CELJE, Pot na Lavo 22

Podpis mentorja

Zupanc

Podpis odgovorne osebe

za Lazič

*

POJASNILO

V skladu z 20. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje je potrebno podpisano izjavo mentorja (-ice) in odgovorne osebe šole vključiti v izvod za knjižnico, dovoljenje za objavo avtorja (-ice) fotografskega gradiva, katerega ni avtor (-ica) raziskovalne naloge, pa hrani šola v svojem arhivu.