SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 445

Izgradnja ručnog računala

Davor Cihlar

Zagreb, lipanj 2012.

SADRŽAJ

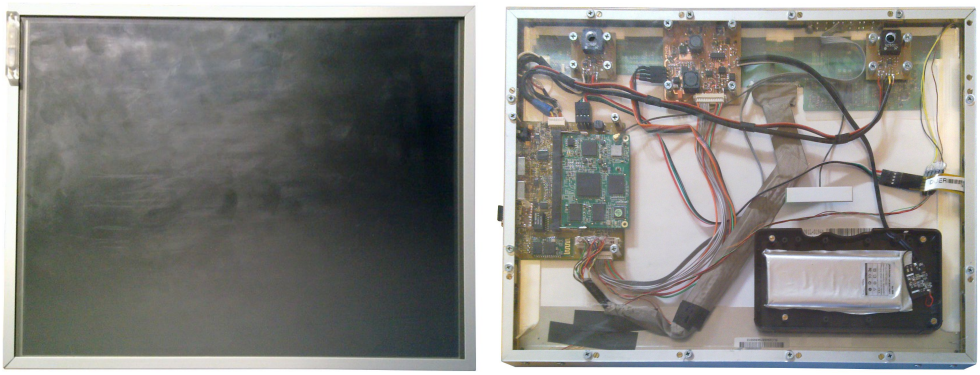
1. Uvod

Veća ručna računala (engl. *tablet*) su postala jako popularna dolaskom Appleovog iPada na tržište, a prijenosna računala poput Tablet-PC-a su zasjenjena. Kako je moja želja oduvijek bila napraviti prijenosno računalo poput Tablet-PC-a, za ovaj diplomski rad sam odlučio otići korak dalje i ispitati mogućnosti izrade ručnog računala pomoću dostupnih komponenti te usporediti rezultate sa stvarnim sustavima.

Komponente su birane prema mogućnostima, cijeni i potrošnji. U nekim slučajevima su odabrane komponente veće efikasnosti (manje potrošnje) i više cijene, dok su u nekim drugim slučajevima odabrane komponente s najmanjom cijenom jer ih je potrebno puno ili su same po sebi jako skupe.

Kao operacijski sustav je odabran Linux zbog svoje otvorenosti, jednostavne prilagodbe i velikih mogućnosti, a kao srce cijelog sustava je odabran procesorski modul TDM-3730 firme TechNexion zbog visokih performansi, lake dostupnosti i relativno niske cijene te zbog toga što dolazi s predinstaliranim operacijskim sustavom Linux.

Funkcionalni prototip je s obje strane prikazan na slici 1.1.



(a) Prednja strana **(b)** Stražnja strana

Slika 1.1: Gotov prototip

U prvom dijelu rada je prikazan blok-dijagram i po funkcijama podijeljen opis pojedinih sklopovskih komponenti, dok je u drugom dijelu prikazana osnovna i periferna programska podrška s prikazom pojedinih programskih rješenja. U dodatku A se nalaze sheme sklopova, a u dodatku B se nalazi nacrt za pleksi-staklo.

2. Sklopovska podrška

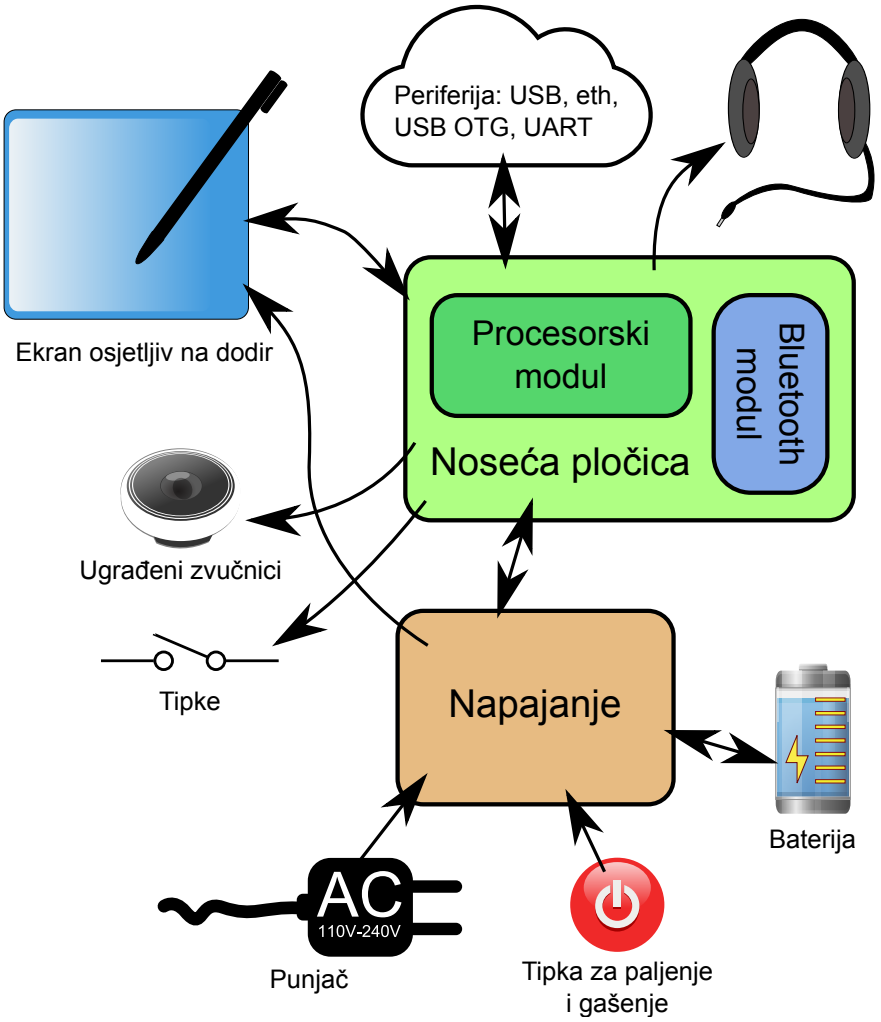
Sklopovska podrška je modularna i sastoji se od nekoliko komponenti:

- procesorski modul,
- noseća pločica,
- napajanje,
- baterija,
- LCD s panelom osjetljivim na dodir,
- pojačala sa zvučnicima,
- pločica s tipkama.

Blok-dijagram povezanosti komponenti je prikazan na slici 2.1.

Za procesorski i Bluetooth modul su odabrani komercijalno dostupni gotovi moduli jer bi njihova konstrukcija izlazila iz okvira ovog diplomskog rada, međutim noseća pločica koja mehanički i električki povezuje ostale module, te modul za napajanje, pojačala sa zvučnicima i pločica s tipkama su namjenski razvijeni za ovaj uređaj.

Odabir operacijskog sustava je na neki način određivao i odabir procesorskog modula koji u većini slučajeva dolazi s predinstaliranim operacijskim sustavom.



Slika 2.1: Blok-dijagram komponenti

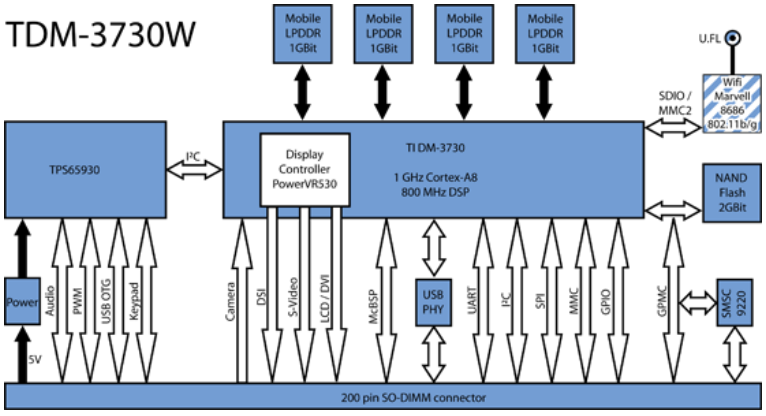
2.1. Procesorski modul TDM-3730

Kao osnova ovog računala je korišten procesorski modul TDM-3730 firme Tech-Nexion. Osnova samog modula je Texas Instrumentsov procesor DM3730 [3] iz OMAP3 serije s pomoćnim čipom TPS65930[1] [8]. Na modulu se nalaze i NAND *flash* i LPDDR RAM memorije te modul za bežični Internet i kontroler za *ethernet*. Blok dijagram procesorskog modula je prikazan na slici 2.2, a fizički izgled modula je prikazan na slici 2.3.

Procesorski modul je moguće priključiti na noseću pločicu preko standardnog SO-DIMM konektora. Preko konektora idu svi podatkovni signali i napajanja. Iz modula je moguće dobiti i napajanje od 1.8V.

Pomoćni čip TPS65930 između ostaloga ima i analogni ulaz koji je na modulu

[1] za napajanje, USB, zvuk i tipkovnicu



Slika 2.2: Blok dijagram procesorskog modula TDM-3730

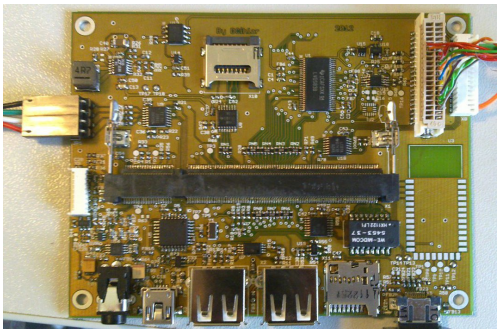
Slika 2.3: Fizički izgled procesorskog modula TDM-3730

spojen na otporničko dijelilo između reguliranih 4.2V i 0V. Dijelilo nije dokumentirano i smetalo je pri mjerenju otpora za prepoznavanje priključenog kabla te su stoga oba otpornika otklonjena.

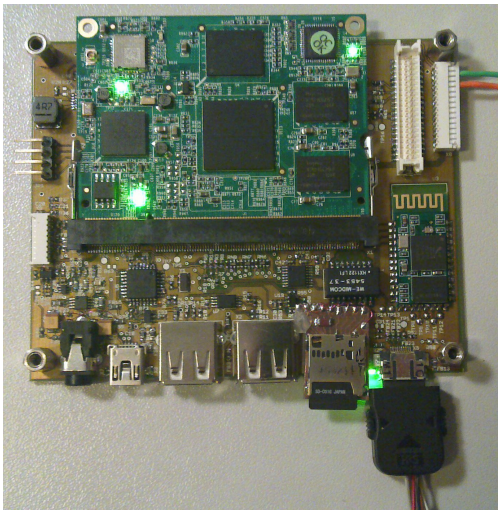
2.2. Noseća pločica

Noseća pločica (engl. *baseboard*) na slici 2.4 je središnji dio cijelog sustava na kojem se nalazi procesorski modul TDM-3730, Bluetooth modul, izvor napajanja od 3.3V i razne periferne jedinice. Sheme se nalaze u dodatku A.2.

Izvor napajanja od 3.3V je prekidačkog *buck-boost* tipa što znači da može spuštati i podizati napon. Takav tip izvora napajanja je potreban jer se 3.3V nalazi između minimalnog i maksimalnog napona baterije pa nije dovoljno samo podizanje ili samo spuštanje napona. Za ovu svrhu je odabran čip LTC3536 (U12 u dodatku A.20).



(a) Bez procesorskog i Bluetooth modula



(b) S procesorskim i Bluetooth modulom

Slika 2.4: Noseća pločica

Čip SN75LVDS83 (U1 u dodatku A.11) pretvara paralelni prijenos podataka prema LCD-ekranu u serijski diferencijalni prijenos (engl. *LVDS – low-voltage differential signaling*). Prednost ovog čipa je ugrađeni pretvornik logičkih naponskih razina koji omogućuje direktno spajanje na procesorski modul.

Čip za očitanje otporničkog panela osjetljivog na dodir TSC2046 također ima prednost ugrađenog pretvornika logičkih naponskih razina (U9 u odjeljku A.9). Osim očitanja otporničkog panela, TSC2046 omogućuje i mjerenje stanja baterije. Mjerenje stanja baterije je vrlo efikasno riješeno tako da otporničko dijelilo nije konstantno razapeto na napajanje nego je pozitivni dio stalno spojen, a uzemljuje se samo kada je potrebno izmjeriti napon baterije. Na taj način otporničko dijelilo ne troši nepotrebno struju dok se ne koristi.

Na nosećoj pločici se nalazi i USB-razdjelnik (engl. *USB-hub*) pune brzine (engl. *full speed*) koji omogućuje proširenje ovog sustava preko dva priključka USB (u dodatku A.25). Postojeću programsku podršku za priključak USB visoke brzine (engl. *high speed*) nije bilo moguće iskoristiti, a programska podrška za USB priključak pune brzine nije u potpunosti implementirana pa se ovaj sklop još ne koristi.

Osim dva priključka USB na pločici se nalazi i priključak USB-OTG (engl. *OTG – On-The-Go*) koji omogućuje povezivanje sustava s računalom (u dodatkuA.24). Tranzistor U13 (u dodatku A.23) omogućuje punjenje iz priključka USB-OTG ukoliko je zatraženo signalom $\overline{\texttt{OTG\_CHRG\_E}}$, odnosno ukliko računalo može dati dovoljno veliku struju potrebnu za punjenje (sporo ili brzo) i rad cijelog sustava.

Za otkrivanje prisustva slušalica ili podatkovnog kabla te prekidanje procesora u

oba slučaja osmišljena su dva posebna sklopa s uspoređivačem analognih signala. U dodatku A.16 je prikazan sklop za prekidanje procesora prilikom priključivanja podatkovnog kabla. Podatkovni kabl omogućuje mrežni (engl. *ethernet*) i serijski priključak. Kako bi bilo moguće prepoznati tip priključka uveden je izvod za identifikaciju. Unutar samog priključka na strani kabla identifikacijski izvod mora biti spojen na otpornik. Vrijednost otpornika tako određuje o kojem tipu kabla se radi. Ukoliko je taj otpor beskonačan, kabl nije priključen i izlaz iz uspoređivača je logička jedinica. Priključivanjem otpornika manjeg od 1MΩ izlaz iz uspoređivača prelazi u logičku nulu. Kako bi mjerenje otpornika bilo moguće, identifikacijski izvod je spojen na ulaz A/D pretvornika na procesorskom modulu.

Priključak za slušalice nije namijenjen samo za zvučne signale. Četvrti kontakt je spojen na izlaz kompozitne slike (engl. *composite video*). Kako bi se čip TPS65930 manje opteretio, između čipa i slušalica i je ubačeno pojačalo snage (u dodatku A.14).

Sklop za otkrivanje prisustva slušalica je prikazan u dodatku A.13. Ovaj sklop nije pouzdan zbog velike RC konstante. Naime, prilikom isključivanja slušalica potrebno je puno vremena da se izlazni elektroliti napune i analogni uspoređivač prebaci stanje na izlazu. Nije bilo moguće nabaviti priključak s četiri kontakta i sklopkom za otkrivanje prisustva utikača pa je stoga napravljen ovaj pokušaj elektroničkog otkrivača prisustva slušalica.

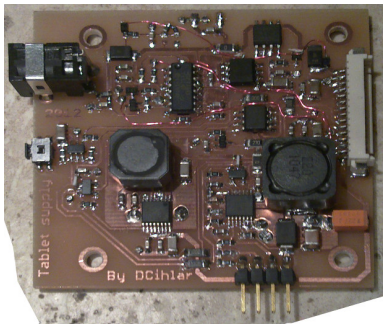
Procesor DM3730 ima digitalne izlaze koji rade isključivo na 1.8V pa je na nekim mjestima bilo potrebno postaviti pretvornike naponskih razina (odjeljci A.22, A.27 i A.28). Kao što je već prije spomenuto, u nekoliko je slučajeva bilo moguće koristiti čipove s ugrađenim pretvornikom naponskih razina.

2.3. Napajanje

Posebna pažnja je posvećena oblikovanju sklopa za napajanje kako bi uređaj radio pouzdano i sa što manje gubitaka.

Visoka efikasnost sustava za napajanje je nužna kako bi baterija što duže trajala. Stoga su sva napajanja prekidačkog tipa te ih je moguće isključiti dok nisu potrebna.

U dodatku A.1 se nalaze sheme sklopova za napajanje, a na slici 2.5 je prikazana fizička implementacija sklopa.



Slika 2.5: Implementacija sklopa za napajanje

2.3.1. Automatsko određivanje izvora

Sustav za napajanje mora samostalno (bez intervencije programske podrške) odrediti što će koristiti kao glavni izvor napajanja. U protivnom punjenje baterije ne bi bilo moguće dok je sustav ugašen ili bi se sustav ugasio prilikom isključivanja punjača ako programska podrška nije dovoljno brza.

Baterija se koristi samo ako niti jedan od punjača nije prisutan. U nastavku su tri tipa podržanog izvora, poredani prema prioritetu:

- brzi punjač (iz gradske mreže): omogućuje struju punjenja od 1A,
- spori punjač (iz USB OTG-a): omogućuje struju punjenja od 0.5A koju je programski moguće povećati na 1A,
- baterija.

Diode D1 i D2 dovode vanjsko napajanje u jedan vod koji se kasnije koristi kao vanjski izvor napajanja. Kroz diode može poteći struja do 2A, a diode D4 i D5 imaju ulogu logičkih ILI vrata za napajanje. One će provesti ukoliko je prisutan vanjski izvor napajanja, a ukoliko niti jedan vanjski izvor nije prisutan logičku nulu će dovesti otpornik R3. Zbog relativno velike reverzne struje u diodama D1 i D2 otpornik R3 mora biti maksimalno 1kΩ.

Kada je prisutan vanjski izvor napajanja tranzistori Q4 i Q5 su aktivni, a u suprotnom je aktivan tranzistor Q1. Aktivacijom tranzistora Q1 odabire se baterija kao izvor napajanja, a aktivacijom tranzistora Q5 odabire se vanjski izvor napajanja. Tranzistor Q4 je u ulozi logičkog sklopa NE i osigurava da Q1 i Q5 nikada nisu aktivni istovremeno.

2.3.2. Punjač baterije

Za punjenje baterije je zadužen čip MCP73861 (U2). Otpornik R6 određuje pretpostavljenu struju punjenja od 0.5A, a tranzistori Q2 i Q3 su u sklopu logičkih vrata ILI koja kada su aktivna pritežu ulaz PROG na 0V i tako mijenjaju struju punjenja na 1A. Q3 se aktivira priključivanjem vanjskog punjača, a Q2 je moguće programski aktivirati ako USB priključak na kojeg je uređaj spojen može dati 2A.

Dijagnostički izlazi $\overline{\texttt{CHRG\_STAT1}}$ i $\overline{\texttt{CHRG\_STAT2}}$ su spojeni na prekidne ulaze procesora kako ne bi trebalo radnim čekanjem dohvaćati stanje punjača. Dijagnostički izlaz $\overline{\texttt{CHRG\_STAT1}}$ označava da li se baterija puni (0V ako se puni) ili je puna (1Hz), a izlaz $\overline{\texttt{CHRG\_STAT2}}$ označava grešku.

2.3.3. Baterija

Korištena je Li-Ion baterija od 8000mAh originalno namijenjena za igraću konzolu Nitendo 3DS. Iz baterije je izbačen punjač, ali je ostavljen nadzorni čip.

Nazivni napon baterije je 3.6V i pri potrošnji od 4W (npr. 2W za procesorski modul i 2W za pozadinsko osvjetljenje) sustav bi u idealnom slučaju mogao neprestano raditi oko 7h što je dovoljno za uređaj ovog tipa.

Uz optimizaciju potrošnje unutar procesorskog modula vijek baterije bi se mogao značajno povećati.

2.3.4. Pozadinsko osvjetljenje

Tradicionalno se (i ponekad još uvijek) za pozadinsko osvjetljenje u većim LCD-ekranima koristi CCFL (engl. *cold cathode fluorescent lamp*) cijev koja je relativno neefikasna, dok se u svim LCD-ekranima namijenjenima za mobilne uređaje koriste bijele svjetleće diode.

Za ovaj uređaj je korišten LCD-ekran iz prijenosnog računala sa CCFL cijevi koja je zamijenjena s 30 jakih bijelih dioda. Umjesto 5W za pozadinsko osvjetljenje je sada potrebno samo 2W uz nešto malo slabije maksimalno osvjetljenje.

Pozadinsko osvjetljenje izvedeno svjetlećim diodama zahtjeva i poseban izvor napajanja. U prijenosnim se računalima za CCFL cijev koristi poseban modul koji napon baterije pretvara u nekoliko kilovolti i takav nikako nije pogodan za svjetleće diode. Svjetleće diode zahtijevaju strujni izvor, a napon ovisi o broju svjetlećih dioda spojenih u seriju te o padu napona na pojedinoj diodi.

Za ovaj ekran korištena su tri paralelna snopa od deset bijelih svjetlećih dioda spojenih u seriju.

Kao regulator odabran je čip TPS61500 (U3). Taj čip je jedan od rijetkih koji mogu uz niski napon iz baterije (min. 3V) dati relativno visoku snagu potrebnu za pozadinsko osvjetljenje. TPS61500 omogućuje i prigušenje intenziteta osvjetljenja pulsno širinskom modulacijom (engl. *PWM – pulse width modulation*).

Slijedilo (engl. *buffer*) U6 gasi pozadinsko osvjetljenje dok je sustav ugašen.

Slika 2.6: LED zamjena za CCFL

2.3.5. Napajanje za USB *host* priključke

Na nosećoj pločici nalaze se dva USB *host* priključka i odlučeno je da svaki mora biti u mogućnosti dati 500mA. Stoga je potrebno napajanje koje iz minimalno 3V može dati 5V pri potrošnji od 1A.

Kao regulator odabran je čip TPS61032 (U1) zbog svoje vrlo visoke efikasnosti (preko 90%) i mogućnosti rada pri niskim ulaznim naponima. Uz TPS61032 je potreban minimalan broj dodatnih komponenti (samo zavojnica i kondenzatori). Povratna veza je ugrađena u samom čipu i fiksno određena za izlaz od 5V.

Slijedilo U5 ima istu ulogu kao i U6 – za gašenje regulatora dok je sustav ugašen.

2.3.6. Sustav za paljenje i gašenje

Tranzistor Q6 spaja i odspaja napajanje noseće pločice u ovisnosti o signalu $\overline{\texttt{TABLET\_ON}}$, a tranzistor Q7 prazni kondenzatore na napajanju prilikom gašenja kako bi se sustav prije i pravilnije ugasio. RC član R23 i C12 malo odgađaju aktivaciju tranzistora Q6 kako bi Q7 manje smetao prilikom prijelaza stanja signala $\overline{\texttt{TABLET\_ON}}$ u aktivno (manji je problem ako smeta prilikom gašenja).

Sustav za paljenje i gašenje mora omogućiti korisniku prisilno paljenje i gašenje pomoću tipke, ali mora omogućiti i programsko gašenje. Dodatno, sustav za paljenje i gašenje mora ugasiti sustav ako se jezgra nije upalila. Na taj se način iskorištava komparator unutar TPS65930 (na procesorskom modulu) koji onemogućuje paljenje ukoliko je napon na bateriji prenizak.

Sve navedene zahtjeve omogućuje sklop sa samo dva logička sklopa NE (U4A i U4B) i jednim slijedilom (U18 na nosećoj pločici u dodatku A.2).

Uz pretpostavku da je sustav ugašen ($\overline{\texttt{TABLET\_ON}}$ = 5V), napajanje od 1.8V je ugašeno pa slijedilo ne može postaviti signal POWEROFF na 0V. Između signala $\overline{\texttt{TABLET\_ON}}$ i POWEROFF nalaze se dva logička sklopa NE i moguće je uočiti da je to stabilno stanje.

Tipka S5 je odspojena pa se kondenzator C11 preko otpornika R18 napuni na 5V. Pritiskom i držanjem tipke S5 kondenzator C11 se prazni preko otpornika R17, ali tokom nekog vremena će ulaz sklopa U4B biti prebačen na logičku jedinicu čime će izlaz biti prebačen u logičku nulu. Aktivacija signala $\overline{\texttt{TABLET\_ON}}$ aktivira tranzistor Q6 te noseća pločica dobije napajanje i počne generirati napon od 1.8V. Kako su izvodi od neinicijalizirani podsustava procesora u stanju visoke impedancije, otpornik R70 definira nisku razinu na ulazu slijedila koje zatim postavlja POWEROFF signal u logičku nulu i samim time sustav za paljenje i gašenje dolazi u drugo stabilno stanje kada je sustav upaljen. Sada je moguće otpustiti tipku i sustav će ostati upaljen.

Ukoliko napajanje od 1.8V nije proradilo (napon baterije je prenizak) signal POWEROFF ne bi nikad prešao u logičku nulu i napajanje bi se ugasilo nakon što bi se ispraznio kondenzator C11.

Prilikom određivanja vrijednosti komponenti C11, R17 i R18 bitno je da je $R18 \gg R17$ te da vremenska konstanta $C11 \cdot R17$ bude dovoljno duga kako bi se stigla uspostaviti povratna veza. Nusprodukt je velika vremenska konstanta $C11 \cdot R18$ koja određuje minimalno vrijeme između pokušaja paljenja i gašenja.

Programsko gašenje je moguće postavljanjem izvoda 148 u logičku jedinicu. Izvod mora imati paralelno spojen kondenzator koji zadržava logičku jedinicu jer prilikom gašenja izvod prelazi u stanje visoke impedancije i ako ne bi bilo kondenzatora, signal POWEROFF bi se deaktivirao i započeo ponovno paljenje sustava.

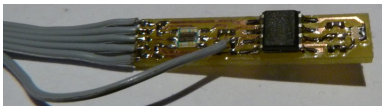
2.4. Pločica sa svjetlećom RGB-diodom

Pločica sa svjetlećom RGB (engl. *red, green, blue*) diodom i senzorom osvjetljenja je samostalan računalni sustav koji se brine samo za svjetlosne obavijesti (npr. obavijest o pristigloj poruci ili praznoj bateriji i sl.) i mjerenje intenziteta osvjetljenja okoline. Shema se nalazi u dodatku A.3.

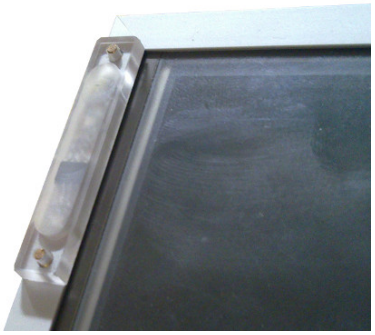
Pločica se nalazi u gornjem lijevom kutu uređaja smještena unutar kućišta od pleksi stakla kao što je prikazano na slici 2.7b. Na taj je način zaštićena od vanjskih utjecaja, ali svjetlost i dalje može nesmetano prolaziti. Kako bi stala na sam rub širine 8mm

pločica je morala biti vrlo uska.

Na pločici (prikazana na slici 2.7a) se nalazi svjetlosni senzor APDS9303 (lijevo) i mikrokontroler ATtiny25 [7] (na sredini) koji upravlja sa svjetlećom RGB-diodom (desno).



(a) Nezaštićena s privremenom žicom za programiranje



(b) Pričvršćena i zaštićena

Slika 2.7: Pločica sa svjetlećom RGB-diodom

2.4.1. Komunikacija s pločicom

Pločica je spojena na napajanje od 3.3V, sabirnicu I^{2}C te jedan prekidni signal. Svjetlosni senzor i mikrokontroler su spojeni na sabirnicu I^{2}C, a prekidni signal je spojen samo na svjetlosni senzor.

Svjetlosni senzor se nalazi na adresi $29_{(16)}$, a mikrokontroler na adresi $12_{(16)}$. Protokol za komunikaciju sa svjetlosnim senzorom je opisan u njegovoj dokumentaciji [5], a protokol za komunikaciju s mikrokontrolerom i upravljanje RGB-diodom je posebno osmišljen i opisan u nastavku ovog poglavlja.

Protokol za upravljanje RGB-diodom preko I^{2}C sabirnice nije dizajniran u stilu uobičajenih I^{2}C protokola gdje se prvo šalje adresa podatka koji se zapisuje pa nakon toga podaci nego se šalju samo podaci. Redni broj okteta određuje koji se podatak zapisuje kao što je prikazano u tablici 2.1.

Nije uvijek potrebno poslati paket sa svim oktetima. Moguće je poslati paket koji sadrži samo sve tri komponente boje (prva tri okteta) nakon čega slijedi oznaka kraja komunikacije na I^{2}C sabirnici. Moguće je i poslati prvih šest okteta kako bi se odredila vremena, ali bez izmjene načina rada.

Tablica 2.1: Značenje okteta pri komunikaciji s mikrokontrolerom

Red. br.	Bitovi	Značenje
1	3..0	Intenzitet crvene boje
2	3..0	Intenzitet zelene boje
3	3..0	Intenzitet plave boje
4	7..0	Vremenski razmak između paljenja ($N \times 60$ms)
5	7..0	Trajanje upaljenog stanja ($N \times 15$ms)
6	7..6	Broj uzastopnih paljenja ($N - 1$)
	5..0	Brzina postavljanja željenog intenziteta ($N - 1$)
7	1..0	Način rada: **0:** Spavanje **1:** Konstantno svijetlo **2:** Bljeskanje

2.4.2. Načini rada

Mikrokontroler za upravljanje RGB-diodom podržava tri načina rada: spavanje, konstantno svijetlo i bljeskanje.

Za vrijeme spavanja sve tri komponente RGB-diode su ugašene i mikrokontroler je u stanju spavanja bez *watchdoga* pa je potrošnja na cijeloj pločici 13μA.

U načinu rada konstantnog svijetla na RGB-diodi je prikazana odabrana boja. Potrošnja u ovom načinu rada je konstantna i maksimalno iznosi 8.6mA. Ovaj način rada zbog velike potrošnje nije praktično primjenjiv, ali koristan je prilikom odabira željene boje.

Posljednji način rada je bljeskanje. U tom se načinu rada RGB-dioda povremeno pali. Vrijeme ugašenog stanja i trajanje svjetlećeg stanja je moguće podesiti po želji.

Paljenje RGB-diode u načinu rada bljeskanje je postepeno, tj. glatko. Pri najkraćem trajanju upaljenog stanja postepeno paljenje traje 75ms (15 stanja intenziteta $\times$ 15ms). Brzinu postepenog paljenja je također moguće podesiti. Najbrža postavka je 15 i u tom je slučaju paljenje RGB-diode trenutno. Trenutno paljenje RGB-diode je moguće koristiti ukoliko je potrebno grubo (digitalno) bljeskanje.

Potrošnja za vrijeme čekanja u ugašenom stanju u načinu rada bljeskanje je nešto veća nego u načinu rada spavanja i iznosi 18μA. Razlog tomu je rad *watchdog* sklopa koji mjeri proteklo vrijeme.

2.4.3. Upravljanje svjetlećom RGB-diodom

Zbog veće uštede energije mikrokontroler je morao imati sklopovsku podršku za sabirnicu I^{2}C kako bi ga početak komunikacije mogao probuditi iz stanja spavanja. Zbog dimenzijskih zahtjeva je mikrokontroler morao biti i vrlo malen pa je stoga ATtiny25 bio najbolji izbor[2].

Određivanje intenziteta svjetlosti pojedine diode je moguće pulsno širinskom modulacijom (u daljnjem tekstu PWM). Nažalost, odabrani mikrokontroler ima samo dva sklopovska generatora PWM signala od kojih je jedan multipleksiran s izvodom za I^{2}C sabirnicu. Iz tog je razloga PWM generator implementiran programski prema odsječku 2.1.

Odsječak 2.1: Prekidna rutina vremenskog sklopa 0

```
ISR(TIM0_COMPA_vect) {
    static uint8_t brojac;
    brojac = (brojac + 1) & 0x1F; /* 0..31 */

    if (brojac < zelena_komponenta) upali_zelenu();
    else ugasi_zelenu();

    /* ... preostale komponente */
}
```

Kako bi se manje energije gubilo na predotporima, postavljeni su predotpori manjih vrijednosti, odnosno, maksimalna struja kroz svjetleću diodu je iznad maksimalne prosječne, ali ispod maksimalne vršne. Radni ciklus (engl. *duty cycle*) sada mora biti maksimalno 50% i iz tog razloga statična varijabla brojač broji do 31 i uspoređuje se s komponentama koje mogu biti maksimalno 15.

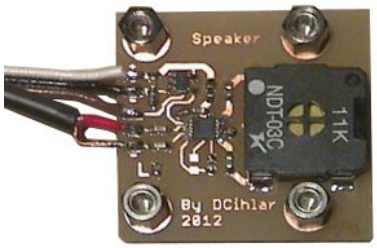
Vremenski sklop mora pokretati prekidnu rutinu dovoljno brzo da ljudsko oko ne primijeti treperenje, ali opet ne prebrzo kako bi procesor stigao obraditi prekid i ostale poslove. Uz pretpostavku da je 50 perioda u sekundi dovoljno brzo i poznato trajanje jedne periode od 32 prekida, vremenski sklop mora prekidati procesor 1600 puta u sekundi, odnosno svakih 625μs.

Unutar 625μs i pri taktu od 8MHz AVR jezgra može izvršiti 5000 jednociklusnih instrukcija i to se pokazalo dovoljnim za prekidnu rutinu s programskim PWM-om i za ostale zadatke (komunikaciju i bljeskanje).

[2] ATtiny dolazi i u manjem pakiranju od SO8, ali nije bio dobavljiv

2.5. Pojačala za ugrađene zvučnike

Za potpuni multimedijski doživljaj ovaj uređaj mora imati ugrađene zvučnike koji su dovoljno glasni, a opet dovoljno mali. Zvučnici moraju imati i pripadajuće pojačalo klase D zbog što veće efikasnosti. Pojačalo mora raditi i na niskom naponu. Sve te zahtjeve zadovoljava odabrano pojačalo LM4667. U dodatku A.4 se nalazi shema sklopa, a na slici 2.8 je prikazana fizička implementacija.



Slika 2.8: Jedno od dva pojačala

2.6. Ekran osjetljiv na dodir

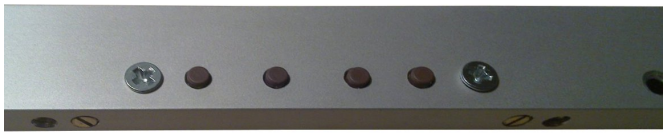
Ekran osjetljiv na dodir sastoji se od dvije odvojene komponente: LCD-ekrana i otporničkog panela osjetljvog na dodir.

LCD-ekran je LP150X08-TLC1, razlučivosti 1024×768 piksela, dijagonale 15 inča i 16 bita dubine boja. Dimenzije LCD-ekrana zbog svoje veličine su odredile dimenzije cijelog uređaja.

2.7. Pločica s tipkama

Za paljenje i gašenje ekrana, prebacivanje na početni ekran te brzi pristup kontroli glasnoće je ugrađena pločica s četiri tipke u matričnom spoju. Tipke očitava sklopovski čip TPS65930. Smještena je u prednjem lijevom dijelu kućišta kao što je prikazano na slikama 2.9a i 2.9b, a fizički izgled pločice je prikazan na slici 2.9c. Shema se nalazi u dodatku A.5.

(a) Smještaj



(b) Prednja strana

(c) Stražnja strana

Slika 2.9: Pločica s tipkama

3. Osnovna programska podrška

Procesorski modul TDM-3730 dolazi s instaliranom osnovnom programskom podrškom (*bootloaderom* i operacijskim sustavom Linux 2.6.37). *Bootloader* je izveden u nekoliko faza [2]. Prilikom paljenja se pokreće izvršni kôd iz ROM-a unutar samog procesora koji izvodi najosnovnije inicijalizacije te pokreće X-Loader. X-Loader određuje osnovne uloge izvoda procesora (postavi multipleksiranje izvoda), obavi dodatne inicijalizacije te pokreće U-Boot. U-Boot je zadnja faza *bootloadera* u kojem se izvodi posljednje postavljanje multipleksiranja izvoda i inicijalizacija sklopovlja koje omogućuje daljnje učitavanje sustava (npr. *ethernet*).

Konačno, operacijski sustav Linux prilikom svog pokretanja mora inicijalizirati svo sklopovlje, pa čak i ono koje ne koristi kako bi tom sklopovlju isključio takt i omogućio nesmetan prelazak načina rada u spavanje. Osim toga, Linux prilikom pokretanja mora prijaviti sve uređaje koje nije moguće automatski detektirati kako bi ih kasnije bilo moguće koristiti. Linux također može podešavati multipleksiranje izvoda, ali nije potrebno jer je sve već postavljeno unutar U-Boota.

X-Loader nije mijenjan i ostao je isti koji je dobiven s procesorskim modulom. U U-Bootu je bilo potrebno promijeniti postavke multipleksiranja izvoda kako bi se omogućilo sučelje SPI te neki GPIO izvodi.

3.1. U-Boot

Datoteke vezane uz platformu su odvojene od originalnih TechNexionovih. Mapa `board/technexion/tdm3730` je kopirana u `board/technexion/tacna` i u toj kopiji su napravljene potrebne izmjene. Kopirana je i datoteka `include/configs/tdm3730.h` u `include/configs/tacna.h`. Konačno, obnovljena je datoteka `boards.cfg` kako bi bilo moguće prevesti U-Boot za novu platformu Tacna[1].

[1] Tacna je kodno ime ovog uređaja

U datoteci `board/technexion/tacna/tdm3730.h` se nalazi popis svih izvoda i odabranih uloga. Primjer koji umjesto UART-a na izvode dovodi GPIO je prikazan u odsječku 3.1. Prvi parametar makroa `MUX_VAL` je naziv izvoda, a drugi parametar određuje svojstvo i ulogu izvoda.

Odsječak 3.1: Primjer postavki multipleksiranja u U-Bootu

```
/*UART2 x 4 PIN*/                                                    \
MUX_VAL(CP(MCBSP3_DX),    (IDIS | PTD | DIS | M4))                  \
        /*(4)GPIO_140 -> (1)UART2_CTS*/                             \
MUX_VAL(CP(MCBSP3_DR),    (IDIS | PTD | DIS | M4))                  \
        /*(4)GPIO_141 -> (1)UART2_RTS (ear_mute)*/                  \
MUX_VAL(CP(MCBSP3_CLKX), (IDIS | PTD | DIS | M4))                   \
        /*(0)McBSP3_CLKX -> (4)GPIO_142 -> (1)UART2_TX */           \
MUX_VAL(CP(MCBSP3_FSX),   (IDIS | PTD | DIS | M4))                  \
        /*(0)McBSP3_FSX -> (4)GPIO_143 -> (1)UART2_RX (spk_mute)*/ \
```

Svojstvo izvoda određuje da li je omogućeno čitanje (`IEN`) ili nije (`IDIS`) te da li su omogućeni *pull-up* (`PTU | EN`) ili *pull-down* (`PTD | EN`) otpornici ili su onemogućeni (`DIS`).

Uloga izvoda može biti `M1`, `M2`, `M3` ili `M4`. Moguće uloge su navedene u dokumentaciji procesora, ali su navedene i unutar nekih komentara kako bi bilo lakše odrediti ulogu.

3.2. Inicijalizacija sklopovlja u Linuxu

U mapi `arch/arm/mach-omap2/` se nalaze svi izvorni kodovi vezani uz arhitekturu OMAP2 na dalje, a sve datoteke unutar te mape koje počinju s `board-` su vezane uz specifičnu platformu, tj. pločicu (engl. *board*).

Ovdje je platforma `tdm3730` također kopirana u platformu `tacna`. Kopirana je datoteka `board-taotdm.c` u datoteku `board-tacna.c` te su izmijenjene datoteke `Kconfig` i `Makefile`. Konačno, u konfiguraciji prevođenja operacijskog sustava Linux onemogućeno je prevođenje za platformu `tdm3730`, a omogućeno je prevođenje za novu platformu `tacna`.

U nastavku ovog poglavlja su opisane inicijalizacije koje nije implementirala firma TechNexion.

3.2.1. Matrična tipkovnica

Pomoćni čip TPS65930 (kompatibilan s TWL4030) na procesorskom modulu ima ugrađeno sklopovsko skeniranje matrične tipkovnice. Sklopovsko skeniranje, za razliku od programskog skeniranja, omogućuje veću uštedu energije jer procesor može spavati dok minimalno sklopovlje skenira za pritiske i prekida (tj. budi) procesor pri svakom novom pritisku.

U odsječku 3.2 su prikazani opisnici potrebni za konfiguraciju matrične tipkovnice. U opisniku `tacna_kp_twl4030_data` se nalazi broj redaka i stupaca tipkovnice te pokazivač na opisnik uloga pojedinih tipaka.

Ugrađena tipkovnica ima četiri tipke spojene u matricu 2×2. Predviđene uloge su im za podešavanje glasnoće, povratak na glavni ekran te paljenje i gašenje, odnosno, zaključavanje ekrana. U listi `board_keymap` su određene uloge prema lokaciji tipke u matrici.

Uvjet `CONFIG_KEYBOARD_TWL4030` osigurava da je prilikom prevođenja uključen modul za tipkovnicu.

Odsječak 3.2: Opisnici tipkovnice

```
#ifdef CONFIG_KEYBOARD_TWL4030
static uint32_t board_keymap[] = {
    KEY(0, 0, KEY_VOLUMEUP),
    KEY(0, 1, KEY_VOLUMEDOWN),
    KEY(1, 0, KEY_HOME),
    KEY(1, 1, KEY_POWER),
};

static struct matrix_keymap_data board_map_data = {
    .keymap         = board_keymap,
    .keymap_size    = ARRAY_SIZE(board_keymap),
};

static struct twl4030_keypad_data tacna_kp_twl4030_data = {
    .keymap_data    = &board_map_data,
    .rows           = 2,
    .cols           = 2,
    .rep            = 1,
};
#endif
```

U strukturu za inicijalizaciju čipa TWL4030 (`taotdm_twldata`) je bilo potrebno dodati odsječak 3.3 kako bi opisnik tipkovnice bio poznat prilikom inicijalizacije čipa TWL4030.

Odsječak 3.3: Priključivanje tipkovnice

```
#if defined(CONFIG_KEYBOARD_TWL4030)
    .keypad         = &tacna_kp_twl4030_data,
#endif
```

3.2.2. Kartica μSD

Ovaj sustav je predviđen da ima dvije memorijske kartice. Jednu za potrebe sustava, a drugu za potrebe korisnika. Memorijsku karticu za potrebe korisnika je moguće mijenjati jer je lako dostupna na rubu uređaja dok memorijsku karticu sustava nije moguće mijenjati jer je fizički nedostupna. Osim toga memorijsku karticu za potrebe sustava nije niti dopušteno mijenjati jer bez nje sustav ne može raditi.

Memorijska kartica za potrebe korisnika je spojena na sabirnicu HSMMC koja je i namijenjena za memorijske kartice. Druga sabirnica HSMMC nije bila dostupna pa je memorijska kartica za potrebe sustava spojena na zasebnu sabirnicu SPI. Za tu je sabirnicu bilo potrebno u U-Bootu postaviti ispravno multipleksiranje izvoda.

Opisnik sabirnice SPI je prikazan u odsječku 3.4. U opisniku sabirnice SPI potrebno je navesti sve uređaje priključene na sabirnicu jer ih nije moguće automatski otkriti. Na ovu sabirnicu SPI je spojena samo kartica MMC te je stoga u opisniku samo jedan element koji povezuje upravljački modul `mmc_spi` s uređajem na ovoj sabirnici SPI koji se odaziva na nultoj adresi.

Upravljačkom modulu `mmc_spi` je potrebno dati dodatni opisnik u kojem je navedeno koje napone kartica mora podržavati.

Sabirnicu SPI je potrebno prijaviti sustavu pozivom na funkciju `spi_register_board_info` unutar funkcije `tdm3730_init` kao što je prikazano u odsječku 3.5. Funkciji se kao argument zadaje lista sa svim uređajima na sabirnici.

Odsječak 3.4: Opisnik SPI sabirnice i modula za memorijsku karticu

```
#if defined(CONFIG_MMC_SPI) || defined(CONFIG_MMC_SPI_MODULE)
#include <linux/spi/mmc_spi.h>
static struct mmc_spi_platform_data mmc_spi_data = {
    .ocr_mask = MMC_VDD_32_33 | MMC_VDD_33_34,
};
#endif

static struct spi_board_info spi3_board_info[] __initdata = {
#if defined(CONFIG_MMC_SPI) || defined(CONFIG_MMC_SPI_MODULE)
    {
        .modalias = "mmc_spi",
        .max_speed_hz = 50000000,
        .bus_num = 3,
        .chip_select = 0,   /* nulta adresa */
        .platform_data = &mmc_spi_data,
        .mode = SPI_MODE_0,
    },
#endif
};
```

Odsječak 3.5: Inicijalizacija SPI sabirnice

```
spi_register_board_info(spi3_board_info,
                        ARRAY_SIZE(spi3_board_info));
```

3.2.3. Ekran osjetljiv na dodir

Na nosećoj pločici se nalazi isti čip (ADS7846) za mjerenje otporničkog panela (engl. *resistive touch panel*) kao i na demonstracijskoj pločici firme TechNexion tako da nije bilo potrebno puno izmjena.

U opisniku čipa ADS7846 je bilo potrebno zapisati podatke poput otpora panela na X-osi, maksimalan dopušteni otpor pritiska te postavke za istitravanje.

3.2.4. Sabirnica I^2C

Na zasebnu sabirnicu I^2C su spojeni akcelerometar, senzor osvjetljenja i mikrokontroler za upravljanje RGB-diodom. Na sabirnici I^2C također nije moguće automatski prepoznati priključene uređaje pa je potrebno napraviti listu sa svim priključenim ure-

đajima kao što je prikazano u isječku 3.6. Lista povezuje adresu uređaja s upravljačkim modulom.

Odsječak 3.6: Lista uređaja priključenih na sabirnicu I^2C

```
static struct i2c_board_info __initdata taotdm_i2c_2_boardinfo[] = {
    {
        .type = "mma8450",
        .addr = 0x1C,
        .irq = OMAP_GPIO_IRQ(18),
    },
    {
        I2C_BOARD_INFO("tinyRGB", 0x12),
    },
    {
        .type = "apds9303",
        .addr = 0x29,
        .irq = OMAP_GPIO_IRQ(13),
    }
};
```

Sabirnicu I^2C je potrebno prijaviti sustavu pozivom na funkciju `omap_register_i2c_bus` unutar funkcije `taotdm_i2c_init` kao što je prikazano u odsječku 3.7. Funkciji se kao argument zadaje redni broj sabirnice, maksimalni takt u kHz te lista sa svim uređajima na sabirnici.

Odsječak 3.7: Inicijalizacija sabirnice I^2C

```
omap_register_i2c_bus(2, 100, taotdm_i2c_2_boardinfo,
                              ARRAY_SIZE(taotdm_i2c_2_boardinfo));
```

3.2.5. Ulazno/izlazni izvodi opće namjene

Uz inicijalizaciju sklopovlja je potrebno prijaviti ulazno/izlazne izvode opće namjene (engl. *GPIO – general purpose input/output*) te ih postaviti na početnu vrijednost. Popis svih izvoda GPIO i njihovih lokacija na modulu i procesoru se nalazi u tablici 3.1.

U odsječku 3.8 su prikazani primjeri inicijalizacije ulaznog i izvoda. Funkcija `gpio_request` prijavljuje zadani izvod sa zadanim opisom, a funkcije `gpio_direction_input` i `gpio_direction_output` određuju smjer izvoda. Funk-

Tablica 3.1: GPIO izvodi

Izvod na modulu	Izvod na procesoru	Smjer	Funkcija
47	143	izl.	Gasi zvučnike
49	141	izl.	Gasi slušalice
65	137	ul.	Slušalice prisutne
68	12	izl.	Omogući punjenje preko USB-a
69	19	ul.	Podatkovni kabl prisutan
72	13	ul.	Prekid iz senzora osvjetljenja
74	18	ul.	1. prekid iz akcelerometra
90	20	ul.	2. prekid iz akcelerometra
142	126	izl.	Omogući akcelerometar
144	98	izl.	Omogući 5V napajanje za USB
146	97	izl.	Omogući brzo punjenje
148	101	izl.	Gasi cijeli sustav
150	103	ul.	Greška prilikom punjenja
152	102	ul.	Status punjenja

cija `gpio_direction_output` osim broja izvoda na procesoru kao drugi argument prima početno stanje izvoda.

Funkcijom `gpio_export` se omogućuje pristup izvodima GPIO iz korisničkog adresnog prostora preko mape iz sustavskog datotečnog sustava *sysfs*. Mapa se nalazi na lokaciji `/sys/class/gpio/` i sadrži datoteku `value` preko koje je moguće očitavati i mijenjati stanje izvoda.

Ukoliko je potrebno, za ulazni izvod GPIO je moguće zatražiti prekid funkcijom `request_irq`. Identifikacijski broj prekida je moguće odrediti makroom `OMAP_GPIO_IRQ`.

Odsječak 3.8: Inicijalizacija ulaznog i izlaznog izvoda

```
/* ulazni izvod */
if (!gpio_request(137, "EAR_PLUGED") &&
    !gpio_direction_input(137)) {
        gpio_export(137, 0);
        printk("ear_pluged ok\n");
}

/* izlazni izvod */
if (!gpio_request(126, "ACC_EN") &&
    !gpio_direction_output(126, 1)) {
        gpio_export(126, 0);
        printk("acc_en ok\n");
}
```

3.2.6. Funkcija za gašenje sustava

Globalna varijabla `pm_power_off` je pokazivač na funkciju za gašenje cijelog sustava. Pretpostavljena vrijednost te varijable je `NULL` što znači da se sustav ne može ugasiti. U tom će slučaju jezgra operacijskog sustava samo ostati u beskonačnoj petlji.

Ovaj sustav podržava automatsko gašenje kao što je opisano u poglavlju 2.3.6 i potrebno je izmijeniti globalnu varijablu `pm_power_off` ukoliko prijava izvoda GPIO prođe uspješno. Funkcija čiju se adresu zapisuje u globalnu varijablu `pm_power_off` se nalazi u odsječku 3.9. Jedini zadatak te funkcije je da na izvod za gašenje sustava zapiše logičku jedinicu.

Odsječak 3.9: Funkcija za gašenje sustava

```
static void tacna_power_off(void) {
    gpio_set_value(101, 1);
}
```

3.2.7. Mjerenje stanja baterije

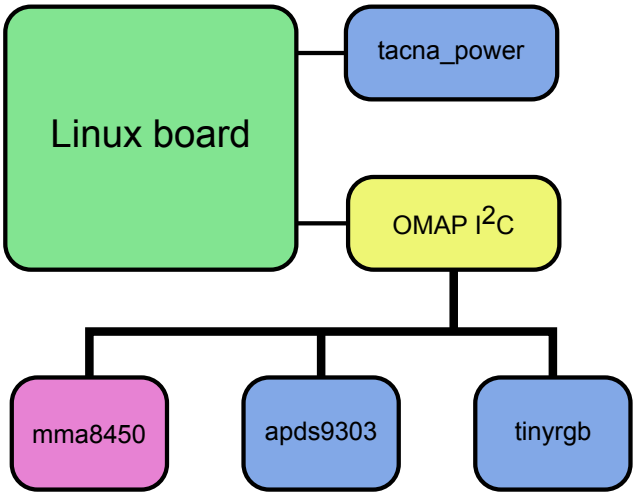
Dodatni analogni ulaz čipa ADS7846 je spojen na napon dobiven izravno iz baterije kako bi bilo moguće izmjeriti njeno trenutno stanje. No nažalost, upravljački modul za čip ADS7846 nema funkciju za očitanje analogne vrijednosti koju je moguće direktno pozvati unutar jezgre operacijskog sustava. Stoga je potrebno koristiti ugrađeno mjere-

nje baterije na čipu TPS65930 dok se ne nadogradi upravljački modul za čip ADS7846. Čip TPS65930 se nalazi iza linearnog regulatora na kojem pad napona uvelike ovisi o potrošnji sustava tako da ta mjerenja nisu pouzdana.

Kako bi bilo moguće koristiti pretvornik A/D iz čipa TPS65930 potrebno je uključiti upravljački modul `twl4030_madc` i dodati opisnik u listu svih priključenih uređaja `omap3_tdm3730_devices`.

4. Periferna programska podrška

Za potpunu funkcionalnost uređaja je bilo potrebno implementirati tri upravljačka modula [9] [10] i prebaciti podršku za akcelerometar iz novije jezgre operacijskog sustava Linux. Dijagram modula je prikazan na slici 4.1. Plavom bojom su označeni implementirani moduli.



Slika 4.1: Dijagram modula

Također, bilo je potrebno korisniku uređaja omogućiti odabir da li će zvuk biti preusmjeren na slušalice ili na ugrađene zvučnike. Napravljeni su i demonstracijski programi za akcelerometar, senzor intenziteta svjetlosti i za upravljanje RGB-diodom.

Na nosećoj pločici se nalazi Bluetooth modul koji je spojen na jedno od serijskih sučelja. Sva programska podrška niske razine je već ugrađena u operacijski sustav Linux i bilo je potrebno samo povezati serijsko sučelje s Bluetoothom u init-skripti.

4.1. Upravljački moduli

4.1.1. Virtualni datotečni podsustav za upravljanje i nadzor

Virtualni datotečni podsustav za upravljanje i nadzor (*sysfs*) je tipično mapiran (engl. *mount*) u mapi `/sys`, a služi za komunikaciju procesa u korisničkom adresnom prostoru s uređajima, odnosno upravljačkim modulima uređaja [6]. Puno je jednostavniji za korištenje od *procfs*-a jer nije potrebno brinuti se za otvaranje i zatvaranje datoteka, nego je samo potrebno pročitati odnosno zapisati podatak u međuspremnik.

U mapi `/sys` su uređaji grupirani na razne načine simboličkim poveznicama. Primjerice, u mapi `/sys/class` su grupirani po svom tipu, a u `/sys/bus` se nalaze sve sabirnice.

Svaki uređaj prilikom prijavljivanja dobiva vlastitu mapu u kojoj ima svojstva u obliku datoteka koje je moguće čitati i/ili pisati. Prozivanjem (engl. *polling*) svojstva je moguće i čekati na novu vrijednost.

Grupa specifičnih svojstava uređaja se definiraju prilikom prijavljivanja uređaja funkcijom `sysfs_create_group` kao što je prikazano u odsječku 4.1. Prilikom odjave uređaja grupu sa svojstvima je potrebno izbrisati funkcijom `sysfs_remove_group`. Obje funkcije kao prvi argument primaju lokaciju grupe u *sysfsu*, a kao drugi opisnik grupe svojstava.

Odsječak 4.2 prikazuje opisnik grupe `uredjaj_attr_grp`, listu svih atributa `uredjaj_attrs`, te opisnik atributa "svojstvo". U opisniku grupe nužno je navesti NULL-terminiranu listu sa svim atributima, a po potrebi je moguće zadati i ime grupe, odnosno podmape u kojoj će se svojstva nalaziti. Opisnici svojstva se jednostavno definiraju makroom `DEVICE_ATTR` čiji prvi argument određuje ime svojstva, drugi prava pristupa, a treći i četvrti redom pokazivače funkcija za očitanje i izmjenu svojstva.

Konačno, funkcije za očitanje i izmjenu svojstva su prikazane u odsječku 4.3. Na temelju argumenta `dev` je moguće doznati o kojem se točno uređaju radi (ako je više istih uređaja), a argument `attr` omogućuje da više različitih svojstava ima iste funkcije. Argument `buf` je međuspremnik.

Uz pretpostavku da se primjer smjesti u mapu `/sys/class/power/test`, svojstvu "svojstvo" je moguće pristupiti kroz datoteku `/sys/class/power/svojstvo`. Stanje varijable `podatak` je sada moguće pročitati naredbom `cat`, a naredbom `echo` je moguće mijenjati stanje varijable (ali samo ukoliko se zapiše cijeli broj).

Odsječak 4.1: Primjer definiranja i brisanja grupe svojstva upravljačkog modula

```
/* funkcija za prijavu uredjaja */
static __devinit int uredjaj_probe(struct platform_device *pdev) {
    int err;
    /* ... inicijalizacija */

    /* stvaranje svojstava */
    err = sysfs_create_group(&pdev->dev.kobj, &uredjaj_attr_grp);
    if (err < 0) {
        dev_err(&pdef->dev, "Nije moguce inicijalizirati sysfs!\n");
        goto err_sysfs;
    }

    /* ... finalizacija */
    return 0;

    /* ... oporavak od greske */
err_sysfs:
    return err;
}

/* funkcija za odjavu uredjaja */
static __devexit int uredjaj_remove(struct platform_device *pdev) {
    /* brisanje svojstava */
    sysfs_remove_group(&pdev->dev.kobj, &uredjaj_attr_grp);
    /* ... deinicijalizacija */
}
```

Odsječak 4.2: Primjer opisnika grupe svojstava i opisnika svojstva

```
/* opisnik svojstva "svojstvo", moze mu pristupiti bilo tko */
static DEVICE_ATTR(svojstvo,
                   S_IRUGO | S_IWUGO,
                   svojstvo_show, svojstvo_store);

/* lista sa svojstvima grupe */
static struct attribute *uredjaj_attrs[] = {
    &dev_attr_svojstvo.attr,
    /* ostala svojstva */
    NULL
};

/* opisnik grupe svojstva */
static struct attribute_group uredjaj_attr_grp = {
    .attrs = uredjaj_attrs,
    /* ostali elementi strukture su opcionalni */
};
```

Odsječak 4.3: Primjer funkcija za očitanje i izmjenu svojstava

```
static unsigned long podatak;

/* funkcija za prikaz svojstva */
static ssize_t svojstvo_show(struct device *dev,
                             struct device_attribute *attr,
                             char *buf) {
        return sprintf(buf, "%lu\n", podatak);
}

/* funkcija za izmjenu svojstva */
static ssize_t svojstvo_store(struct device *dev,
                              struct device_attribute *attr,
                              const char *buf, size_t len) {
    if (strict_strtoul(buf, 0, &podatak))
        return -EINVAL;
    return len;
}
```

Prozivanje događaja datotečnog podustava za upravljanje i nadzor

Prozivanje *sysfs* događaja omogućuje blokiranje procesa do promjene svojstva uređaja. Upravljački modul mora obavijestiti *sysfs* o izmjeni svojstva funkcijom `sysfs_notify`.

U odsječku 4.4 je prikazan primjer prozivanja nad datotekom svojstva `/sys/class/power/test/svojstvo`. Svojstva nisu tokovi podataka pa je potrebno prije svakog čitanja pokazivač vratiti na početak datoteke.

Odsječak 4.4: Primjer prozivanja *sysfs* događaja

```
int main() {
    struct pollfd pfd;
    char buf[MAX_BUF_LEN];

    pfd.fd = open("/sys/class/power/test/svojstvo", O_RDONLY);
    if (pfd.fd < 0) err(1, "open");

    pfd.events = POLLPRI;   /* prioritetni dogadjaj */
    for (;;) {
        if (poll(&pfd, 1, -1) < 0)
            err(1, "poll");
        lseek(pfd.fd, 0, SEEK_SET);
        if (read(pfd.fd, buf, MAX_BUF_LEN) < 0)
            err(1, "read");

        /* ... obrada medjuspremnika buf */
    }
}
```

4.1.2. Modul za akcelerometar

Upravljački modul `mma8450` za akcelerometar nije podržan u Linuxu 2.6.37 nego tek u 3.1. Uz minimalne izmjene je kopiran u 2.6.37 i dodan u pripadajuće datoteke `Makefile` i `Kconfig`. Modul ima nekoliko grešaka i nedostataka koje bi trebalo ispraviti. Primjerice, ne postoji podrška za signal za paljenje i gašenje akcelerometra. Iz tog razloga je prilikom inicijalizacije sustava bilo potrebno ručno izvod GPIO za paljenje akcelerometra fiksno postaviti na logičku jedinicu. Osim toga, modul vraća vrijednosti bez predznaka umjesto s predznakom tako da je u korisničkim aplikacijama potrebno dodavati predznak prema odsječku 4.5.

Odsječak 4.5: Ispravljanje predznaka u rezultatu mjerenja akcelerometra

```
struct input_event ev;

/* citanje vrijednosti s akcelerometra */
if ((rd = read(fd, &ev, sizeof(ev))) < sizeof(ev))
    perror_exit("read()");
if (ev.type == EV_ABS) {
    /* ispravljanje predznaka */
    if (ev.value & 0x800) {
        ev.value |= ~0x7ff;
    }
    /* ... */
}
```

4.1.3. Modul za praćenje stanja baterije

Modul za praćenje stanja baterije, kao što mu samo ime kaže, omogućuje praćenje stanja baterije korisniku uređaja. Nalazi se u `drivers/power/tacna_power.c`.

Zbog greške u sklopovlju (ispravljena u shemama u dodatku) nije se moguće pouzdati u stanja signala punjača. Umjesto toga se isključivo na temelju napona napajanja zaključuje da li je priključen vanjski izvor napajanja ili ne. Maksimalni izmjereni napon na bateriji bez punjača je 4V, tako da je pri naponu većem od 4.2V moguće zaključiti da je priključen vanjski izvor napajanja.

Modul prijavljuje sustavu izvor napajanja tipa baterija prema odsječku 4.6. Globalno polje `tacna_bat_props` sadrži popis svih svojstava koje je moguće dohvatiti preko funkcije `tacna_bat_get_prop`. Implementirana je podrška za najosnovnija svojstva potrebna za podsustav APM:

POWER_SUPPLY_PROP_STATUS: vraća status baterije (ili se puni ili se prazni),

POWER_SUPPLY_PROP_VOLTAGE_NOW: trenutni napon baterije (mjeri se pomoću funkcije `twl4030_get_madc_conversion(12)` dostupne iz modula `twl4030_madc`),

POWER_SUPPLY_PROP_VOLTAGE_MAX: vraća maksimalni napon na bateriji koji iznosi 4.2V,

POWER_SUPPLY_PROP_VOLTAGE_MIN: vraća minimalni napon na bateriji koji iznosi 2.7V,

POWER_SUPPLY_PROP_PRESENT: uvijek vraća istinu jer je baterija uvijek prisutna, tj. nije izmjenljiva,

POWER_SUPPLY_PROP_TECHNOLOGY: uvijek vraća tip baterije Li-ion.

Odsječak 4.6: Prijava izvora tipa baterija

```
bat->name = "Kruh␣baterija";
bat->type = POWER_SUPPLY_TYPE_BATTERY;
bat->properties = tacna_bat_props;
bat->num_properties = ARRAY_SIZE(tacna_bat_props);
bat->get_property = tacna_bat_get_prop;
ret = power_supply_register(&pdev->dev, bat);
if (ret)
    goto err_psureg1;
```

Iz korisničkog adresnog prostora komponentama ovog modula je moguće pristupiti preko datoteka u mapi `/sys/class/power_supply`.

Podsustav APM

Podsustav APM (engl. *Advanced Power Manager*) je apstrakcijski sloj za upravljanje potrošnjom namijenjen za IBM-kompatibilna računala [1]. U ovom se sustavu koristi samo kako bi grafičko sučelje GPE moglo očitati stanje baterije jer ono ne podržava drugi način. Bilo ga je potrebno uključiti u postavkama jezgre operacijskog sustava.

Podsustav APM periodički pretražuje priključene baterije i dojavljuje njihovo stanje korisničkoj aplikaciji.

4.1.4. Modul za mjerenje intenziteta svjetlosti

Upravljački modul za senzor APDS9303 ne postoji niti u jednoj verziji jezgre, te ga je bilo potrebno implementirati.

APDS9303 podržava dva načina mjerenja: automatsko periodično mjerenje i ručno mjerenje. Ručno mjerenje je potrebno samo za iznimne slučajeve i stoga ono nije implementirano. Nije implementirana niti mogućnost izmjene periode mjerenja, ali je predviđena za daljnji razvoj.

Programi unutar korisničkog adresnog prostora mogu očitavati vrijednost sa senzora putem datoteke `lux0_input` unutar mape `/sys/class/i2c-adapter/`

`i2c-2/2-0029/`. Prozivanjem datoteke je moguće dohvaćati svaku novu vrijednost.

Funkcija za očitanje svojstva `lux0_input` blokira izvođenje procesa do prekida sa senzora. U prekidu se senzoru šalje potvrda prekida i čitaju se izmjerene vrijednosti `CH0` i `CH1` (vidljiva i infracrvena svjetlost).

Koristi se dretveni prekid što znači da prekid pokreće dretvu koja se izvodi kad dođe na red. Dretvu, za razliku od obične prekidne funkcije, je moguće blokirati prilikom komunikacije preko I^2C sabirnice.

Iz obje vrijednosti se izračunava osvjetljenje u mili luksima prema funkciji u odsječku 4.7. Funkcija koristi isključivo cjelobrojne operacije (engl. *fixed point*). Operacija potenciranja $x^{1.4}$ je implementirana pomoću pregledne tablice (engl. *lookup table*). Formule za izračun intenziteta svjetlosti u luksima u uputama čipa APDS9303 koriste decimalne faktore. Faktori su pomnoženi s 100000 kako bi se dobili cijeli brojevi. Rezultat je stoga u zapisu 100×mLuks kojeg se prilikom prikaza dijeli sa 100 kako bi se dobio prikaz u mili luksima sa dvije decimalne znamenke.

Odsječak 4.7: Izračun osvjetljenja u mili luksima

```
// /100 za  mili lukse
static u32 apds9303_calc_lux(u16 ch0, u16 ch1) {
    static const u16 expx_14[16] =
        {2, 6, 11, 17, 23, 30, 37, 45,
         53, 61, 70, 79, 88, 98, 107, 118};
    const u32 ratio = (ch0 > 0) ? ch1 * 1024 / ch0 : 2048;

    if (ratio <= (u32)(0.52 * 1024)) {
        return 3150 * ch0 - 46 * ch0 * expx_14[(ratio >> 6) & 0x07];
    } else if (ratio <= (u32)(0.65 * 1024)) {
        return 2290 * ch0 - 2910 * ch1;
    } else if (ratio <= (u32)(0.80 * 1024)) {
        return 1570 * ch0 - 1800 * ch1;
    } else if (ratio <= (u32)(1.30 * 1024)) {
        return 338 * ch0 - 260 * ch1;
    } else {
        return 0;
    }
}
```

4.1.5. Modul za upravljanje RGB-diodom

Ovaj modul je zadužen za komunikaciju s mikrokontrolerom koji upravlja RGB-diodom. Modul se sastoji od tri sučelja za svjetleće diode (opisano u poglavlju 4.1.5) te od svojstva za upravljanje načinom rada. Modul je smješten u mapi `drivers/leds/`, datoteci `leds-tinyrgb.c`.

Sučelje za svjetleće diode

Sučelje za svjetleće diode omogućuje vrlo jednostavnu implementaciju upravljačkih modula za svjetleće diode [4]. Posebno jednostavan slučaj su svjetleće diode spojene na izvod GPIO. U tom slučaju nije potrebno raditi posebni upravljački modul.

Pomoću okidača (engl. *trigger*) jezgra operacijskog sustava može sama prema potrebi paliti i gasiti svjetleće diode, ali moguće je i imati svjetleće diode namijenjene ručnom, odnosno korisničkom upravljanju.

Iz korisničkog adresnog prostora svim svjetlećim diodama je moguće pristupiti unutar mape `/sys/class/leds`. Svaka svjetleća dioda ima svoju mapu unutar koje se nalaze datoteke `brightness` (za određivanje intenziteta osvjetljenja) i `max_brightness` (za dohvaćanje maksimalnog intenziteta).

U odsječku 4.8 je prikazana pojednostavljena prijava svjetleće diode jezgri operacijskog sustava. Postavljanje opisnika svjetleće diode kao globalnu varijablu nije poželjno i ovdje je postavljeno zbog jednostavnosti. U praktičnom rješenju opisnik se mora nalaziti u podacima vezanim uz uređaj.

Opisnik svjetleće diode `cdev` ima dva bitna elementa koja je potrebno inicijalizirati: naziv diode oblika "ime:boja:uloga" i funkciju za postavljanje intenziteta. Po potrebi je moguće odrediti maksimalan intenzitet (pretpostavljena vrijednost je 255) i još razne druge elemente koji nisu bitni za ovaj uređaj.

Kada okidač ili korisnik zatraži izmjenu intenziteta poziva se *callback* funkcija `ledica_set_brightness`. Primjer funkcije je prikazan u odsječku 4.9. Jedini zadatak te funkcije je slanje naredbe sklopovlju za izmjenu intenziteta. Ukoliko komunikacija sa sklopovljem može blokirati, potrebno je koristiti pozadinske zadatke i samo funkcijom `schedule_work` pokrenuti zadatak za izmjenu intenziteta. U tom pozadinskom zadatku se onda izvodi komunikacija sa sklopovljem.

Odsječak 4.8: Prijava sučelja za svjetleću diodu

```
struct led_classdev cdev;   /* opisnik svjetlece diode */

/* funkcija za prijavu uredjaja */
static __devinit int uredjaj_probe(struct platform_device *pdev) {
    int err;
    /* ... inicijalizacija */

    /* prijava sucelja za svjetlecu diodu */
    cdev.name = "s-dioda:zelena";
    cdev.brightness_set = ledica_set_brightness;
    err = led_classdev_register(&pdev->dev, &cdev);
    if (err < 0) {
        dev_err(&pdef->dev, "Nije moguce prijaviti svj. diodu!\n");
        goto err_led;
    }

    /* ... finalizacija */
    return 0;

    /* ... oporavak od greske */
err_led:
    return err;
}

/* funkcija za odjavu uredjaja */
static __devexit int uredjaj_remove(struct platform_device *pdev) {
    /* brisanje svojstava */
    sysfs_remove_group(&pdev->dev.kobj, &uredjaj_attr_grp);
    /* ... deinicijalizacija */
}
```

Odsječak 4.9: Funkcija za izmjenu intenziteta svjetlosti diode

```
static void ledica_set_brightness(struct led_classdev *cdev,
                                  enum led_brightness brightness) {
    /* posalje sklopovlju naredbu za izmjenu intenziteta */
    postavi_intenzitet(brightness);
}
```

Upravljanje RGB-diodom iz korisničkog adresnog prostora

Iz korisničkog adresnog prostora svakoj komponenti RGB-diode je moguće pristupiti preko datoteke `/sys/class/leds/tinyRGB:boja/brightness` ("boja" može biti "r" za crvenu, "g" za zelenu ili "b" za plavu komponentu).

Sam uređaj u *sysfsu* se nalazi u mapi `/sys/class/i2c-adapter/i2c-2/2-0012/`. Definirana su sljedeća svojstva:

mode: način rada bljeskanja; može biti "off" (ugašeno), "static" (konstantno svijetlo) i "fade" (automatsko bljeskanje),

period: razmak između paljenja,

duration: trajanje jednog bljeska,

fade_speed: brzina postepenog paljenja,

num_fade: broj uzastopnih bljeskova unutar jednog paljenja.

Komunikacija s mikrokontrolerom

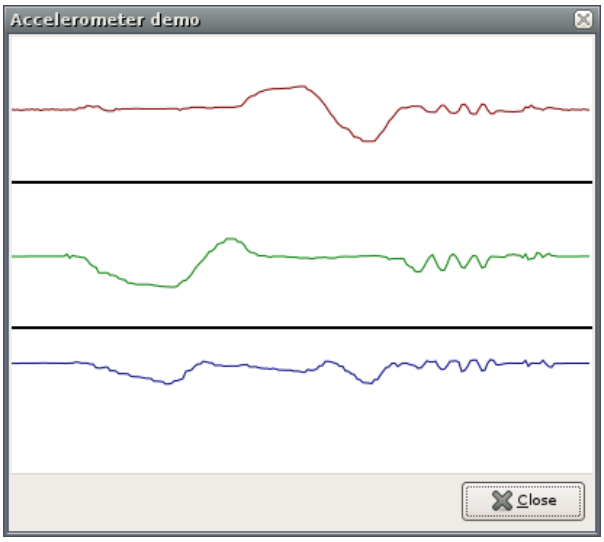
Kao što je već prije spomenuto, komunikacija s mikrokontrolerom odvija se preko sabirnice I^2C. Za slanje naredbi se koristi funkcija `i2c_master_send`. Ta funkcija omogućuje sirovo slanje podataka koje je potrebno zbog neuobičajenog protokola.

Implementirana je posebna funkcija za slanje naredbi `trgb_write`. Funkcija slaže paket s naredbama i unutar kritičnog odsječka poziva gore navedenu funkciju za slanje naredbi kako ne bi došlo do kolizije. Ovisno o parametru `level` šalju se naredbe boje, boje i vremena ili boje, vremena i način rada.

4.2. Demonstracijski programi

4.2.1. Demonstracija akcelerometra

Na slici 4.2 je prikazano grafičko sučelje za demonstraciju funkcionalnosti akcelerometra. Sučelje je ostvareno pomoću biblioteke GTK+ i prozivanjem datoteke `/dev/input/event2`. Upravo ta je datoteka u kojoj je moguće dohvatiti događaje s akcelerometra.



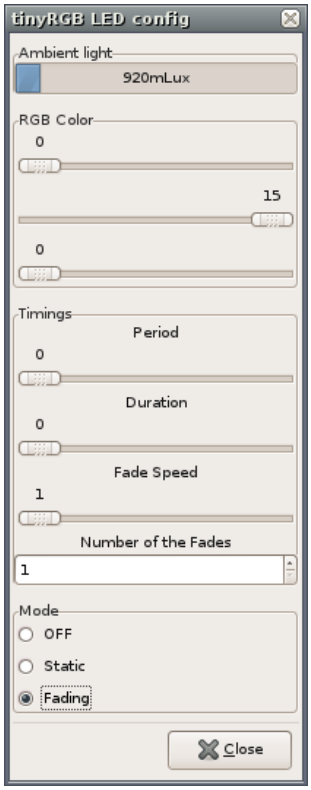
Slika 4.2: Grafičko sučelje za očitanje akcelerometra

4.2.2. Demonstracija senzora intenziteta svjetlosti i RGB-diode

Na slici 4.3 je prikazano grafičko sučelje za demonstraciju očitanja senzora intenziteta svjetlosti i upravljanja RGB-diodom. Sučelje je ostvareno pomoću biblioteke GTK+.

Trenutno stanje intenziteta svjetlosti okoline dohvaća se prozivanjem datoteke `/sys/class/i2c-adapter/i2c-2/2-0029/lux0_input`.

Sučelje omogućuje izmjenu svih svojstava mikrokontrolera za upravljanje RGB-diodom, a prilikom pokretanja dohvaćaju se prethodna stanja svih svojstava.



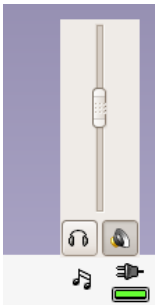
Slika 4.3: Grafičko sučelje za očitanje intenziteta svjetlosti i upravljanje RGB-diodom

4.3. Preusmjeravanje zvuka

Zbog greške u sklopovlju za otkrivanje prisustva slušalica bilo je potrebno napraviti aplikaciju koja korisniku omogućuje ručno usmjeravanje zvuka na slušalice i/ili ugrađene zvučnike.

Napravljena je GTK+ aplikacija koja se prilikom pokretanja smjesti u područje obavijesti (engl. *system tray*). Pritiskom na ikonu se pojavljuje prozorčić za podešavanje glasnoće s dva dodatna preklopna gumba (engl. *toggle button*) za paljenje i gašenje pojedinog izlaza (prikazan na slici 4.4).

Aplikacija omogućuje i onemogućuje izlaze upravljanjem izvoda GPIO preko *sysfs*-a, a glasnoću podešava pomoću biblioteke Alsa.



Slika 4.4: Prozorčić za podešavanje glasnoće i preusmjeravanje zvuka

4.4. Sučelje za podešavanje tipaka

Sučelje za podešavanje tipaka je dio grafičkog sučelja GPE i potrebno je bilo samo napisati datoteku s postavkama `/etc/gpe/key-layout`.

Datoteka s postavkama je u zapisu INI. U odsječku 4.10 je prikazan primjer postavki za jednu tipku.

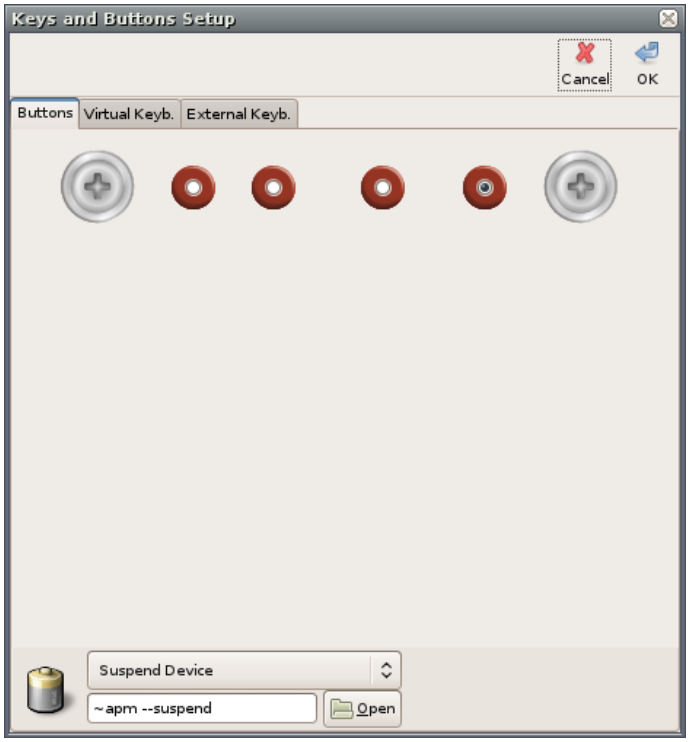
Odsječak 4.10: Primjer postavki za jednu tipku

```
[Global]
image = /usr/share/pixmaps/tacna-keys.png
offset_x = 30
offset_y = 10

[Power]
title = Power Button
symbol = gpe-config-apm.png
key = XF86PowerDown
command = apm --suspend
xpos = 315
ypos = 30
```

U odjeljku `Global` se određuje pozadinska slika i njene koordinate, a u ostalim odjeljcima čiji su nazivi ime tipke se određuje redom naslov, slika akcije, naziv događaja u X11, naredba koja će se pokrenuti pritiskom na tipku i konačno lokacija tipke na slici.

Na slici 4.5 je prikazan izgled sučelja za podešavanje tipaka, a položaj tipaka na samom kućištu je prikazan na slici 2.9a. Slika unutar sučelja za podešavanje je nacrtana prema fizičkim tipkama i vijcima.



Slika 4.5: Sučelje za podešavanje tipaka

4.5. Bluetooth

Korišteni Bluetooth modul je bio isporučen u načinu rada prilagodbe Bluetootha serijskom sučelju za mikrokontrolere. Takav način rada je previše ograničen za potrebe ručnog računala. Stoga je Bluetooth modul prebačen u BCSP (engl. *BlueCore Serial Protocol*) način rada. U tom načinu Linux može izravno slati HCI (engl. *host/controller interface*) naredbe i koristiti Bluetooth modul za bilo koji protokol, a ne samo za serijsko sučelje preko Bluetootha.

Za pokretanje Bluetooth podsustava je napravljena posebna pokretačka skripta (engl. *init script*) pojednostavljeno prikazana u odsječku 4.11. Program `hciattach` povezuje serijsko sučelje s Bluetooth uređajem. Kao prvi argument prima putanju do datoteke s tokom podataka serijskog sučelja. Drugi argument je naziv protokola koji će se koristiti za komunikaciju s Bluetooth modulom, a treći argument određuje brzinu prijenosa podataka. Ukoliko komunikacija s Bluetooth modulom uspije, program `hciattach` stvara virtualni Bluetooth uređaj `hci0`.

Program `hciconfig` podešava i pokreće Bluetooth modul. Kao prvi argument se zadaje kojem Bluetooth uređaju se šalje naredba, a drugi argument je upravo naredba koju se šalje.

Odsječak 4.11: Bluetooth pokretačka skripta

```
#!/bin/sh

BT_UART=/dev/ttyO2

bt_start() {
    hciattach "$BT_UART" bcsp 115200
    hciconfig hci0 up
}

bt_stop() {
    hciconfig hci0 down
    killall hciattach
}

case "$1" in
  start)
    echo "Starting␣Bluetooth␣support"
    bt_start
  ;;
  stop)
    echo "Stopping␣Bluetooth␣support"
    bt_stop
  ;;
esac

exit 0
```

5. Mehanička konstrukcija

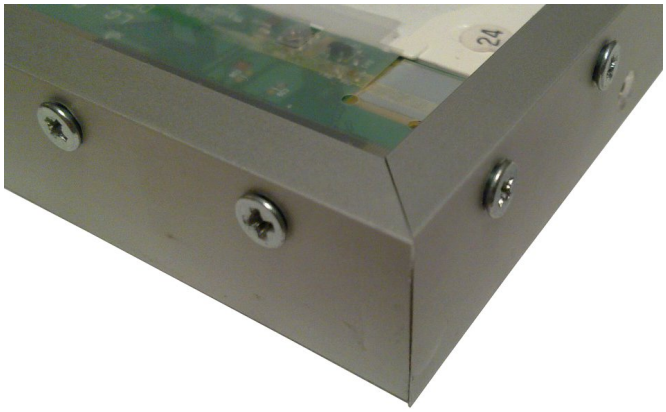
LCD-ekran s panelom osjetljivim na dodir je najviše utjecao na konstrukciju kućišta. Čitavo kućište je ustvari dimenzije ekrana s time da je deblje za oko 9 mm kako bi ispod ekrana bilo mjesta za elektroniku.

S obzirom da se radi o prototipu čija je osnovna svrha bila prikazati mogućnosti takvog računala, kućište je konstruirano tako da se što jednostavnije može izraditi. Bitno je bilo i da se sastoji od lako dostupnih materijala. Uobičajen izgled za ovaj tip uređaja nije bio prioritet.

Kućište se sastoji od gornje strane koja je određena panelom osjetljivim na dodir ispod kojeg je smješten LCD-ekran, aluminijskog okvira i zaštite s donje strane.

Rubovi kućišta koji nose priključke i tipke te spajaju gornju i donju plohu računala izrađeni su iz eloksiranog aluminija U-profila. Međusobno su povezani kutnicima od mesinga u kojima su urezani navoji za vijke koji drže čitavu konstrukciju (slika 5.1).

Donji dio kućišta koji štiti elektroniku je od pleksi-stakla kako bi unutrašnjost računala bila vidljiva. Nacrt je prikazan u dodatku B.



(a) Donja strana okvira

(b) Gornja strana okvira

(c) Unutarnja strana okvira

Slika 5.1: Konstrukcija okvira

6. Zaključak

Ovaj diplomski rad prikazuje izgradnju jednog prijenosnog računala u obliku funkcionalnog prototipa. U okviru programske podrške zasnovane na operacijskom sustavu Linux razvijeni su i dorađeni potrebni sustavski moduli te je razvijen i pokoji program, prvenstveno u svrhu ispitivanja ili prikaza mogućnosti računala.

Naravno, ovaj rad nije završio proizvodom koji se može odmah komercijalizirati. U tu svrhu bi trebalo uložiti znatne napore kao bi se doradila sklopovska rješenja, dizajn samog uređaja, mehanička konstrukcija te naravno doradila programska podrška za što bolji korisnički doživljaj te razvili ili iskoristili postojeći korisnički programi.

Zanimljivo je da je ukupna cijena utrošenog materijala za izradu prototipa usporediva sa cijenom komercijalnih uređaja istih mogućnosti.

Međutim, primjena ovdje opisane programske i sklopovske implementacije dokazala je da se relativno malim naporima u malom vremenu i za malo uloženih sredstava može razviti upotrebljivo prijenosno računalo.

LITERATURA

[1] *Advanced power management*. URL http://en.wikipedia.org/wiki/Advanced_power_management.

[2] *Bootloader Project*. URL http://omapedia.org/wiki/Bootloader_Project.

[3] *AM/DM37x Multimedia Device - Technical Reference Manual*. URL http://www.ti.com/lit/ug/sprugn4p/sprugn4p.pdf.

[4] *LED handling under Linux*. URL http://kernel.org/doc/Documentation/leds/leds-class.txt.

[5] *APDS9303*, 2009. URL http://www.avagotech.com/docs/AV02-2299EN.

[6] *The filesystem for exporting kernel objects*, 2010. URL http://kernel.org/doc/Documentation/filesystems/sysfs.txt.

[7] *ATtiny25/45/85*, 2011. URL http://www.atmel.com/Images/doc2586.pdf.

[8] *TPS65930/TPS65920 OMAP Power-Management and System Companion Devices - Technical Reference Manual*, 2011. URL http://www.ti.com/lit/ug/swcu052g/swcu052g.pdf.

[9] Davor Cihlar. *Upravljački modul za operacijski sustav Linux*. 2010.

[10] Davor Cihlar. *Prenamjena ugradbenog računalnog sustava*. 2011.

Izgradnja ručnog računala

Sažetak

Rad se sastoji od izvedbe prijenosnog računala *tablet* zajedno s prilagodbom operacijskog sustava temeljenog na jezgri Linux. Koristeći dobavljive komponente i vlastitu projektiranu pločicu, izvedeno je računalo s ekranom osjetljivim na dodir, baterijom, punjačem, akcelerometrom, svjetlosnim senzorom, zvučnicima i podrškom za slušalice. U radu se opisuju izvedene promjene jezgre Linux, opisuje se hardverska izvedba, te su navedene sheme sklopovskog dijela rada.

Ključne riječi: tablet, linux, ekran osjetljiv na dodir, akcelerometar, svjetlosni senzor, baterija, napajanje, otvoreni sustav, otvoreni kod, slobodna programska podrška, jezgra

Building a tablet computer

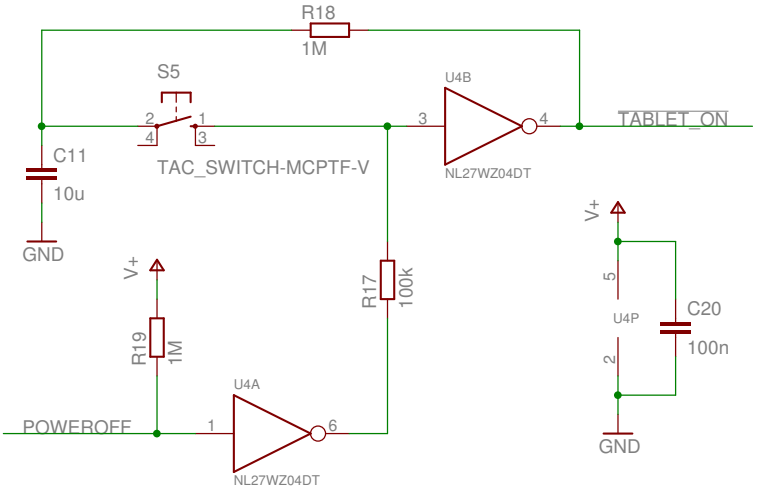
Abstract

This study consists of engineering a tablet computer and modification of an operating system based on the Linux kernel. Using off-the-shelf components as well as the self-engineered board, this computer contains a touchscreen, a battery, a charger, an accelerometer, a light sensor, as well as speakers, along with support for a headset. This study describes the modifications to the Linux kernel, the hardware, and it also contains the schematics for the hardware.

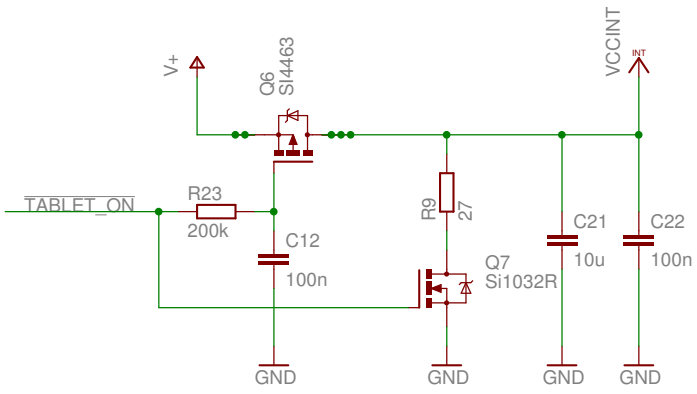
Keywords: tablet, linux, touchscreen, accelerometer, light sensor, battery, power supply, open systems, open source, free software, kernel

Dodatak A
Sheme sklopova

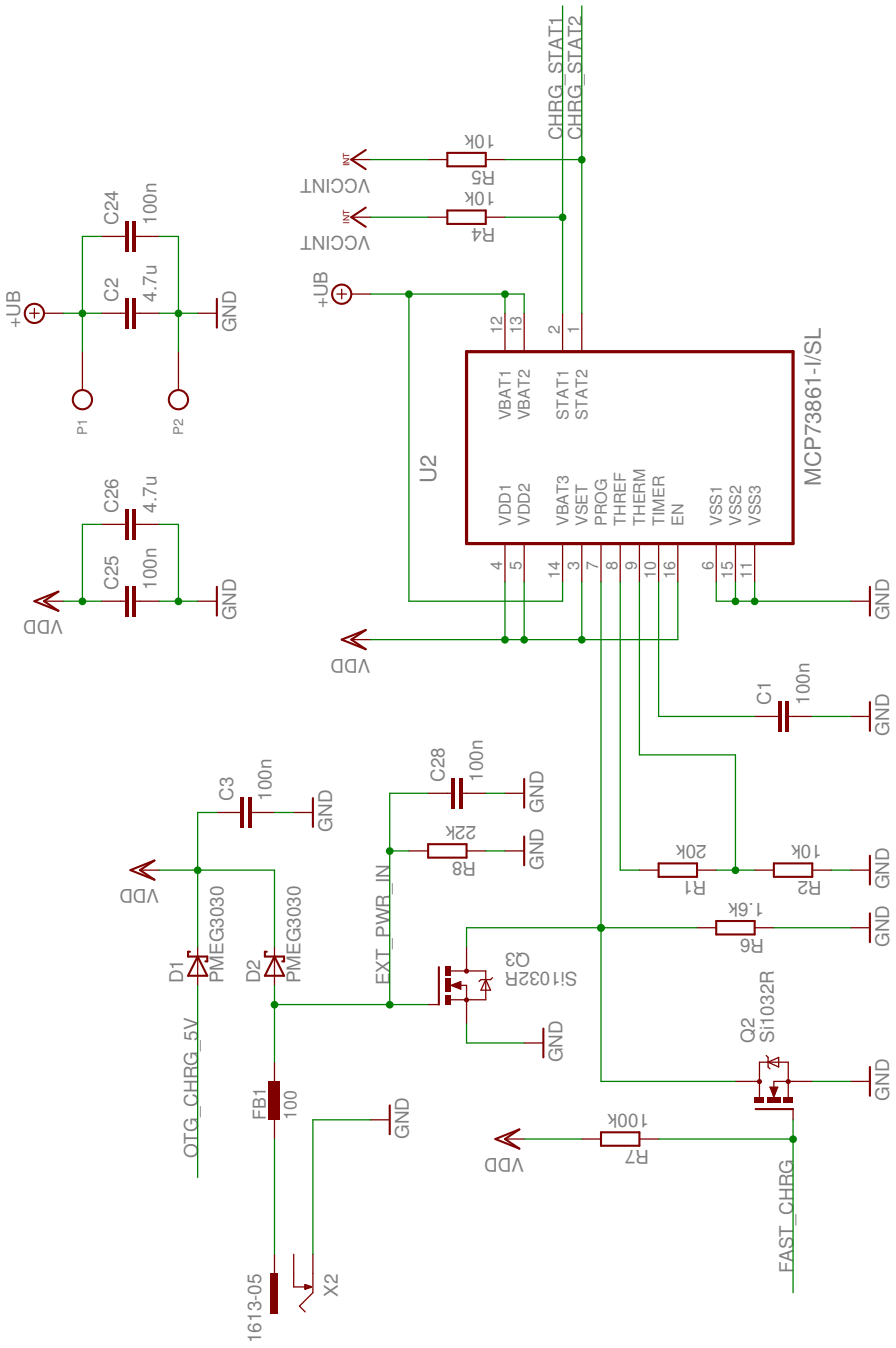
A.1. Napajanje



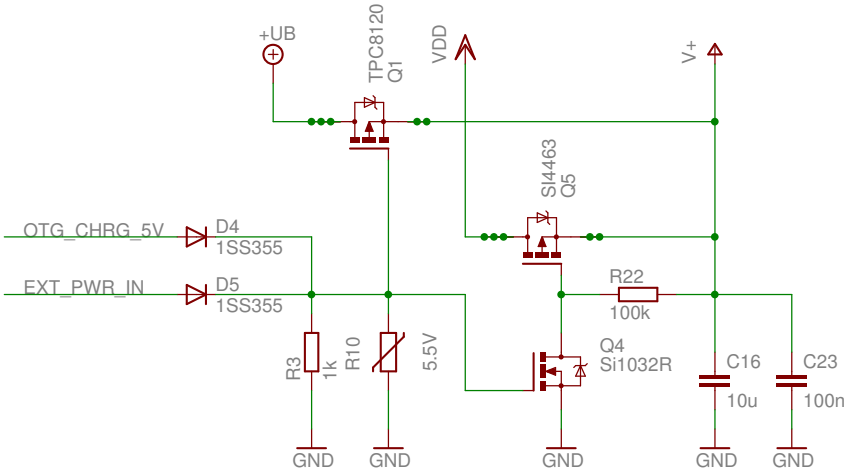
Slika A.1: Tipka za paljenje i gašenje



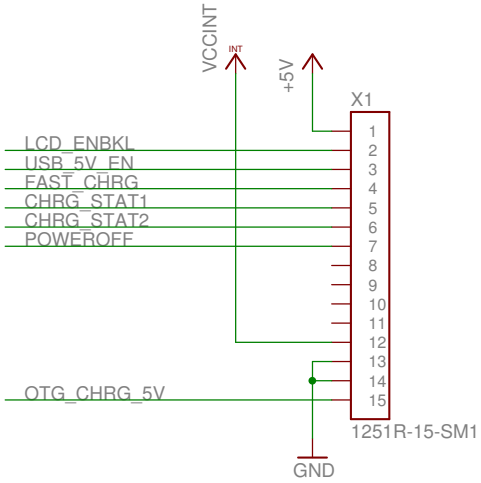
Slika A.2: MOSFET-i za paljenje i gašenje



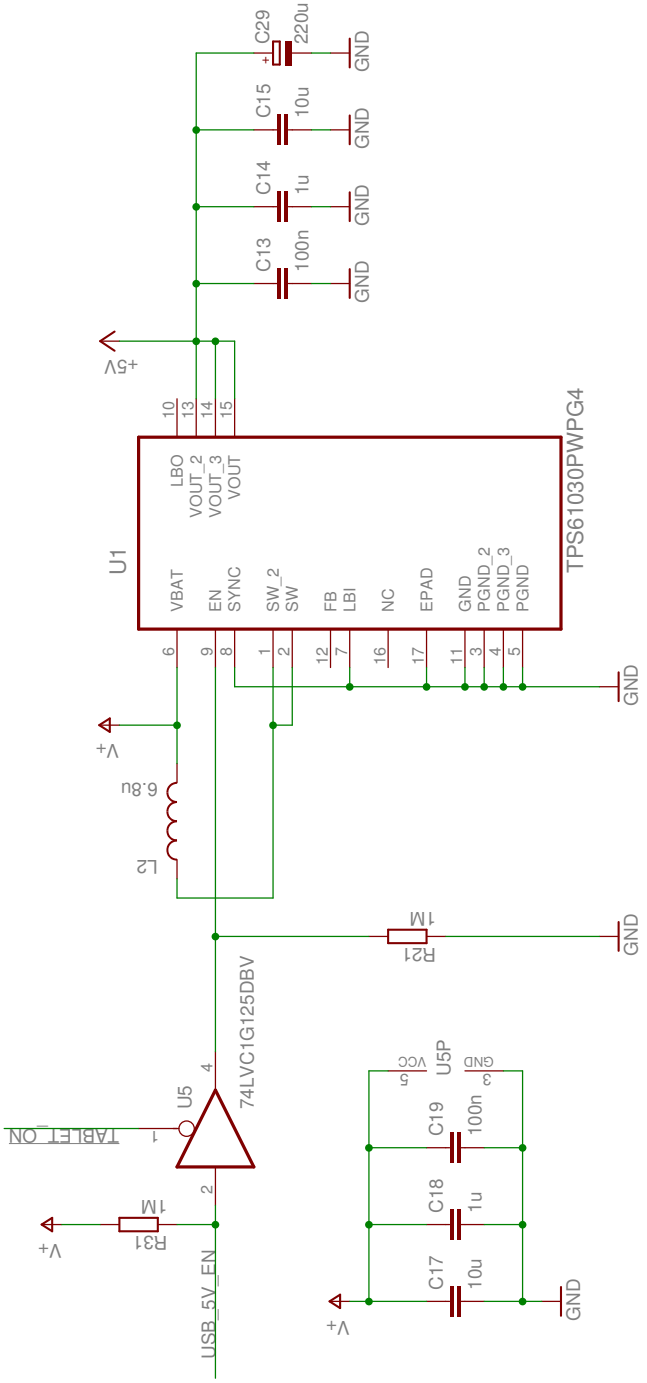
Slika A.3: Punjač baterije



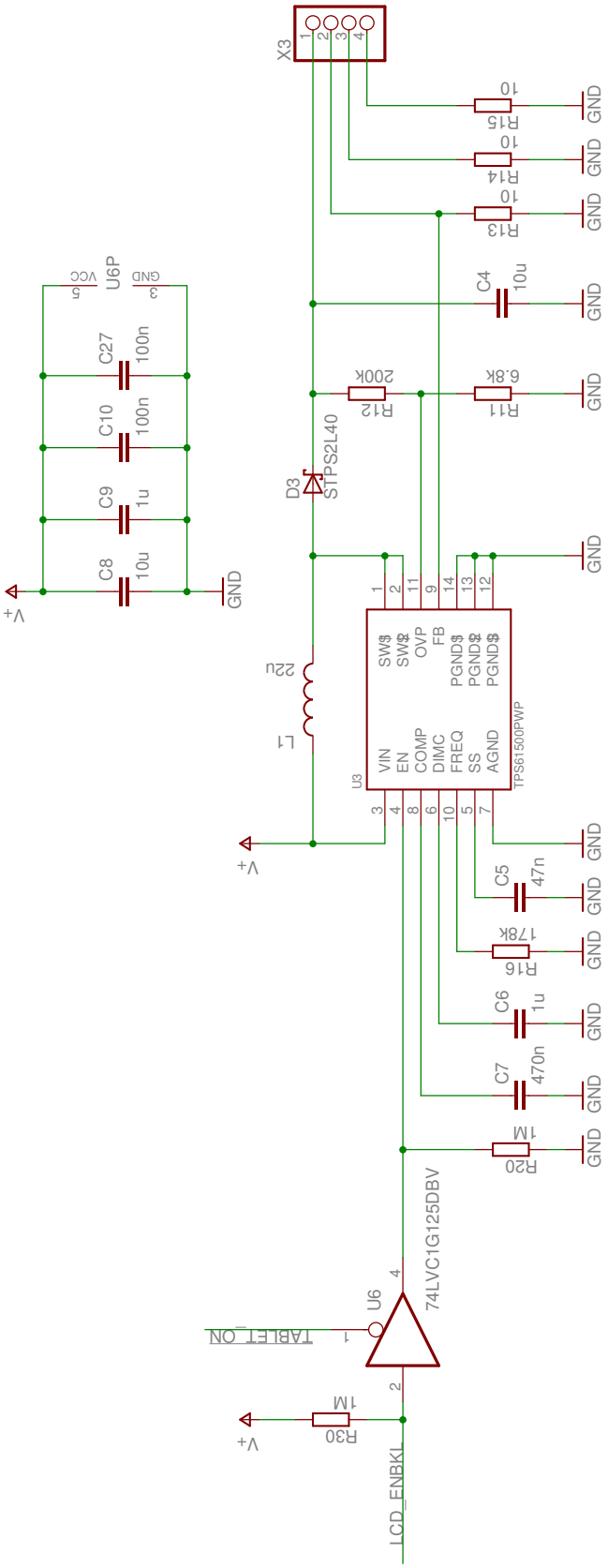
Slika A.4: Odabir izvora napajanja



Slika A.5: Konektor za pločicu s napajanjem



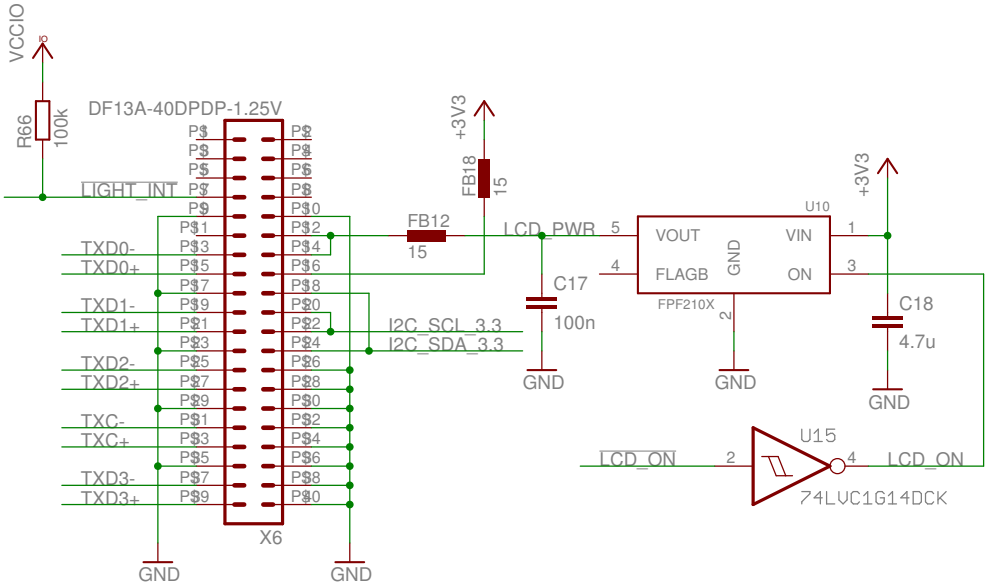
Slika A.6: Izvor napajanja od 5V



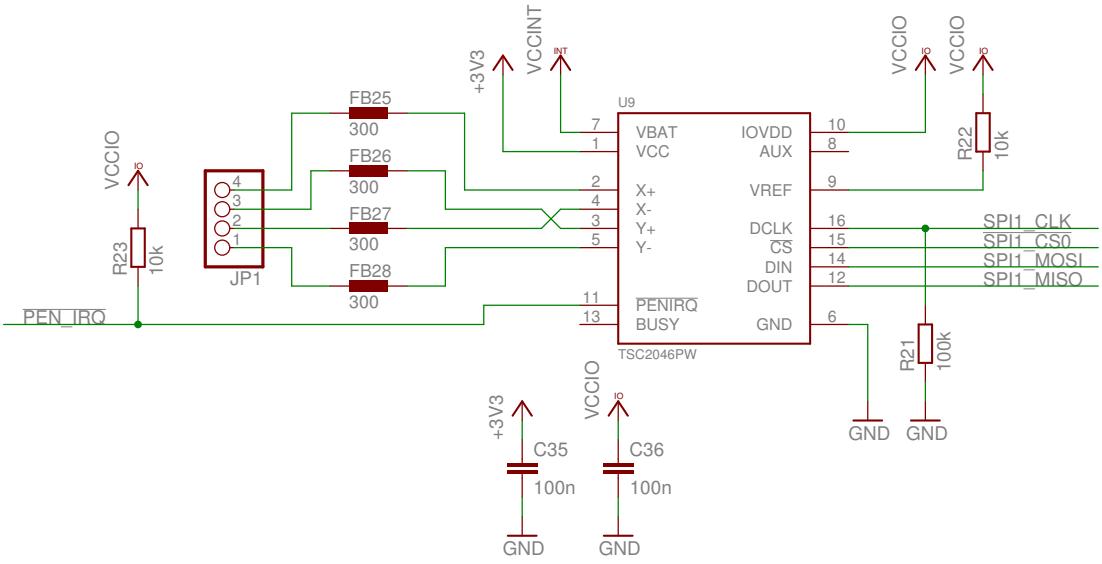
Slika A.7: Napajanje za pozadinsko osvjetljenje

A.2. Noseća pločica

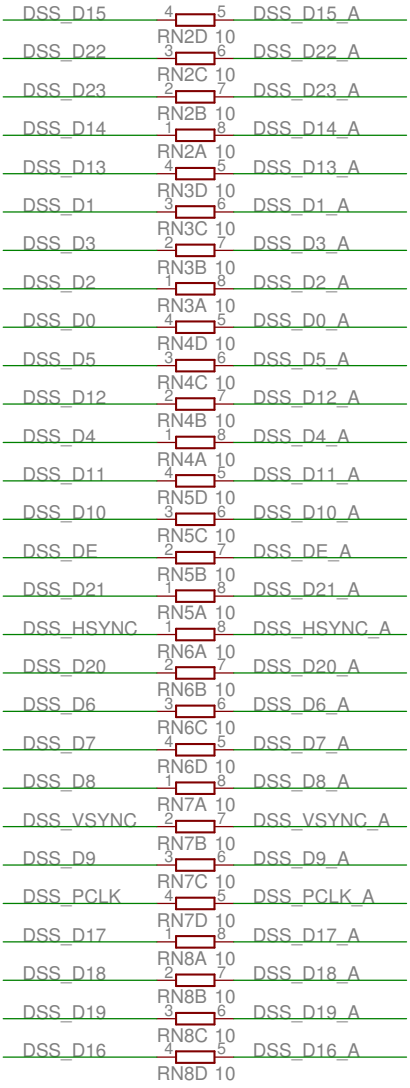
A.2.1. LCD-ekran



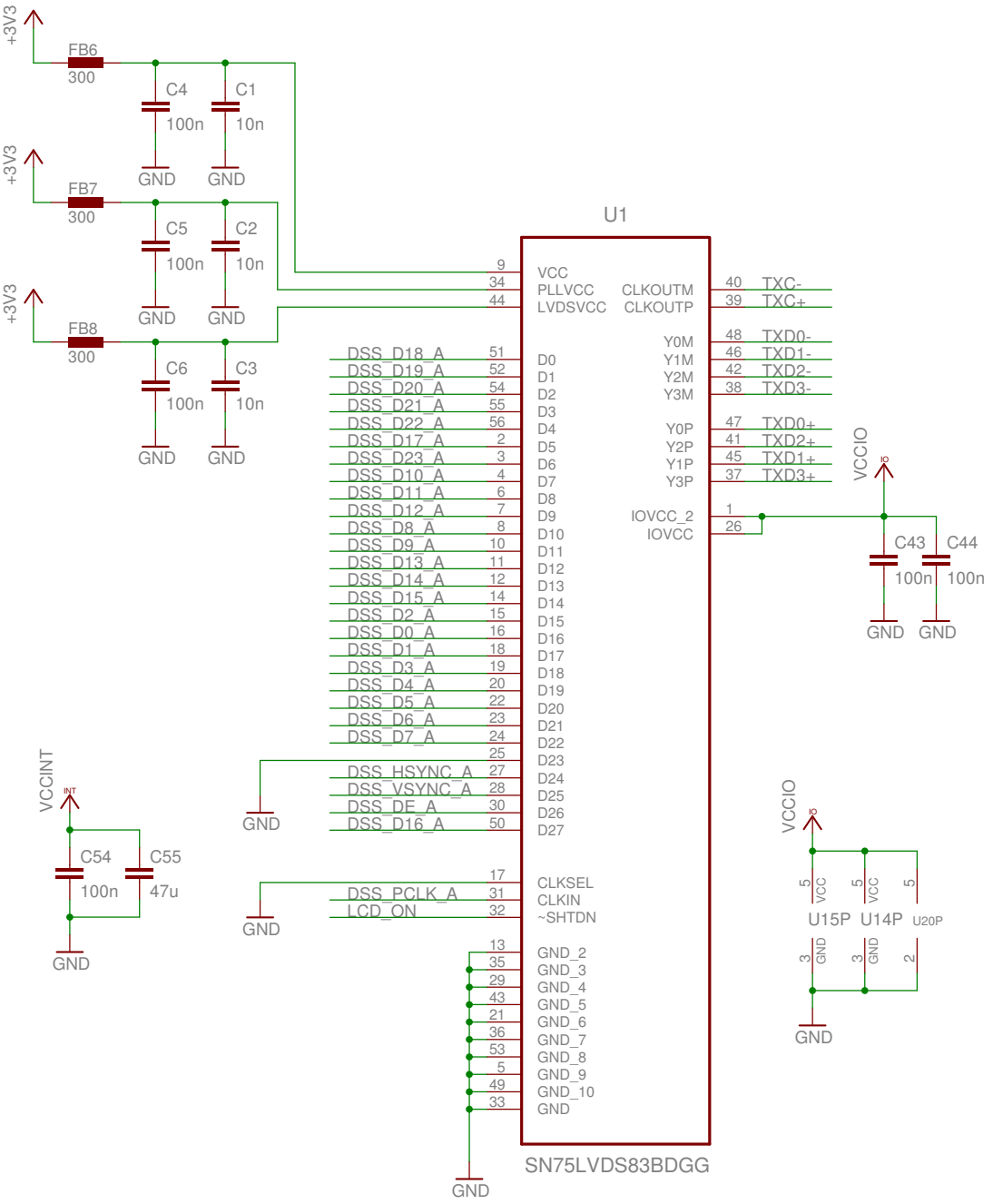
Slika A.8: Priključak za LCD ekran



Slika A.9: Čip za očitanje otporničkog panela osjetljivog na dodir

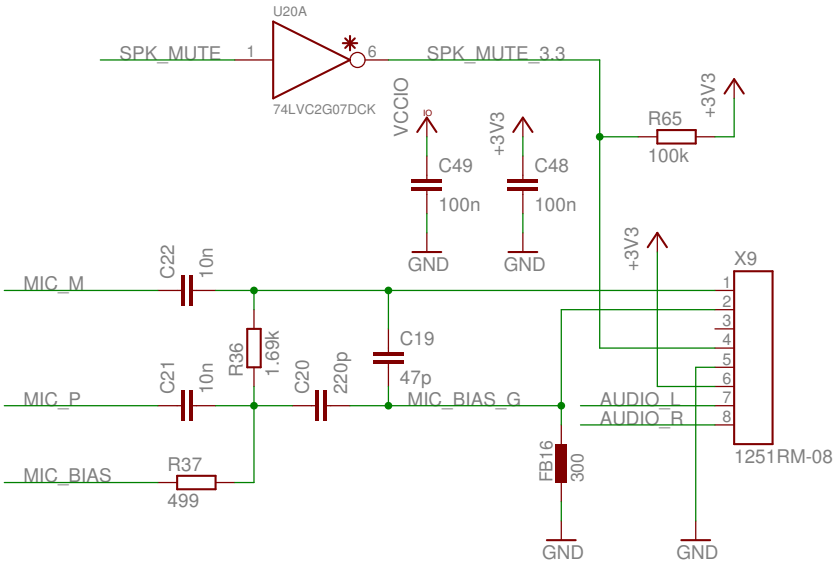


Slika A.10: Otpornici za ublaživanje naglih struja

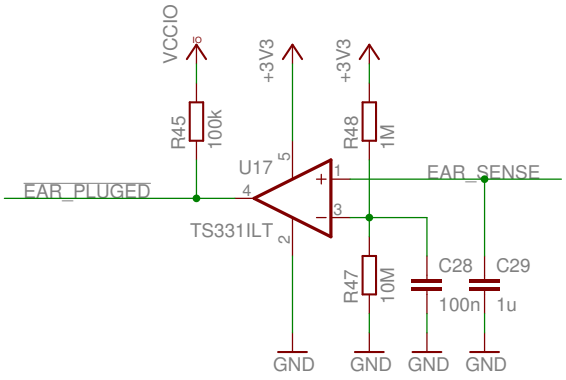


Slika A.11: LVDS pretvornik

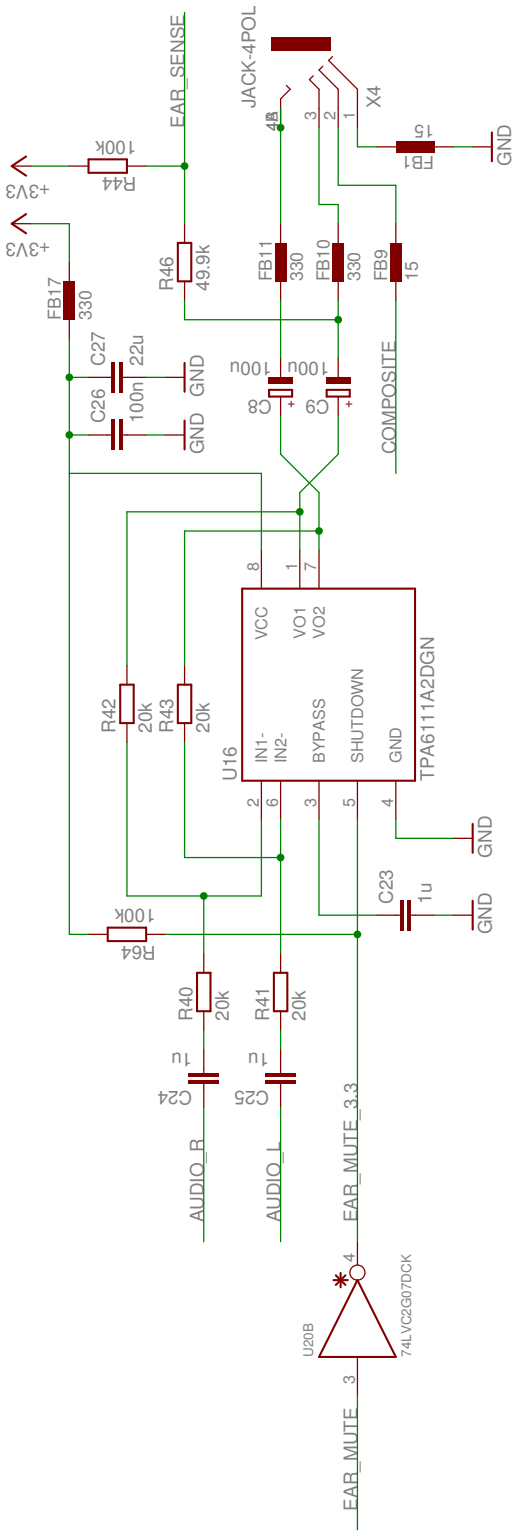
A.2.2. Audio



Slika A.12: Priključak za zvučnike i mikrofon

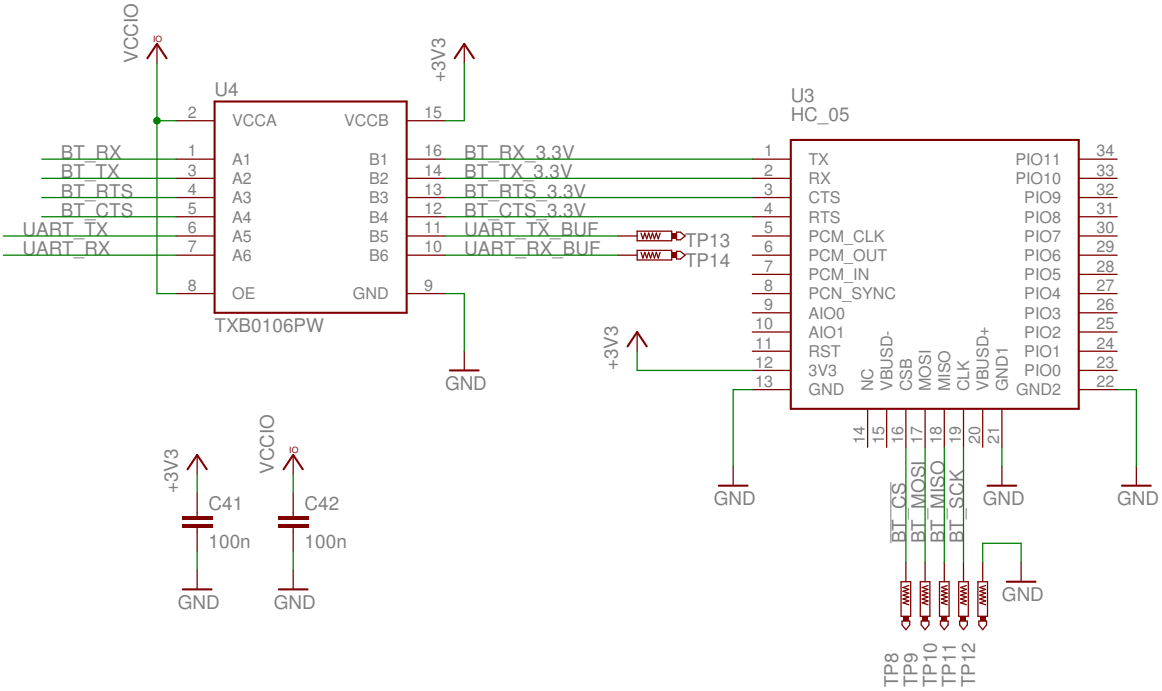


Slika A.13: Sklop za otkrivanje prisustva slušalica (loš)

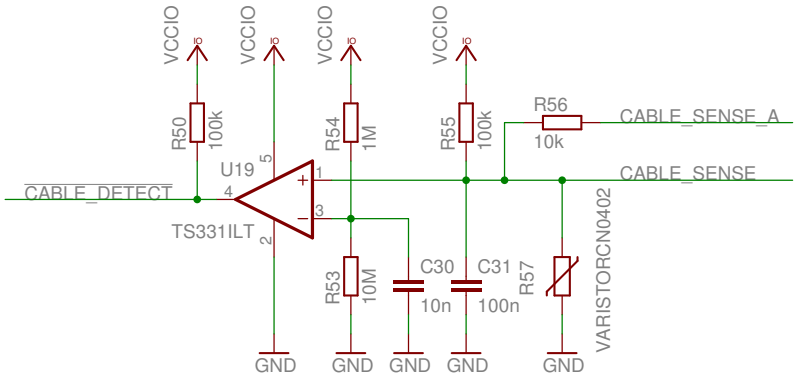


Slika A.14: Pojačalo za slušalice

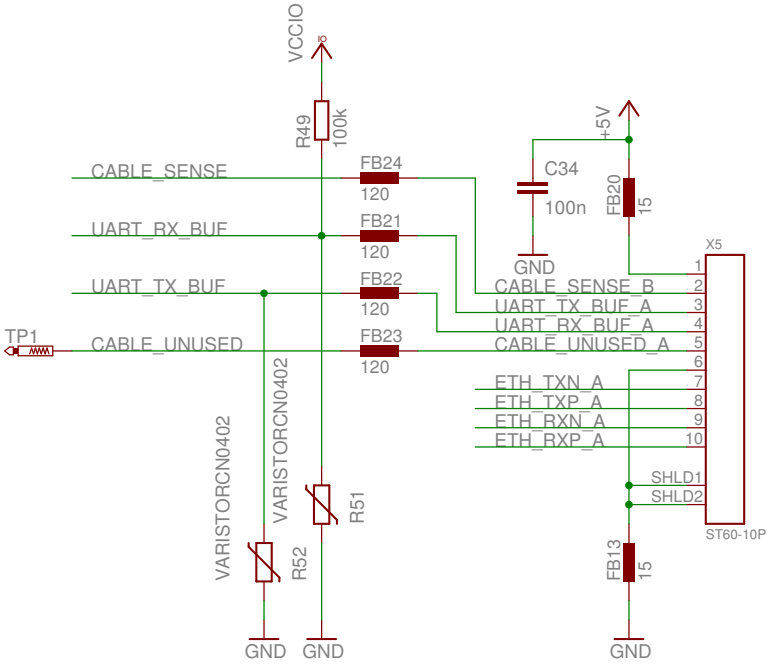
A.2.3. Mreža



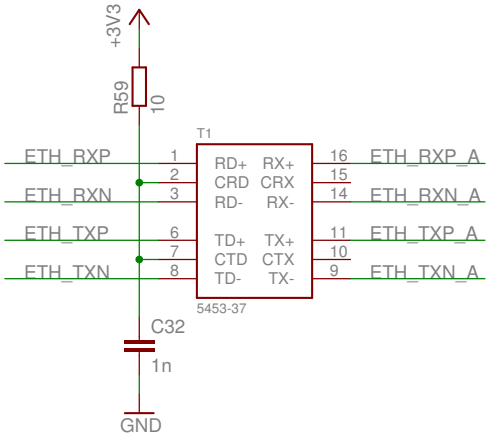
Slika A.15: Bluetooth modul s pretvaračem naponskih razina



Slika A.16: Sklop za otkrivanje prisustva podatkovnog kabla

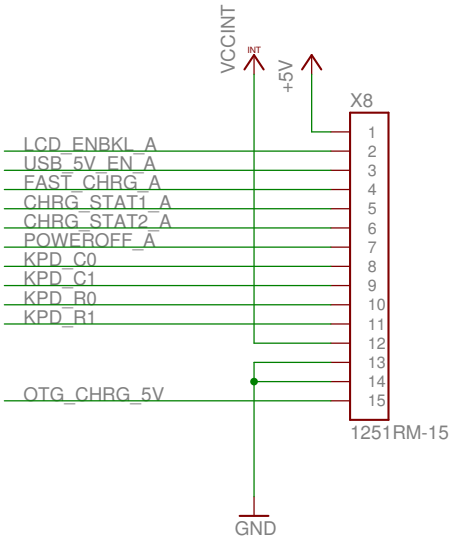


Slika A.17: Podatkovni priključak

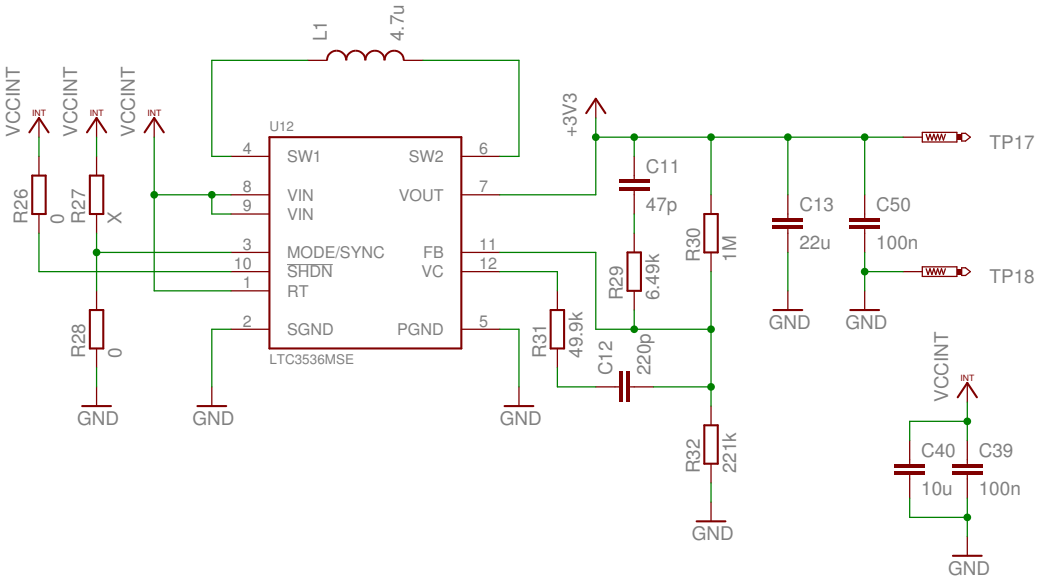


Slika A.18: Transformator za *ethernet*

A.2.4. Napajanje

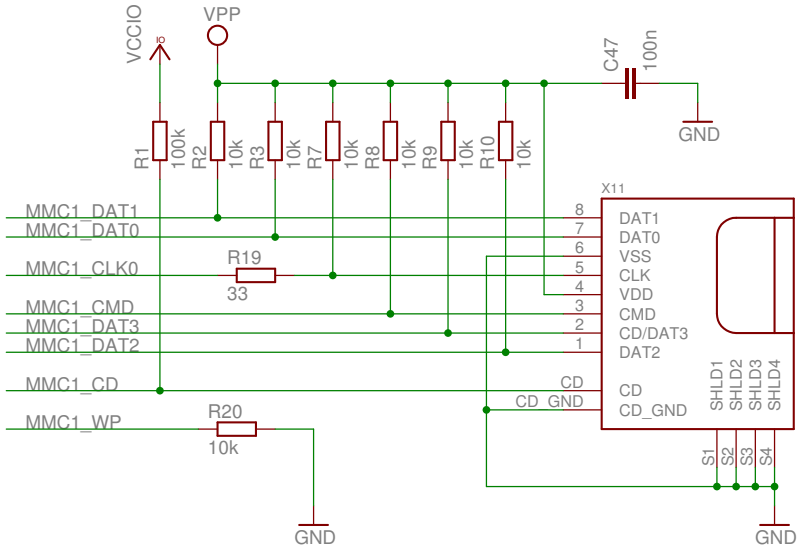


Slika A.19: Priključak za napajanje

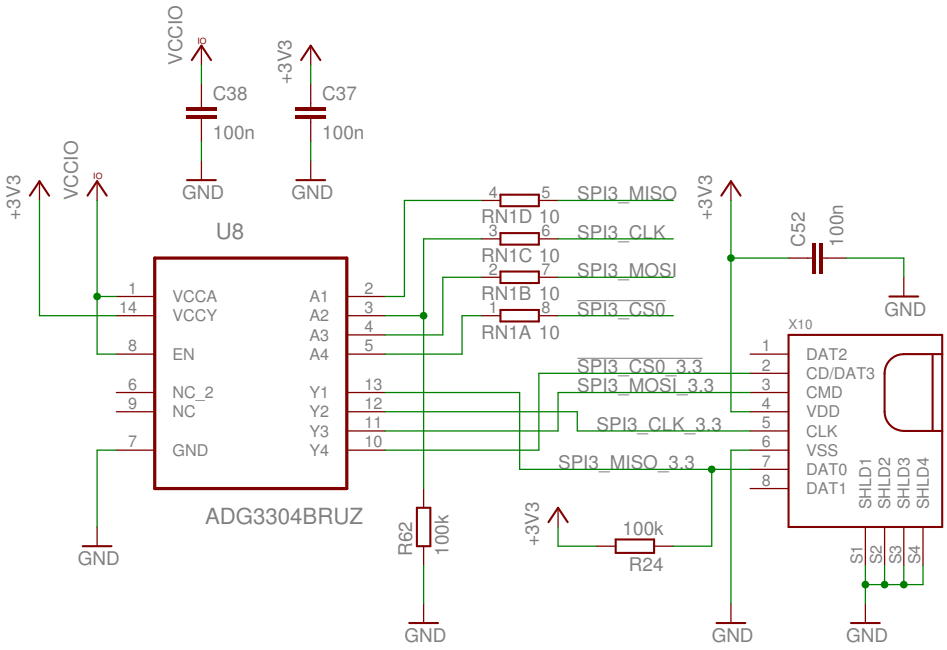


Slika A.20: Izvor napajanja od 3.3V

A.2.5. Memorijske kartice

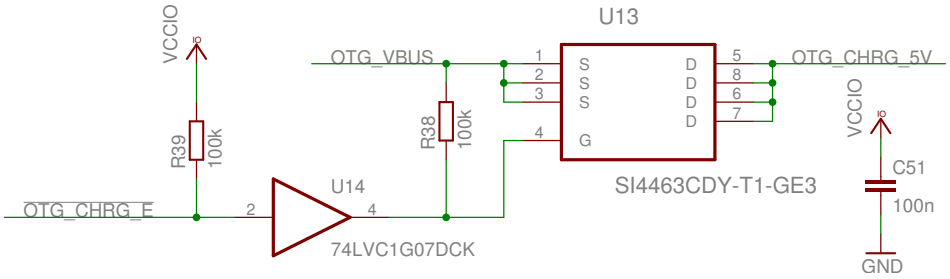


Slika A.21: Memorijska kartica MMC spojena na uobičajenu sabirnicu

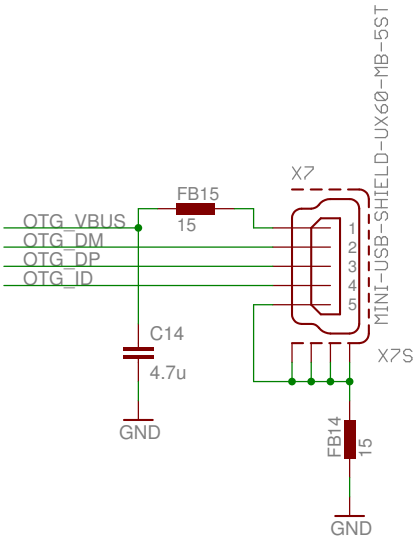


Slika A.22: Memorijska kartica MMC spojena na sabirnicu SPI

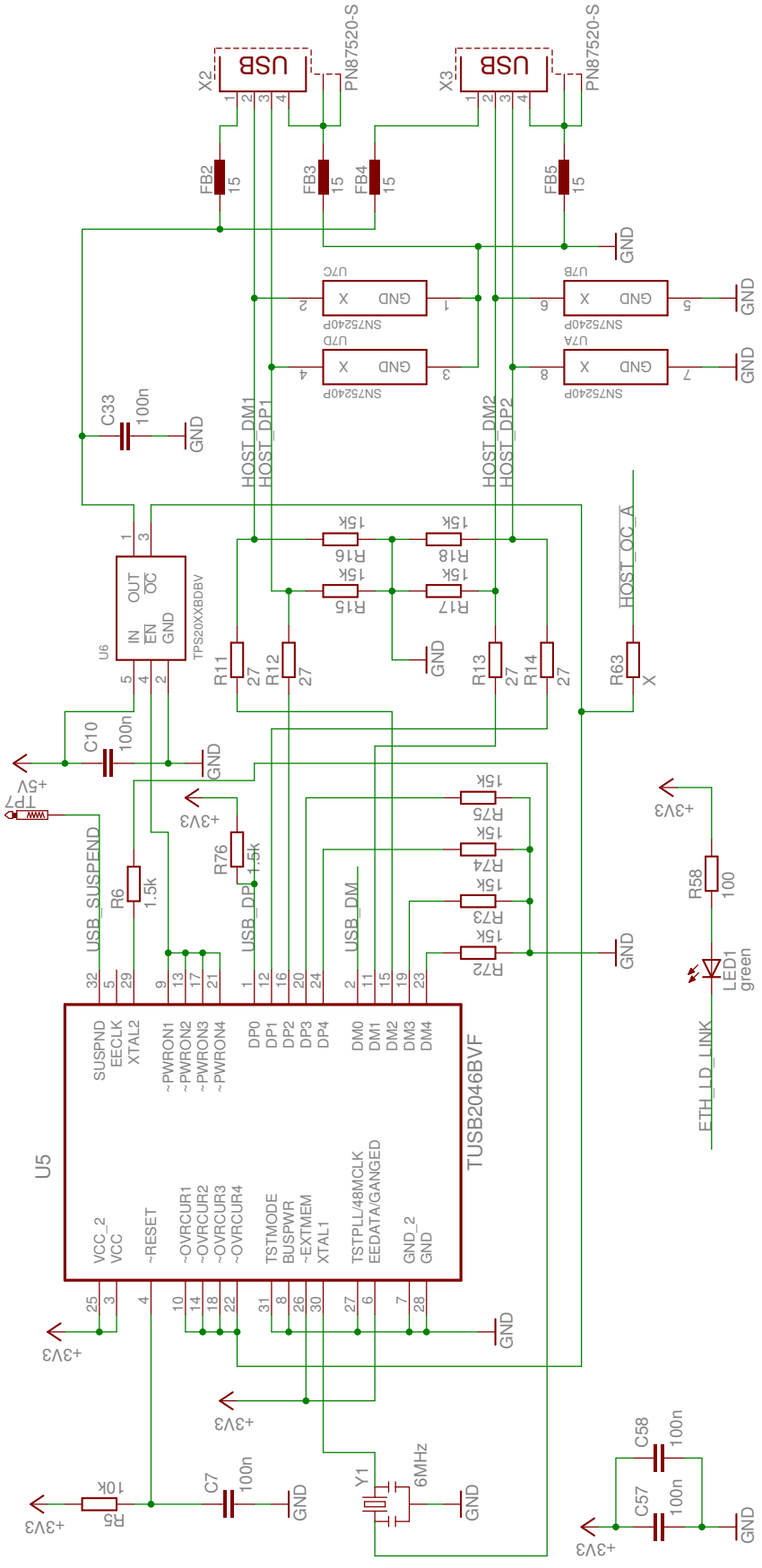
A.2.6. USB



Slika A.23: Prosljeđivanje izvora napajanja iz USB-a

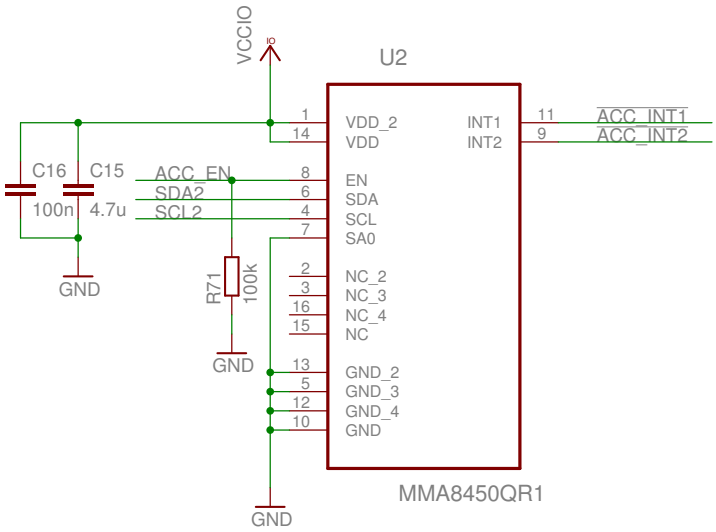


Slika A.24: USB OTG

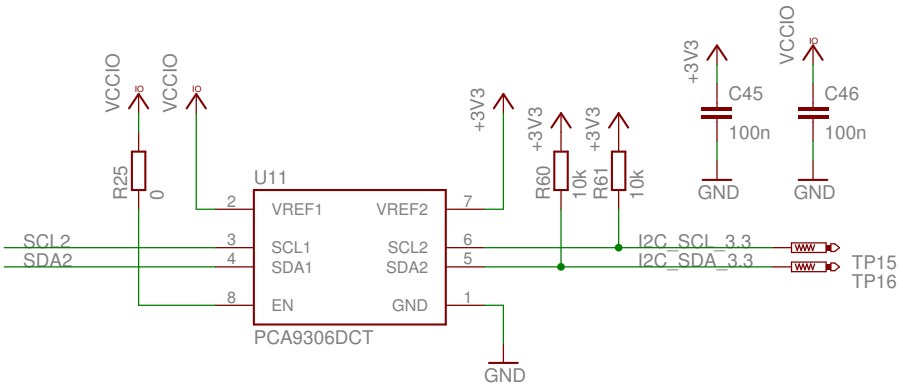


Slika A.25: USB *hub*

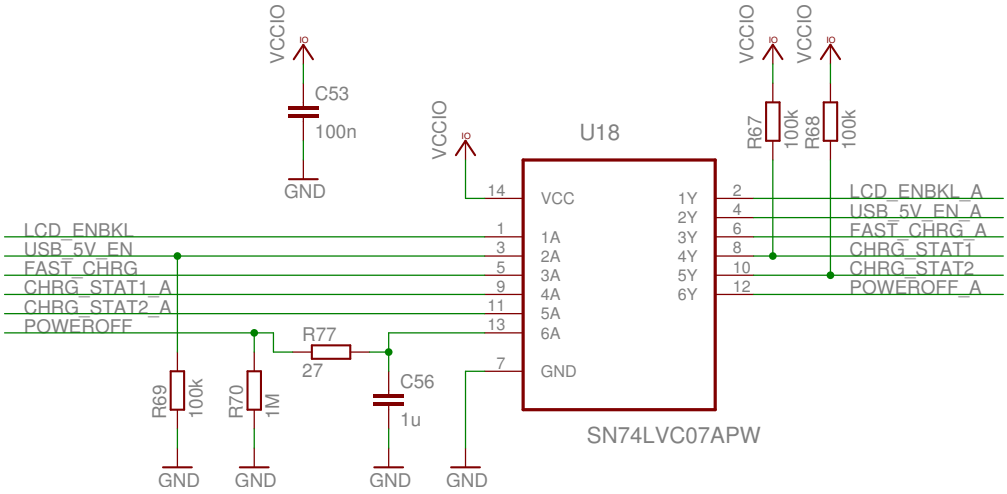
A.2.7. Razno



Slika A.26: Akcelerometar

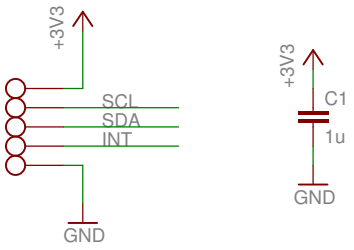


Slika A.27: Pretvarač naponskih razina za sabirnicu I^2C

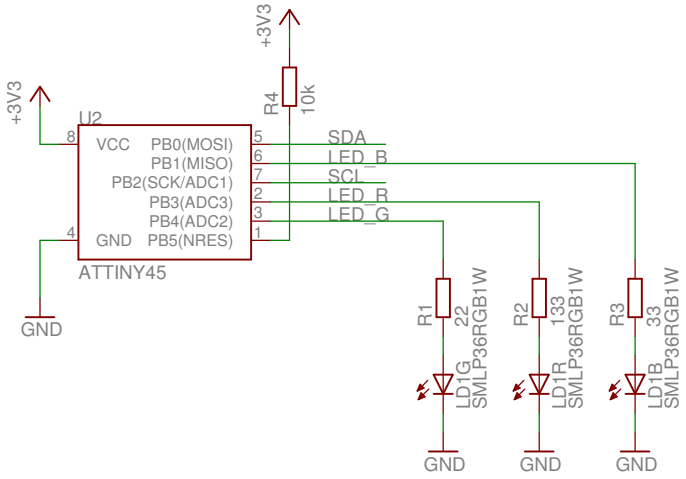


Slika A.28: Pretvarač naponskih razina

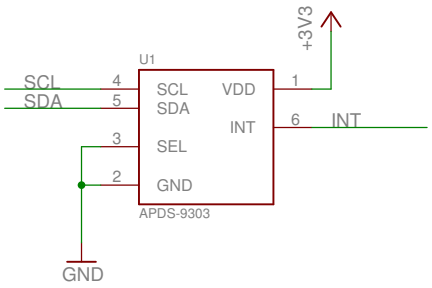
A.3. Pločica sa svjetlećom RGB-diodom



Slika A.29: Priključak

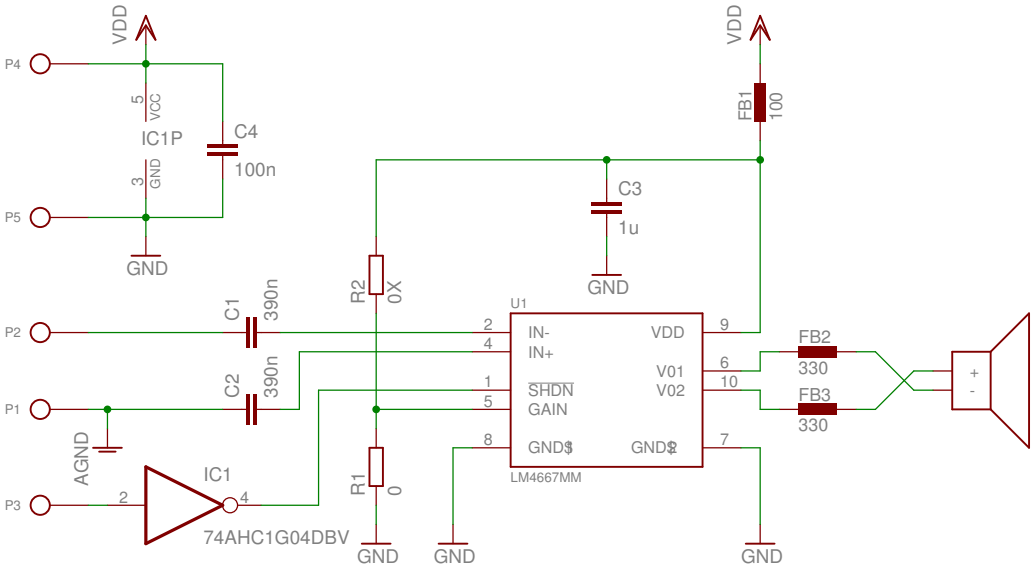


Slika A.30: Mikrokontroler s RGB-diodom



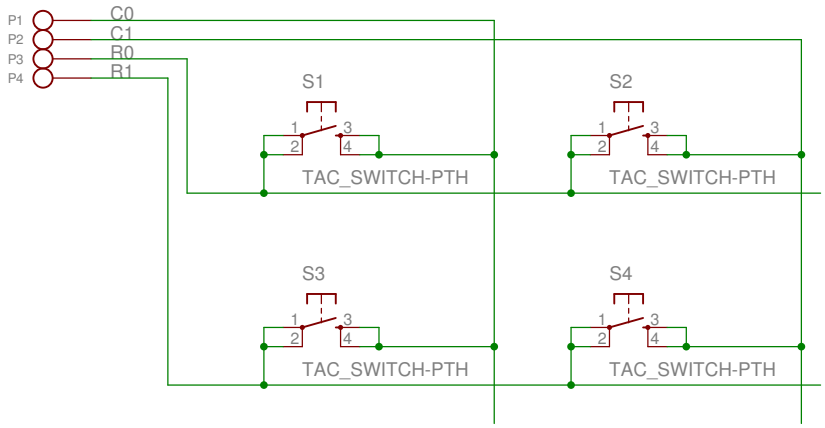
Slika A.31: Senzor intenziteta svjetlosti

A.4. Pojačalo sa zvučnikom



Slika A.32: Pojačalo sa zvučnikom

A.5. Pločica s tipkama



Slika A.33: Pločica s tipkama

Dodatak B

Nacrt za pleksi-staklo