KERROSNÄYTTÖMODUULIN SUUNNITTELU

Andrei Mehiläinen

Opinnäytetyö
Huhtikuu 2011
Tietotekniikka
Sulautetut järjestelmät ja elektroniikka
Tampereen ammattikorkeakoulu

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma, sulautetut järjestelmät ja elektroniikka

Tekijä	Andrei Mehiläinen
Työn nimi	Kerrosnäyttömoduulin suunnittelu
Sivumäärä	41 sivua + 3 liitesivua
Valmistumisaika	28.5.2011
Työn ohjaaja	yliopettaja Kai Poutanen
Työn tilaaja	Pikotec Oy

TIIVISTELMÄ

Tässä opinnäytetyössä suunniteltiin ja toteutettiin kerrosnäyttömoduuli, joka voidaan asentaa hissin koriin tai kerrokselle. Työ tehtiin Pikotec Oy:lle, joka on Kangasalla toimiva elektroniikan suunnittelu- ja valmistusyritys.

Työssä suunniteltiin laitteen elektroniikka ja ohjelmisto. Laitteen päätarkoitus on näyttää kerroksen numero, missä hissi kullakin hetkellä sijaitsee. Kerroksen numeron lisäksi taustalla voidaan näyttää erilaisia kuvia ja videoita. Laite käyttää JPEG-kuvatiedostoja ja AVI-videotiedostoja, jotka voidaan tallentaa laitteen microSD-muistikortille.

Laitteesta kehitettiin versiot 3,5- ja 5,7-tuumaisille TFT-näytöille. Näytön ohjaukseen käytetään monipuolista näytönohjainpiiriä, jonka ominaisuuksiin kuuluvat muun muassa laitteistopohjainen JPEG-koodekki ja SD-muistikorttiliitäntä. Ohjelmisto kehitettiin C-ohjelmointikielellä 16-bittiselle mikrokontrollerille, joka toimii laitteen pääprosessorina.

Työn tuloksena saatiin toimiva sarjatuotantoversion prototyyppi, joka voidaan esittää asiakkaille. Prototyyppiä voidaan muokata asiakkaan tarpeiden mukaan ja lisätä siihen tarvittaessa uusia ominaisuuksia.

Avainsanat: kerrosnäyttö, videon toisto, tiedostomuodot, TFT, AVI, JPEG, SD, FAT

Tampere University of Applied Sciences
Department of Computer Engineering, Embedded Systems and Electronics

Author	Andrei Mehiläinen
Title	Elevator display design
Number of pages	41 pages + 3 appendix pages
Graduation time	28.5.2011
Thesis supervisor	Senior lecturer Kai Poutanen
Commissioned by	Pikotec Oy

ABSTRACT

The main purpose of this thesis was to design and implement elevator display module, which can be installed either to the car or to a level. This project was done for electronics design and manufacturing company Pikotec Oy.

This thesis focuses on the electronics and software of the device. The main purpose of the device is to show the floor number where elevator is located. In addition to the floor number images and videos can be displayed on the background. The device supports JPEG image files and AVI video files. Files can be stored in the microSD memory card of the device.

The device has been developed in two versions: for 3.5” and 5.7” TFT displays. Highly integrated graphics controller chip is used to control the TFT display. The features of the chip include, among others, the hardware accelerated JPEG codec and SD memory card interface. The software is developed in C programming language for 16-bit microcontroller, which is the device’s main processor.

The project has resulted in a prototype of the series production version, which can be presented to customers. The prototype can be customized according to customer's needs and new features can be added to it if necessary.

Keywords: elevator display, video playback, file formats, TFT, AVI, JPEG, SD, FAT

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma, sulautetut järjestelmät ja elektroniikka

ALKUSANAT

Kesällä 2010 olin harjoittelijana Pikotec Oy:n tuotekehitysosastolla. Pikotec Oy on ostanut vuonna 2009 Vuolas Oy:n, jonka päätuotteina ovat olleet erilaiset hissinäytöt. Yhtenä projektinani on ollut TFT-näytöllä varustetun kerrosnäyttömoduulin uudistus. Tavoitteena oli saada laitteesta monipuolisempi ja halvempi valmistaa. Tehtävänäni oli suunnitella laitteen elektroniikka ja ohjelmisto. Haluan kiittää Kimmo Mannista ja muita Pikotec Oy:n työntekijöitä, jotka auttoivat minua tässä projektissa.

Tampereella 28. huhtikuuta 2011

Andrei Mehiläinen

SISÄLLYSLUETTELO

1 JOHDANTO

Työssä esitetään hisseihin asennettavan kerrosnäyttömoduulin kehityksen vaiheita suurin piirtein samassa järjestyksessä, kuin miten kehitystyö alun perin etenikin. Aluksi esitetään määrittelyt ja vaatimukset, joita laitteelle kehityksen alussa asetettiin. Kerrotaan lyhyesti laitteen edellisen version ominaisuuksista ja puutteista, joita uuden version oli tarkoitus korjata. Tämän jälkeen perehdytään TFT-paneeleihin, niiden liitäntöihin ja ohjaustapoihin.

Luvussa 4 kerrotaan kehitystyön alkuvaiheista ja kerrosnäyttömoduulin ensimmäisen prototyypin laitteistosta sekä perehdytään käytössä olleeseen kehitysalustaan ja sen komponentteihin.

Työ on ohjelmistopainotteinen, joten laitteen ohjelmistoa tarkastellaan luvussa 5 yksityiskohtaisesti. Ohjelmisto on vahvasti sidoksissa laitteistoon, joten samassa luvussa kerrotaan myös joidenkin komponenttien ominaisuuksista.

Laitteen konfigurointia varten kehitettiin PC-sovellus, josta kerrotaan luvussa 6. Sovellus ei kuitenkaan ole tämän työn pääaihe, joten luvussa perehdytään vain sovelluksen ominaisuuksiin, muttei erityisesti sen kehitykseen.

Luvussa 7 kerrotaan laitteen sarjatuotantoversion prototyypistä, käydään läpi muutokset ja lisäykset ensimmäisen prototyypin nähden sekä perehdytään tässä versiossa käytettyihin komponentteihin. Tämän lisäksi kerrotaan lyhyesti laitteen testauksesta ja pohditaan jatkokehitysmahdollisuuksia.

Lopuksi tarkastellaan työn tuloksia ja pohditaan, kuinka hyvin kehityksen alussa asetetut tavoitteet on saavutettu.

2 MÄÄRITTELY JA VAATIMUKSET

Vuolas Oy on valmistanut vuodesta 2005 lähtien TFT-näytöllä varustettuja kerrosnäyttöjä. Vanhan version ominaisuudet ovat vanhentuneet ja asiakkaat kaipaavat parempaa vastinetta rahoille. Uuden version on ollut tarkoitus pitää sisällään kaikki nykyisen version tärkeimmät ominaisuudet ja mahdollisesti lisätä joitakin uusia ominaisuuksia.

Laitteesta oli myös tarkoitus saada huomattavasti halvempi valmistaa. Nykyinen versio perustuu Alteran valmistamaan FPGA-piiriin ja se ympärille rakennettuun kytkentään. Pelkästään FPGA-piirin hinta sadan kappaleen erissä on 15 euroa, tämän lisäksi tarvitaan vielä erilliset FLASH- ja RAM-muistipiirit. Koko kytkennän hinta ilman TFT-paneelia nousee näin ollen noin 100 euroon. Myös ohjelmiston kehitys FPGA:lle laitteistonkuvauskielellä on hankalaa ja koodin ylläpidettävyys vaikeaa.

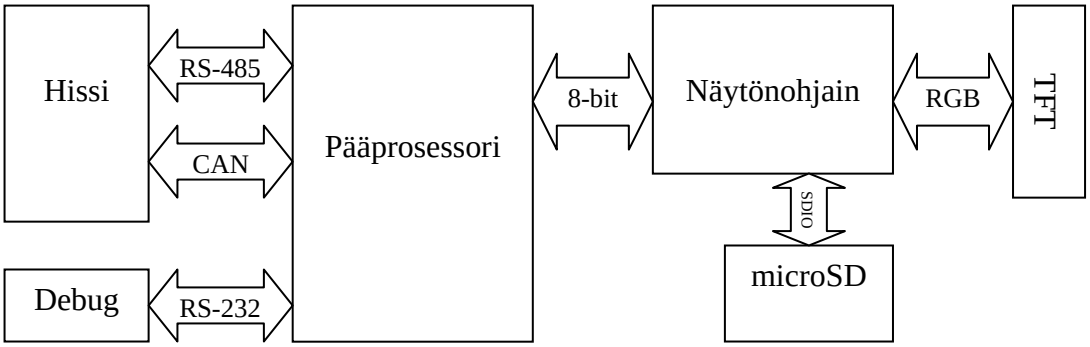
Vanha versio tukee vain yhtä TFT-paneelityyppiä, paneelin pitää olla kooltaan 5,7 tuumaa ja sen liitynnän on oltava 18-bittinen RGB. Uuteen versioon haluttiin lisätä tuki myös pienemmälle 3,5 tuuman paneelille. Sekä nykyisen että uuden version paneelien resoluutio on QVGA eli 320 pikseliä leveä ja 240 pikseliä korkea.

Laitteesta haluttiin saada myös helppokäyttöisempi. Esimerkiksi vanhassa versiossa asetuksien säätäminen tapahtuu kirjoittamalla käsin hankala konfiguraatiotiedosto, jossa ei saa olla yhtään virhettä. Uudessa versiossa tiedosto luodaan automaattisesti PC-konfiguraatio-ohjelman avulla.

Yhtenä vanhan version ominaisuutena on mahdollisuus asettaa taustakuvia näytölle. Kuvien on oltava BMP-muodossa, ja ne siirretään sarjaliikenneväylän kautta laitteeseen asennuksen yhteydessä. Uusien kuvien pävittäminen on hankalaa, koska asentaja joutuu käymään kannettavan tietokoneen kanssa erikseen jokaisen näytön luona. Tähän voi helposti kulua useita tunteja, varsinkin jos rakennuksessa on monta kerrosta ja hissiä. Kuvien päivittämistä haluttiin helpottaa, ja uudessa versiossa ne sijaitsevatkin erillisellä muistikortilla. Asentajan ei tarvitse kuin ladata uudet kuvat muistikorteille ja käydä vaihtamassa ne näyttöihin. Jatkossa on tarkoitus lisätä mahdollisuus muistikortin sisällön päivittämiseen hissin dataväylän kautta.

Tällä hetkellä kaikista yleisin hissidataväylä on rinnakkainen 15-bittinen. Kuitenkin on hyvin todennäköistä, että lähitulevaisuudessa hissien valmistajat siirtyvät käyttämään sarjamuotoista CAN-väylää. Yhtenä uuden version vaatimuksena olikin CAN-väylä–valmius.

Kaikki nämä vaatimukset huomioon ottaen suunnitteluvaiheessa päätettiin kehittää uusi versio puhtaalta pöydältä eikä yrittää päivittää nykyistä versiota. Uusi versio perustuu mikrokontrollerin ympärille rakennettuun kytkentään (kuva 1). Mikrokontrollerissa pyörii laitteen ohjelmisto, ja TFT-paneelia ohjataan erillisellä näytönohjainpiirillä. Lisäksi uusi versio sisältää muistikortin, jossa kaikki kuvat ja konfiguraatiotiedosto sijaitsevat.



Kuva 1. Kerrosnäyttömoduulin lohkokaavio

2.1 Näytettävä sisältö

Kerrosnäyttömoduulia voidaan verrata tavalliseen digitaaliseen valokuvakehykseen, joka on muokattu hissikäyttöä varten. Yksinkertaisesti laitteen tarkoitus on näyttää kerroksien numeroita jonkin taustakuvan tai -videon päällä. Jokaiselle kerrokselle voidaan määrittää oma taustakuva, jolloin kuva vaihtuu silloin, kun hissi liikkuu kerroksesta toiseen. Vaihtoehtoisesti voidaan määrittää soittolista, jolloin kuvat ja videot vaihtuvat kerroksesta riippumatta tietyn ajan välein. Soittolistaa kannattaa käyttää erityisesti silloin, kun laite ei ole asennettu hissin koriin vaan sijaitsee kerroksessa. Taustakuvat ovat JPEG-muodossa ja videot AVI-muodossa (tästä tarkemmin ”Videon toisto”-luvussa). Kaikki kuvat ja videot sijaitsevat muistikortin juurihakemistossa. Konfiguraatiotiedostossa määritellään, mikä kuva näytetään missäkin kerroksessa. Myös soittolista määritellään siellä.

Kerroksien numerot ovat käytännössä kuvia, jotka myös tallennetaan muistikortille. Ennen muistikortille tallentamista nämä kuvat on käännettävä näytönohjainpiirille sopivaan muotoon. Tämä voidaan tehdä PC-konfiguraatio-ohjelman avulla. Oletuksena on, että muistikortille on tallennettu numeroiden kuvat nollasta yhdeksään ja ne on nimetty muotoon <numero>.cur (esimerkiksi 5.cur). Tällöin laite osaa automaattisesti hakea muistikortilta oikeat tiedostot. Kaksinumeroiset luvut muodostetaan myös automaattisesti näistä kuvista. Konfiguraatiotiedostossa on kuitenkin mahdollisuus määrittää jokaiselle kerrokselle erillinen ”numerokuva”. Tämä voi olla kätevää, jos esimerkiksi halutaan näyttää rakennuksen alimmalla tai ylimmällä kerroksella kuvia, jotka eivät sisällä numeroita vaan esimerkiksi kirjaimia tai tekstiä.

Taustakuvien ja kerroksien numeroiden lisäksi näytöllä näytetään ylös- ja alaspäin osoittavia nuolien kuvia. Nämäkin kuvat on käännettävä sopivaan muotoon PC-ohjelman avulla. Kuvien tiedostonimet ovat up.cur, down.cur ja updown.cur ja ne sijaitsevat muistikortin juurihakemistossa. Nuolien kuvat voidaan myös animoida: ylänuolen kuvaa vieritetään ylöspäin ja alanuolen kuvaa taas alaspäin. Vierityksen nopeus on säädettävissä konfiguraatiotiedostossa. Tämän lisäksi konfiguraatiotiedostossa voidaan määrittää koordinaatit, missä numerokuvat ja nuolien kuvat näytöllä sijaitsevat. Liitteessä 2 on esimerkki konfiguraatiotiedoston sisällöstä.

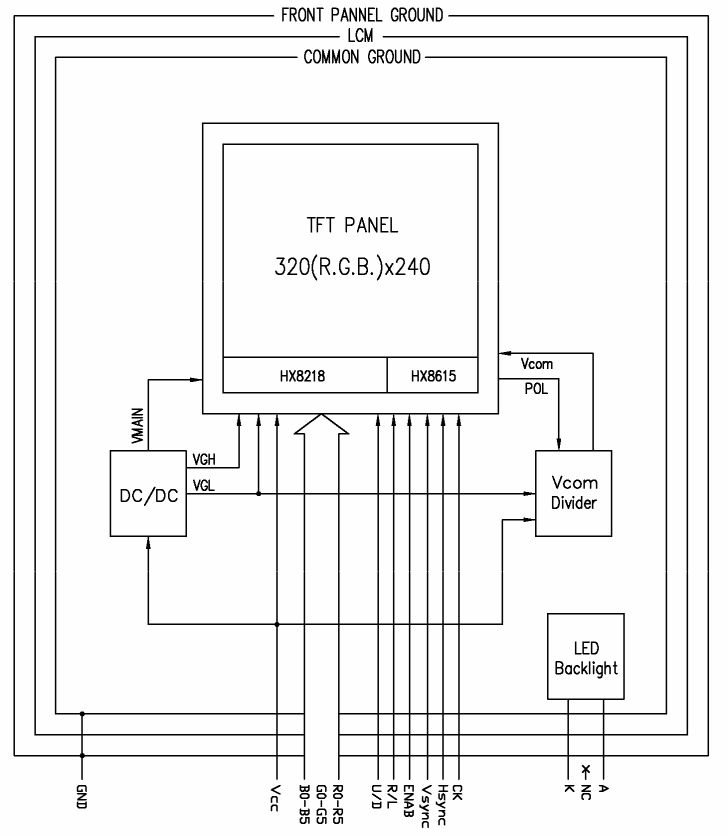
3 TFT-PANEELIT

Erilaisten TFT-paneelien ohjaustavat riippuvat paneelin koosta, resoluutiosta ja paneelin ohjainpiiristä. Tässä työssä tarkastellaan lähinnä 3,5- ja 5,7-tuumaisten QVGA-resoluutiolla varustettujen paneelien ohjausta. Yhtenäistä kaikille paneeleille on se, että niiden ohjaus perustuu VGA-standardiin.

3.1 Paneelien liitännät

Normaalisti VGA-liitännän värisignaalit ovat analogisia, mutta TFT-paneelien liitännät ovat yleensä digitaalisia (3,3 V tai 5 V TTL). Paneelien liitännät sisältävät yleensä

RGB-väridatalinjat, synkronointisignaalien linjat sekä pikselikello- ja data enable -linjat. Myös paneelin käyttöjännite ja maa ovat yleensä samassa liitännässä. Paneelien sisällä sijaitsee yksi tai useampia ohjainpiirejä, jotka varsinaisesti ohjaavat pikseleitä. Gate driver -piiri ohjaa paneelin rivejä ja source driver -piiri taas rivin yksittäisiä pikseleitä. Esimerkiksi 5,7-tuumaisissa paneeleissa suosittu kombinaatio on Himax HX8615A gate-ohjain ja HX8218-A source-ohjain (kuva 2). 3,5-tuumaisissa paneeleissa käytetään yleensä yhtä ohjainpiiriä, joka sisältää sekä source- että gate-ohjaimet. Suosittu ohjainpiiri 3,5-tuumaisissa paneeleissa on Himax HX8238A.



Kuva 2. 5,7” TFT-paneelin lohkokaavio [19, s. 11]

Pikseleiden väridata lähetetään rinnakkaisen RGB-väylän kautta. Väylän leveys riippuu paneelin maksimivärimäärästä. Esimerkiksi jos paneelin maksimivärimäärä on 16,7 miljoonaa väriä, RGB-väylän on oltava 24-bittinen. 18-bittisen väylän avulla voidaan ilmaista 262 tuhatta väriä. Monet 3,5-tuumaiset paneelit toimivat niin sanotulla sarjamuotoisella RGB:llä. Tässä moodissa RGB-väylää käytetään 8-bittisenä. Pikselien värit lähetetään peräkkäin: ensin punainen, sitten vihreä ja lopuksi sininen.

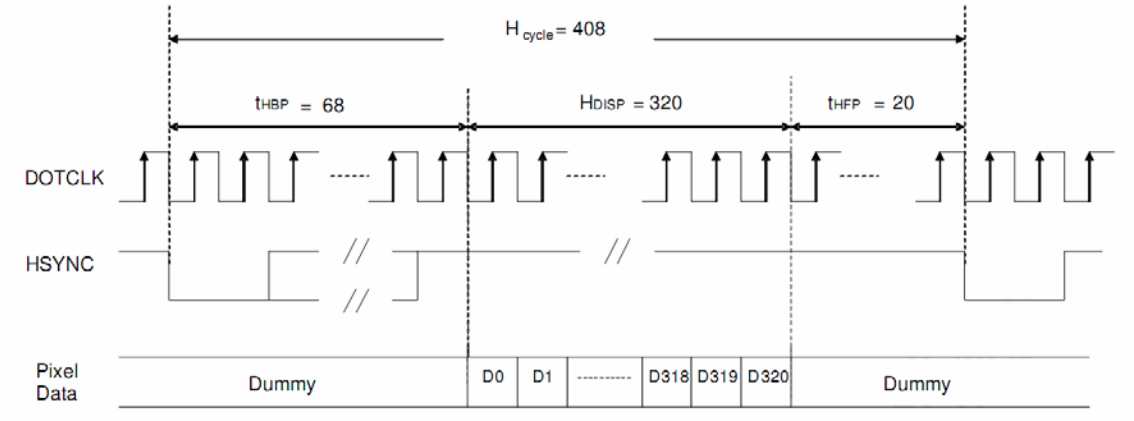
Datasignaalien lisäksi liitännässä voi olla joitakin ohjauslinjoja. Esimerkiksi 5,7-tuumaisissa paneeleissa on yleensä linjat kuvan kääntöä (vaaka- ja pystysuunnassa) varten. 3,5-tuumaisissa paneeleissa käytetty HX8238A-ohjainpiiri sisältää asetusrekisterit ja SPI-väylän, jonka kautta rekistereihin voidaan kirjoittaa dataa. Näiden rekistereiden avulla voidaan esimerkiksi määrittää RGB-väylän moodi (sarja/tavallinen), kääntää kuva vaaka- ja pystysuunnassa sekä säätää kuvan värejä.

Paneelien taustavalo on yleensä tehty ledeillä. Ne voivat olla sarjassa tai rinnan kytkettyjä. Yleensä 3,5-tuumaisissa paneeleissa on kuusi valkoista ledia sarjaan kytkettynä, jolloin taustavalon jännitteeksi tulee noin 19 – 20 V. 5,7-tuumaisissa paneeleissa ledien kytkennät vaihtelevat, ja taustavalon jännite voi olla 6 – 24 V.

3,5-tuumaisissa paneeleissa on yleensä yksi liitin, jossa kulkevat kaikki edellä mainitut linjat. Myös taustavalon ja mahdollisen kosketuspaneelin signaalit kulkevat samassa liittimessä. 5,7-tuumaisissa paneeleissa taustavalon ja kosketuspaneelin liittimet ovat yleensä erotettu dataliittimestä.

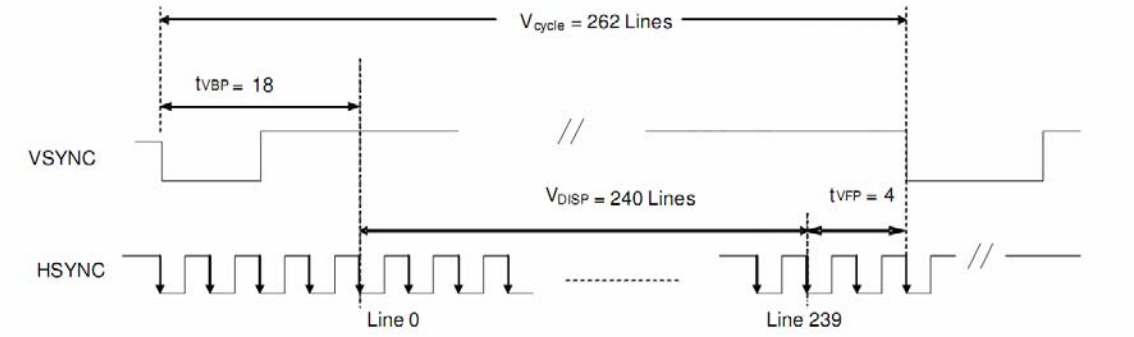
3.2 Paneelien ohjaus

Kuvainformaatio lähetetään paneelille pikseli kerrallaan alkaen vasemmasta yläreunasta. Ennen uuden rivin alkua näytönohjain asettaa vaakasynkronointilinjan (Hsync) aktiiviseksi ja odottaa horizontal back porch -ajan. Kuvan 3 mukaisessa signaloinnissa aktiivitila on alhaalla. Kun tämä aika on kulunut, voidaan lähettää varsinainen kuvainformaatio. Rivin pikselit lähetetään peräkkäin ilman erillistä synkronointia. Kun kaikki rivin pikselit on lähetetty, näytönohjain odottaa horizontal front porch -ajan lähettämättä taas mitään kuvainformaatiota. Tässä vaiheessa Hsync-linjan on oltava ei-aktiivisessa tilassa. Kun tämä aika on kulunut, alkaa uusi rivi.



Kuva 3. Ajoituskaavio yhden rivin lähettämiseksi [2, s. 57]

Ennen ensimmäisen rivin alkua näytönohjain asettaa pystysynkronointilinjan (Vsync) aktiiviseksi ja odottaa vertical back porch -ajan (kuva 4). Kun tämä aika on kulunut, voidaan siirtyä rivin ohjaukseen. Viimeisen rivin jälkeen näytönohjain odottaa vertical front porch -ajan. Tässä vaiheessa Vsync-linjan on oltava ei-aktiivisessa tilassa. Kun tämä aika on kulunut, alkaa uusi kuva.



Kuva 4. Ajoituskaavio yhden kuvan lähettämiseksi [2, s. 57]

Taulukko 1. HX8238A-ohjainpiirin ajoitukset näytönpäivitystaajuudella 60 Hz [2, s. 57]

horizontal back porch (t_{HBP})	68 pikseliä	10.6 µs
yksi rivi (H_{DISP})	320 pikseliä	49.9 µs
horizontal front porch (t_{HFP})	20 pikseliä	3.1 µs
vertical back porch (t_{VBP})	18 riviä	1.145 ms
yksi kuva (V_{DISP})	240 riviä	15.27 ms
vertical front porch (t_{VFP})	4 riviä	254.5 µs
pikselikello	1 pikseli	156 ns (6.414 MHz)

Taulukossa 1 on laskettu ajat, kun näytönpäivitys tapahtuu 60 kertaa sekunnissa. Pikselikellotaajuus voidaan laskea seuraavasti:
$(t_{HBP} + H_{DISP} + t_{HFP}) * (t_{VBP} + V_{DISP} + t_{VFP})$ * virkistystaajuus
= 6.414 MHz (jaksonaika 156 ns)

Jos käytössä on sarjamuotoinen RGB, pikselikellotaajuuden on oltava kolme kertaa suurempi, kuin tavallisessa RGB-väylässä.

Näiden signaalien lisäksi liitännässä kulkee data enable -linja. Tämä linja asetetaan aktiiviseksi, kun näytönohjain lähettää kuvainformaatiota. Toisin sanoen data enable -linja on aktiivisena H_{DISP}- ja V_{DISP}-aikoina. Jotkut paneelit osaavat synkronoitua pelkästään DE-linjan avulla, jolloin Hsync- ja Vsync-linjoja ei tarvitse kytkeä.

VGA-standardi ei määrittele ajoituksia VGA:ta (640x480) pienemmille resoluutioille, joten back porch ja front porch -ajat vaihtelevat eri paneeleilla. Oikeat ajat kannattaa tarkistaa paneelin datalehdestä. Yleensä paneeleilla, joissa on sama ohjainpiiri, myös ajoitukset ovat samoja.

4 ENSIMMÄINEN PROTOTYYPPI

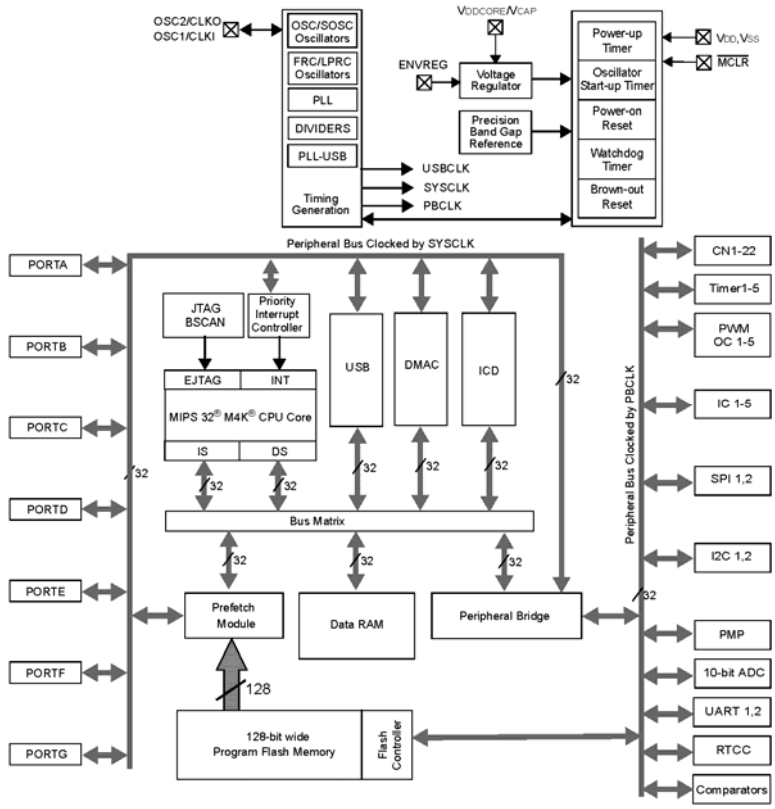
Ensimmäistä prototyyppiä lähdettiin rakentamaan valmiin kehitysalustan päälle, joka tilattiin kiinalaisesta verkkokaupasta techtoys.com.hk. Kehitysalusta koostuu prosessorikortista, näytönohjainkortista ja TFT-paneelista. Tähän kehitysalustaan päädyttiin, koska se sisältää tarpeeksi tehokkaan prosessorin ja sopivan näytönohjainpiirin. Prosessorina alustassa on Microchipin PIC32-sarjan yleiskäyttöinen mikrokontrolleri ja näytönohjainpiirinä Solomon Systech:n SSD1928-kuvaprosessori. Paneelina on standardi-RGB-liitännällä oleva 3,5-tuumainen TFT, jonka ohjainpiirinä on Himax HX8238A.

4.1 PIC32-prosessori

Microchip julkaisi vuonna 2007 uuden PIC32MX-tuoteperheen, joka koostuu 32-bittisistä mikrokontrollereista. PIC32 perustuu MIPS-arkkitehtuuriin ja M4K-ytimeen. Uuden tuoteperheen piirit ovat ainakin osittain pinniyhteensopivia vanhempien 16-

bittisten PIC24- ja dsPIC-piirien kanssa. Kaikkien PIC32-mikrokontrollereiden maksimikellotaajuus on 80 MHz, joka voidaan muodostaa sisäisen PLL:n avulla. PIC32:n erikoisominaisuuksia (joita ei löydy muista Microchipin prosessoreista) ovat esimerkiksi flash-ohjelmamuistin puskurointi, mahdollisuus suorittaa ohjelma RAM-muistista ja varjorekisterit, jotka muun muassa mahdollistavat lyhyet keskeytysten vasteajat.
[8, s. 2], [9]

Kehitysalustan prosessorina on PIC32MX360F512L, joka sisältää 512 kB ohjelmamuistia ja 32 kB datamuistia. Kuvassa 5 on esitetty prosessorin yleinen rakenne. Kuvasta käy hyvin ilmi, mitä oheislaitteita piiri sisältää.



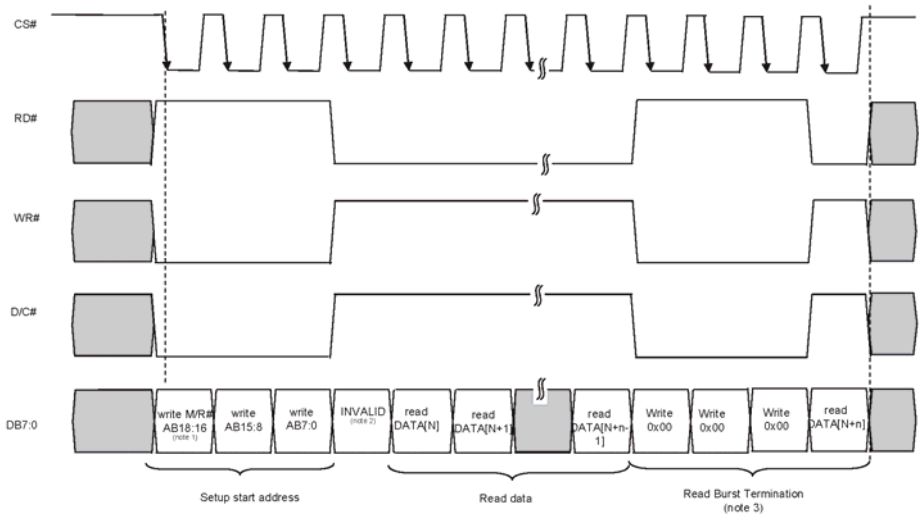
Kuva 5. PIC32MX3XX/4XX-piirin lohkokaavio [7, s. 13]

Microchipin valmistama prosessori valittiin sen takia, koska sekä Pikotec Oy:n suunnittelijoilla että tämän työn kirjoittajalla on kaikista eniten kokemusta kyseisen valmistajan prosessoreista. Heti alussa oli selvä, että kehitysalustan prosessori on hieman ylimitoitettu kehitettävään sovellukseen. Tämä ei kuitenkaan ollut ongelma, sillä koodin portattavuus Microchipin prosessoreiden (ainakin 16- ja 32-bittisten) välillä on suhteellisen hyvä.

4.2 SSD192x-näytönohjainpiiri

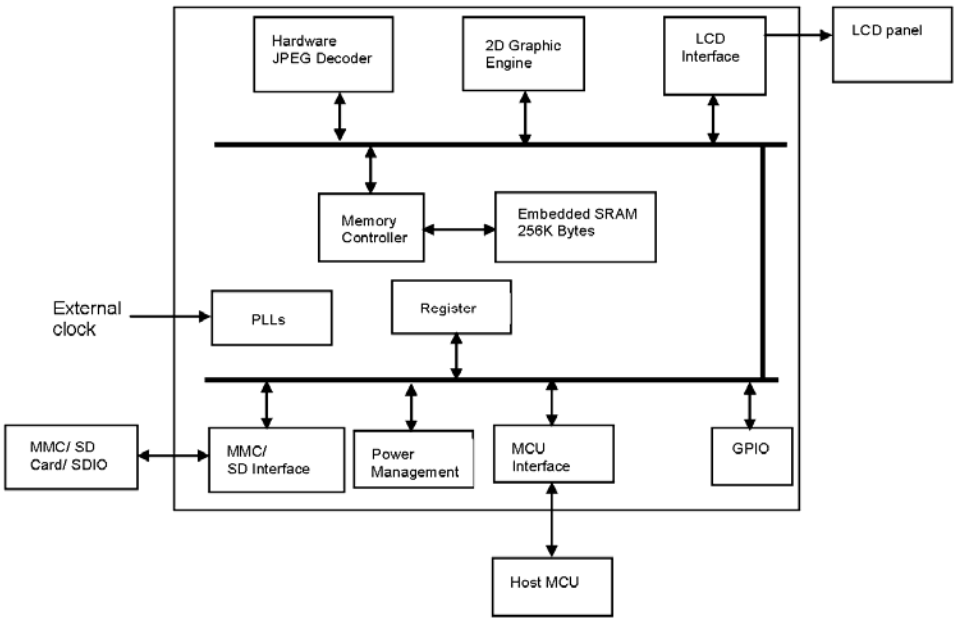
Solomon Systechin valmistama SSD192x on korkeasti integroitu kuvaprosessori, joka on ensisijaisesti tarkoitettu käytettäväksi LCD-näyttöjen ohjaamiseen mikrokontrolleri-sovelluksissa. Piirin avulla voidaan ohjata mustavalkoisia STN-paneeleja ja TFT-väripaneeleja. Piiri sisältää 256 kilotavua RAM-muistia, rautapohjaisen JPEG-koodekin, SD-muistikorttiliitännän ja se tukee 2D-grafiikan kiihdytystä. [24, s. 6-8]

Piiriä voidaan ohjata joko 8- tai 16-bittisen rinnakkaisen väylän kautta käyttäen joko suoraa tai epäsuoraa osoitusta. Tässä projektissa piiriä ohjataan 8-bittisen väylän kautta, epäsuora osoitus -moodissa. Datalinjojen lisäksi ohjaukseen tarvitaan RD- ja WR-linjat, jotka asetetaan aktiiviseksi joko luku- tai kirjoitusoperaation ajaksi. Näiden lisäksi D/C-linja asetetaan 0-tilaan, kun piirille lähetetään komentoja ja 1-tilaan, kun lähetetään tavallista dataa. CS-linjan tilan muutos ykkösestä nollaksi ilmoittaa piirille, että data on luettavissa väylältä tai että piiri voi itse kirjoittaa uutta dataa väylälle. Kuvassa 6 on esitetty esimerkki lukuoperaatiosta epäsuora osoitus -moodissa. Kuvassa AB-bitit ovat osoitebittejä (18-bittinen osoiteavaruus) ja M/R-bitin avulla valitaan, osoitetanko piirin rekistereitä vai muistia.



Kuva 6. Esimerkki lukuoperaatiosta [24, s. 27]

Piiristä on olemassa kaksi versiota: SSD1928 ja SSD1926. SSD1928 sisältää CMOS-kameraliitännän ja 8-bittisen RGB-väylän TFT-paneelin ohjaukseen. SSD1926:ssa ei ole kameraliitäntää, mutta RGB-väylä on piirissä 24-bittinen. Muuta eroavaisuutta piiri-versioiden välillä ei ole. Kuvassa 7 on esitetty SSD1926-piirin lohkokaavio.



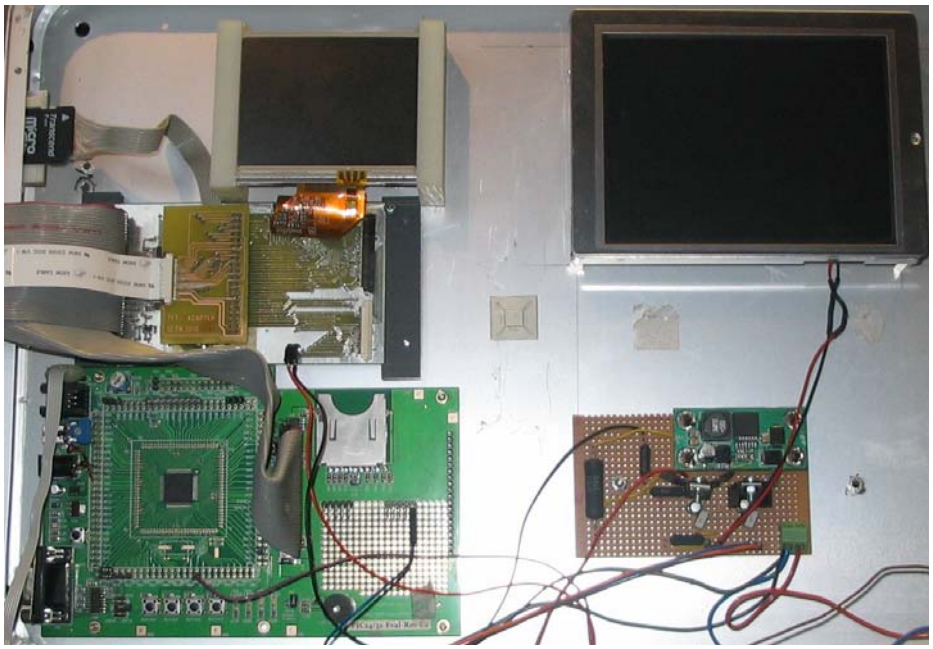
Kuva 7. SSD1926-piirin lohkokaavio [24, s. 9]

4.3 Oma näytönohjainkortti

Erilaisten TFT-paneelien testaukseen päätettiin rakentaa oma näytönohjainkortti. Kortti sisältää SSD1926-piirin, micro-SD-muistikorttilukijan ja FPC-liittimet erilaisille TFT-paneeleille. Piiriksi valittiin SSD1926 sen 24-bittisen RGB-väylän takia. Piirin näytekappaleet tilattiin suoraan Solomon Systech:ltä. Tämä kortti voidaan kytkeä prosessorikorttiin 40-napaisen nauhakaapelin avulla. Kortissa on laajennusmahdollisuus ja siihen voidaan kytkeä myöhemmin tehtyjä adaptereita uusille paneeleille. Näytönohjainkortin kytkentäkaavio on liitteessä 1.

Kehityksen aikana testattavana olivat 3,5- ja 5,7-tuumaiset TFT-paneelit, joiden resoluutio oli QVGA (320x240). Paneelien näytekappaleet tilattiin sekä suomalaisilta että ulkomaisilta elektroniikkatoimittajilta. Kaikki testatut paneelit oli valmistettu Kiinassa. Testauksen aikana havaittiin monien paneelien dokumentaation olevan puutteellista ja epätäsmällistä. Esimerkiksi monien valmistajien datalehdistä puuttui tieto paneelin ohjainpiiristä.

Kuvassa 8 on esitetty ensimmäisen prototyypin laitteisto: prosessorikortti, näytönohjainkortti, virtalähde ja TFT-paneelit.



Kuva 8. Ensimmäinen prototyyppi

5 OHJELMISTO

Laitteen ohjelmisto kirjoitettiin C-ohjelmointikielellä käyttäen Microchip:n kehitysympäristöä ja kääntäjiä. Ensimmäisen prototyypin prosessori oli 32-bittinen, joten kääntäjänä oli C32. Toisessa prototyypissä prosessori vaihdettiin 16-bittiseen ja kääntäjänä käytettiin C30 (tästä kerrotaan tarkemmin luvussa ”Sarjatuotantoversion prototyyppi”). Projektissa käytettiin kääntäjien ilmaisversioita, jotka eroavat täysversioista lähinnä optimointien osalta. Ilmaisversioiden korkein sallittu optimointitaso on 1/3 [6]. Tämä optimointitaso osoittautui kuitenkin täysin riittäväksi sekä koodin koon että suoritusnopeuden osalta, eikä täysversioiden lisenssejä tarvinnut hankkia.

Ohjelmisto kehitettiin bottom-up-menetelmällä. Aluksi luotiin kaikki tarvittavat alemman tason rutiinit, ja lopuksi niistä rakennettiin toimiva kokonaisuus. Tämän menetelmän ansiosta ohjelmiston kehitys pystyttiin aloittamaan jo siinä vaiheessa, kun täydellinen määrittely ei ollut vielä valmis. Esimerkiksi kehityksen alussa ei ollut täysin selvää, minkälainen pääohjelman pitäisi olla. Kuitenkin oli heti tiedossa, minkälainen näytönohjain laitteeseen tulee, joten ensimmäisenä kehitettiin kyseiselle piirille sopiva ajuri. Tässä luvussa kerrotaan ohjelmiston kehityksen vaiheista suurin piirtein samassa järjestyksessä, kuin miten ohjelmisto alun perin kehitettiinkin.

5.1 Näytönohjaimen ajuri

SSD192x-piirin ajurin pohjana käytettiin Microchipin grafiikkakirjastosta löytyvää ajuria. Kirjaston koodia laajennettiin ja muutettiin paremmin projektiin sopivaksi. Ajuri koostuu kahdesta osasta: alemman ja ylemmän tason funktioista. Alemman tason funktiot mahdollistavat kommunikoinnin näytönohjaimen kanssa. Funktioiden avulla voidaan lukea ja kirjoittaa 8-bittisen väylän kautta näytönohjaimen sisäisiä rekistereitä ja muistia. Myös piirin alustusfunktioita voidaan pitää alemman tason funktioina. Ajurin ylemmäksi tasoksi voidaan lukea kaikki funktiot, jotka mahdollistavat SSD192x-piirin ominaisuuksien käytön. Nämä ovat esimerkiksi JPEG-koodekin käyttö, muistikortin sektoreiden luku/kirjoitus ja grafiikan piirto.

5.1.1 Rekisterit ja muisti

SSD192x-piiri sisältää yli tuhat erilaista 8-bittistä rekisteriä, joista noin puolet on JPEG-koodekin hakutaulukkoja. Rekistereiden avulla ohjataan piirin toimintaa, kuten esimerkiksi säädetään piirin kellotaajuusasetuksia sekä RGB-liitännän asetuksia ja ajoituksia, ohjataan JPEG-koodekkia sekä luetaan ja kirjoitetaan muistikorttia.

Jotkut rekisterit koostuvat monesta 8-bittisestä rekisteristä. Näiden rekistereiden käytön helpottamiseksi on tehty omat luku/kirjoitus-funktiot. Kaikkien rekistereiden kirjoitus-funktiot ottavat parametreina rekisterin osoitteen ja kirjoitettavan datan.

```
void SSD192x_write_reg_BYTE(WORD addr, BYTE data);
void SSD192x_write_reg_WORD(WORD addr, WORD data);
void SSD192x_write_reg_DWORD(WORD addr, DWORD data);
```

Lukufunktiot ottavat parametreina pelkästään rekisterin osoitteen ja palauttavat luetun datan.

```
BYTE SSD192x_read_reg_BYTE(WORD addr);
WORD SSD192x_read_reg_WORD(WORD addr);
DWORD SSD192x_read_reg_DWORD(DWORD addr);
```

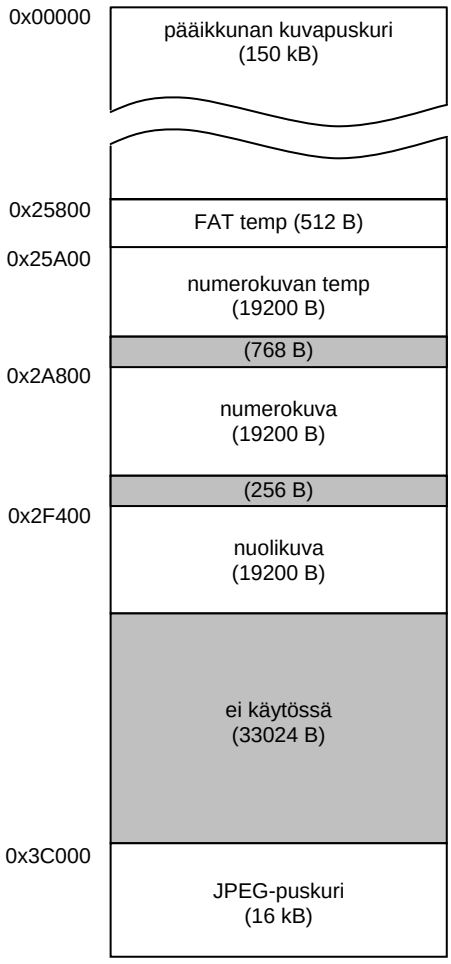
Piirin sisäistä RAM-muistia voidaan lukea ja kirjoittaa joko yksi tai monta tavua kerrallaan. Yhden tavun luku/kirjoitus-funktiot toimivat samalla periaatteella, kuin rekistereiden luku/kirjoitus-funktiot. Monen tavun luku/kirjoitus-funktiot ottavat parametreina muistiosoitteen, josta luku/kirjoitus alkaa, osoittimen taulukkoon, joka sisältää kirjoitet-

tavan datan tai johon luettava data tallennetaan ja luettavien/kirjoitettavien tavujen määrän.

```
void SSD192x_write_mem_BYTE(DWORD addr, BYTE data);
void SSD192x_write_mem_array(DWORD addr, BYTE *data, WORD size);

BYTE SSD192x_read_mem_BYTE(DWORD addr);
void SSD192x_read_mem_array(DWORD addr, BYTE *data, WORD size);
```

Piirin muistia käytetään moneen eri tarkoitukseen (kuva 9). Suuremman osan muistista vie pääikkunan kuvapuskuri, jonka koko on 150 kB. Kuvapuskurin jälkeen on 512 tavun kokoinen paikka, johon voidaan väliaikaisesti lukea yksi sektori muistikortilta. Tämän paikan merkitys on selitetty tarkemmin luvussa 5.2.2 (Datan lukeminen). Tämän lisäksi muistissa on varattu paikat kerrosnumero- ja nuolikuville. Näistä kuvista kerrotaan tarkemmin luvussa 5.6 (Grafiikan näyttäminen). Muistialueen loppupäässä sijaitsee JPEG-koodekin FIFO-puskuri, jonka koko on 16 kB. Puskurin merkitys on selitetty tarkemmin luvussa 5.4 (JPEG-kuvien näyttäminen).



Kuva 9. SSD192x-piirin muistikartta

5.1.2 SSD192x-piirin alustus

Aluksi alustusfunktiossa säädetään mikrokontrollerin rinnakkaisportin asetukset ja määritetään muut ohjauslinjat (RD, WR, CS, D/C ja RESET) ulostuloiksi. Seuraavaksi SSD192x-piiri resetoidaan asettamalla RESET-linja noin 50 millisekunniksi 0-tilaan. Tämän jälkeen mikrokontrolleri voi kirjoittaa ja lukea dataa piiriltä.

SSD192x-piiri käyttää kellolähteenä ulkoista 4 MHz kidettä. Alustuksessa piirin sisäinen kellotaajuus muodostetaan PLL:n avulla. Tässä projektissa näytönohjainpiiri toimii 85 MHz kellotaajuudella, mikä on kyseisen piirin maksimikellotaajuus [24, s. 20]. Seuraavaksi alustusfunktiossa säädetään TFT-paneelin ohjaukseen liittyviä asetuksia.

SPI-ohjausväylä

Luvussa 3.2 (Paneelien ohjaus) kerrotaan, että monet 3,5-tuumaiset paneelit sisältävät erillisen SPI-ohjausväylän. Tämän väylän kautta alustuksessa määritetään paneelin RGB-väylän moodi (sarja/tavallinen), käännetään kuva tarvittaessa vaaka- ja pystysuunnassa sekä säädetään kuvan värejä.

SSD192x-piiri sisältää viisi yleiskäyttöistä GPIO-linjaa, joiden tilaa voidaan muuttaa piirin rekistereiden avulla. Tässä projektissa SPI-ohjausväylää käytetään näiden GPIO-linjojen avulla. Sarjamuotoinen data ja kellopulssi muodostetaan ohjelmallisesti kirjoittamalla SSD192x-piirin rekistereitä. SPI-ohjausväylää käytetään yksisuuntaisesti: data lähetetään näytönohjainpiiriltä paneelin ohjainpiirille. Datan siirtonopeudella ei tässä tapauksessa ole suurta väliä, sillä ohjausväylää käytetään vain alustuksen yhteydessä. Toinen, ehkä perinteisempi tapa, olisi kytkeä paneelin ohjausväylä suoraan mikrokontrollerin SPI-väylään.

Ajoitukset

Ajoitusasetukset vaihtelevat eri TFT-paneeleilla. Näytönohjaimen rekistereiden avulla voidaan määrittää esimerkiksi RGB-väylän moodi (sarja/tavallinen), kuvan koko (pikseleinä), back ja fron porch -ajat, synkronointisignaalit ja pikselikellotaajuus. Tässä projektissa kuva päivitetään 60 kertaa sekunnissa.

Pääikkuna

SSD192x-piiri tukee pääikkunan ja liikkuvan ikkunan käyttöä. Pääikkuna osoittaa näytönohjaimen RAM-muistissa olevaan kuvapuskuriin (frame buffer) ja näkyy aina koko näytön kokoisena. Liikkuva ikkuna näkyy aina pääikkunan päällä. Jos liikkuva ikkuna on käytössä, sille on oltava muistissa oma kuvapuskuri. [23, s. 87]

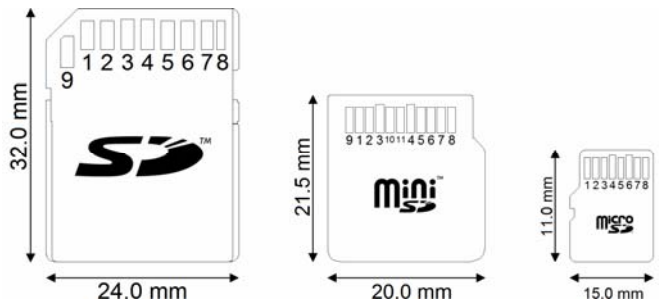
Alustuksessa määritellään, mistä muistiosoitteesta pääikkunan kuvapuskuri alkaa. Lisäksi määritellään ikkunan värimoodi, joka voi olla 8-, 16- tai 32-bittinen. Värimoodi määrittelee, miten pikseleiden värit on tallennettu kuvapuskurissa. Tässä projektissa käytetään 16-bittistä värimoodia, jolloin yhden pikselin punaisen värisävyn ilmaisemiseen käytetään sanan ensimmäiset 5 bittiä, vihreän keskimmäiset 6 bittiä ja sinisen loput 5 bittiä. Tässä värimoodissa pääikkunan kuvapuskuri vie 150 kB:
320 * 240 * 2 = 153600 B = 150 kB

Taulukko 2. Yhden pikselin sisältö

punainen					vihreä						sininen				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

5.2 SD-muistikortti

Laitteessa käytetään microSD-muistikorttia kuvien, videoiden ja konfiguraatiotiedoston säilyttämiseksi. SD (Secure Digital) on SD-järjestön kehittämä flash-muistikorttityyppi. Kortin tekniikka perustuu Toshiban kehittämään MMC-korttiin ja ne ovatkin osittain yhteensopivia. SD-kortteja on olemassa kahta eri tyyppiä: tavallinen ja High Capacity. Molemmasta tyypistä on olemassa kuvan 10 mukaisesti kolme fyysisesti erikokoista versiota: SD, miniSD ja microSD. [21], [22, s. 17]



Kuva 10. SD-muistikorttien versiot

Tässä projektissa käytetään tavallisia (ei SDHC) microSD-kortteja niiden hyvän hinta/kapasiteetti/koko-suhteen takia. microSD-kortin pinnijärjestys ei ole yhteensopiva SD- ja miniSD-korttien kanssa, mutta liitännän signaalit ovat kuitenkin kaikissa versioissa samat. SD-kortteja voidaan ohjata joko SD- tai SPI-moodissa. SD-moodissa dataväylä voi olla joko 1- tai 4-bittinen.

Taulukko 3. SD-liitännän signaalit [20, s. 23-25]

SD-moodi		SPI-moodi		SD	mini SD	micro SD
VSS	Maa	VSS	Maa	3, 6	3, 6	6
VDD	Käyttöjännite (3.3 V)	VDD	Käyttöjännite (3.3 V)	4	4	4
CLK	Kello	SCLK	Kello	5	5	5
CMD	Komento/vastaus	DI	Datan vastaanotto	2	2	3
DAT0	4-bittinen dataväylä	DO	Datan lähetys	7	7	7
DAT1		-		8	8	8
DAT2		-		9	9	1
DAT3/CD		CS	Piirinvalintasignaali	1	1	2
NC	Ei kytketty	NC	Ei kytketty	-	10, 11	-

5.2.1 Muistikortin alustus

SSD192x-piiri sisältää SD-korttiliitännän, joka voidaan asettaa toimimaan joko 1- tai 4-bittiseenä. SPI-liitäntä ei ole tuettu. Tässä projektissa käytetään 4-bittistä moodia sen nopeuden takia. Muistikortin ohjaus tapahtuu lähettämällä kortille komentoja ja odottamalla kortin vastausta niihin. SSD192x-piirissä on rekisterit komentojen ja datan lähettämistä ja vastaanottoa varten. Näitä rekistereitä käyttämällä voidaan esimerkiksi alustaa muistikortti (tässä tapauksessa alustuksella ei tarkoiteta muistikortin formatointia, vaan muistikortin ohjainpiirin alustusta) sekä lukea ja kirjoittaa sektoreita muistikortille.

Ennen kuin kortille voidaan lähettää komentoja tai dataa, on säädettävä liitännän moodi ja nopeus kohdalleen. SD-liitännän CLK-linjan taajuus muodostetaan SSD192x-piirin sisäisestä taajuudesta, joka on tässä projektissa 85 MHz. SD-muistikortin maksimikellotaajuus on 25 MHz [22, s. 17]. Maksimi alle 25 MHz taajuus, joka voidaan muodostaa piirin sisäisten jakajien avulla, on 21.25 MHz. Vaikka tällä taajuudella tiedonsiirtonopeus muistikortin ja SSD192x-piirin välillä ei ole maksimi mahdollinen, se on kuitenkin osoittautunut riittäväksi tässä projektissa.

Kun liitäntä on alustettu, voidaan lähettää alustuskomentoja kortille. Tässä projektissa SD-kortin alustussekvenssi on otettu Microchipin ajurin koodista. Luettavan ja kirjoitettavan sektorin koko on 512 tavua.

5.2.2 Datan lukeminen

SSD192x-piirin avulla data voidaan lukea kortilta kahdella eri tavalla: joko datarekisterin kautta mikrokontrollerin muistiin tai käyttäen DMA:ta (suoramuistiosoitus), näytönohjaimen muistiin. Ensin valitaan asetusrekisterin avulla, mitä moodia käytetään ja sitten kortille lähetetään sektorinlukukomento. Kun data on luettavissa, tilarekisterissä vaihtuu lukubitin tila. Tavallista lukumoodia käyttäessä kaikki sektorin tavut luetaan samasta rekisteristä. Rekisterin sisältö vaihtuu automaattisesti jokaisen lukuoperaation jälkeen. Funktio, joka toteuttaa tämän operaation, ottaa parametreina muistikortin sektorin osoitteen ja osoittimen mikrokontrollerin muistiin, johon data luetaan.

```
BYTE SDSectorRead(DWORD sector_addr, BYTE *buffer);
```

Vaikka tämä on kätevä ja helppo tapa lukea data prosessorin muistiin, se ei kuitenkaan ole kovin nopea. Jos halutaan suurempaa siirtonopeutta, on käytettävä DMA-lukua.

DMA-moodissa lukukomennon lähettämisen jälkeen data siirretään kortilta automaattisesti SSD192x-piirin RAM-muistiin. Funktio, joka toteuttaa tämän operaation, ottaa parametreina muistikortin sektorin osoitteen ja osoitteen SSD192x-piirin muistiin, johon data luetaan.

```
BYTE SDSectorDMARead(DWORD sector_addr, DWORD dma_addr, WORD num_blk);
```

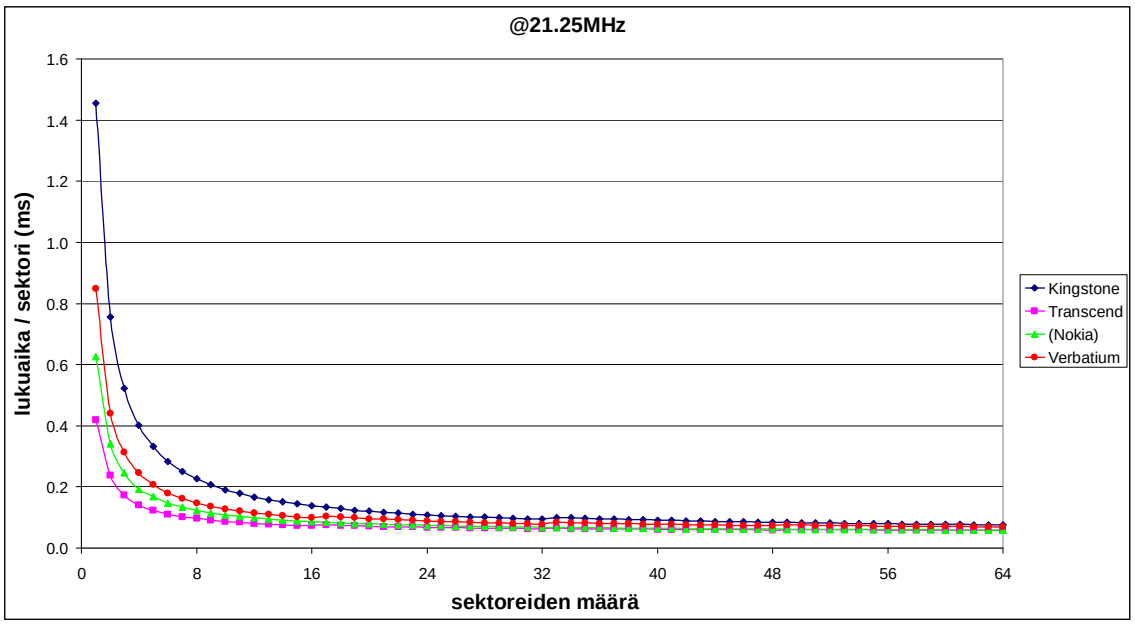
Kun kaikki sektorin tavut on siirretty, tilarekisterissä vaihtuu lukubitin tila. Tämän jälkeen data on luettavissa piirin muistista käyttäen esimerkiksi monen tavun muistinlukufunktiota.

```
SDSectorDMARead( sector, SDcard_temp, 1 );
SSD192x_read_mem_array( SDcard_temp, buff, 512 );
```

Vaikka tämä tapa näyttää monimutkaisemmalta, se on kuitenkin huomattavasti nopeampi, kuin tavallinen (rekisterin kautta) lukumoodi. DMA-lukumoodin ainoana rajoitteena on se, että piirin muistissa on oltava 512 tavun kokoinen paikka, johon data voidaan väliaikaisesti lukea. Tämä ei kuitenkaan ole kovin iso ongelma, sillä piirissä on yhteensä 256 kilotavua muistia.

Joissain tapauksissa dataa ei edes tarvitse välittää mikrokontrollerille asti, vaan riittää, että se luetaan SSD192x-piirin muistiin. Esimerkiksi dekoodatessa JPEG-kuvia kuvatiedosto kannattaa lukea suoraan JPEG-koodekin FIFO-puskuriin (joka sijaitsee muistissa). Tässä tapauksessa kontrolleri ei ota kantaa kuvatiedoston sisältöön. Pakkaamattomat kuvatiedostot voidaan joissain tapauksissa lukea suoraan kuvapuskuriin, jos tiedoston muoto on sopiva.

Data voidaan lukea muistikortilta myös monta sektoria kerrallaan. Ennen lukukomentoa kortille kerrotaan, montako sektoria halutaan lukea. Kun kaikki sektorit on siirretty SSD192x-piirin muistiin, tilarekisterissä vaihtuu lukubitin tila. Lukemalla monta sektoria kerrallaan saavutetaan paras mahdollinen tiedonsiirtonopeus.



Kuva 11. Muistikorttien lukunopeudet

Kuvassa 11 on esitetty eri valmistajien muistikorttien sektorinlukunopeudet, kun data luetaan kortilta käyttäen monen sektorin lukua. X-akselilla on luettujen sektoreiden määrä ja Y-akselilla on yhden sektorin lukemiseen käytetty aika. Kuvasta huomataan, että mitä enemmän sektoreita luetaan kerrallaan, sitä vähemmän aikaa keskimäärin kuluu yhden sektorin lukemiseen. Parhaimmillaan päästään aikoihin 60 µs/sektori, mikä tarkoittaa noin 8 MB/s:ssa.

5.3 FAT-tiedostojärjestelmä

Muistikortilla on oltava tiedostojärjestelmä, jotta kortin sisältö voitaisiin helposti muokata tietokoneen avulla. Tässä projektissa käytetään Microsoftin kehittämää FAT-tiedostojärjestelmää. FAT (File Allocation Table) on alun perin kehitetty MS-DOS-käyttöjärjestelmää varten, mutta myöhemmin sitä on käytetty monen Windows-käyttöjärjestelmän päätiedostojärjestelmänä. Nykyään USB-muistitikuissa ja muistikorteissa käytetään yleensä FATtia sen yksinkertaisuuden ja hyvän käyttöjärjestelmätuen vuoksi. [11]

On olemassa monia FAT-tiedostojärjestelmää tukevia kirjastoja. Jotkut niistä on kehitetty erityisesti sulautetuille järjestelmille ja vaativat mahdollisimman vähän resursseja prosessorilta. Tässä projektissa käytettiin vapaan lähdekoodin FatFs-kirjastoa, joka tukee sekä FAT16- että FAT32-tiedostojärjestelmiä. Myös FAT32:n pitkät tiedostonimet ovat tuettuja. Kirjasto tarvitsee toimiakseen muistikortin sektorin luku- ja kirjoitusfunktiot alemmalta tasolta. Kirjaston avulla voidaan esimerkiksi avata tiedostoja luku- ja kirjoitusmoodissa, luoda uusia tiedostoja ja hakemistoja, selata hakemistoja ja suorittaa muita hyödyllisiä toimintoja tiedostojärjestelmän sisällölle. Tässä projektissa prosessorin ei ole tarkoitus kirjoittaa muistikortille mitään, joten FAT-kirjastoa käytetään lukumoodissa. Tällöin kirjasto vie vähemmän muistia, koska kirjoitusfunktiot jäävät pois käännöksestä. [1]

Taulukko 4. FatFs-kirjaston muistinkulutus

konfiguraatio	minimointi-taso	C32 (PIC32)	C30 (PIC24/dsPIC)
vain lukufunktiot	3	4.5 kB	2.7 kB
vain lukufunktiot	2	5.1 kB	3.1 kB
vain lukufunktiot + pitkän tied.nimet	2	6.6 kB	3.8 kB
luku- ja kirjoitus	2	9.2 kB	5.7 kB
luku- ja kirjoitus	1	10.0 kB	6.1 kB
luku- ja kirjoitus	0	12.8 kB	7.8 kB
luku- ja kirjoitus + pitkän tied.nimet	2	11.3 kB	6.7 kB
luku- ja kirjoitus + pitkän tied.nimet	1	12.6 kB	7.3 kB
luku- ja kirjoitus + pitkän tied.nimet	0	15.5 kB	9.1 kB

Taulukossa 4 on esitetty FatFs-kirjaston viemät ohjelmamuistin määrät eri konfiguraatioilla ja kääntäjillä. Minimointitasojen merkitykset on selitetty kirjaston dokumentaatiossa ja lähdekoodin kommenteissa. Molempien kääntäjien optimointitasona on ollut 1/3.

5.4 JPEG-kuvien näyttäminen

Hissinäytössä jokaiselle kerrokselle voidaan määrittää oma taustakuvansa. Kuvat ovat JPEG-tiedostoja ja ne sijaitsevat muistikortin juurihakemistossa. JPEG on Joint Photographic Experts Group -komitean kehittämä kuvien tallennusmuoto. JPEG-pakkaus on häviöllinen, ja se soveltuu parhaiten pehmeitä reunoja sisältävien kuvien (esimerkiksi valokuvien) pakkaamiseen. Monet kuvankäsittelyohjelmat tukevat JPEG-tiedostoja ja digitaalikameroiden valmistajat käyttävät sitä valokuvien tallennusmuotona. [3], [4]

SSD192x-piiri sisältää rautapohjaisen JPEG-koodekin, jonka avulla voidaan dekoodata JPEG-pakattuja kuvia. Koodekille voidaan syöttää dataa FIFO-puskurin kautta. Puskuri sijaitsee SSD192x-piirin muistissa, ja sen paikka siellä on vapaasti määritettävissä. Puskurin kokoa voidaan säätää välillä 4 – 256 kB neljän kilotavun askelin. Prosessori voi kirjoittaa dataa FIFO-puskuriin esimerkiksi käyttämällä muistiinkirjoitusfunktioita, mutta koska tässä projektissa kaikki kuvat sijaitsevat muistikortilla, luetaan ne suoraan sieltä DMA-lukufunktioiden avulla. Koodekki on hyvin nopea, ja pullonkaulaksi jää muistikortin lukunopeus. Esimerkiksi kuvan, jonka resoluutio on 2048x1536 ja koko 1.5 MB, näyttäminen kestää noin 200 ms. Mitä isompi FIFO-puskuri on, sitä enemmän sektoreita voidaan lukea siihen kerrallaan. Kuvasta 11 huomataan, että puskurin koon on hyvä olla vähintään 8 kB. Tässä projektissa puskurin kooksi on asetettu 16 kB.

Koodekki sisältää rautapohjaisen skaalaimen, jonka avulla kuvat voidaan pienentää näytölle sopiviksi. Skaalaimen asetukset ovat tosin hieman rajoittuneet, sillä kuvaa voidaan pienentää vain 1/2-, 1/4- ja 1/8-kokoon alkuperäisestä.

Koodekille voidaan määrittää paikka muistissa, johon se kirjoittaa dekoodattua kuvadataa. Tässä projektissa kuvat dekoodataan suoraan pääikkunan kuvapuskuriin, jolloin ne ilmestyvät näytölle heti, kun koodekki on saanut kirjoitetuksi dataa muistiin.

Funktio, joka hoitaa JPEG-kuvien näyttämisen, ottaa parametreina JPEG-tiedoston ensimmäisen sektorin osoitteen ja tiedoston koon tavuina.

```
BYTE show_jpeg(DWORD file_sector, DWORD file_size);
```

Kuvan dekoodaus tapahtuu kahdessa vaiheessa. Ensin koodekille on lähetettävä tiedoston otsikko-osa (header), josta koodekki hakee kuvan parametrit. Lähettäjän ei tarvitse tietää, missä vaiheessa otsikko-osa loppuu, koska koodekin tilarekisteristä näkee, kun koodekki on saanut kaikki tarvittavat tiedot kuvasta. Kun otsikko-osa on käsitelty, koodekin rekistereistä saadaan kuvan resoluutio selville. Resoluution perusteella säädetään skaalaimen asetuksia. Skaalaimessa pyritään käyttämään sellaista jakajaa, jolla kuva saataisiin pienenetyksi mahdollisimman lähelle näytön resoluutiota (320x240). Tämän jälkeen voidaan siirtyä kuvainformaation dekoodaukseen. Yksinkertaisesti koko tiedosto, mukaan lukien otsikko-osa, lähetetään koodekille. Koodekin tilarekisteristä näkee, onko dekoodauksen aikana tapahtunut virheitä.

5.5 Videon toisto

Taustakuvien lisäksi hissinäytössä on mahdollisuus toistaa videoita soittolistan avulla. Videon tiedostomuotona käytetään AVIa ja itse video on pakattu MJPEG-pakkauksella.

5.5.1 AVI-tiedostomuoto

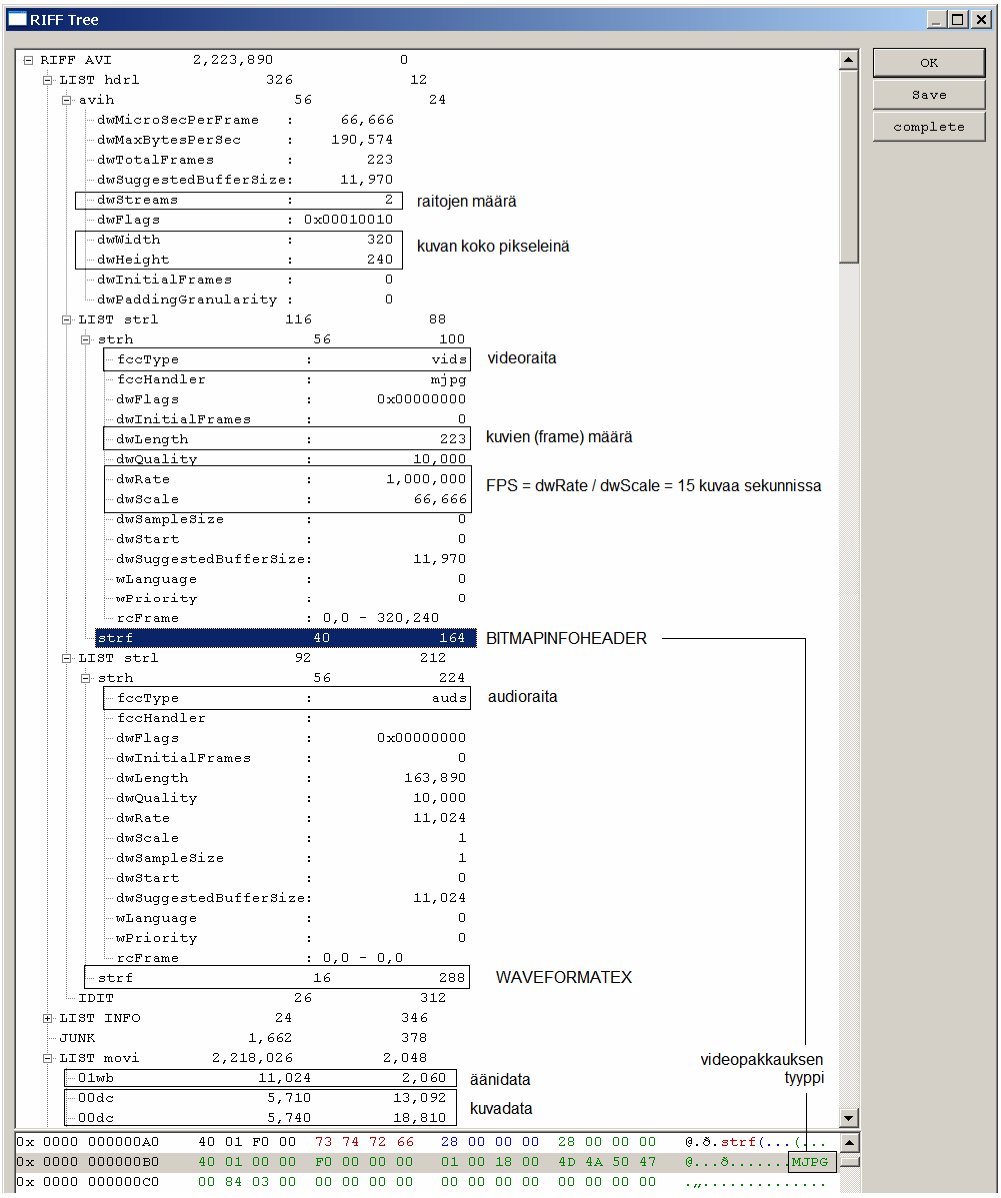
AVI (Audio Video Interleave) on Microsoftin kehittämä tiedostomuoto, joka voi sisältää ääni- ja videoinformaatiota. AVI on RIFF-tyyppinen tiedostomuoto. RIFF puolestaan pohjautuu Electronic Arts:n kehittämään IFF-tiedostomuotoon. IFF-tiedostot koostuvat datalohkoista (CHUNK). Jokainen lohko alkaa ID-tunnuksella, jota sanotaan myös FourCC-tunnukseksi. Tunnuksen avulla ohjelma, joka lukee tiedostoa, tietää, minkälainen lohko on kyseessä ja mitä sille pitäisi tehdä. Tunnuksen jälkeen seuraa nelitavuinen luku, joka kertoo lohkon koon tavuina. Kun ohjelma tietää, minkä kokoinen lohko on kyseessä, se voi päätellä, mistä alkaa seuraava lohko. Kokotavujen jälkeen alkaa lohkon dataosa. [10]

```
typedef struct {
    DWORD dwFourCC
    DWORD dwSize
    BYTE data[dwSize]
} CHUNK;
```

Lohkojen lisäksi AVI-tiedosto sisältää listoja (LIST). Lista pitää sisällään muita listoja ja lohkoja.

```
typedef struct {
    DWORD dwList    //tämä voi olla ”LIST” tai ”RIFF”
    DWORD dwSize
    DWORD dwFourCC
    BYTE data[dwSize-4]
} LIST;
```

AVI-tiedosto alkaa otsikko-osalla, joka sisältää kaikki tarvittavat tiedot videosta. Tiedot on jaettu lohkoihin, jotka ovat listojen sisällä. Alexander Noe on tehnyt kätevän ohjelman [16], jonka avulla AVI-tiedoston sisällön tutkiminen on todella helppoa.



Kuva 12. AVI-tiedoston sisältö

Kuvassa 12 ohjelman avulla tutkitaan Canonin digitaalikameran tuottamaa AVI-tiedostoa, joka on pakattu MJPEG-pakkauksella. Tiedosto alkaa otsikkolistalla, jonka sisällä kaikki otsikkolohkot sijaitsevat. Ensimmäinen otsikkolohko on tiedoston pääotsikko (avih, AVI header). Tästä lohkosta ohjelman kannattaa lukea tiedoston raitojen määrä ja videokuvan koko. Vaikka muutkin lohkon tiedot ovat hyödyllisiä, niitä ei kannata lukea tästä lohkosta. Enkooderiohjelmistot kirjoittavat aina tiedot videon toistonopeudesta ja kuvien (frame) määrästä videoraidan otsikkolohkoon, muttei välttämättä pääotsikkoon. [15, s. 8-10]

Pääotsikkolohkon jälkeen seuraa lista raitojen otsikkolohkoista (strh, stream header). Esimerkkitapauksessa ensimmäinen raita on videoraita. Raidan tyypin jälkeen tuleva fccHandler-arvo kertoo, mitä koodekia lukuohjelman tulisi käyttää videon toistamiseen. Tästä arvosta ei kuitenkaan kannata päätellä, mitä pakkausta videossa on käytetty. Esimerkiksi suosittu ffdshow-enkooderi voi joskus (riippuen asetuksista) kirjoittaa tähän oman FourCC-tunnuksen (ffds), mikä ei siis kerro, millä koodekilla video on pakattu. Koodekin tyypin saa selville strf (stream format) -lohkosta.

Videoraidan otsikkolohkosta kannattaa tarkistaa, kuinka monta kuvaa video yhteensä sisältää. Tämän lisäksi dwRate- ja dwScale-arvojen avulla voidaan laskea, kuinka monta kuvaa sekunnissa on näytettävä. Tässä projektissa ohjelmistossa on varattu oma ajastin videon toistoa varten. Ajastin viritetään aiheuttamaan keskeytys silloin, kun on uuden kuvan näyttämisen aika. Esimerkiksi jos prosessorin kellotaajuus on 80 MHz, ajastimen jakaja 256 ja videon toistonopeus 15 kuvaa/s, voidaan ajastimen laskurin raja-arvo laskea seuraavasti:
yhden kuvan aika = 1 / (dwRate / dwScale)
yksi ajastimen ”tikki” = 1 / (80 MHz / 256)
laskurin raja-arvo = yhden kuvan aika / yksi ajastimen ”tikki”

Raidan otsikkolohkon jälkeen seuraa raidan formaattilohko (strf, stream format), jonka sisältö riippuu raidan tyypistä. Videoraidan tapauksessa sisältö vastaa Microsoftin MSDN-kirjaston BITMAPINFOHEADER-tietotyyppiä. Tästä lohkosta kannattaa lukea videopakkauksen tyypin FourCC-tunnus. Yleisimpien pakkausten tunnukset voidaan tarkistaa Microsoftin MSDN-sivuilta [12]. Esimerkiksi MJPEG-pakkauksen FourCC-tunnus on ”MJPG”.

Videoraitalistan jälkeen alkaa lista mahdollisista ääniraidoista. Ääniraidan otsikko-osassa ei ole sellaisia tietoja, joita lukuohjelman tulisi välttämättä tietää. Äänipakkauksen tyypin saa selville formaattilohkosta, joka ääniraidan tapauksessa vastaa Microsoftin MSDN-kirjaston WAVEFORMATEX-tietotyyppiä. Pakkauksen tyyppi ilmoitetaan numerotunnuksen avulla. Yleisimpien pakkausten tunnukset voidaan tarkistaa Microsoftin MSDN-sivuilta ja MSDN-kirjaston mmreg.h-tiedostosta [25]. Esimerkiksi PCM-formaatin (ei pakkausta) tunnus on 0x0001 ja MP3-pakkauksen tunnus 0x0055.

Otsikkolohkojen jälkeen alkaa lista datalohkoista (movi). AVI-tiedoston datalohkot voivat sisältää kuvadataa, ääntä tai tekstiä. Näiden lohkojen FourCC-tunnukset koostuvat kahdesta ASCII-koodatusta heksadesimaaliluvusta ja kahdesta kirjaimesta. Numerot ilmaisevat raidan (stream) numeron ja kirjainyhdistelmä kertoo, minkä tyyppinen lohko on kyseessä.

Taulukko 5. AVI-tiedoston datalohkojen tunnukset

FourCC-tunnus	raita	tyyppi
00dc	ensimmäinen	videodata
01wb	toinen	äänidata
02tx	kolmas	teksti

On huomattava, että vaikka raidat ovat erityyppisiä, numerointi on kaikille yhteinen. Yhdessä tiedostossa ei siis voi olla esimerkiksi kahta ensimmäistä raitaa videolle ja äänelle.

5.5.2 MJPEG-pakkaus

Motion JPEGillä pakatussa videossa jokainen kuva (frame) on pakattu erikseen JPEG-pakkauksella. MJPEG:n pakkaussuhde on huono verrattuna kehittyneempiin pakkausalgoritmeihin, kuten esimerkiksi MPEG1/2/4:een. MJPEGin hyvä puoli on kuitenkin se, että laitteistolta vaaditaan vähemmän tehoa sen dekoodaukseen. Tässä projektissa apuna käytetään rautapohjasta JPEG-dekooderia, mutta tarpeeksi tehokkaalla prosessorilla/mikrokontrollerilla dekoodaus onnistuu myös ohjelmallisesti. [13]

MJPEG-pakatussa AVI-tiedostossa jokainen kuva on tallennettu omaan videolohkoon (00dc). Jokainen kuva on oma JPEG-tiedosto, jossa on mukana sekä otsikko- että data-

osat. Otsikko-osasta kuitenkin puuttuu Huffman-taulukko. Ilmeisesti enkooderit jättävät taulukon pois tilan säästämiseksi, sillä taulukko on sama kaikille videon kuville [14]. Tämän takia dekooderille on lähetettävä kyseinen taulukko erikseen ennen muuta videolohkon sisältöä. Taulukko voidaan ottaa mistä tahansa JPEG-tiedostosta. Muuten kuvien dekoodaus tapahtuu samalla periaatteella, kuin tavallisen JPEG-kuvan dekoodaus, joka on selitetty luvussa 5.4 (JPEG-kuvien näyttäminen). Skaalaimen asetukset säädetään vain kerran videon ensimmäisen kuvan kohdalla.

Laitteeseen sopivat videotiedostot voidaan luoda erilaisten videon editointiohjelmistojen avulla. Verkosta löytyy monta ilmaista ohjelmaa, joiden avulla voidaan muuntaa olemassa olevan videon tallennusmuotoa ja pakkausta. Kehityksen aikana testivideot luotiin VirtualDub-ohjelman avulla käyttäen ffdshow-enkooderia. Käyttäjäystävällisempi vaihtoehto on esimerkiksi ilmainen MediaEncoder-ohjelmisto, joka tukee monia tiedostomuotoja ja pakkauksia. Ohjelman asetuksista valitaan tiedoston tallennusmuodoksi AVI ja pakkaustyypiksi MJPEG. Tämän lisäksi videon resoluutioksi kannattaa asettaa 320x240, jotta laitteen ei tarvitse ajon aikana erikseen skaalata videota. Videoita ei välttämättä tarvitse luoda tietokoneella, vaan esimerkiksi Canonin digitaalikameroiden tuottamat videot toimivat laitteessa suoraan.

5.6 Grafiikan näyttäminen

Hissinäytön ehkä tärkein tehtävä on ilmaista, minkä kerroksen kohdalla hissi kullakin hetkellä sijaitsee. Tämän lisäksi nuolien avulla voidaan ilmaista, mihin suuntaan hissi on menossa. Kerrosnumerot ja nuolet näytetään aina taustalla olevan kuvan tai videon päällä. Ongelmaksi muodostuu se, ettei SSD192x-piiri tue varsinaista OSD (On-screen display) -grafiikkaa. Piiri tukee kuitenkin hiiren (tai jonkin muun osoitinlaitteen) kursorin piirtämistä kuvan päällä. Kursoreita voi olla kaksi kappaletta ja niiden kokoa ei ole rajoitettu mitenkään. Kursori voi siis olla vaikka koko näytön kokoinen. Kursorit ovat aina kaiken muun grafiikan päällä. Kursoreiden kuvainformaatio ei sijaitse kuvapuskurissa vaan muualla piirin muistissa. Kursorikuvan paikka näytössä on vapaasti säädettävissä. Kursorikuvalle on kuitenkin omat rajoitteensa: kuva voi sisältää yhtä aikaa maksimissaan vain neljä eri väriä, joista yksi on läpinäkyvä väri. Tässä projektissa ensimmäinen kursori näyttää kerrosnumerokuvat ja toinen kursori nuolien kuvat.

Kerrosnumeroiden ja nuolien kuvat sijaitsevat muistikortilla. Kuville on kehitetty oma tiedostomuoto, jotta ne voitaisiin lukea muistikortilta suoraan piirin muistiin DMA-lukufunktioiden avulla. Tiedostot luodaan PC-konfiguraatio-ohjelman avulla BMP- ja PNG-kuvista. Prosessorin pitää kuitenkin ensin lukea tiedoston otsikko-osa, jotta se voi säätää kursorin asetukset kohdalleen. Otsikko-osa sisältää kuvan mitat pikseleinä ja kuvassa käytetyt värit. Heti otsikko-osan jälkeen alkaa kuvainformaatio. Pikselit on tallennettu peräkkäin alkaen kuvan vasemmasta yläkulmasta. Jokainen pikseli vie 2 bittiä eli yhteen tavuun mahtuu 4 pikseliä. Kerrosnumero- ja nuolikuville on piirin muistissa varattu 19200 tavun kokoiset paikat. Kuvat voivat siis maksimissaan olla koko näytön kokoisia: 320 * 240 / 4 = 19200.

Nuolikuvia voidaan animoida. Animointi tapahtuu siten, että ylänuoli vieritetään ylöspäin ja alanuoli alaspäin. Animoinnin nopeus on säädettävissä konfiguraatiotiedostossa. Animointi hoidetaan ajastimen ylivuotokeskeytyksen avulla. Kun keskeytys tapahtuu, nuolen kuvaa siirretään yksi pikseli ylös- tai alaspäin.

5.7 Pääohjelma

Laitteen pääohjelmassa yhdistetään kaikki edellä kerrotut ohjelmiston osat yhdeksi kokonaisuudeksi. Pääohjelma alkaa laitteiston alustuksella. Laitteen asetukset luetaan konfiguraatiotiedostosta. Ne asetukset, jotka puuttuvat konfiguraatiotiedostosta, saavat vakioarvoja. Tämän jälkeen kerrosnumero- ja nuolikuvien tiedostojen ensimmäisten sektoreiden osoitteet tallennetaan muistiin. Näin myöhemmin suorituksen aikana ei tarvitse enää kutsua FAT-kirjaston funktioita, mikä nopeuttaa kuvien näyttämistä.

Riippuen asetuksista pääohjelman suoritus jatkuu joko normaali- tai soittolistamoodissa. Normaalimoodissa taustakuva vaihdetaan silloin, kun hissi liikkuu kerroksesta toiseen. Soittolistamoodissa kuvat ja videot vaihtuvat kerroksesta riippumatta tietyn ajan välein.

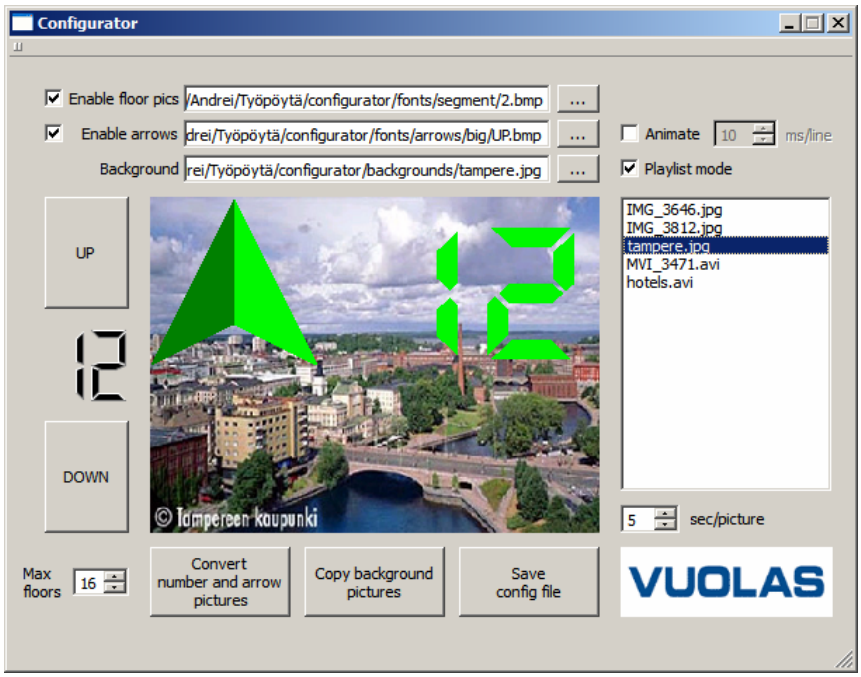
Pääohjelma on tehty siten, että se olettaa saavansa suorituksen aikana tietoja hissin tilasta. Tiedot voidaan vastaanottaa joko RS-485- tai CAN-väylän kautta. Protokolla on sovitettavissa asiakkaan/hissivalmistajan tarpeiden mukaiseksi. Minimivaatimuksena voidaan pitää sitä, että ohjelma saa tiedon hissin sijainnista. Tämän lisäksi ohjelma osaa

hyödyntää tiedon nuolien tilasta (alas-, ylös- ja tuplanuoli) sekä ottaa huomioon mahdollisia hälytyksiä.

Liitteessä 3 on pääohjelman pseudokielinen esitys, josta saa hyvän kuvan siitä, mitä asioita ja missä järjestyksessä pääohjelmassa tehdään.

6 KONFIGUROINTISOVELLUS

Kerrosnäyttömoduulin konfigurointia varten kehitettiin PC-sovellus. Sovellus kehitettiin C++ -kielellä käyttäen hyväksi Qt-kirjastoa. Qt on alun perin Trolltechin kehittämää alustariippumaton graafinen käyttöliittymäkirjasto. Qt:lla voi helposti tehdä tehokkaita natiiveja sovelluksia useammille käyttöjärjestelmille kirjoittamalla vain yksi yhteinen lähdekoodi. [17]



Kuva 13. Konfigurointisovelluksen käyttöliittymä

Sovelluksen avulla voidaan simuloida kerrosnäyttömoduulin näkymää eri kerroksissa. Ohjelman käyttöliittymän (kuva 13) keskellä sijaitsevan simulointi-ikkunan koko on 320x240 pikseliä, mikä on myös laitteessa käytettävien TFT-paneeleiden resoluutio.

Sovelluksen käyttäjä voi lisätä jokaiselle kerrokselle oman taustakuvan tai vaihtoehtoisesti määrittää soittolistan. Taustakuvien on oltava JPEG-muodossa ja soittolistan videoiden AVI-muodossa. Soittolistan tiedostojen järjestys voidaan muuttaa raahaamalla tiedostot hiirellä haluttuun paikkaan listassa. Tiedostot voidaan poistaa listasta klikkaamalla niiden päälle hiiren oikealla näppäimellä. Lisäksi käyttäjä voi määrittää soittolistan tiedostojen vaihtovälin sekunteina.

Käyttäjä voi lisätä kerrokselle kuvan, joka sisältää kerroksen numeron. Kuvan ei välttämättä tarvitse olla numero, vaan se voi sisältää esimerkiksi kirjaimet ja muut kerroksen tunnukset. Kuvat voivat olla BMP- tai PNG-muodossa. Jos avattavan kuvatiedoston nimi on 0.bmp/png … 9.bmp/png, ohjelma osaa automaattisesti hakea muut numerotiedostot samasta kansiosta ja muodostaa niistä loput numerot (>=10). Käyttäjä voi valita, käytetäänkö kerrosnumerokuvia vai ei (päälle/pois toiminto). Käyttäjä voi valita kerrosnumerokuvan paikan näytöllä painamalla hiiren vasenta nappia simulointi-ikkunan päällä.

Käyttäjä voi lisätä nuolien kuvia ohjelmaan. Nuolia on kolmea eri tyyppiä: ylös, alas ja kaksikärkinen nuoli. Jokaisesta nuolesta on oltava oma kuvansa. Kuvat voivat olla BMP- tai PNG-muodossa, ja niiden nimien on oltava up.bmp/png, down.bmp/png ja updown.bmp/png. Nuolien kuvat ovat kaikilla kerroksilla samat. Käyttäjä voi valita, käytetäänkö nuolikuvia vai ei (päälle/pois toiminto). Käyttäjä voi valita nuolikuvien paikat hissinäytöllä painamalla hiiren oikeata nappia simulointi-ikkunan päällä. Jokaisella nuolen kuvalla voi olla oma paikkansa. Lisäksi käyttäjä voi kytkeä nuolien animoinnin päälle/pois. Animoinnissa ylänuoli liikkuu ylöspäin ja alanuoli alaspäin. Animoinnin nopeus on säädettävissä. Tuplanuolia ei animoida.

Kun kaikki tarvittavat tiedostot on lisätty ohjelmaan ja parametrit asetettu, käyttäjä voi generoida ohjelman avulla konfiguraatiotiedoston, joka sisältää kaikki tarvittavat asetustiedot ja tiedostojen nimet. Lisäksi kerrosnumero- ja nuolikuvat voidaan tässä vaiheessa kääntää laitteen näytönohjainpiirille sopivaan muotoon. Asetetut taustakuvat ja videot voidaan kopioida suoraan haluttuun paikkaan. Konfiguraatiotiedosto, kuvat ja videot voidaan tallentaa esimerkiksi laitteen muistikortille.

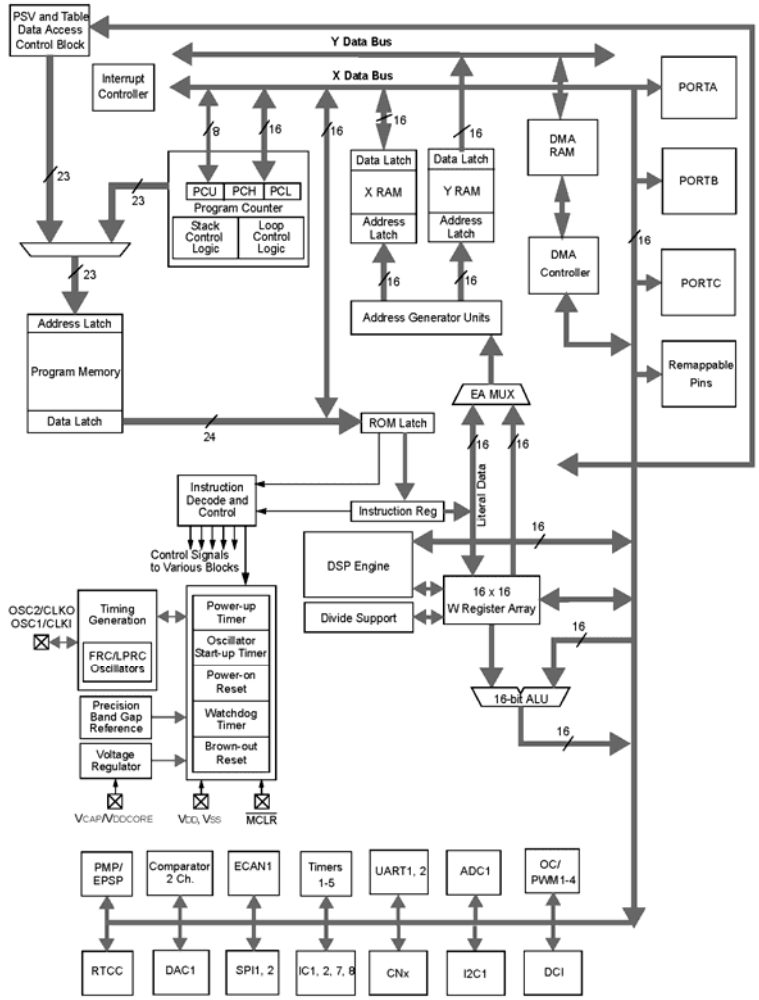
7 SARJATUOTANTOVERSION PROTOTYYPPI

Sarjatuotantoversiota alettiin kehittää heti, kun ohjelmisto oli saatu toimimaan ensimmäisen prototyypin laitteistolla. Tässä luvussa kerrotaan lähinnä laitteistomuutoksista versioiden välillä, sillä molempien versioiden ohjelmisto on periaatteessa sama. Tärkein muutos on ehkä se, että tässä versiossa kaikki tarvittava elektroniikka sijaitsee samalla piirilevyllä. Laitteen prosessori vaihdettiin 16-bittiseen dsPIC-sarjan mikrokontrolleriin, joka on edullisempi, kuin ensimmäisessä versiossa käytetty PIC32. Tämän lisäksi virtalähde suunniteltiin kokonaan uudelleen, sillä ensimmäisen version virtalähde oli alun perin tarkoitettu vain väliaikaiseksi ratkaisuksi.

7.1 DSPIC33FJ64GP804-prosessori

Kun sarjatuotantoversiota alettiin kehittämää, ohjelmisto oli käytännössä valmis. Tämän takia pystyttiin määrittelemään tarkasti uudelta prosessorilta vaadittavat ominaisuudet, kuten esimerkiksi ohjelma- ja datamuistimäärät. DSPIC33FJ64GP804-prosessori sisältää 64 kB ohjelmamuistia ja 16 kB RAM-muistia [5, s. 5]. Tälle prosessorille käännettynä ohjelmisto vie 37 kB eli noin 58 % prosessorin ohjelmamuistista. Datamuistia tarvitaan hieman yli 3 kB, mikä on noin 20 % RAM-muistista. Kääntäjän optimointitasona on ollut 1/3.

Tämän prosessorin yksi tärkeä ominaisuus on asetettavissa olevat oheislaitteiden IO-nastat. Yleensä mikrokontrollerin oheislaitteiden tulot- ja lähdöt on määritetty kiinteästi joihinkin piirin IO-nastoihin. Tässä prosessorissa käyttäjä pystyy itse määrittelemään joidenkin oheislaitteiden sisään- ja ulostulonastat. Kaikkien oheislaitteiden IO:t eivät kuitenkaan ole käyttäjän määritettävissä, esimerkiksi AD- ja DA-muuntimien sekä rinnakkaisportin nastat on kiinteästi määritetty. Kaikkien sarjaliikenneyksiköiden nastat ovat määritettävissä. Uudelleenmäärittely voidaan tehdä ohjelmassa ajon aikana niin monta kertaa, kuin käyttäjä haluaa. Tämän ominaisuuden ansiosta on mahdollista valmistaa järkevästi piirejä, joissa on suhteellisen vähän nastoja, mutta silti monipuoliset oheislaitteet. Käyttäjän kannalta ominaisuus helpottaa esimerkiksi piirilevysuunnittelua. [5, s. 161-163]



Kuva 14. DSPIC33FJ64GP804-piirin lohkokaavio [5, s. 16]

Kuvassa 14 on esitetty prosessorin yleinen rakenne. Kuvasta käy hyvin ilmi, mitä oheislaitteita piiri sisältää.

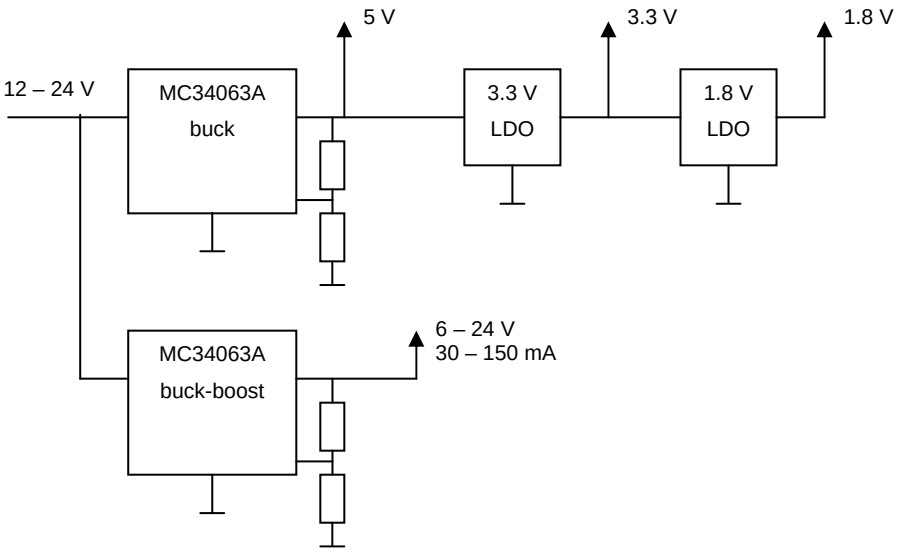
7.2 Virtalähde ja muut lisäykset

Laitteen käyttöjännitteeksi on määritetty 12 – 24 V. Laitteen sisäiset komponentit tarvitsevat toimiakseen kolmea eri jännitetasoa: 5 V, 3,3 V ja 1,8 V. Tämän lisäksi virtalähteellä voidaan tuottaa erilaisia jännitteitä TFT-paneelien taustavaloille.

Virtalähde koostuu kahdesta hakkurikytkennästä (kuva 15): Ensimmäinen hakkuripiiri toimii buck-hakkurina ja alentaa laitteen käyttöjännitteen (12 – 24 V) viiteen volttiin. Tämä jännite alennetaan LDO-regulaattoreiden avulla 3,3 ja 1,8 volttiin.

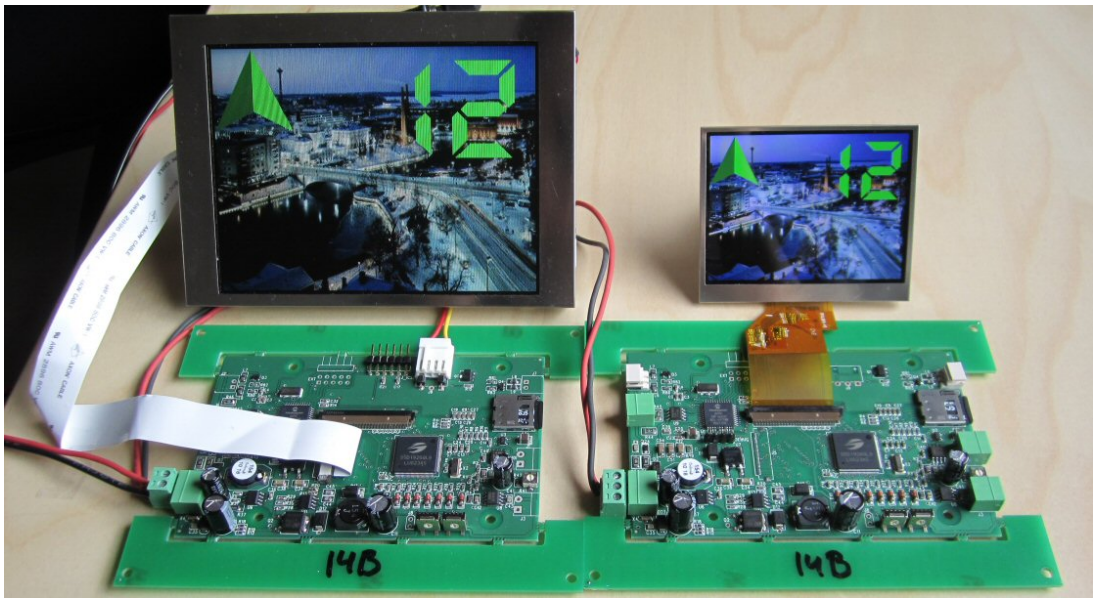
Toinen hakkurikytkentä on taustavaloa varten, ja sen hakkuripiiri toimii buck-boost -hakkurina. Tulojännite otetaan käyttöjännitteestä ja lähtö kytketään suoraan TFT-paneelin taustavaloliittimeen. Luvussa 3.1 (Paneelien liitännät) todettiin, että paneeleiden taustavalojännitteet vaihtelevat 6:sta 24 volttiin. Virrankulutus vaihtelee välillä 30 – 150 mA. Hakkurikytkennän lähtöjännite voidaan säätää vaihtamalla kytkennän komponentteja. Itse kytkentä pysyy kuitenkin samana, joten erilaisille TFT-paneeleille ei tarvitse tehdä omia piirilevyjä.

Molemmissa hakkurikytkennässä piirinä käytetään alun perin Motorolan kehittämää MC34063A:ta. Piirin toimintajännite on 3 – 40 V ja maksimivirranantokyky 1.5 A [18].



Kuva 15. Virtalähteen periaatteellinen rakenne

Muita samalla piirilevyllä sijaitsevia komponentteja ovat muun muassa RS-485-muunnin, CAN-ohjain, niin sanottu Gong-äänipiiri, ulkoisten nuolien ohjauselektroniikka ja kiertokytkimet. Jos laite on asennettu kerrokseen (ei hissin koriin), voi olla hyödyllistä, että laite tietää, missä kerroksessa se sijaitsee. Asennuskerros asetetaan kiertokytkimien avulla. Jos laite asennetaan hissin koriin, kytkimiä ei kalusteta. Gong-äänipiiri tuottaa kahta soittokelloa muistuttavaa ääntä. Kaiutin voidaan kytkeä suoraan piirin lähtöön. Tätäkään piiriä ei välttämättä tarvitse kalustaa, jos laite on asennettu hissin koriin. Kuvassa 16 on esitetty laitteen sarjatuotantoversion prototyypin piirilevyt sekä niihin kytketyt TFT-näytöt.



Kuva 16. Sarjatuotantoversion prototyyppi (vasemmalla 5,7” ja oikealla 3,5” näyttö)

7.3 Testaus ja jatkokehitys

Tämän työn kirjoittamisen aikana laitteen täydellistä testausta ei oltu vielä suoritettu. Kuitenkin kehityksen aikana laitteen ohjelmiston yksittäisiä osakokonaisuuksia testattiin heti niiden kirjoittamisen jälkeen. Ilmenneitä ongelmia pyrittiin korjaamaan saman tien. Ohjelmistoa voidaan siis pitää ainakin osittain testattuna ja toimivaksi todettuna. Laitteistolle on jatkossa suoritettava rasitustestaus ja EMC-mittaukset.

Kehityksen aikana laite ja ohjelmisto testattiin Vuolas Oy:n oman VEBUS-sarjaliikenneprotokollan ja hissiä simuloivan testauskortin avulla. Tulevaisuudessa ohjelmistoon on tarkoitus lisätä tuki eri hissivalmistajien protokollille.

Sarjatuotantoversion piirilevyllä on olemassa laajennusliitin, johon on johdatettu prosessorin käyttämättä jääneet nastat. Tähän liittimeen voidaan jatkossa kehittää erilaisia laajennuskortteja. Yksi tällainen kortti voi esimerkiksi olla äänikortti. Tällä hetkellä laite tukee AVI-videotiedostoja. Yleensä AVI-tiedoston ääniraita on joko PCM-muodossa tai pakattu MP3-muotoon. Laitteen prosessori ei kykene reaaliaikaiseen MP3-dekoodaukseen (ainakaan laadukkaasti pakattujen ääniraitojen kanssa). Äänikortissa on siis oltava oma rautapohjainen MP3-dekooderipiiri. Yksi tällainen piiri voi esimerkiksi olla tamperelaisen VLSI Solution Oy:n valmistama VS1053. Tämä piiri tukee monia äänenpakkausmuotoja ja sisältää muun muassa stereo-DA-muuntimen.

8 YHTEENVETO

Työn tavoitteena oli modernisoida alun perin Vuolas Oy:n kehittämä ja tällä hetkellä Pikotec Oy:n valmistama kerrosnäyttömoduuli vastaamaan nykypäivän vaatimuksiin. Tavoitteena oli saada laitteesta monipuolisempi ja halvempi valmistaa. Myös laitteen helppokäyttöisyyttä oli tarkoitus parantaa. Käytännössä koko laite suunniteltiin alusta asti uudelleen käyttäen edullisempia komponentteja ja rakenteita, kuin alkuperäisessä versiossa.

Uuden version valmistuskustannukset ovat huomattavasti vanhaa versiota alhaisemmat. Laitteeseen on kuitenkin sisällytetty vanhan version toiminnallisuus ja lisätty joitakin uusia ominaisuuksia, kuten esimerkiksi videon toisto. Käytettävyyttä parannettiin kehittämällä mahdollisimman helppokäyttöinen konfigurointisovellus. Kaiken kaikkiaan kehityksen alussa asetettuja tavoitteita voidaan siis pitää saavutettuina.

Tässä työssä kehitetty laite ei tietenkään ole ainutlaatuinen, sillä monella kilpailijalla on olemassa ominaisuuksiltaan vastaavanlaisia laitteita. On kuitenkin hyvin yleistä, että jos vastaava laite pystyy toistamaan videota, se myös sisältää huomattavasti tehokkaampaa ja kalliimpaa laitteistoa. Todennäköisesti kilpailijat käyttävät omissa laitteissaan vähintään Linux-tasoista käyttöjärjestelmää ja ARM9-tasoista prosessoria. Toisaalta tehokkaampi järjestelmä mahdollistaa paremman tiedostoformaattituen ja helpomman muokattavuuden. Tällaisen järjestelmän ohjelmiston sovellusosa voidaan kehittää korkean tason kielillä, kuten esimerkiksi C++:lla ja Javalla.

Multimediapainotteisissa laitteissa ei yleensä käytetä pieniä ja halpoja mikrokontrollereita pääprosessoreina. Tämä työ kuitenkin todistaa, että käyttämällä tarpeeksi monipuolisia oheiskomponentteja on mahdollista valmistaa edullisesti järjestelmä, joka kykenee kilpailemaan huomattavasti tehokkaampien prosessoreiden/järjestelmien kanssa. Tämän työn tekemiseen Pikotec Oy antoi erinomaiset puitteet: kehitystyön aikana oli täysin vapaat kädet valita tekniikat, komponentit ja ohjelmistot halutun lopputuloksen saavuttamiseksi.

LÄHTEET

[1] ELM - FatFs Generic FAT File System Module. http://elm-chan.org/fsw/ff/00index_e.html (31.3.2011)

[2] Himax Technologies Inc. HX8238A Datasheet. www.olimex.com/dev/pdf/ARM/LPC/HX8238A.pdf (31.3.2011)

[3] JPEG Committee. JPEG homepage. http://www.jpeg.org/jpeg/index.html (31.3.2011)

[4] JPEG File Extension. http://www.fileinfo.com/extension/jpeg (31.3.2011)

[5] Microchip Technology Inc. dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, and dsPIC33FJ128GPX02/X04 Data Sheet. ww1.microchip.com/downloads/en/devicedoc/70292d.pdf (31.3.2011)

[6] Microchip Technology Inc. MPLAB C Compiler for PIC32 MCUs. http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2615&dDocName=en532454 (31.3.2011)

[7] Microchip Technology Inc. PIC32MX3XX/4XX Data Sheet. ww1.microchip.com/downloads/en/DeviceDoc/61143G.pdf (31.3.2011)

[8] Microchip Technology Inc. PIC32 Microcontroller Families. http://ww1.microchip.com/downloads/en/DeviceDoc/DS-39904j.pdf (31.3.2011)

[9] Microchip Technology Inc. PIC32 Product Overview. http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2607 (31.3.2011)

[10] Microsoft. AVI RIFF File Reference. http://msdn.microsoft.com/en-us/library/dd318189%28v=VS.85%29.aspx (31.3.2011)

[11] Microsoft. FAT File System. http://msdn.microsoft.com/en-us/library/ee489982.aspx (31.3.2011)

[12] Microsoft. Miscellaneous Video Subtypes. http://msdn.microsoft.com/en-us/library/dd390688%28v=vs.85%29.aspx (31.3.2011)

[13] Microsoft. Multimedia Technical Note: JPEG DIB Format. http://www.fileformat.info/format/bmp/spec/b7c72ebab8064da48ae5ed0c053c67a4/view.htm (31.3.2011)

[14] MJPEG (Motion JPEG) Video Codec.
http://www.digitalpreservation.gov/formats/fdd/fdd000063.shtml (31.3.2011)

[15] Noe, Alexander. AVI File Format.
www.alexander-noe.com/video/documentation/avi.pdf (31.3.2011)

[16] Noe, Alexander. AVI-Mux GUI.
http://www.alexander-noe.com/video/amg/ (31.3.2011)

[17] Nokia Corporation. Qt Creator IDE and tools.
http://qt.nokia.com/products/developer-tools (31.3.2011)

[18] ON Semiconductor. MC34063A Datasheet.
www.onsemi.com/pub_link/Collateral/MC34063A-D.PDF (31.3.2011)

[19] Palm Technology. PT0573224-A202 Datasheet.
www.gst-lcd.com/spec/tftspec/PT0573224-A202.pdf (31.3.2011)

[20] SanDisk Corporation. SanDisk SD Card Product Family.
http://app.arrownac.com/aws/pg_DataSheetView?docid=45986484S8052843N2401 (31.3.2011)

[21] SD Association. SD Technology Overview.
http://www.sdcard.org/developers/tech/ (31.3.2011)

[22] SD Group. SD Card Simplified Specification.
www.sandisk.com/Assets/File/OEM/Manuals/SD_Physical_specsv101.pdf (31.3.2011)

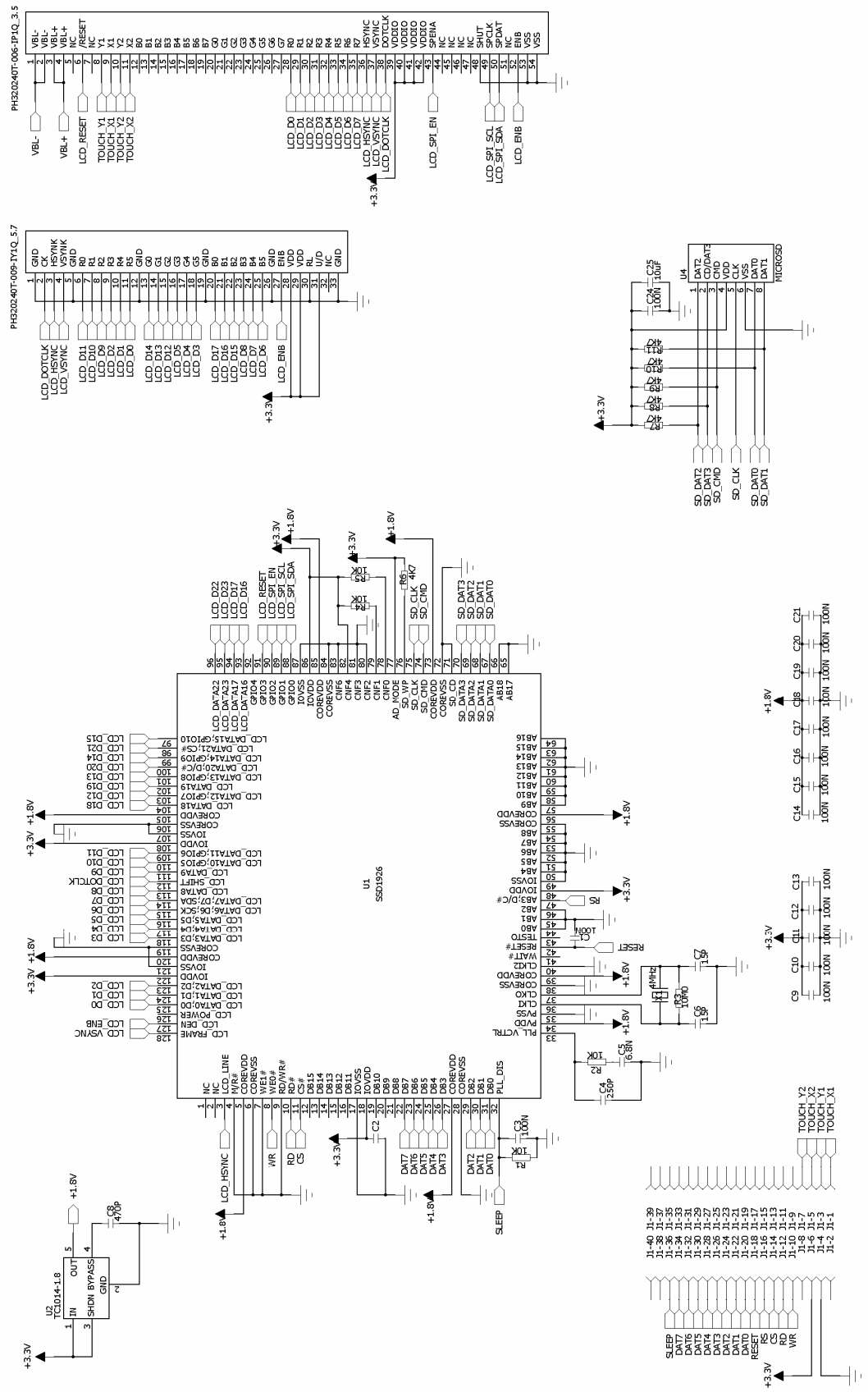
[23] Solomon Systech. SSD1926 Application Note.
www.solomon-systech.com/pdf/AppNote_SSD1926_10.pdf (31.3.2011)

[24] Solomon Systech. SSD1926 Datasheet.
www.solomon-systech.com/pdf/SSD1926R1_2.pdf (31.3.2011)

[25] wFormatTag Waveaudio types.
http://www.tek-tips.com/viewthread.cfm?qid=470674&page=32 (31.3.2011)

LIITEET

Liite 1. Näytönohjainkortin kytkentäkaavio



Liite 2. Esimerkki konfiguraatiotiedostosta

```
show_numbers                näytetään kerrosnumerot
numbers_pos: 310 10         kerrosnumeron sijainti näytössä (oikea yläkulma)

show_arrows                 näytetään nuolet
up_arrow_pos: 80 10         ylänuolen sijainti (oikea yläkulma)
down_arrow_pos: 80 150      alanuolen sijainti (oikea yläkulma)
updown_arrow_pos: 80 40     tuplanuolen sijainti (oikea yläkulma)

scroll_arrows               animoidaan nuolet
scroll_delay: 10            animoinnin nopeus (ms/rivi)

exc_floor00 P_big.cur       kerroksessa 0 näytetään P-pysäköintimerkin kuva
exc_floor05 blank.cur       kerroksessa 5 ei näytetä kerrosnumeroa (tyhjä kuva)
                            muissa kerroksissa näytetään kerrosnumerokuvat
                            normaalisti

floor00 black.jpg           kerroksien taustakuvat
floor01 tampere.jpg
floor02 tallinn.jpg
floor03 paris.jpg
floor04 budapest.jpg
floor05 berlin.jpg

playlist_delay: 30          soittolistan tiedostojen vaihtoväli (s)

//PLAYLIST:                 soittolista on käytössä, jos PLAYLIST sanan edessä
denmark.jpg                 ei ole kauttaviivoja
hotels.avi
czech.jpg
london.avi
monaco.jpg
earth.avi
prague.jpg
lake.avi
LOOP
```

Liite 3. Pääohjelman pseudokielinen esitys

```
prosessorin kellotaajuuksien asetus
oheislaitteiden alustus (ajastimet, UARTit ja keskeytykset)
rinnakkaisportin alustus
näytönohjainpiirin muistin tyhjennys
näytönohjainpiirin ja TFT-paneelin alustus
muistikortin alustus (ei formatointi)
FAT-kirjaston alustus
vakioasetusten määritys
luetaan asetukset konfiguraatiotiedostosta
ladataan muistiin kerrosnumero- ja nuolikuvatiedostojen osoitteet

soittolistamoodi:
  if(hälytys päällä)
    if(uusi hälytys)
      päivitä hälytyskuva
    goto soittolistamoodi

  if(aika vaihtaa taustakuva tai –videotiedosto JA video ei käynnissä)
    lue seuraavan tiedoston nimi soittolistasta
    if(JPEG-tiedosto)
      aseta taustakuvaksi
    else if(AVI-tiedosto)
      aloittaa videon toisto
    if(ollaan listan lopussa)
      hyppää listan alkuun

  if(hissi on liikkunut uuteen kerrokseen JA kerrosnumerot käytössä)
    päivitä kerrosnumerokuva

  if(nuolikuvat käytössä JA uusi nuoliohjaus)
    päivitä nuolen kuva
    if(animointi käytössä)
      aloita nuolen animointi

  goto soittolistamoodi

perusmoodi:
  if(hälytys päällä)
    if(uusi hälytys)
      päivitä hälytyskuva
    goto perusmoodi

  if(hissi on liikkunut uuteen kerrokseen)
    päivitä taustakuva
    if(kerrosnumerot käytössä)
      päivittää kerrosnumerokuva

  if(nuolikuvat käytössä JA uusi nuoliohjaus)
    päivitä nuolen kuva
    if(animointi käytössä)
      aloita nuolen animointi

  goto perusmoodi
```