PC/104-teollisuustietokoneiden uusiminen

Opinnäytetyö

Marko Möttönen

Elektroniikan koulutusohjelma
Sulautetut järjestelmät

Hyväksytty ___.___._____ _____

SAVONIA-AMMATTIKORKEAKOULU TEKNIIKKA KUOPIO

Koulutusohjelma

Elektroniikan koulutusohjelma

Tekijä

Marko Möttönen

Työn nimi

PC/104-teollisuustietokoneiden uusiminen

Työn laji	Päiväys	Sivumäärä
Opinnäytetyö	20.05.10	40 + 20

Työn valvoja

Yliopettaja Väinö Maksimainen

Tiivistelmä

Tämän opinnäytetyön aiheena oli uusia Savonia-ammattikorkeakoulun tekniikan Kuopion yksikön elektroniikan laboratorion PC/104-teollisuustietokonelaitteisto, kehitellä uusille laitteille laboratorioharjoituksia sekä ohjelmoida pieni hälytysjärjestelmä FreeRTOS-reaaliaikakäyttöjärjestelmälle.

Työssä pyydettiin eri laitevalmistajilta tarjouksia kahdeksasta PC/104-prosessorikortista. Laitteet tilattiin parhaan tarjouksen antaneelta yritykseltä ja koottiin käyttökuntoon Savonia-ammattikorkeakoulun tekniikan Kuopion yksikön elektroniikan laboratoriossa.

Uusia laitteita sekä laadittuja laboratorioharjoituksia on määrä käyttää elektroniikka-alan opinnoissa tulevilla vuosikursseilla tutustuttaessa sulautettujen järjestelmien ratkaisuihin ja sovelluksiin.

Työssä ohjelmoitiin myös pieni hälytysjärjestelmä FreeRTOS-reaaliaikakäyttöjärjestelmälle. Ohjelman oli tarkoitus antaa hälytys kun rinnakkaisportin tulo kääntyy. Ohjelmaa simuloitiin omatekoisella rinnakkaisporttiin liitetyllä laitteella, jonka kytkimillä ohjattiin rinnakkaisportin tuloja.

Avainsanat

PC/104, teollisuustietokone

Luottamuksellisuus

julkinen

SAVONIA UNIVERSITY OF APPLIED SCIENCES

Degree Programme

Electronics

Author

Marko Möttönen

Title of Project

Renewal of PC/104 industrial computers

Type of Project	Date	Pages
Final Project	20.05.10	40 + 20

Academic Supervisor

Mr Väinö Maksimainen, Senior Lecturer

Abstract

The aim of this thesis was to renew Savonia polytechnics Kuopios technology units PC/104 industrial computing laboratory devices, to develop some exercises for new devices and program a small alarm system for FreeRTOS real time operating system.

Offers for eight PC/104 cpu cards were asked from different manufacturers in this thesis. Devices were ordered from the company who gave the best offer and then devices were assembled for use at Savonia polytechnics Kuopios technology units laboratory.

New devices and designed exercises are planned to be used at electronics studies in coming year courses when getting to know solutions and application of embedded systems.

A little alarm system for FreeRTOS real time operating system was also programmed in this thesis. The program was designed to give the alarm when parallel ports input is turned. This program was simulated with a self-made device which was connected to computers parallel port. Parallel port inputs was controlled with switches on the device.

Keywords

PC/104, Industrial computer

Confidentiality

public

ALKUSANAT

Tämä opinnäytetyö on tehty kevään 2010 aikana Kuopiossa. Haluan kiittää Savonia-ammattikorkeakoulun informaatiotekniikan kehitysyksikköä, ohjaavaa opettajaa yliopettaja Väinö Maksimaista, laboratorioinsinööri Pertti Kainulaista sekä kaikkia muita työssäni minua auttaneita.

Kuopiossa 20.5.2010

Marko Möttönen

SISÄLTÖ

1 JOHDANTO

Tämän opinnäytetyön aiheena on tutustua teollisuustietokonestandardiin PC/104 ja uusia Savonia-ammattikorkeakoulun tekniikan Kuopion yksikön elektroniikan laboratorion PC/104-laitteet sekä laatia uusiin laitteisiin liittyviä laboratorioharjoituksia.

Ensimmäisenä tavoitteena on hankkia PC/104-teollisuustietokoneet, jotka vastaavat tarpeita ja ovat sopivan hintaisia. Hankitut laitteet asennetaan käyttöön käyttäen hyväksi entisten laitteiden virtalähteitä ja koteloita, joita tulee hieman muokata tarpeisiin sopiviksi.

Laitteisiin liittyvät laboratorioharjoitukset laaditaan Savonia-ammattikorkeakoulun Kuopion tekniikan yksikön kurssille Tietokonetekniikan työt 1. Kyseisellä kurssilla on käytetty tähän asti edellisiä PC/104-laiteita, joihin on olemassa omat harjoituksensa. Nyt tarkoituksena on päivittää harjoitukset uusien laitteiden myötä.

Työssä on tavoitteena myös tutustua FreeRTOS-reaaliaikakäyttöjärjestelmään ja laatia esimerkkiohjelmana reaaliaikainen valvontajärjestelmä, jota käytetään PC/104-teollisuustietokoneen rinnakkaisportin kautta. Ohjelman järjestelmää testataan rinnakkaisporttiin liitetyllä yksinkertaisella kytkennällä, jossa on LEDejä sekä kytkimiä, joilla seurataan ohjelman toimintaa ja pystytään simuloimaan hälytystilannetta.

2 PC/104-TEOLLISUUSSTANDARDI

PC/104 on sulatetuissa järjestelmissä yleisesti käytetty mikrotietokonestandardi, jota hallinnoi PC/104 Consortium. PC/104-standardi on tarkoitettu PC-yhteensopiville moduuleille, joita voidaan pinota yhteen ja luoda erilaisia tarpeita vastaavia sulautettuja tietokonejärjestelmiä. PC/104-laitteet on tarkoitettu käytettäväksi sulautetuissa järjestelmissä, joissa on hyvin tärkeää saada järjestelmän sovelluksien keräämä data luotettavasti talteen vaativista ympäristöoloista huolimatta. Niinpä PC/104-standardin laitteita käytetään usein erilaisissa tehtaissa, laboratorioissa sekä ohjelmoitavissa seurantajärjestelmissä. [1]

PC/104-järjestelmät ovat hyvin samankaltaisia kuin tavalliset työpöytä-PC-järjestelmät, mutta eroavat näistä fyysisiltä mitoiltaan sekä suorituskyvyltään. Lyhenne PC/104 perustuu sen samankaltaisuuteen tavallisten PC-järjestelmien kanssa (PC) ja standardin mukaisen liitäntäväylän pinnimäärään (104). Nämä järjestelmät ovat kokoonsa nähden hyvin tehokkaita, ja niitä voidaan ohjelmoida samoilla työkaluilla kuin normaaleja tietokonejärjestelmiä. PC/104-järjestelmät on suunniteltu vastaamaan sulautettujen järjestelmien tarpeita eli pientä virrankulutusta, pientä kokoa, kestävyyttä ja laajennettavuutta.

Tätä nykyä PC/104-moduuleita on saatavilla hyvin useita eri käyttökohteita ja -tarkoituksia varten. Perustarpeita vastaavista moduuleista mainittakoon esimerkkeinä mm. prosessorikortit, I/O-kortit ja näytönohjaimet. Mutta myös erikoisempiin tarkoituksiin on saatavissa moduuleita, esimerkiksi GPS-moduuleita. Jokainen moduuli voi olla osa suurempaa järjestelmää, tai se voidaan koteloida yksittäisenä komponenttina suorittamaan jotain tiettyä tehtävää.

PC/104-standardin kokoluokan kehitti ensimmäisenä vuonna 1987 Ampro Computers Californiassa, mutta myöhemmin vuonna 1992 PC/104 Consortium standardoi sen. Nykyään reilusti yli 200 jälleenmyyjää tarjoaa PC/104-moduuleita, ohjelmistoja ja tuotetukea. [1]

Toisin kuin PCI-väylää hyödyntävässä yleisimmässä tietokoneen emolevytyypissä ATX:ssä, PC/104-kortissa ei ole liittimiä vakiona. Tämä mahdollistaa moduulien liittämisen toisiinsa pinoamalla ne toistensa päälle käyttäen apuna korottimia, jotka voidaan kiinnittää kortin kiinnitysreikiin.

PC/104-standardin piirilevyn mitat ovat 3,55 x 3,775 tuumaa (90,17 x 95,89 mm); korkeus riippuu käytettyjen komponenttien korkeudesta. Valmistajat kiinnittävät huomiota komponenttien korkeuteen, ettei komponenttien koko aiheuta häiriöitä toiminnassa tai muita on-

gelmia pinottaessa moduuleja päällekkäin. Komponenttien maksimikorkeudet on määritetty standardien spesifikaatioissa. [1]

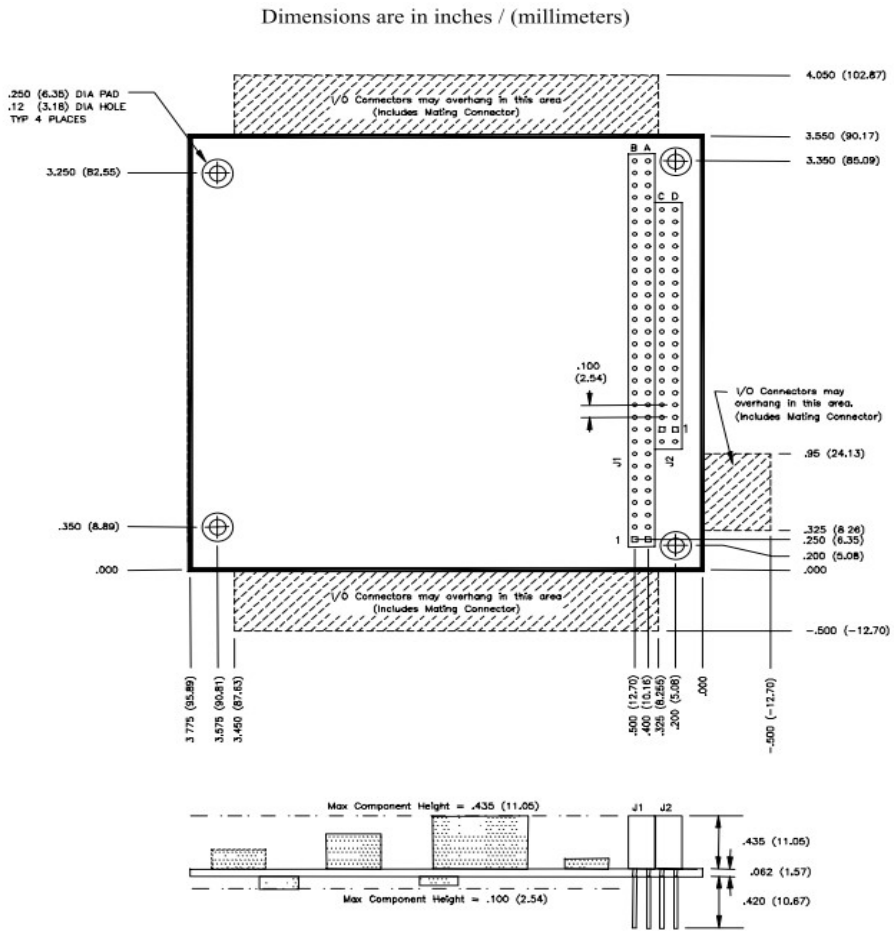
PC/104-standardi kattaa yhteensä viisi eri standardia, joita PC/104:n lisäksi ovat PC/104-Plus, PCI-104, PCI/104-Express ja PCIe/104. Lisäksi PC/104 Consortium on standardoinut neljä sulautettujen järjestelmien emolevyn kokostandardia, jotka ovat EBX, EBX Express, EPIC ja EPIC Express. [1]

3 PC/104-STANDARDIT

3.2 PC/104

PC/104-standardi sisältää viisi eri standardia, joista ensimmäisenä voidaan mainita standardeista yksinkertaisin ja ensimmäisenä julkistettu PC/104. Tämän standardin moduuleissa on liitäntäväylänä ainoastaan PC/104-väylä sekä moduulikohtaiset muut liitännät ja liittimet.

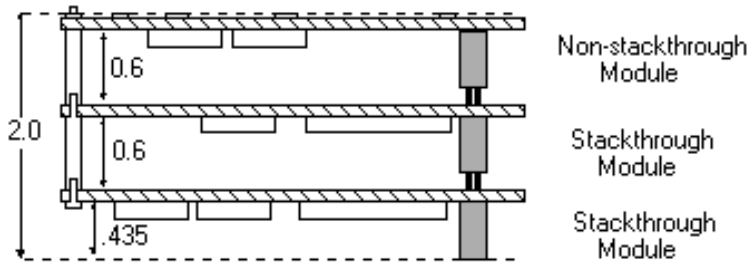
PC/104-standardin moduuleissa on 104-pinninen PC/104-väylä, jossa on kaikki ISA (Industry Standard Architecture) -väylän linjat. Väylään on kuitenkin lisätty yksi maadoituspinni, jolla pyritään varmistamaan väylän luotettavuus. Väylän signaalien ajoitukset ja jännitetasot ovat identtiset ISA-väylän kanssa, mutta virrankulutus on pienempi. Eli PC/104-väylää voidaan myös kutsua ISA-väyläksi. Tämä ISA-väylän liitäntä kattaa siis yhteensä 104 pinniä, jotka on jaettu 64- ja 40-pinniseen liittimeen. 8-bittisissä PC/104-korteissa on vain 64-pinninen liitin, kun 16-bittisissä korteissa on molemmat liittimet eli yhteensä 104 pinniä.



Kuva 1. 16-bittisen PC/104-moduulin mekaaniset mitat. [5]

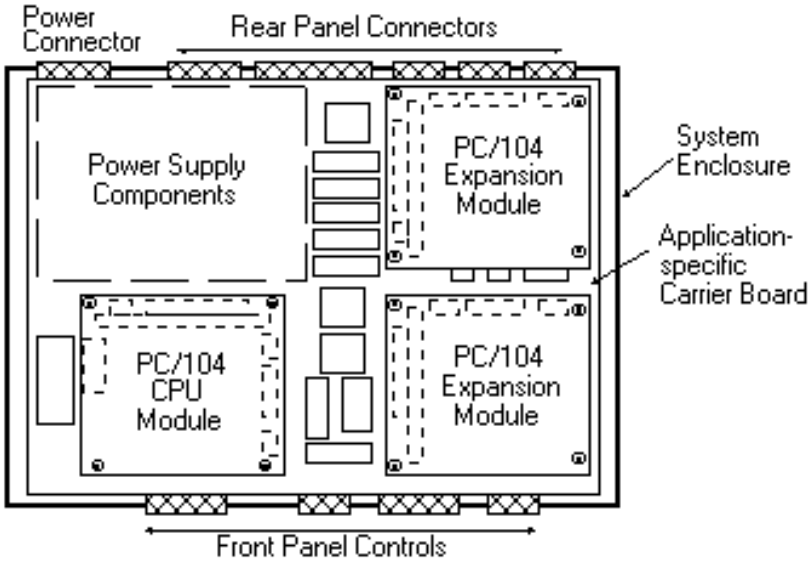
Markkinoilla on eri käyttötarpeisiin lukuisia standardin moduuleita, joita voidaan yhdistellä toisiinsa PC/104-väylän avulla ja koota omia tarpeita vastaava järjestelmä. Esimerkkinä voidaan mainita PC/104-prosessorimoduuli, joka on käytännössä kuin tavallinen tietokone pienessä koossa, joten se toimii usein PC/104-järjestelmän perustana. Tällä prosessorimoduulilla voidaan ajaa jotain käyttöjärjestelmää ja sen PC/104-väylään voidaan liittää tarvittavia lisäkortteja, kuten esimerkiksi ohjelmoitavia I/O-moduuleja.

Vaikka erilaisia moduuleita on saatavilla valtavia määriä ja PC/104-järjestelmien käyttökohteet ovat miltei rajattomat, käytössä on kuitenkin kaksi eniten yleistynyttä menetelmää moduulien liittämiseksi toisiinsa. Ensimmäinen on ns. pinomalli, jossa kortit liitetään toisiinsa pinoamalla (kuva 2).



Kuva 2. Moduulit pinottuna. [2]

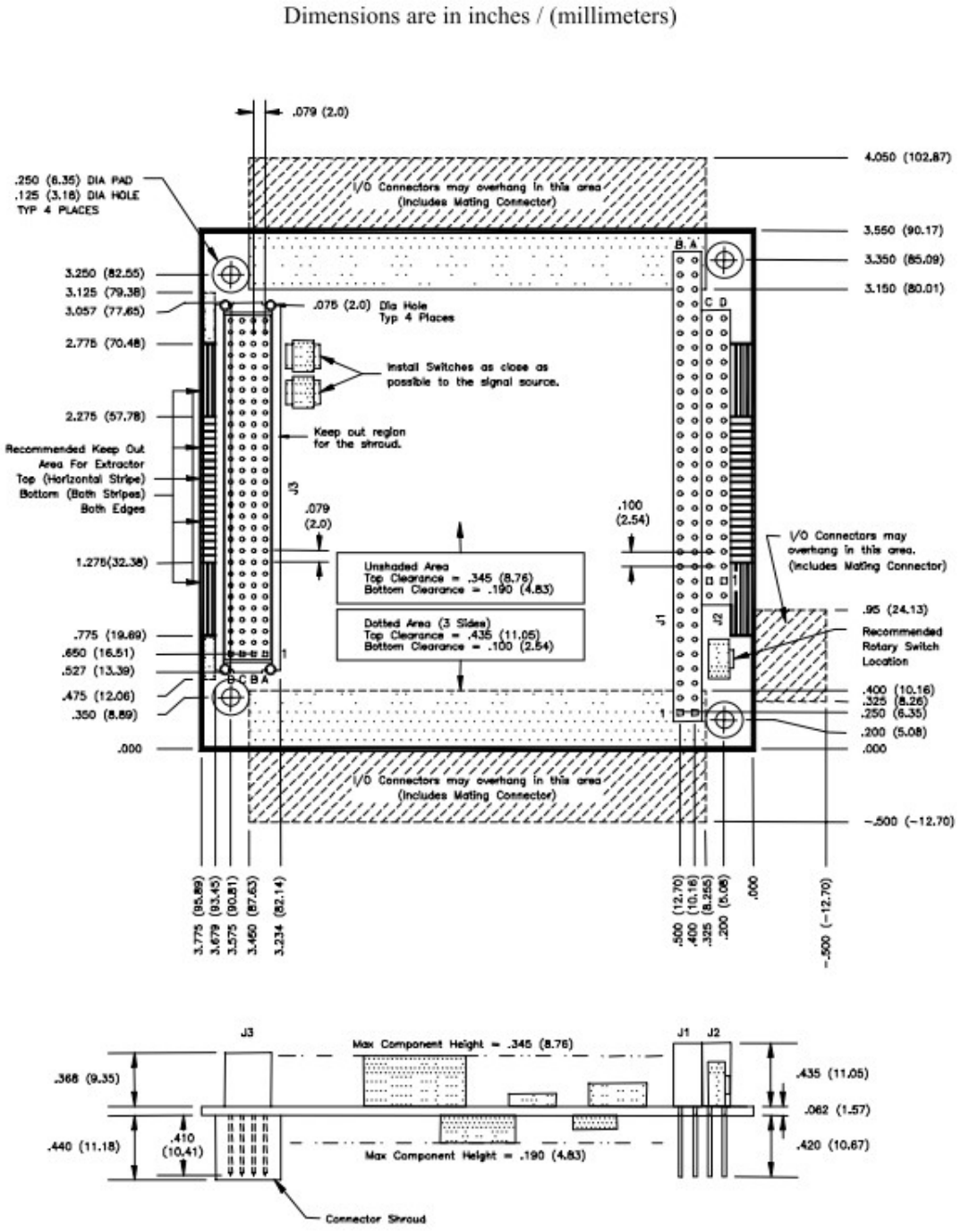
Toinen yleistynyt tapa on komponenttimainen liitäntätapa, jossa moduulit kiinnitetään johonkin alustaan (kuva 3), jossa ne sitten liitetään toisiinsa ja mahdollisiin ulkoisiin liittimiin.



Kuva 3. Moduulit sijoiteltuna toiselle alustalle. [3]

3.3 PC/104-Plus

PC/104-Plus-standardi ei eroa paljoa PC/104-standardista. Erona on, että siihen on ISA-väylän lisäksi lisätty PCI-väylä. PCI-väylän lisääminen PC/104-laitteeseen tuo mukanaan useita etuja käyttäjälle, kuten esimerkiksi nopean tiedonsiirtonopeuden PCI-väylän kautta ja matalat kustannukset PC/104:n uniikin väylämallin takia..



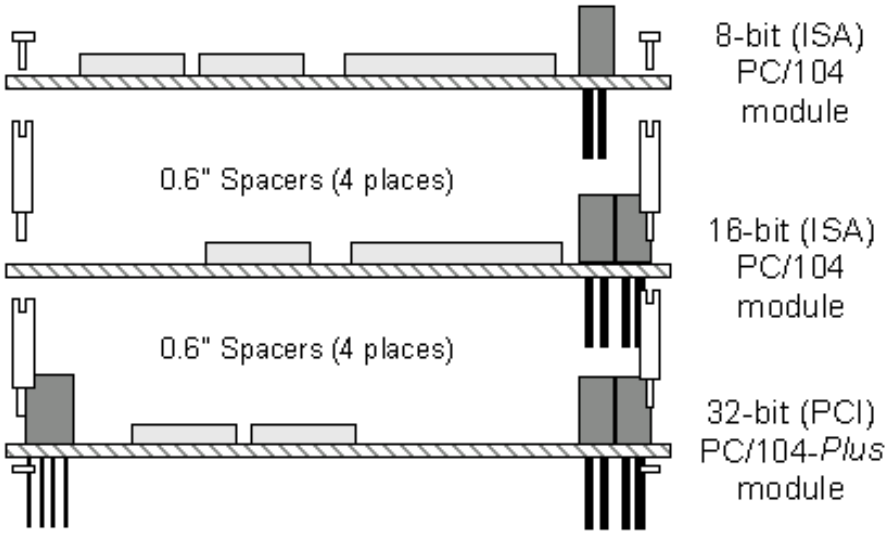
Kuva 4. PC/104-Plus-kortin mekaaniset mitat. [6]

PC/104-Plus-version kehitti alun perin Ampro Computers Inc ja tarjosi sitä PC/104 Consortiumille syyskuussa vuonna 1996. PC/104 Consortiumin työryhmän tarkastelun ja viimeistelyn jälkeen PC/104-Plus-standardi hyväksyttiin helmikuussa vuonna 1997. [1]

Päätavoitteina PCI-väylän lisäämisessä PC/104:ään oli säilyttää sen tärkeimmät ominaisuudet, kuten kompakti koko, pinoamismahdollisuus, samat liitosreiät, pieni tehonkulutus ja PC-yhteensopivuus. Tärkeänä ominaisuutena oli myös pinoamismahdollisuus PC/104-laitteiden kanssa, mikä mahdollisti PC/104-Plus-laitteiden liittämisen mihin tahansa lukuisiin jo olemassa oleviin PC/104-laitteisiin. [1]

PC/104-Plus-kortin 32-bittinen PCI-väylä on toteutettu 120-pinnisellä liittimellä, jota suunniteltaessa on otettu huomioon pinoon liittämisen mahdollisuus PC/104-korttien kanssa. Lisäyksenä on myös yksi ohjauspinni, jolla on pyritty helpottamaan uros-naarasliitoksia PCI-väylän liitoksissa sekä suojelemaan PCI-väylän hieman ohuempia pinnejä.

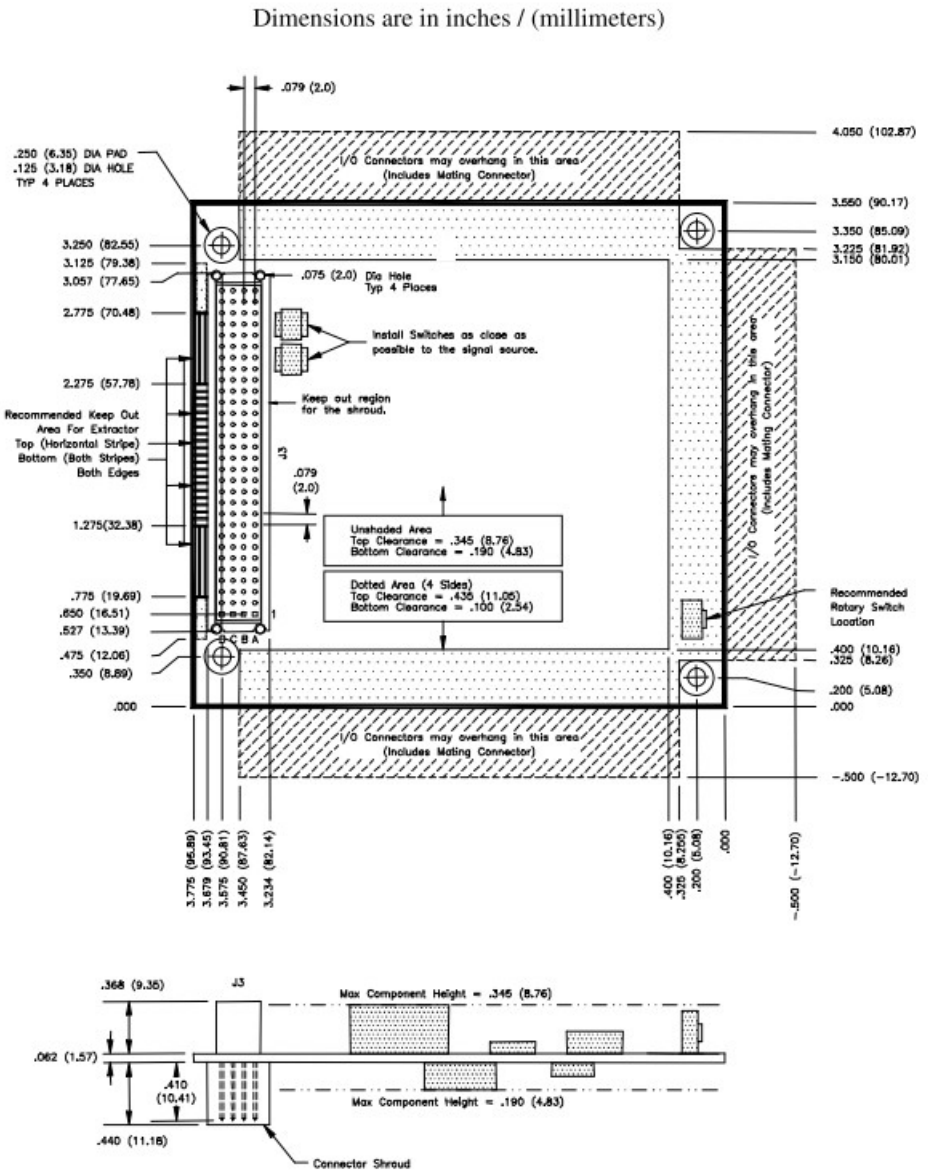
PC/104-Plus on siis mahdollista liittää toiseen PC/104-Plus-moduuliin tai PC/104-moduuliin pinoamalla (kuva 5). Mikäli liitetään useampi PC/104-Plus-moduuli, käytetään liitäntätapana miltei aina nopeampaa PCI-väylää, mutta on myös mahdollista käyttää ISA-väylää. PC/104-Plus-korttien ISA-väylä on yleensä käytössä vain liitettäessä PC/104-moduuli PC/104-Plus-moduuliin.



Kuva 5. PC/104-Plus-moduuli pinossa kahden PC/104-moduulin kanssa. [4]

3.4 PCI-104

Koska PCI-väylä on syrjäyttämässä ISA-väylän, PCI-104-standardissa on luovuttu ISA-väylästä kokonaan ja käytetty ainoastaan PCI-väylää. PCI-104-kortilla ei ole enää 104-pinnistä ISA-väylän liitintä, joka on edeltäjiensä PC/104- ja PC/104-Plus-standardin korteissa. Kortilla on vain 120-pinninen PCI-väylän liitin, joka on myös PC/104-Plus-korteissa. Tarvittaessa kuitenkin ISA-väylän tuki on mahdollinen käyttäen PCI-ISA-siltausta.

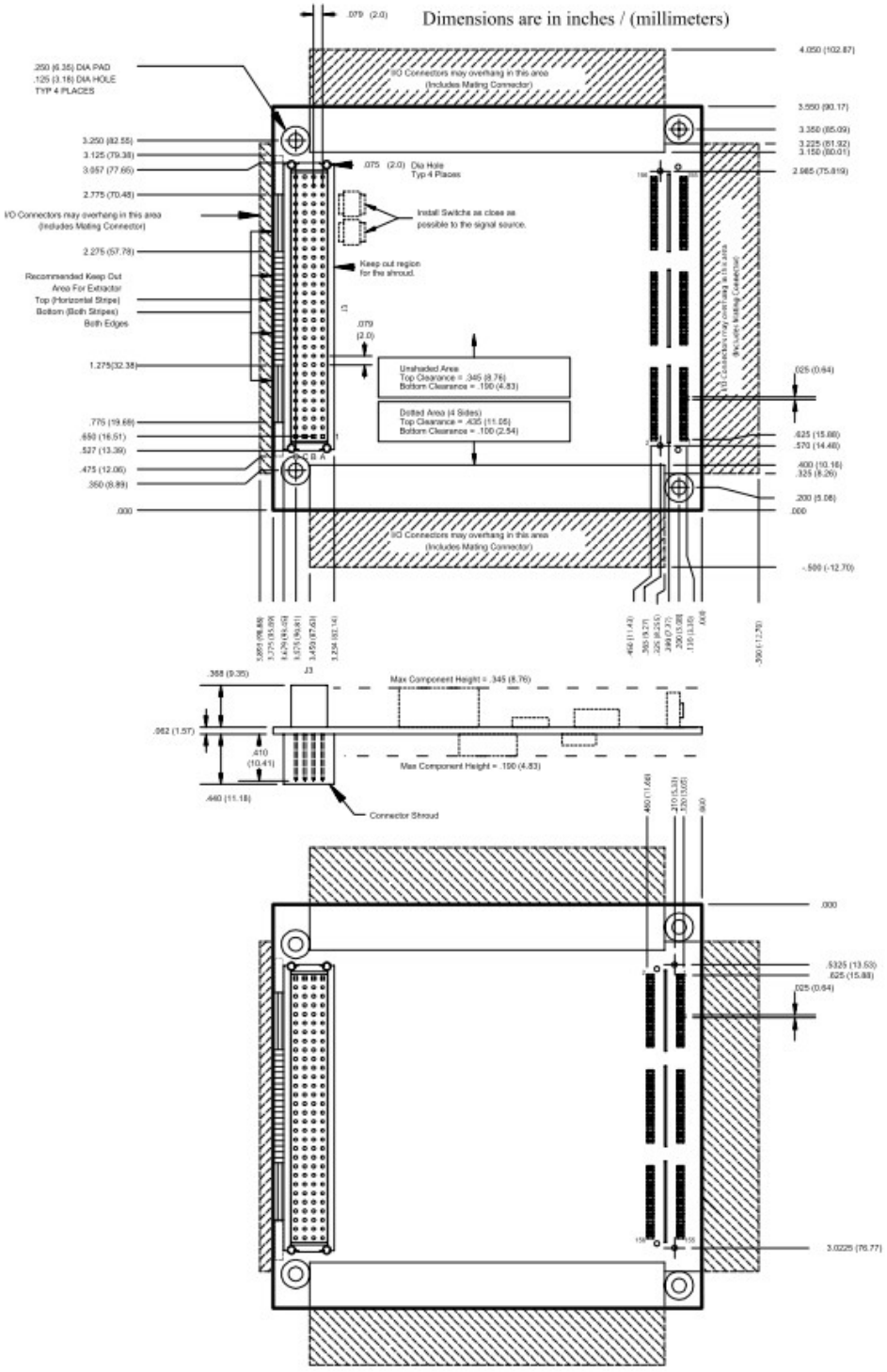


Kuva 6. PCI-104-kortin mekaaniset mitat. [7]

PC/104 Consortium otti PCI-104-standardin käyttöön vuonna 2003. Standardin koko ja kiinnitystapa on pysynyt samana edeltäjiensä mukaan, mutta sen korkeus on hieman pienentynyt ISA-väylästä luopumisen takia. [1]

3.5 PCI/104-Express

PCI/104-Express-standardin julkaisi PC/104 Consortium maaliskuussa 2008 tarkoituksena hyödyntää korkeanopeuksista PCI Express -väylää sulautettujen järjestelmien sovelluksissa. Tämä standardi toi mukanaan uuden korkean suorituskyvyn omaavan fyysisen rajapinnan kuitenkin säilyttäen ohjelmistojen yhteensopivuuden PCI-väylän laitteisiin. [1]



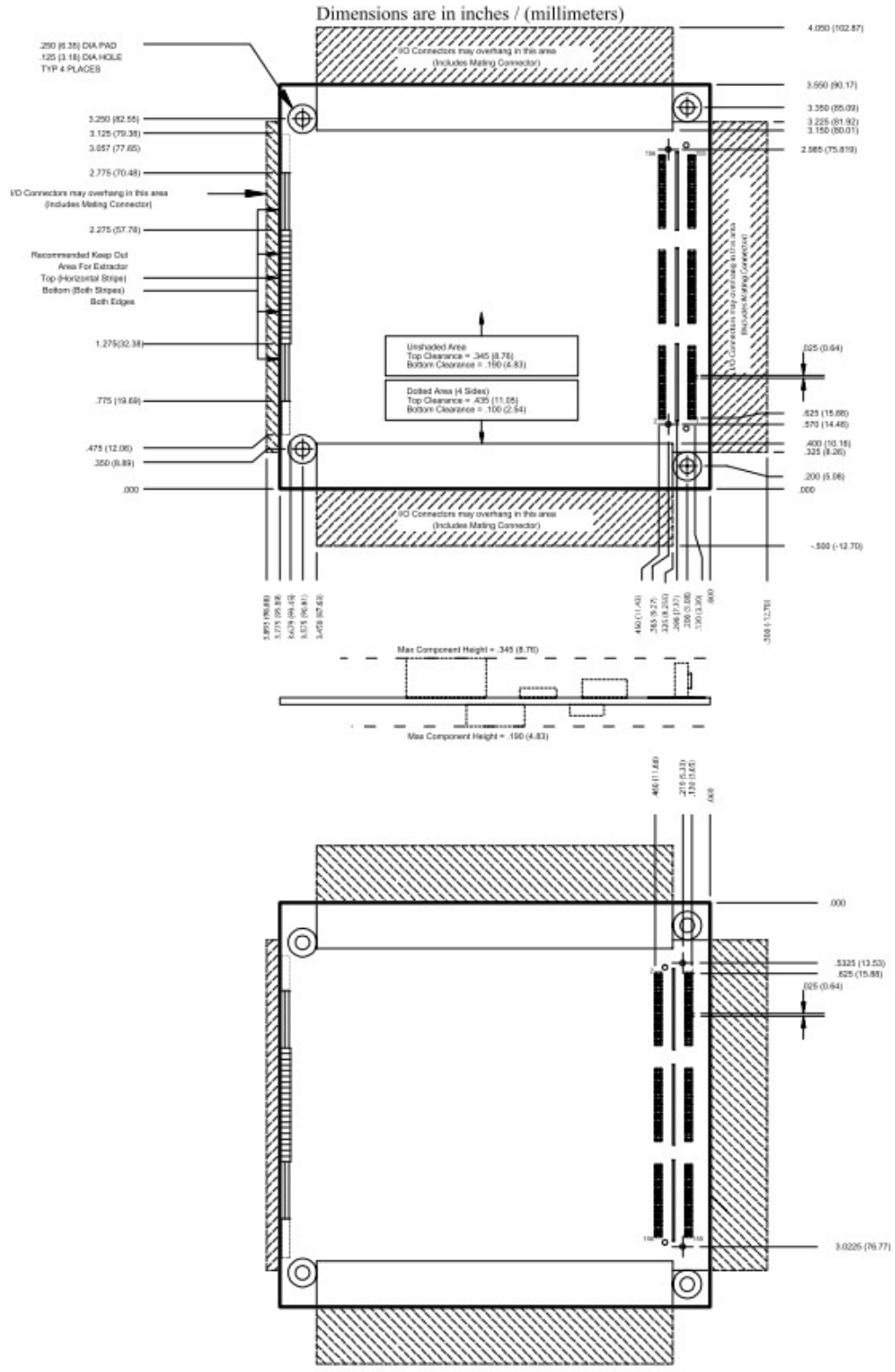
Kuva 7. PCI/104-Express-kortin mekaaniset mitat. [8]

Käytännössä katsoen PCI/104-Express-laitteet eivät eroa PCI-104-laitteista muulla tavalla kuin, että niissä on PCI-väylän lisäksi PCI Express -väylä. PCI/104-Express-standardin PCI Express-väylän liitin on suunniteltu erityisesti PC/104 Consortiumille vastaamaan PC/104-standardin pinoamiskorkeuden ja korotusholkkien vaatimuksia.

PCI/104-Express-moduulit ovat alaspäin yhteensopivia standardien PC/104-Plus ja PCI-104 kanssa PCI-väylän kautta ja ylöspäin standardin PCIe/104 kanssa PCI Express -väylän kautta.

3.6 PCIe/104

PCIe/104 on käytännössä sama kuin PCI/104-Express ilman PCI-väylää. PCIe/104-moduulit ovat siis suoraan yhteensopivia vain PCI/104-Express-moduulien kanssa. Mutta koska PCI Express -väylä perustuu PCI-väylään voidaan PCI- ja PCI Express -väylän laitteet sillata helposti yhteen. [1]

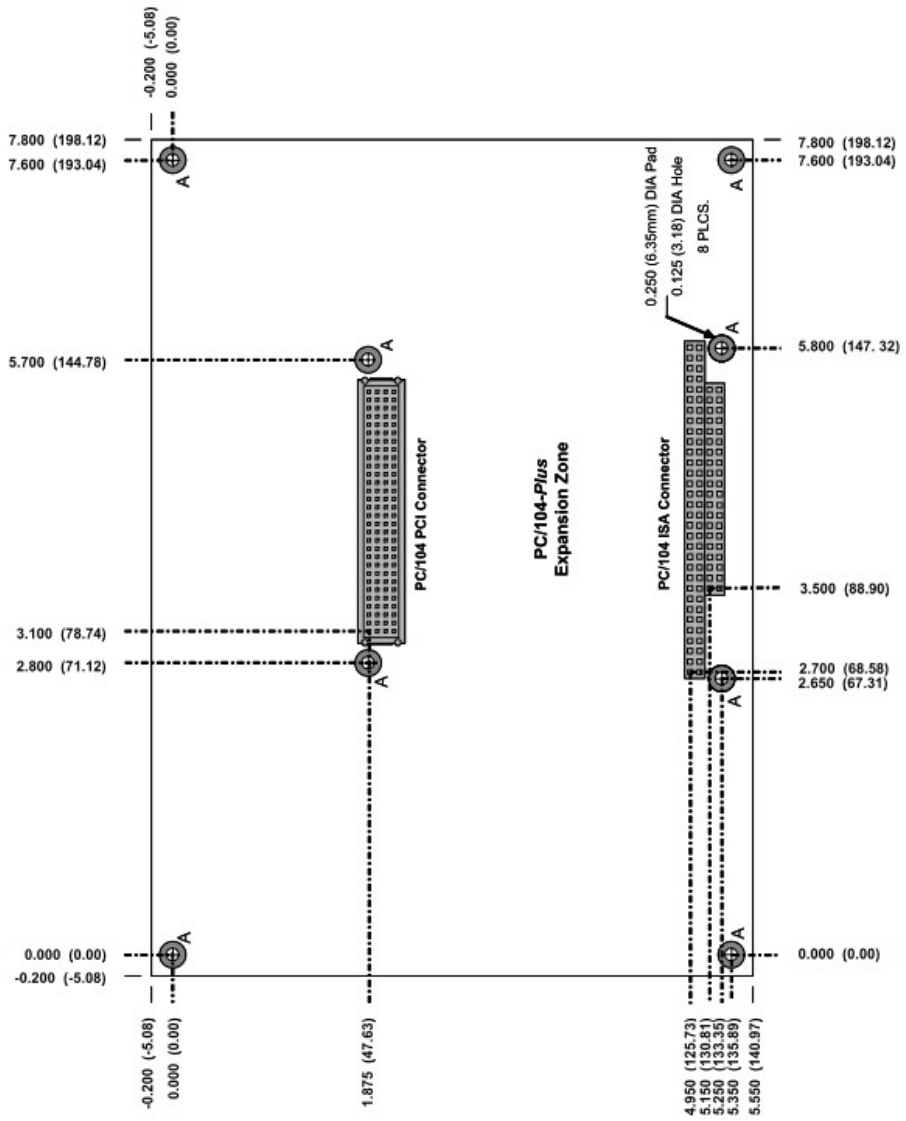


Kuva 8. PCIe/104-kortin mekaaniset mitat. [8]

3.7 EBX & EBX Express

Ennen sulautettujen järjestelmien suunnittelijoilla oli ongelmana se, että jouduttiin valitsemaan emolevyt mallistoista jotka oli suunniteltu työpöytäkäyttöön. Näiden emolevyjen koko ja tehonkulutus olivat suurin taakka suunniteltaessa sulautettujen järjestelmien ratkaisuja. [1]

EBX (Embedded Board eXpandable) on sulautettuja järjestelmiä varten luotu emolevystandardi. Se on sulautettujen järjestelmien alan johtavien yritysten yhteistyössä luoma standardi, joka on tarpeeksi pieni sulautettuihin järjestelmiin ja tarpeeksi suuri pitämään sisällään kaikki sulautetun tietokonejärjestelmän tarvittavat ominaisuudet: prosessori-, keskusmuisti-, massamuisti rajapinnat, näytönohjaimen, sarjaportit, rinnakkaisportit ja muut liitännät. [1]



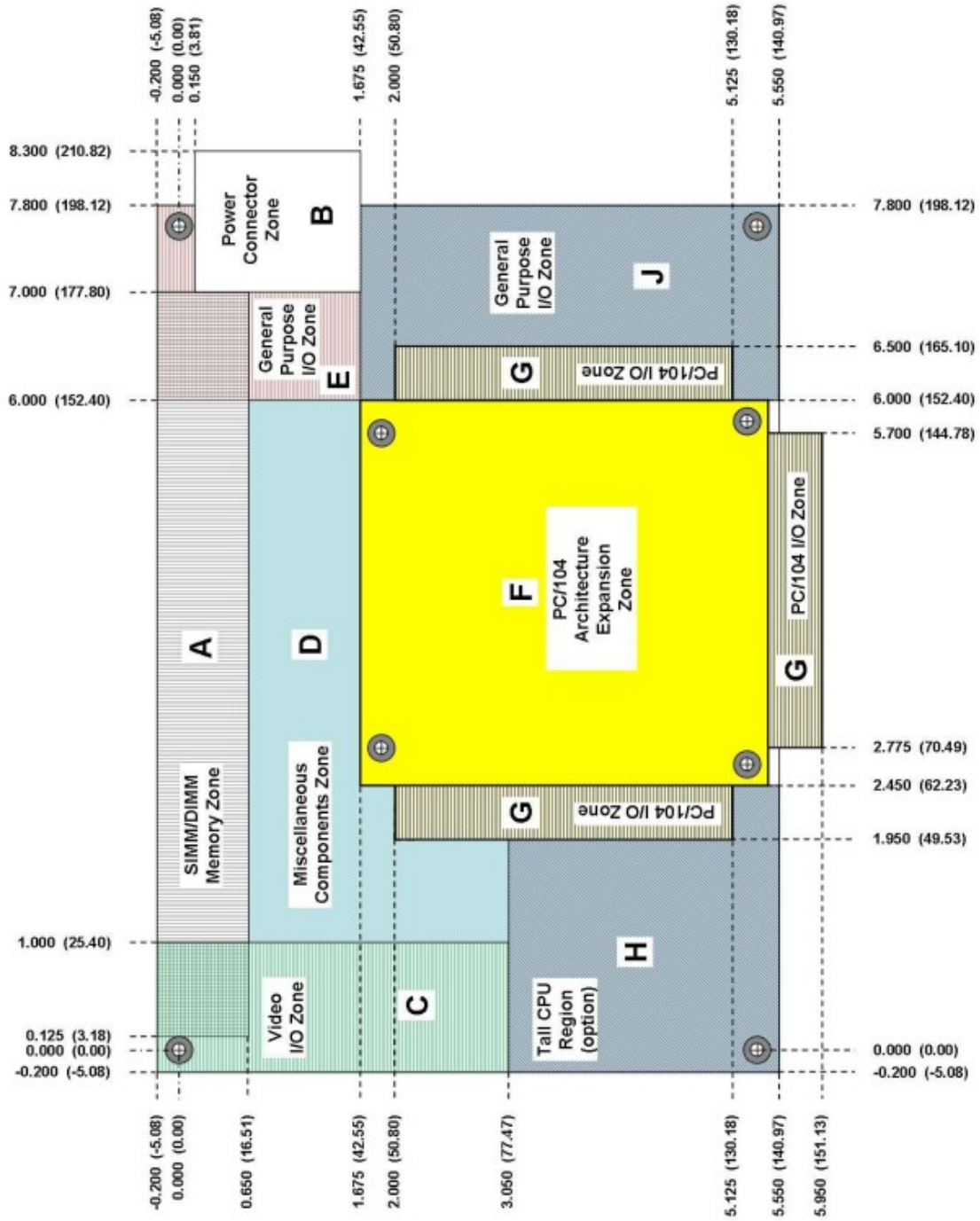
Kuva 9. EBX-emolevyn mekaaniset mitat. [9]

Käytännössä EBX on tavallinen emolevy pienemmässä koossa ja siihen on lisätty liittimet halutuille standardoiduille laajennuskorteille joita ovat PC/104, PC/104-Plus, PCI-104, PCI/104-Express, PCIe/104, PC Card ja ExpressCard. PC/104-moduuleita siis voidaan pinota EBX-levylle useita päällekkäin ja lisää laajennettavuutta tuo PC Card ja ExpressCard-väylä. Myös itse EBX-kortti on mahdollista liittää johonkin suurempaan alustaan pinoamalla samalla tavalla kuin PC/104-moduulit.

EBX-standardin emolevyt ovat kooltaan 5,75 x 8,00 tuumaa (146 x 203 mm). Levyllä on kahdeksan kiinnitysreikää, joista neljä sijaitsee levyn kulmissa ja toiset neljä PC/104-kortin sijoituspaikalla. EBX-levyt on jaettu useaan alueeseen, joista kukin on tarkoitettu tietylle rajapinnalle ja sen komponenteille. Nämä alueet on listattu taulukkoon 1 ja niiden sijainnit EBX-emolevyllä on esitetty kuvassa 10.

Taulukko 1. EBX-emolevyjen alueiden selitykset. [9]

Zone	Description	Max. Component Height Inch (mm)
A	Memory expansion	1.5” (38.1mm)
B	Power connector	0.5” (12.7mm)
C	Video I/O (option) (includes mating connectors)	0.75” (19.1mm)
D	Misc. primary side components	0.75” (19.1mm)
E	General purpose I/O, tall region (includes mating connectors)	0.75” (19.1mm)
F	PC/104 stackable expansion location (Primary and secondary side)	See PC/104-*Plus* or PCI/104-Express specification
G	PC/104 architecture module I/O areas	0.600” (15.24mm)
H	Tall CPU (option) (includes heat sink)	1.2” (30.5mm)
I	PC Card slot (option)	0.6” (15.2mm)
J	General purpose I/O, low profile region (includes mating connectors)	0.5” (12.7mm)
---	Secondary side components	0.19” (4.8mm)
---	Board thickness	0.062” (1.57mm)



Kuva 10. EBX-emolevyjen alueet ja mekaaniset mitat. [9]

EBX Express -standardin emolevyt ovat muuten samanlaisia kuin EBX-levyt, mutta niissä on ISA-väylän liittimen sijaan PCI Express -väylän liitin.

3.8 EPIC & EPIC Express

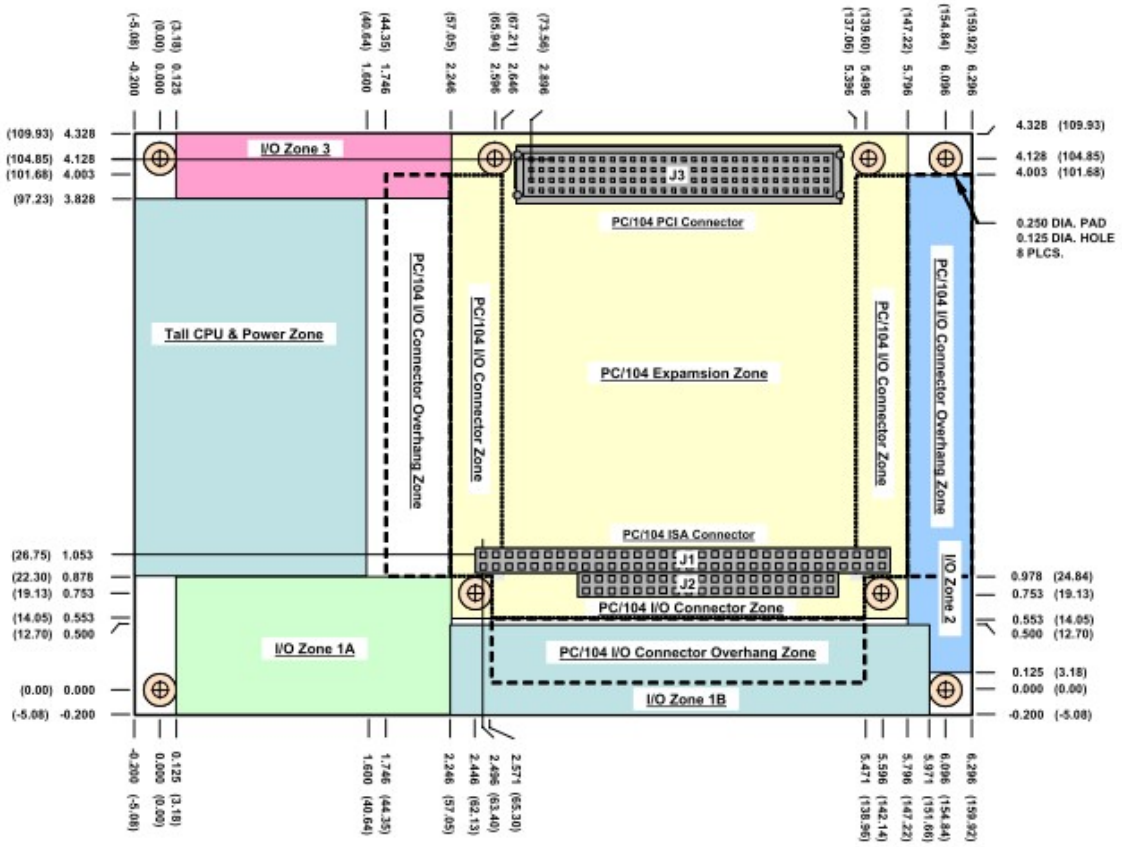
EPIC (Embedde Platform for Industrial Computing) on sulautettujen järjestelmien emolevystandardi, joka EBX:n tavoin tukee PC/104-moduuleita mutta se on kooltaan hieman pienempi. EPIC-korttien koko on 4,528 x 6,496 tuumaa (115,00 x 165,00 mm). Erona EBX:ään on myös se, että EPIC-emolevyissä on panostettu enemmän I/O-liitäntöihin, jotka voidaan toteuttaa pinniliitäntöinä tai PC-maailmasta tutuin liittimin.

Käytännössä EPIC ei eroa juurikaan EBX:stä. EPIC-kortissa on kahdeksan kiinnitysreikää, neljä kortin nurkissa ja toiset neljä PC/104-kortin liitospaikalla. EBX:n tapaan EPIC-emolevylle voidaan pinota useita PC/104-laajennusmoduuleita. EPIC-kortti on mahdollista liittää johonkin suurempaan alustaan PC/104-moduulien tapaan pinoamalla.

Kuten EBX-emolevyt, on myös EPIC-emolevyt jaettu eri alueisiin eri rajapinnoille ja niiden komponenteille. EPIC-korteissa on neljä eri I/O-aluetta, prosessori- ja virta-alue ja PC/104-laajennusalueet. Nämä alueet on listattu taulukkoon 2 ja niiden sijainnit EPIC-emolevyllä on esitetty kuvassa 11.

Taulukko 2. EPIC-emolevyjen alueiden selitykset. [10]

Zone	Max. Component Height (in.)
I/O Zone 1A	None
I/O Zone 1B	None except in overlap area (See below)
I/O Zone 2	None except in overlap area (See below)
I/O Zone 3	None except in overlap area (See below)
Tall CPU and Power Zone	None
PC/104 Expansion Zone	0.345 in. (8.76 mm)
PC/104 I/O Connector Zone	0.345 in. (8.76 mm)
PC/104 I/O Connector Overhang Zone	0.600 in. (15.24 mm)



Kuva 11. EPIC-emolevyjen alueet ja mekaaniset mitat. [10]

EPIC-emolevystandardista on olemassa myös EPIC Express -standardi, joka EBX Expressin tapaan ei eroa itse EPIC-standardista muulla tavalla kuin, että siinä on ISA-väylän liittimen sijaan PCI Express -väylän liitin.

4 TYÖN TOTEUTUS

4.1 Työn lähtökohdat

Tämän opinnäytetyön lähtökohtana oli uusia Savonia-ammattikorkeakoulun elektroniikan ja tietotekniikan laboratoriossa käytössä olleet PC/104-teollisuustietokoneet ja suunnitella uusia laboratorioharjoituksia. Nämä harjoitukset on esitetty liitteessä 1.

Tarkoituksena oli etsiä uudet PC/104 SBC (Single Board Computer) -prosessorikortit, joiden hinta olisi kohtuullinen ja ominaisuudet vastaisivat paremmin tämänhetkisiä tarpeita. Myös laitteiden koteloinnin toteutusta mietittiin jo tässä vaiheessa, käytetäänkö samoja koteloita kuin edellisten korttien kanssa vai etsitäänkö uudet. Edellisissä koteloissa oli asialliset virtalähteet, joiden käyttöä myös uusien korttien virtalähteinä pidettiin viisaana ja taloudellisena ratkaisuna.

Myös kaapeloinnin toteutus tuli miettiä ja selvittää, saisiko tuleville korteille tilattua myös kaapelisarjat vai jouduttaisiinko ne tekemään itse kuten edellisissä laitteissa.

Projektissa aikataulullisesti korkeimmalle priorisoitiin uusien laitteiden etsiminen, tilaaminen ja kokoaminen. Tavoitteena oli saada muutama laite mahdollisimman nopeasti käyttöön, jotta laboratoriossa olisi käytettävänä PC/104-laitteisto. Mainittakoon, että edelliset kortit oli annettu pois ja laboratoriossa oli jäljellä vain laitteiden kotelot sekä virtalähteet. Tästä syystä ensimmäisten laitteiden kokoamisella oli hieman kiire.

Laboratorioharjoituksien laatimisella ei ollut kiire, koska uudet tehtävät tulisivat käyttöön vasta myöhempänä ajankohtana. Harjoituksiin tuli laatia paitsi teoriatehtäviä PC/104-standardista ja käytössä olevasta kortista, myös käytäntöä, tässä tapauksessa BIOS keskeytyksien koodaamista.

Lisäksi henkilökohtaisena tavoitteena oli tutustua laiteläheiseen ohjelmointiin Rhodeus-LC-kortin avulla. Tarkoituksena oli laatia hälytysjärjestelmä FreeRTOS-reaaliaikakäyttöjärjestelmää käyttäen. Ohjelmalla oli tarkoitus ohjata rinnakkaisporttiin liitettyä kytkentää, jolla pystytään simuloimaan hälytysjärjestelmän toimintaa käytännössä.

Seuraavaksi esitellään elektroniikan laboratoriossa aikaisemmin käytössä olleen Microspace MSM586SEN/SEV/SL PC/104 -kortin keskeisimpiä ominaisuuksia, minkä jälkeen tarkastellaan uusia Rhodeus-LC PC/104 -kortteja.

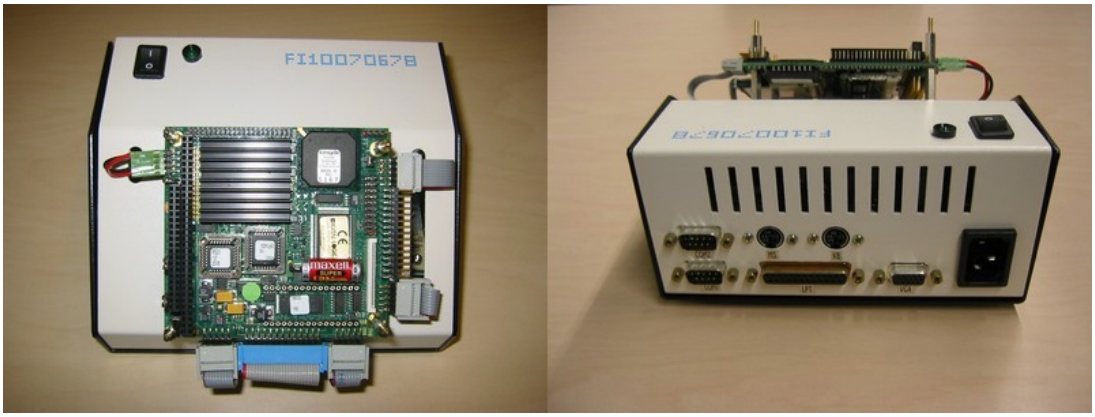
4.2 Microspace MSM586SEN/SEV/SL

Elektroniikan laboratoriossa oli aikaisemmin käytössä Microspace MSM586SEN/SEV/SL PC/104 -kortti. Taulukossa 3 on listattu kyseisen kortin keskeisimmät ominaisuudet.

Taulukko 3. Microspace MSM586SEN/SEV/SL -kortin tekniset tiedot. [12]

Size	PC/104 Board			
CPU	ELAN 520			
Performance	133 Mhz			
2-L Cache	-			
DRAM	16 – 128 MB, SDRAM-SODIMM			
	SL: Soldered DRAM 32/64 MB			
Bootable Flash	DOC2000			
CompactFlash	1 Slot			
Keyboard/Mouse	2xPS/2			
BootDrive	FD, HD, DOC			
Harddisk	44pin IDE			
Floppy	26pin FD			
		RS232	RS422	RS485
Serial	COM1	yes	-	opt.
	COM2	yes	-	opt.
	COM3	yes	-	opt.
	COM4	yes	opt.	opt.
Parallel	LPT1			
USB	-			
LAN (SEV, SEN)	100/10BASE-T, INTEL 82559ER			
SCSI	-			
Audio	-			
Video(SEV)	69000, 2 MB			
	CRT	Standard		
	LCD	TFT/STN		
	Resolution	24 Bit		
	Level	3V/5V		
Video Input	-			
EEPROM	1k			
Watchdog	onboard			
RTC Battery				
Power Management	yes, APM			
Supply	5V			
Consumption	900mA (typ.)			

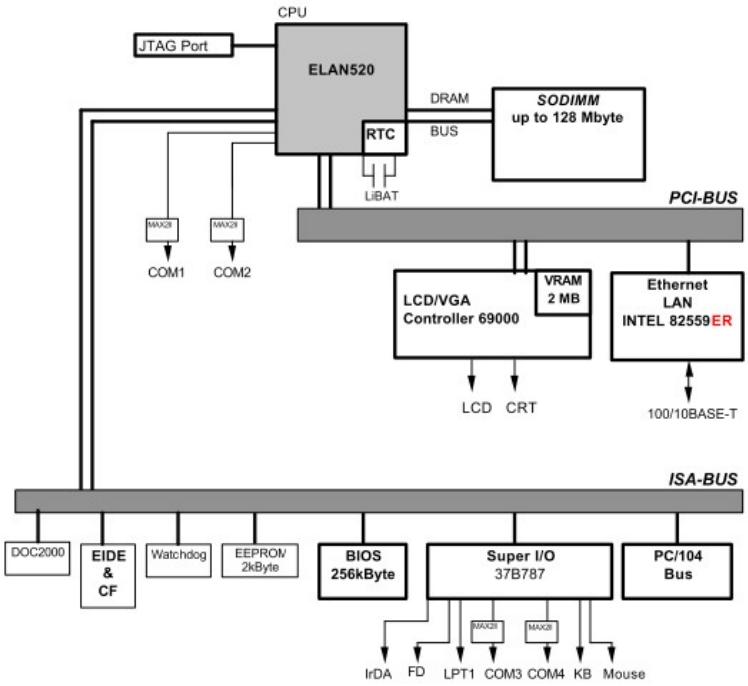
Kuvassa 12 näkyy Microspace MSM586SEN/SEV/SL -kortti kiinnitettynä kotelonsa. Samoja koteloita käytettiin Rhodeus-LC-korteille (ks. luku 4.4). Kortin lohkokaavio on esitetty kuvassa 13.



Kuva 12. Microspace MSM586SEN/SEV/SL -laite kotelossaan.

Kortin datalehdessä korttia kuvaillaan seuraavalla tavalla. Seuraava kappale on suomennettu datalehdestä.

Microspacen PC/104 MSM586SL/SEN/SEV:ssä on kaikki samat rajapinnat kuin tavallisessa standardi PC:ssä ja useita muita lisäominaisuuksia, kuten esimerkiksi neljä COM-porttia, DOC2000 liitäntä, ohjelmoitava vahtikoira, LAN-tuki ja CompactFlash-paikka. Jotkin sarjaliikennerajapinnoista ovat haluttaessa valittavissa joko RS422:ksi, RS485:ksi tai TTL:ksi. Kortille juotetun SDRAM-muistin ansiosta tämä moduuli kestää paremmin tärinän ja iskut. Sen ainutlaatuinen toiminnallinen monimuotoisuus tekee tästä kortista ns. ”all-rounder” -kortin sulautettujen järjestelmien ratkaisuihin. [12]



Kuva 13. MSM586/SEN/SEV/SL:n lohkokaavio. [11]

4.3 Rhodeus-LC

Tavoitteena oli siis löytää uudet PC/104-laitteet, jotka olisivat tarpeeksi suorituskykyisiä ja sopivan hintaisia. Ensimmäiseksi etsittiin internetistä eri PC/104-laitteiden valmistajien ja toimittajien sivustoja, joilta katseltiin sopivaa korttia. Uusien korttien ensimmäinen vaatimus oli, että niissä tuli olla USB-portit. Myös suorituskykyä oli tarkoitus kasvattaa.

Ongelmaksi osoittautui aluksi se, että hyvin harvan valmistajan tai toimittajan sivuilla oli ilmoitettu tarjolla olevien laitteiden hintoja. Mahdolliset kortit rajattiin pariin malliin, joista lähetettiin näille valmistajille tarjouspyynnöt kahdeksasta kortista. Vastaukset saatiin kahdesta paikasta. PC/104-kortit tilattiin lopulta Diamond Systems Corporationista, jonka päämaja sijaitsee Yhdysvalloissa, mutta se toimittaa tavaraa ympäri maailmaa.

Seuraavaksi siis perehdytään Savonia-ammattikorkeakoulun tekniikan Kuopion yksikön elektroniikan laboratorion uusiin PC/104-kortteihin eli Diamond Systems Corporationin Rhodeus-LC -kortteihin.

Rhodeus-LC tarjoaa keskitasoista laskentatehoa kompaktissa, vähävirtaisessa ja edullisessa PC/104-kortissa. Se on hintalaatusuhteeltaan erinomainen valinta esimerkiksi lääkintäväline-, prosessin valvonta- ja instrumentointisovelluksissa. [14]

Rhodeus-LC hyödyntää AMD:n Geode LX800 -prosessoria, joka toimii tuulettimetta 500 MHz:n kellotaajuudella. LX800:ssa on integroitu VGA-näytönohjain 2D-kiihdytyksellä ja tuki korkean resoluution CRT- ja LCD-näytöille. Lisäksi Realtek RTL8100CL Ethernet -ohjain tarjoaa 10/100Mbps verkkotuen sekä wake-on-LAN toiminnon BIOSin avulla. Rhodeus-LC tukee SDRAM-keskusmuistia 1 GB:iin asti, jonka liittimenä on 200-pinninen DDR SO-DIMM. [14]

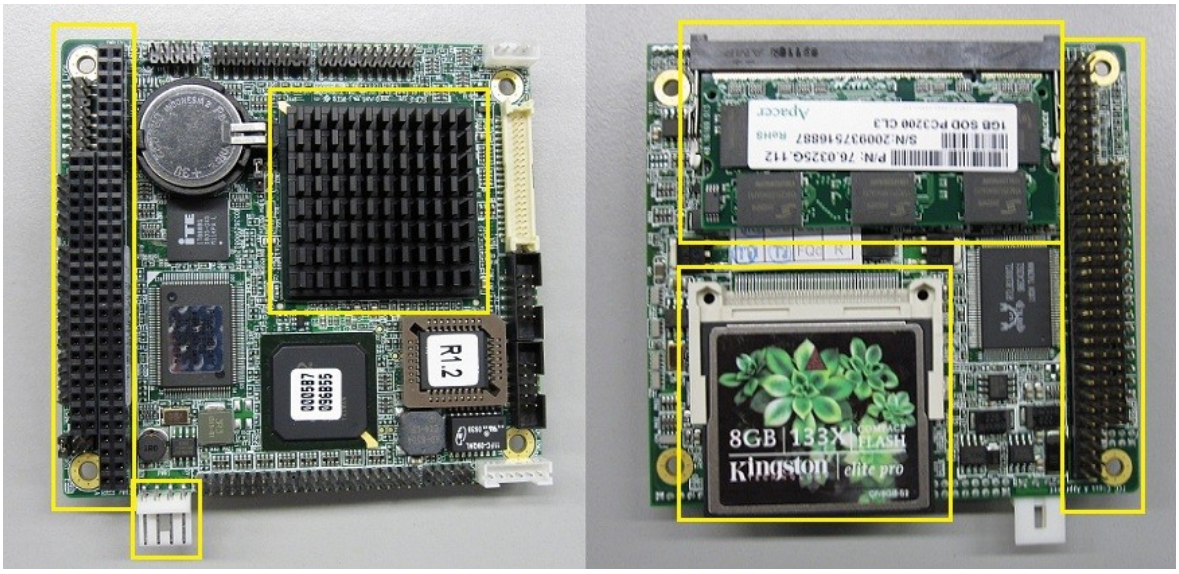
Rhodeus-LC-kortilla on useita I/O portteja minkä tahansa sovelluksen tarpeisiin, kuten esimerkiksi Ethernet, UDMA-33 IDE, CompactFlash, floppy, parallel, ps/2-näppäimistö ja ps/2-hiiri sekä kaksi USB 2.0 -porttia. Kortissa on myös kaksi 16450-yhteensopivaa sarjaporttia, joista toinen käyttää oletuksena RS-232 -protokollaa ja toinen portti voidaan jumpperilla valita käyttämään RS-233-, RS-422- tai RS-485 -protokollaa. Vahtikoira-ajastin tarjoaa suojaa ohjelmien kaatuessa ja sen viiveet ovat ohjelmoitavissa 1 - 255 sekunnin välein tai 1 - 255 minuutin välein. [14]

Rhodeus-LC:n keskeisimmät ominaisuudet on listattu taulukkoon 4. Rhodeus-LC-laitteen lohkokaavio on esitelty kuvassa 16.

Taulukko 4. Rhodeus-LC kortin tekniset tiedot. [14]

Size	PC/104 Board
CPU	AMD Geode^{tm} LX800
Performance	500MHz
Cache	64K L1 cache & 128K L2 cache
Chipset	CS5536
I/O Chip	W83627HG
System Memory	1 x 200 pins DDR SO-DIMM up to 1GB SDRAM
VGA/LCD Controller	AMD Geode LX series CPU integrated VGA controller with 2D engine (shared memory max, 64MB)
LCD	Supports 18/24-bit TTL up to 1280 x 1024
Ethrnet	Realtek 8100CL 10/100 Base-T Ethernet
BIOS	Phoenix-Award BIOS
IDE Interface	1 x Ultra DMA 33, supports 2 IDE devices
Serial Ports	2 x RS-232 ports COM1: RS-232 COM2: RS-232/RS-422/RS-485 selectable
Parallel Port	1 x SPP/EPP/ECP mode
Floppy Port	1 x Floppy connector
Keyboard / Mouse	Standard PS/2 Keyboard and Mouse
Universal Serial Bus	2 x USB 2.0 ports
On-Board Mass Storage	1 x Type II Compact flashdisk socket up to 4GB
Audio	Speaker Out
Hardware Monitor	Integrated in W83627HG
Real-Time Clock	AMD Geode CS5536 built-in RTC with lithium battery
Watchdog Timer	1 – 255 Levels (Sec. Or Min.)
Expansion Bus	PC/104 (ISA) interface
Power Input	+5V +/-5% @ 2.0A typical
Power Consumption	5W typical
Operating Temp.	-20°C to +70°C
Dimension (L x W)	96 x 90 mm (3.775” x 3.550”)

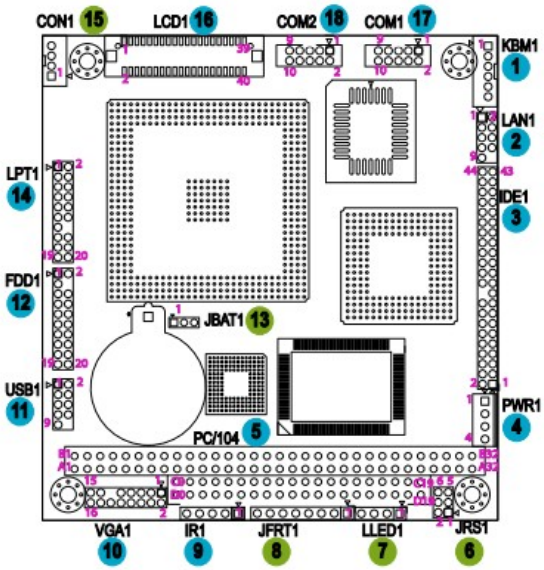
Kuvassa 14 Rhodeus-LC-kortti on kuvattu ylä- ja alapuolelta. Paikallaan ovat myös 1 GB SO-DIMM-muistikampa sekä 8 GB Kingston CompactFlash -muistikortti. Kuvat Rhodeus-LC-kortista kiinnitettynä koteloon esitellään myöhemmin (ks. luku 4.4).



Kuva 14. Rhodeus-LC-kortti kuvattuna ylä- ja alapuolelta.

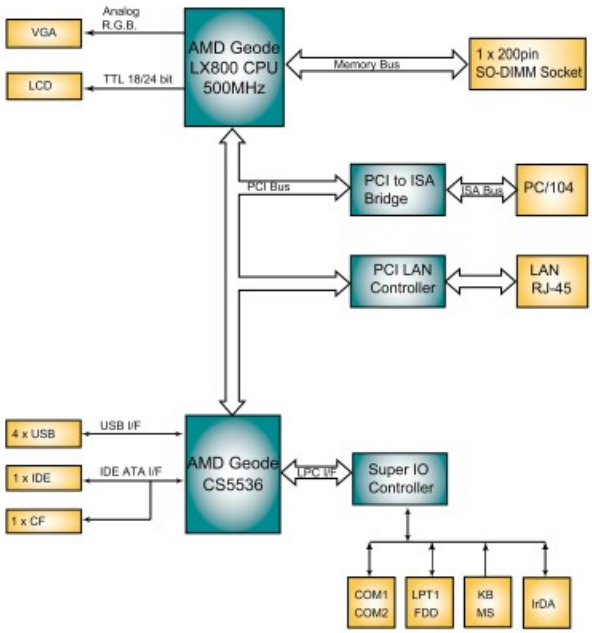
Kuvaan 14 on merkitty oleellisimpia osia. Kuvan vasemmassa laidassa on kuva kortin yläpuolesta, jossa oikealla sijaitsee prosessori, jonka päällä on jäähdytyssiili. Vasemmassa laidassa on ISA-väylän naarasliitin, jonka päälle voidaan liittää laajennuskortteja. Kuvan alareunaan on merkitty virtaliitin. Muut kortin liitännät kuten esimerkiksi näppäimistön, hiiren, sarjaporttien ja näytön liittimet sijaitsevat kortin ulkoreunoilla, näiden liittimien sijainnit ja merkitykset voi tarkastaa kuvasta 15.

Kuvan 14 oikeassa laidassa on kuvattu kortin alapuoli. Tähän kuvaan on merkitty oikealle laidalle ISA-väylän urosliitin, ylös SO-DIMM-liitin ja -muistikampa sekä alalaitaan CompactFlash-liitin ja -kortti.



Kuva 15. Rhodeus-LC-kortin liittimet ja niiden tarkoitukset. [13]

Laboratorion uusien PC/104-järjestelmien kokoonpanoksi tuli siis: Rhodeus-LC SBC-kortti, Kingston 8 GB CompactFlash -muistikortti ja 1 GB DDR SO-DIMM -keskusmuistia. Myös kortille valmistetut kaapelisarjat tilattiin valmiina, koska niitä oli saatavilla. Tämä vähensi huomattavasti työn määrää laitteen koteloinnissa. Kaapelisarjojen kanssa kuitenkin tuli hieman ongelmia, mutta siitä tarkemmin myöhemmin (ks. luku 4.4).



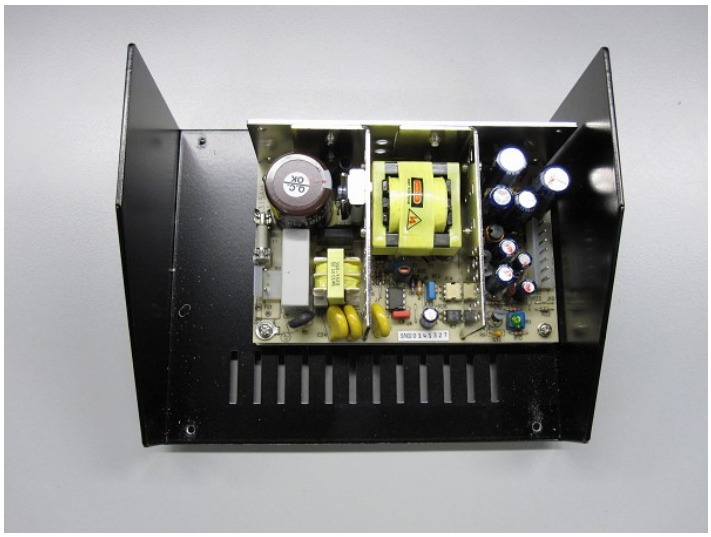
Kuva 16. Rhodeus-LC:n lohkokaavio. [13]

Rhodeus-LC tukee seuraavia käyttöjärjestelmiä: Windows 2000, Windows XP, Windows CE 5.0 ja Linux. [14]

4.4 Laitteen kasaus ja kotelointi

Kun laitteet oli tilattu, oli aika suunnitella laitteen kasausta ja kotelointia. Ensimmäisessä tilauksessa oli mukana kaikki kahdeksan korttia, muistikammat ja muistikortit, mutta vain yksi kaapelisarja. Tarkoituksena tässä vaiheessa oli kasata yksi laite, ja mikäli kotelointiratkaisu tuntuisi järkevältä kasattaisiin loputkin laitteet samalla menetelmällä.

Lähtökohtaisesti tuntui järkevimmältä käyttää edellisten laitteiden koteloita koska niissä oli paikallaan virtalähteet, jotka olivat sopivat myös uusille laitteille. Asiaan vaikutti myös se, että kotelossa oli jo valmiina reiät liittimille, joista suurinta osaa voitaisiin hyödyntää. Ensimmäisen uusi laite päätettiin kasata käyttäen hyväksi vanhan laitteen koteloa ja virtalähdettä (kuva 17).

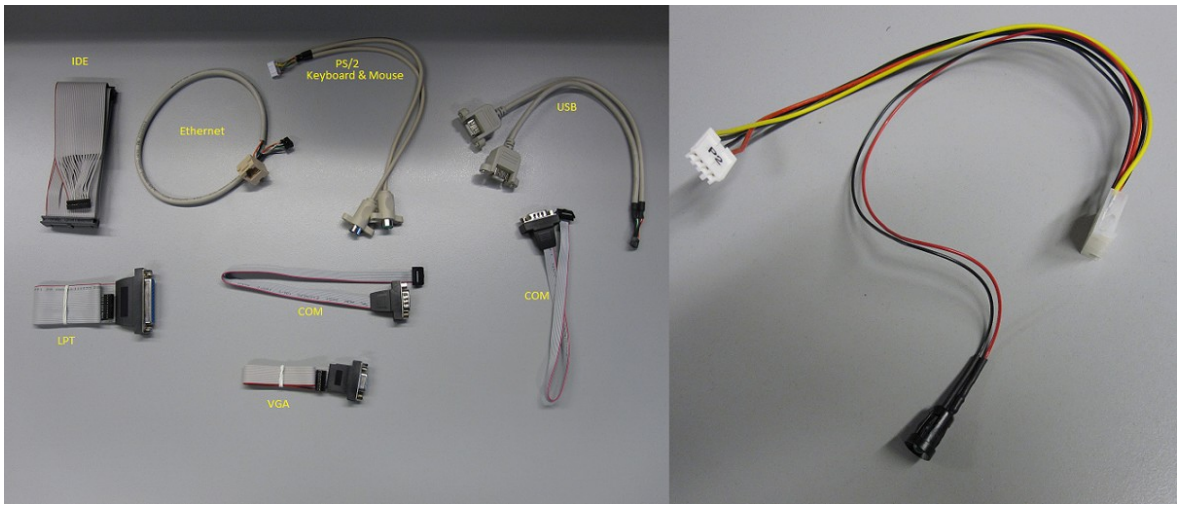


Kuva 17. Virtalähde.

Koska Rhodeus-LC-kortissa on enemmän ulkoisia liitäntöjä kuin Microspacen kortissa, oli koteloa siis pakko hieman muokata, jotta kaikki liittimet saataisiin integroitua koteloon. Kotelossa oli valmiina reiät kahdelle COM-portille, yhdelle LPT-portille, hiirelle, näppäimistölle ja näytölle (kuva 12). Koska nämä liittimet ovat standardien mukaisia oli tavoitteena käyttää näitä valmiita reikiä. Koteloa siis jouduttaisiin muokkaamaan ainakin kahden USB-liittimen ja yhden verkkoliittimen tarpeisiin.

Myös virtakaapelit piti valmistaa itse sopiviksi, johtuen korttien virtaliittimien eroavaisuuksista ja jännitteen tarpeesta. Vanhoissa virtakaapeleissa oli käytetty vain +5V jännitettä, joka oli otettu kolmipaikkaisella molex-liittimellä virtalähteeltä ja kaapelin toisessa päässä oli 2 x 4 paikkainen riviliittimen naarasosa. Kaapeliin oli myös liitetty LED, joka

kertoi laitteen olevan päällä. Koska Rhodeus-LC-kortti käyttää +5 V lisäksi +12 V jännitettä, piti virtalähteen päähän asentaa useampi paikkainen molex-liitin, jotta molemmat jännitetasot saataisiin käyttöön. Myös kortille tuleva virtaliitin piti vaihtaa. Tähän tarkoitukseen kelpasi diskettiaseman virtaliitin. Uusiin kaapeleihin lisättiin myös edelliset virtaLEDit. Uudessa virtakaapelissa on siis virtalähteen päässä viisipaikkainen molex-liitin, josta otettiin myös jännite LEDille ja kortin päässä diskettiaseman virtaliitin. Kuvassa 18 on nähtävissä virtakaapeli, sekä Rhodeus-LC-kortin kaapelisarjan kaapelit (kuvasta puuttuu floppy-kaapeli).



Kuva 18. Rhodeus-LC-kaapelisarjan kaapelit ja virtakaapeli.

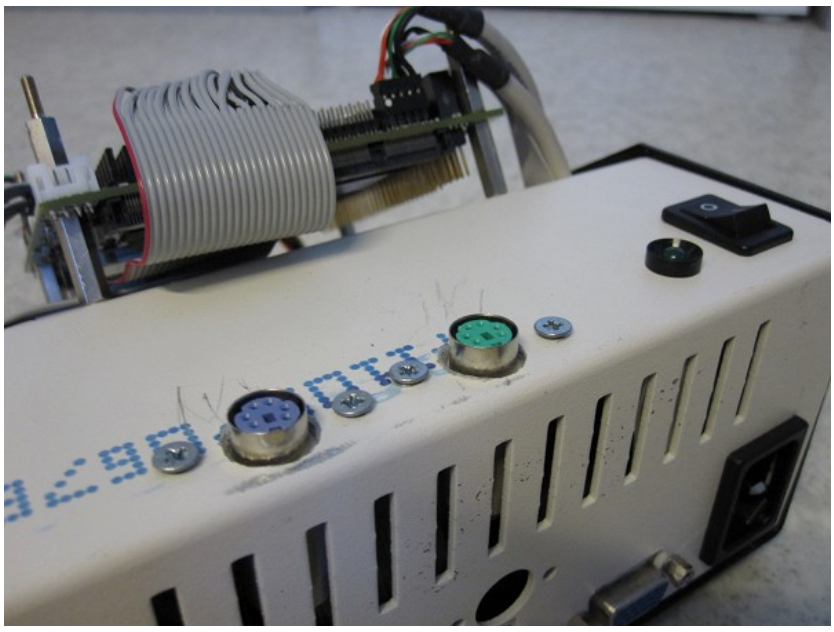
Ensimmäiset ongelmat aiheutuivat tilanpuutteesta kotelon takaseinän ja virtalähteen välissä. Tämä aiheutti sen, että koteloa ei saanut kiinni kun kaapelisarjan kaapelit olivat paikoillaan takaseinässä. Tässä vaiheessa suunniteltiin eri vaihtoehtoja miten kotelon saisi kiinni asti kun kaapelit ovat paikallaan. Aluksi harkittiin virtalähteen siirtämistä hieman eteenpäin kotelon pohjalla, mutta tämä olisi aiheuttanut tilanpuutteen kotelon etuosassa.

Kaapeleiden suojamuoveja tutkailtaessa huomattiin, että niistä oli varaa karsia hieman muovia pois, jolloin ne sopisivat kotelon takaseinän ja virtalähteen väliin. Tätä kokeiltiin ensin yhdellä kaapelilla ja ratkaisu tuntui toimivalta. Modifikaatio ei myöskään aiheutunut mitään mikä olisi voinut vaikuttaa laitteen toimintaan tai sen turvallisuuteen. Tämä muokkaus tehtiin siis rinnakkaisportin, sarjaporttien ja näytön kaapeleille, jonka jälkeen ne menivät koteloon vaivatta.

Seuraava ongelma oli näppäimistön ja hiiren kaapelin kohdalla. Ongelma oli sama kuin aikaisemmin mainittujen kaapeleiden kohdalla, eli tilanpuute. Näiden kaapeleiden kohdalla

itse kaapeleiden vastaava muokkaaminen ei ollut mahdollista, joten oli pakko kehitellä jokin toinen ratkaisu näille kaapeleille.

Koska kotelossa oli rajallinen määrä tilaa ja pohjalevy oli huomattavasti paksumpaa materiaalia kuin kotelon kansilevy, ei hiiren ja näppäimistön liittimille ollut kovin monta sijoitusmahdollisuutta. Pyrkimyksenä oli asentamaan nuo liittimet kansilevyyn sillä edellytyksellä, että USB-portin liittimille jää tilaa kotelon etuosaan ja verkkoliittimelle kotelon takaosaan. Tämä rajasi sijoitusmahdollisuuksia entistä enemmän, joten päädyttiin ratkaisuun, joka tuntui olevan ainoa järkevä vaihtoehto. Näppäimistön ja hiiren liittimet sijoitettiin kotelon kansilevyn päälle. Tässä ratkaisussa ei tullut ongelmia tilanpuutteesta kotelon takaseinän ja virtalähteen välissä, eikä liittimet ole muiden liittimien tiellä. Joten kanteen porattiin reiät liittimille ja niiden kiinnitysruuveille. Miinuksena lopputuloksesta voidaan mainita se, että liittimet osoittavat ylöspäin mutta käytön kannalta tällä ei ole mitään merkitystä. Ratkaisuun voitiin olla tyytyväisiä, liittimet kotelossa näkyy kuvassa 19.



Kuva 19. Hiiren ja näppäimistön liittimet kotelossa.

Seuraavaksi oli käsittelyvuorossa USB-portin liittimet. Näille liittimille oli jo suunniteltu alustavaksi sijoituspaikaksi kotelon etuosaa. Kotelon etuosassa oli nimittäin molemmilla reunoilla sopivan kokoiset alueet joihin liittimet voisi kiinnittää. Näidenkin liittimien kohdalla ongelmia saattaisi aiheuttaa mahdollinen tilanpuute kotelon sisällä. Tarkemmin se, että ottaako toinen USB-liitin kiinni virtalähteeseen kotelon sisällä. Tämä huoli onneksi osoittautui turhaksi, sillä molemmat USB-liittimet mahtuivat vaivattomasti kotelon sisällä.

USB-liittimien paikkojen muokkaamisessa tarvittiin jo hieman enemmän tarkkuutta reikien koossa ja sijoituksessa, koska näille liittimille varattu pinta-ala oli rajallinen. Tässä vaiheessa kiinnitettiin vielä erityishuomiota liittimen ja virtalähteen välimaastoon. Toinen liitin sijoitettiin varmuuden vuoksi mahdollisimman ylös, ettei se ottaisi kiinni virtalähteeseen. Reikien koossa piti myös ottaa huomioon, että reikä ei ole liian pieni sillä se saattaisi aiheuttaa ongelmia joidenkin USB-laitteiden kanssa. Liian iso reikä taas saattaisi näyttää rumalta.

Reikien poraamista varten tehtiin pieni piirtomalli paperista, joka teipattiin sopivaan kohtaan koteloa ja merkittiin reikien paikat. Poran ja viilan avulla koteloon ilmestyi itse liittimien ja kiinnitysruuvien reiät. Seuraavaksi kantta USB-liittimineen koitettiin kiinnittää kotelon alaosaan ja todettiin, että lopputulos oli hyvä. Liittimet eivät olleet kiinni virtalähteessä, kotelo meni kiinni hyvin ja liittimien sijainti oli hyvä. USB-liitin kiinnitettynä koteloon näkyy kuvassa 20.

Kuva 20. USB-liitin kotelossa.

Viimeinen kotelon muokkaamista vaativa liitin oli verkkoliitin. Tälle liittimelle oli aluksi hieman vaikeuksia keksiä sijoituspaikkaa. Aluksi mietittiin, olisiko järkevää jättää kyseinen kaapeli liittimineen kokonaan ulos kotelosta. Vaihtoehtona oli myös, että tehtäisiin verkkokäyttöä varten omat kaapelit, joissa olisi ollut toisessa päässä RJ-45-urosliitin ja toisessa päässä Rhodeus-LC-kortille sopiva liitin. Tästä ajatuksesta päätettiin kuitenkin luopua.

Verkkoliitin oli asennustavaltaan ns. kiskomainen, eli siinä oli ylä- ja alaosissa pienet urat, joiden avulla se pysyisi paikallaan tietynkokoisessa urassa. Käsittelyssä olevan kotelon kansi osoittautui juuri samanpaksuiseksi kuin verkkoliittimen urat. Tästä syystä liittimelle suunniteltiin asennuspaikkaa kotelon kansilevystä.

Tavoitteena oli saada liitin asennettua takalevyyn, koska kotelon päällä oli ennestään jo hiiren ja näppäimistön liittimet eikä enempää liittimiä haluttu osoittamaan ylöspäin. Myös etulevy voitiin unohtaa, sillä se oli varattu USB-liitimille.

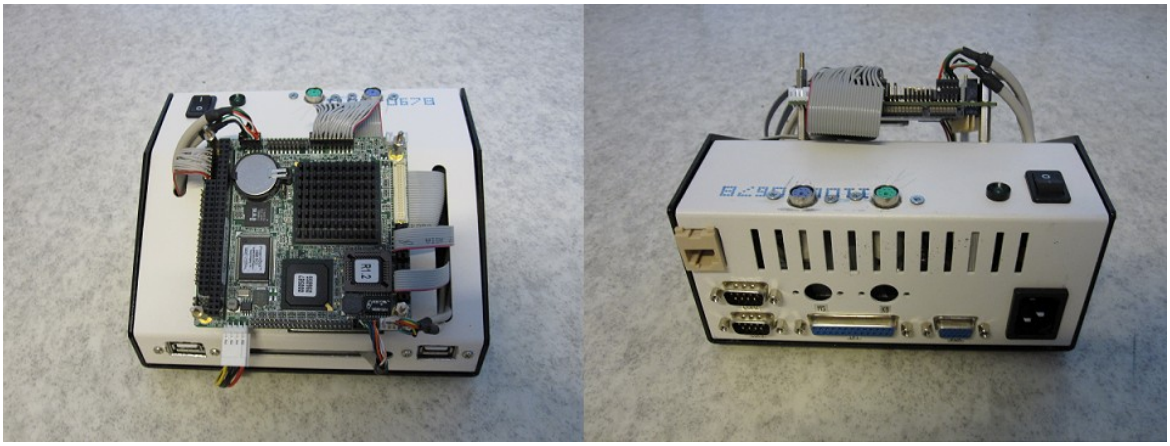
Lopulta sijoituspaikka valittiin kannen takaosasta sarjaporttiliittimien yläpuolelta. Koska liitin ei ollut mekaanisilta mitoiltaan kovin suuri, pystyi sen asentamaan kyseiseen paikkaan ongelmitta. Tähän kohtaan kannen reunaan tehtiin juuri oikeankokoinen lovi niin, että liitin mahtui siihen ikään kuin kiskoille. Loven syvyys oli tarkalleen liittimen levyinen. Tämä mahdollisti sen, että liitin saatiin pysymään paikallaan kotelon alaosan avulla. Kun kotelo suljetaan, pitää kotelon alaosan seinä sitä paikoillaan sen ”kiskolla” eli kotelon reunaan tehdyllä lovella. Ratkaisu oli toimiva ja siihen oltiin tyytyväisiä. Verkkoliitin asennettuna koteloon on nähtävissä kuvassa 21.

Kuva 21. Verkkoliitin kotelossa.

Tässä vaiheessa oli saatu valmiiksi yhden kotelon kansi tarvittavine muokkauksineen. Kun kaikki liittimet asentuivat koteloon hyvin, oli seuraavaksi muokattava loput seitsemän koteloa. Aluksi päätettiin kuitenkin koota yksi laite valmiiksi siltä varalta, että kasaamisessa ilmenisi muita ongelmia.

Seuraavaksi kiinnitettiin kaapelit koteloon ja kytkettiin ne kortille. Lattakaapelit kiinnitettiin kotelon sisäseiniin kaksipuoleisella teipillä, joka toimi myös pienenä vedonpoistona. Kaapeleiden kytkemisessä ei ollut ongelmia, vaan ne menivät kotelon sisälle vaivattomasti. Kotelo ei ollut liian ahdas ja meni kiinni vaivatta.

Kun yksi laite oli kasattu käyttökuntoon ja lopputulos oli tyydyttävä, oli seuraavaksi vuorossa loppujen seitsemän kotelon muokkaaminen. Kuvassa 22 näkyy laiteen kokoamisen lopputulos.



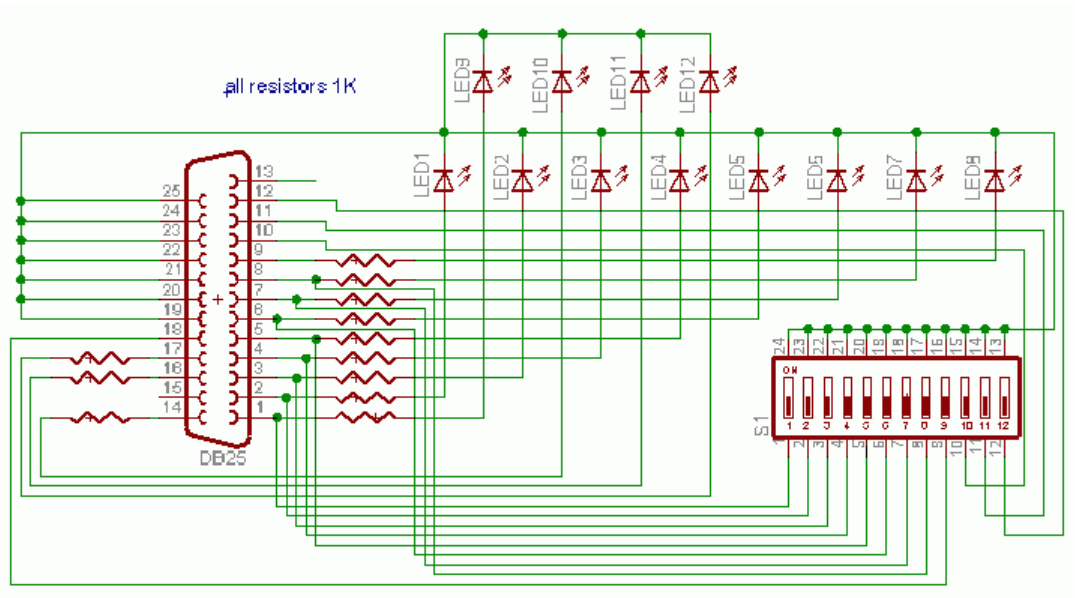
Kuva 22. Laitteen kokoamisen lopputulos.

Loppujen seitsemän kotelon muokkaus ja kaapelointi sujui rutiininomaisesti. Lopputuloksena oli siis kahdeksan laitetta valmiina käyttöön.

4.5 FreeRTOS-hälytysjärjestelmä rinnakkaisportille

Tässä opinnäytetyössä yhtenä tavoitteena oli ohjelmoida hälytysjärjestelmä FreeRTOS-reaaliaikakäyttöjärjestelmälle, joka käyttää hyväksi Rhodeus-LC PC/104 -laitteen rinnakkaisporttia ja siihen liitettyä simulointikytkentää.

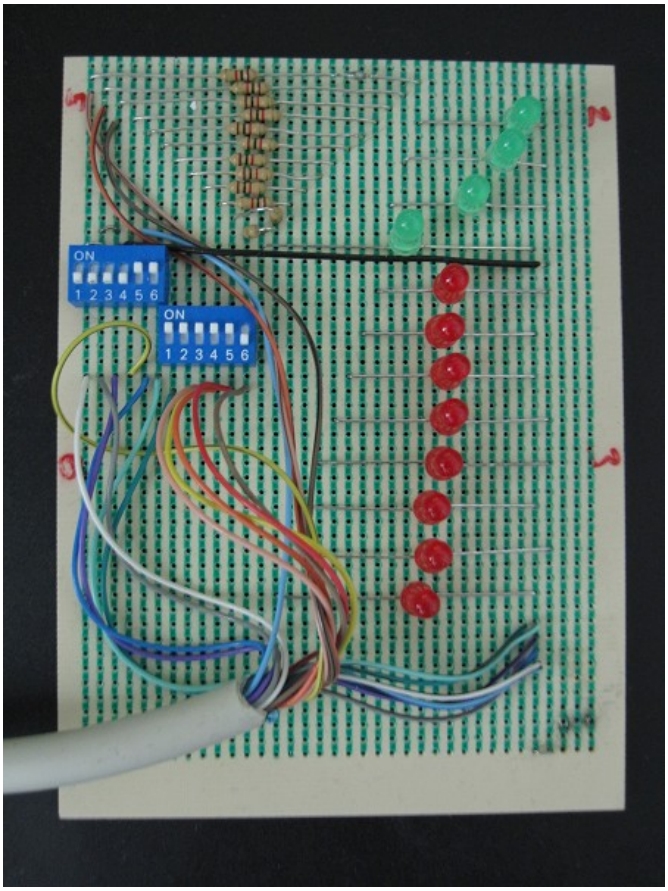
Ensimmäiseksi tuli rakentaa simulointikytkentä rinnakkaisporttiin. Joka helpottaisi huomattavasti ohjelmointia, kun välillä pystyy testaamaan uusia ratkaisuja. Käytettävän kytkennän sain ohjaavalta opettajaltani, joka suositteli sitä tähän tarkoitukseen. Kytkentäkaavio on nähtävissä kuvassa 23.



Kuva 23. Simulointilaitteen kytkentäkaavio.

Simulointilaite koostui rinnakkaisporttiin liitetyistä 12 kytkimestä, 12 LEDistä ja niiden 1 kΩ etuvastuksista. Kytkennässä LEDit 1 - 8 on kytketty rinnakkaisportin datarekisteriin ja niitä voidaan ohjata kytkimillä 2 - 9. LEDit 9 - 12 on kytketty rinnakkaisportin controlrekisteriin eli rinnakkaisportin lähtöihin, joista LEDiä 9 voidaan ohjata kytkimellä 1. Kytkimet 10 - 12 on kytketty rinnakkaisportin statusrekisteriin; näillä kytkimillä voidaan ohjata rinnakkaisportin sisääntuloja. Kytkentäkaaviosta toteutettu kytkentä on nähtävissä kuvassa 24.

Kun kytkentä oli saatu tehtyä, alkoi perehtyminen itse ohjelmointiin ja FreeRTOS -käyttöjärjestelmään. Suunnitteilla oli siis pieni hälytysjärjestelmä. Ohjaavan opettajan kanssa sovimme, että järjestelmä toimisi esimerkiksi niin, että kun rinnakkaisportin tulo kääntyy, tulee hälytys, josta ilmoitetaan LEDejä vilkuttamalla.



Kuva 24. Simulointilaite.

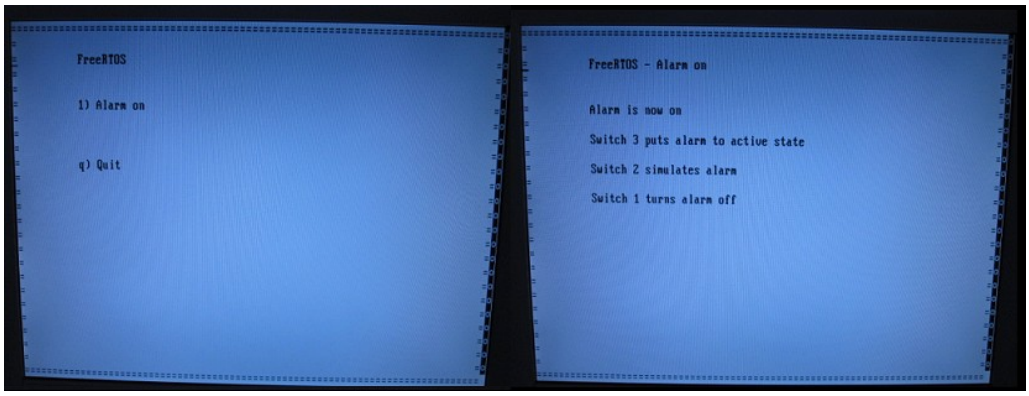
Tätä työstettävää FreeRTOS-ohjelmaa oli siis tarkoitus ajaa FreeDOSin päällä; varsinainen ohjelmointi tehtiin Open Watcom -ohjelmistolla. Aluksi suunniteltiin, että ohjelmalla olisi tekstipohjainen käyttöliittymä, jossa voitaisiin navigoida numeronäppäinten avulla, kytkeä hälytys ja kytkeä hälytys pois ohjelmallisesti ja kytkimillä simuloida hälytyksiä.

Aluksi keskityttiin rinnakkaisportin ohjauksen ohjelmointiin ja kokeiltiin, miten saisi tietyllä kytkimellä tehtyä hälytyksen simuloinnin ja toisella kytkimellä asettaa hälytyksen valmiustilaan. Tämän vaiheen ohjelmointi oli helppoa, mutta suuria ongelmia aiheutuivat ohjelman tehtävien ajastus ja priorisointi.

Ohjelma yritettiin saada toimimaan siten, että kortin kytkimillä olisi ainoastaan tehty hälytyksen simulointi. Ohjelmassa oli siis kolme eri tehtävää: näytön päivitys, näppäimistön luku ja rinnakkaisportin ohjaus. Mutta kun näppäimistöllä yritettiin saada hälytys kytkettyä ja kytkettyä pois sekä samalla käyttää rinnakkaisportin ohjausta, ohjelma kaatui ajastuksien vuoksi. Myös tehtävien prioriteetteja vaihdettiin, mutta tämä ei auttanut. Tämän ongelman ratkaisussa tehtiin kompromissi ja päätettiin käyttää näppäimistöä vain hälytyksen kytkemiseen ja ohjelman sulkemiseen. Kortin kytkimiä käytettiin hälytyksen aktivointiin, simu-

lointiin ja sulkemiseen. Ohjelmaan lisättiin myös yksi tehtävä, jonka oli määrä sammuttaa hälytys. Seuraavaksi on selostettu lopulliseksi jääneen ohjelman toiminta. Ohjelman koodit ovat luettavissa liitteessä 3.

Ohjelma koostuu siis neljästä tehtävästä: näytön päivitys, näppäimistön luku, hälytyksen sammutus ja rinnakkaisportin ohjaus. Ohjelman alussa luodaan kaikki muut tehtävät paitsi rinnakkaisportin ohjaus ja päivitetään näytölle aloitusnäkymä. Aloitusnäkymästä on mahdollista valita hälytyksen kytkentä tai ohjelman sulkeminen. Kun hälytys kytketään, luodaan rinnakkaisportin ohjaustehtävä ja näytölle avautuu hälytysnäkymä, jossa on ohjeistettu kytkimien käyttötarkoitukset.



Kuva 25. FreeRTOS-ohjelman aloitusnäkymä ja hälytysnäkymä.

Kun hälytys on kytketty aloitusnäkymästä, se kytketään aktiiviseksi kytkimellä kolme, jolloin joka toinen punainen LED syttyy palamaan. Kun hälytys on päällä ja aktiivinen, voidaan hälytys simuloida kytkimellä kaksi. Tällöin kaikki punaiset LEDit alkavat vilkkua. Hälytyksen saa kytkettyä pois, palauttamalla kytkin kaksi takaisin alas, jolloin hälytys pysyy päällä ja aktiivisena. Toinen tapa sammuttaa hälytys on nostaa kytkin yksi ylös, jolloin hälytys sammuu, rinnakkaisportin ohjaustehtävä tapetaan ja ohjelma palaa aloitusnäkymään, josta se voidaan kytkeä uudestaan.

5 YHTEENVETO

Tämän opinnäytetyön tavoite oli uusia Savonia-ammattikorkeakoulun tekniikan Kuopion yksikön elektroniikan laboratorion PC/104-teollisuustietokonelaitteisto, laatia uusia laboratorioharjoituksia sekä ohjelmoida pieni hälytysjärjestelmä FreeRTOS-reaaliaikakäyttöjärjestelmälle.

Laboratorion laitteiston uusiminen sujui ilman erityisiä ongelmia, ja uudeksi laitteistoksi tilatut Rhodeus-LC-prosessorikortit ovat suorituskyvyltään ja ominaisuuksiltaan opetuskäyttöön varsin hyvät. Ainoat ongelmat laitteiden kokoamisessa ja uusien korttien sovittamisessa vanhoihin koteloihin olivat puhtaasti mekaanisia, mutta niistä selvittiin kuitenkin helposti.

Laaditut laboratorioharjoitukset olivat pääasiassa teoreettisia, koska näissä laboratorioharjoituksissa keskitytään lähinnä vain itse PC/104-standardiin. Mukaan tuli myös joitakin ohjelmointiharjoituksia.

Henkilökohtaisena ohjelmointitavoitteenani oli tehdä pieni hälytysjärjestelmä käyttäen FreeRTOS-reaaliaikakäyttöjärjestelmää ja Rhodeus-LC-laitteen rinnakkaisporttia. Tähän ohjelmaan rakennettiin rinnakkaisporttiin pieni kytkentä, jolla järjestelmää testattiin. Hälytysjärjestelmä onnistui hyvin, koska se toimi vaaditulla tavalla.

Asetetut tavoitteet ja vaatimukset toteutuivat: laitteet saatiin koottua käyttökuntoon, laboratorioharjoituksia laadittiin ja FreeRTOS-hälytysjärjestelmä saatiin ohjelmoitua toimivaksi. Projektin lopputulokseen voidaan siis olla tyytyväisiä. Projektin aikana on saatu paljon uutta ja hyödyllistä tietoa sulautetuista järjestelmistä.

LÄHTEET

1. PC/104 Embedded Consortium [WWW-sivusto]. [viitattu 1.4.20210]. <www.pc104.org>

2. PC/104 Embedded Consortium [WWW-sivusto]. PC/104 Specifications. Figure 2. [viitattu 1.4.2010]. Saatavissa: <http://www.pc104.org/pc104_specs.php>

3. PC/104 Embedded Consortium [WWW-sivusto]. PC/104 Specifications. Figure 3. [viitattu 1.4.2010]. Saatavissa: <http://www.pc104.org/pc104_specs.php>

4. PC/104 Embedded Consortium [WWW-sivusto]. PC/104-Plus Specifications. Figure 2. [viitattu 1.4.2010]. Saatavissa: <http://www.pc104.org/pc104_plus_specs.php>

5. PC/104 Embedded Consortium. PC/104 Specification. Version 2.6. October 13. 2008. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

6. PC/104 Embedded Consortium. PC/104-Plus Specification. Version 2.3. October 13, 2008. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

7. PC/104 Embedded Consortium. PCI-104 Specification. Version 1.0. November 2003. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

8. PC/104 Embedded Consortium. PCI/104-Express™ & PCIe/104™ Specification. Version 1.0. March 24, 2008. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

9. PC/104 Embedded Consortium. EBX™ and EBX Express™ Specification. Version 3.0. October 23, 2008. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

10. PC/104 Embedded Consortium. EPIC™ and EPIC Express™ Specification. Version 3.0. June 23, 2008. [verkkodokumentti, PDF]. [viitattu 1.4.2010]. Saatavissa: <www.pc104.org>

11. Digital-Logig Ag. Microspace MSM586/SEN/SEV manual. Version 1.4. [verkko-

dokumentti, PDF]. [viitattu 3.5.2010]. Saatavissa: <www.dce.felk.cvut.cz/retobot/doc/pc104.pdf>

12. Microspace PC/104. MSM586SEN/SEV/SL datasheet. Digital-Logig Solution Guide 2002. [viitattu 3.5.2010].

13. Diamond Systems Corporation. Rhodeus-LC Low Power AMD Geode™ LX800 PC/104 CPU Module with CRT/LCD, LAN and CF manual. Version 1.00, 2008. [verkkodokumentti, PDF]. [viitattu 3.5.2010]. Saatavissa: <www.diamondsystems.com>

14. Diamond Systems Corporation. Rhodeus-LC Low Power AMD Geode™ LX800 PC/104 CPU Module with CRT/LCD, LAN and CF datasheet. [verkkodokumentti, PDF]. [viitattu 3.5.2010]. Saatavissa: <www.diamondsystems.com>

Laboratorioharjoitus PC/104

Tässä työssä on tarkoitus tutustua PC/104 standardiin ja laboratorion PC/104 prosessorikorttiin Rhodeus-LC.

Tarvittavat laitteet:

- Rhodeus-LC PC/104 -tietokone
- Näyttö, näppäimistö, hiiri
- Nollamodeemikaapeli
- Muistitikku

Tarvittavat materiaalit

- Rhodeus-LC datasheet (pdf)
- Rhodeus manual (pdf)
- PC/104 specification v2.6 (pdf)
- PC/104 Plus specification v2.3 (pdf)
- PCI-104 specification v1.0 (pdf)
- PCI-104 Express specification v1.0 (pdf)
- EBX specification v3.0 (pdf)
- EPIC specification v3.0 (pdf)
- Open Watcom programmers guide (pdf)

Hyodyllisiä www-sivuja

- www.pc104.org
- www.pc104.com
- www.computerhope.com/msdos.htm
- http://docs.huihoo.com/help-pc/

Tehtävät

1. Tutustu aluksi PC/104 standardeihin. Näitä standardeja on olemassa viisi: PC/104, PC/104-Plus, PCI-104, PCI/104-Express ja PCIe/104. Kerro hieman jokaisesta PC/104 standardista.

2. Seuraavaksi tutustu sulautettujen järjestelmien emolevystandardeihin. Näitä standardeja on olemassa neljä: EBX, EBX Express, EPIC ja EPIC Express. Kerro hieman kustakin standardista.

3. Sitten tarkastellaan laboratorion PC/104 laittetta. Listaa aluksi laitteen keskeisimmät ominaisuudet, apuna voit käyttää kortin datalehteä ja manuaalia. Kytke sitten laitteeseen näyttö, hiiri ja näppäimistö. Tutki vielä laitteen BIOS ja etsi mahdollisia lisäyksiä laitteen ominaisuuksiin.

4. Tee seuraavaksi pieni DOS-ohjelma "Hello World", joka siis tulostaa vain näytölle tekstin "Hello World". Rhodeus-LC laitteiden muistikorteille on asennettu FreeDOS käyttöjärjestelmä, jossa on asennettuna Open Watcom kääntäjä. Tarkoituksena on siis luoda tiedosto "hello.c" ja kääntää se DOSissa Open Watcomilla. Tähän löydät apuja Open Watcomin ohjeesta (programmers guide s.5). DOS:ssa tiedoston luominen ja muokkaaminen onnistuu DOSin editorilla, joka aukeaa kirjoittamalla "edit hello.c". Kun saat ohjelman käännettyä ilman virheitä, voit käynnistää sen kirjoittamalla DOSin komentoriville "hello.exe". Näytä ohjelman toiminta opettajalle.

5. Seuraavaksi ohjelmoidaan hieman BIOS keskeytyksiä. Tavoitteena on BIOS keskeytyksien avulla tulostaa näytölle kellonaika ja päivämäärä. Tässä työssä tarvitset Rhodeus-LC koneen lisäksi toisen koneen johon on asennettu Open Watcom sekä nollamodeemikaapelin.

   Luo aluksi uusi projekti, valitse kohdeympäristöksi DOS – 16-bit ja imagen tyypiksi .exe. Lisää sitten projektiin tiedostot functions.c ja timma.c. Käännä projekti, tämän pitäisi mennä kääntäjästä läpi ilman virheilmoituksia.

   Nyt siirrä muistitikulla kolme tekstitiedostoa (timma.txt, timma1.txt ja timma2.txt) Rhodeus koneeseen kansioon C:\fdos\watcom\binw\, tästä kansiosta löytyy myös serserv.exe jota tarvitset seuraavaksi. Jos tarvitset apuja DOS komentoihin, apuja löydät linkistä (www.computerhope.com/msdos.htm)

   Seuraavaksi liitä koneet yhteen nollamodeemikaapelilla COM1 - COM1 portteihin ja varmista, että Open Watcomin *remote dubug swiches* valikossa on trap fileksi määritelty "/tr=ser". Nyt kun em. tekstitiedostot on siirretty em. kansioon käynnistä samaisessa kansiossa sijaitseva serserv.exe ohjelma, ruudulla pitäisi lukea Open Watcomin ilmoituksia ja lopuksi "Press 'q' to exit" Valitse nyt Open Watcomista *remote debug*, nyt ruudulla pitäisi näkyä perhosia jonka jälkeen aukeaa debuggeri. Laita ohjelma käyntiin (run), jolloin etäkoneen ruudulle pitäisi tulla näkyviin ohjelman aloitusnäkymä. Ohjelma näyttää kellonajan ja päiväyksen nollina.

   Nyt ohjelmoi BIOS keskeytykset timma.c tiedostoon merkittyihin kohtiin. Tarvittavat tiedot tähän löydät täältä (http://docs.huihoo.com/help-pc/). Mallia voit katsoa myös tiedostosta functions.c.

```
//FUNCTIONS.C-----------------------------------------------------------
//Ohjelmassa käytetyt funktiot
/*=====================================================================
*     SetCursorPos     sets the blinking cursor any where on the screen
*     ClearScreen      clears the screen with any color
*     PrintAt          prints text with any color any where on the screen
*     GetCharacter     reads character without echo
*     LoadScreen       loads selected screen (txt-document)
=====================================================================*/
#include "functions.h"
#include "stdio.h"

#define SCREEN_COLOR (BLACK << 4 | WHITE)
#define TEXT_COLOR  (BLACK << 4 | WHITE)

#define BIOS_SET_CURSOR 02H
#define DOS_GET_CHAR    08H
#define BIOS_CALL_10    10H
#define DOS_CALL_21     21H

//---------------------------------------------------------------------
//  sets the cursor position
//---------------------------------------------------------------------
// x        = column number 0..79
// y        = row number    0..24
//---------------------------------------------------------------------
void SetCursorPos(unsigned char x, unsigned char y)
{
    // function uses bios interrupt 10H func 2 to set cursor
    _asm
    {
        mov ah, BIOS_SET_CURSOR
        xor bh, bh
        mov dh, y
        mov dl, x
```

```
        int BIOS_CALL_10
    }
}
//---------------------------------------------------------------------
//  Clears the screen
//---------------------------------------------------------------------
// color  = new background color of the screen
//---------------------------------------------------------------------
void ClearScreen(unsigned char color)
{
    char          szBuffer[ ROW_LEN + 1 ] = {0};
    unsigned char y;
    int           iBuffer;

    for ( iBuffer = 0; iBuffer <= ROW_LEN; iBuffer++ )
        szBuffer[ iBuffer ] = ' ';

    for ( y = 0; y < ROW_COUNT; y++)
        PrintAt( 0, y, color, szBuffer );

    SetCursorPos( 0, 0 );
}
//---------------------------------------------------------------------
//  Displays a string on the screen.
//---------------------------------------------------------------------
// x  =  column number
// y  =  line number
// ca = color attribute
// sz = pointer to the string
//
// - This function does not recognize format specs
//   as supplied by printf.
// - When the function reaches the end of the
//   screen, the screen will not scroll up.
//---------------------------------------------------------------------
```

```
void PrintAt(unsigned char y, unsigned char x, unsigned char ca, char *
sz)
{
    // force the pointer to point to the VGA video RAM
    P_VIDEO_RECORD pVGA = ( P_VIDEO_RECORD ) ( VGA_TEXT_ADDRESS |
                          ( ROW_LEN * x + y ) * sizeof( VIDEO_RECORD ) );

    while( *sz )
    {
        pVGA->ch = *( sz++ ); // Character in video RAM
        pVGA->ca = ca;        // Set character attribute
        pVGA++;
    }
}

//---------------------------------------------------------------------------
//   Loads selected screen from txt-document
//---------------------------------------------------------------------------
void LoadScreen(int number)
{
    FILE *fp;
    int r=0, s=0; //rivi ja sarake
    char text[2] = {0};

    switch(number)
    {
        case 0: fp = fopen("timma.txt","r"); break;
        case 1: fp = fopen("timma1.txt","r"); break;
        case 2: fp = fopen("timma2.txt","r"); break;
    }

    if (fp)
      {
         while(!feof(fp))
         {
```

```
            text[0] = fgetc(fp);
            PrintAt(s,r, TEXT_COLOR, text);
            s++;
            if(s >= 80)
            {
             r++;
             s=0;
            }
        }
      }
}

//END OF FUNCTIONS.C-------------------------------------------------------
//MAIN.C-------------------------------------------------------------------
//BIOS keskeytys ohjelma ajan ja päivämäärän tulostamiseen

#include <stdio.h>
#include "functions.h"

#define SCREEN_COLOR (BLACK << 4 | WHITE)
#define TEXT_COLOR  (BLACK << 4 | WHITE)

#define BIOS_CALL_1A    1AH
#define READ_RTC        02H
#define READ_RTC_DATE   04H

void main(void)
{
    char hour=0, minute=0;
    char day=0, month=0, century=0, year=0;
    int screen_number = 0;

    ClearScreen( SCREEN_COLOR );

    LoadScreen(0);
```

```
    SetCursorPos(25,13);

    scanf("%d",&screen_number);

//-------------------------------------------------------------------------
//  Tässä luetaan kellonaika bioskeskeytyksen avulla
//  ja tulostetaan se näytölle
//-------------------------------------------------------------------------
    if(screen_number == 1)
    {
        _asm
        {
            mov ah, READ_RTC
            int BIOS_CALL_1A
            mov hour, ch
            mov minute, cl
        }

        LoadScreen(1);
        SetCursorPos(28,3);

        if(hour < 0x10 && minute < 0x10)
            printf("0%X : 0%X",hour, minute);
        if(hour > 0x10 && minute > 0x10)
            printf("%X : %X",hour, minute);
        if(minute < 0x10 && hour > 0x0f)
            printf("%X : 0%X",hour, minute);
        if(hour < 0x10 && minute > 0x0f)
            printf("0%X : %X",hour, minute);

        PrintAt(40,20, TEXT_COLOR,"TIETOKONETEKNIIKKA PC/104");
    }
```

```
//-------------------------------------------------------------------------
//  Tässä luetaan päivämäärä bioskeskeytyksen avulla
//  ja tulostetaan se näytölle
//-------------------------------------------------------------------------
    if(screen_number == 2)
    {
        _asm
        {
            mov ah, READ_RTC_DATE
            int BIOS_CALL_1A
            mov century, ch
            mov year, cl
            mov month, dh
            mov day, dl
        }

        LoadScreen(2);
        SetCursorPos(28,3);

        printf("%X.%X.%X%X",day,month,century,year);

        PrintAt(40,20, TEXT_COLOR,"TIETOKONETEKNIIKKA PC/104");
    }
//-------------------------------------------------------------------------
}
//END OF MAIN.C-----------------------------------------------------------
```

```
//DEVICE.C-----------------------------------------------------------------
//Ohjelmassa käytetyt funktiot
/*=========================================================================
*     SetCursorPos     sets the blinking cursor any where on the screen
*     ClearScreen      clears the screen with any color
*     PrintAt          prints text with any color any where on the screen
*     GetCharacter     reads character without echo
*
*  To use these functions, include the file "device.h" to your source
*  code and put the file device.obj to your project list.
=========================================================================*/
#include "device.h"

#define BIOS_SET_CURSOR 0x02
#define DOS_GET_CHAR    0x08
#define BIOS_CALL_10    0x10
#define DOS_CALL_21     0x21
#define LPT_DATA  0x378
#define LPT_STATUS      0x379

int  xCur = 0, yCur = 0; // cursor position

//-------------------------------------------------------------------------
//  sets the cursor position
//-------------------------------------------------------------------------
// x        = column number 0..79
// y        = row number    0..24
//-------------------------------------------------------------------------
void SetCursorPos(unsigned char x, unsigned char y)
{
    P_VIDEO_RECORD pVGA =
        ( P_VIDEO_RECORD ) ( VGA_TEXT_ADDRESS
                           |
                           ( ROW_LEN * x + y ) * sizeof( VIDEO_RECORD )
);
```

```
        pVGA->ca  |= BLINK_ON;        // Set character attribute
        xCur = x;
        yCur = y;
}

//---------------------------------------------------------------------------
//  reads the character from keyboard buffer.
//---------------------------------------------------------------------------
// returns  = ASCII character code
// - function uses dos interrupt 21H func 8H to read the character
//  This function does not work, if dos interrupts are allowed
//---------------------------------------------------------------------------
char GetCharacter()
{
    char chTemp;

    _asm
    {
        ; character available?
        in  al, 64h
        and al,1
        jz  empty
        in  al, 60h  ; read character
        mov chTemp,al
        jmp done
  empty:mov chTemp,0
   done:nop
      //  mov ah, DOS_GET_CHAR
      //  int DOS_CALL_21
      //  mov chTemp, al
    }
    return chTemp;
}
```

```
//---------------------------------------------------------------------
//  Clears the screen
//---------------------------------------------------------------------
// color  = new background color of the screen
//---------------------------------------------------------------------
void ClearScreen(unsigned char color)
{
    char          szBuffer[ ROW_LEN + 1 ] = {0};
    unsigned char y;
    int           iBuffer;

    for ( iBuffer = 0; iBuffer <= ROW_LEN; iBuffer++ )
        szBuffer[ iBuffer ] = ' ';

    for ( y = 0; y < ROW_COUNT; y++)
        PutSAt( 0, y, color, szBuffer );

    xCur = 0;
    yCur = 0;
}

//---------------------------------------------------------------------
//  Displays a string on the screen.
//---------------------------------------------------------------------
// x  =  column number
// y  =  line number
// ca = color attribute
// sz = pointer to the string
//
// - This function does not recognize format specs
//   as supplied by printf.
// - When the function reaches the end of the
//   screen, the screen will not scroll up.
//---------------------------------------------------------------------
```

```
void PutSAt(unsigned char y, unsigned char x, unsigned char ca, char *
sz)
{
    // force the pointer to point to the VGA video RAM
    P_VIDEO_RECORD pVGA =
        ( P_VIDEO_RECORD ) ( VGA_TEXT_ADDRESS
                           |
                           ( ROW_LEN * x + y ) * sizeof( VIDEO_RECORD )
);
    while( *sz )
    {
        pVGA->ch = *( sz++ ); // Character in video RAM
        pVGA->ca = ca;        // Set character attribute
        pVGA++;
        xCur++;
        if (xCur > 79)
        {
            xCur=0;
            yCur++;
        }
    }
}
//---------------------------------------------------------------------------
//  Displays a string on the screen.
//---------------------------------------------------------------------------
// x  =  column number
// y  =  line number
// ca = color attribute
// ascii = ascii character
//
// - This function does not recognize format specs
//   as supplied by printf.
// - When the function reaches the end of the
//   screen, the screen will not scroll up.
//---------------------------------------------------------------------------
```

```
void PutChAt(unsigned char y, unsigned char x, unsigned char ca, char
ascii)


{
    // force the pointer to point to the VGA video RAM
    P_VIDEO_RECORD pVGA =
        ( P_VIDEO_RECORD ) ( VGA_TEXT_ADDRESS
                           |
                           ( ROW_LEN * x + y ) * sizeof( VIDEO_RECORD )
);


        pVGA->ch = ascii;  // Character in video RAM
        pVGA->ca = ca;         // Set character attribute
        xCur++;
        if (xCur > 79)
        {
            xCur=0;
            yCur++;
        }
}


//END OF DEVICE.C-------------------------------------------------------
```

```
//MAIN.C-------------------------------------------------------------------
//FreeRTOS Hälytysjärjestelmän koodi

#include "main.h"

#define SCREEN_COLOR (WHITE << 4 | BLACK)
#define DEFAULT_SCREEN_COLOR (BLACK << 4 | WHITE)
#define INCLUDE_vTaskDelete 1
#define IDM_UPDATE_DISPLAY 1

#define IDM_ALARM_ON 1
#define IDM_ALARM_OFF 0
#define LPT_START 1
#define LPT_DATA 0x378
#define LPT_STATUS 0x379
#define COUNT_OF_DISPLAYS 2

static void vKeyboardHandler( void *pvParameters );
static void vLptHandler(void *pvParameters);
static void vDisplayHandler( void *pvParameters );
static void vAlarmOff( void *pvParameters);

static xQueueHandle xDisplay;
static xQueueHandle xOff;
xTaskHandle xHandle;

static xSemaphoreHandle xUpdate;

typedef struct
        {
        char idMessage;
        char data;
        }DISPLAY_MESSAGE;
```

```
typedef struct
       {
        char data;
        }OFF_MESSAGE;

char   display[COUNT_OF_DISPLAYS][2050] = {0} ;

void LoadDisplays()
{
    FILE *fp;
    int i =0;
    char *pDisplay;
    char fileName[] = "disp0.txt";
    for(i=0; i < COUNT_OF_DISPLAYS; i++)
    {
        fileName[4] ='0'+i;
        fp = fopen(fileName,"r");
        if (fp)
        {
            pDisplay = &display[i][0];
            while(!feof(fp))
            {
                *pDisplay++ = fgetc(fp);
            }
            fclose(fp);
        }
    }
}

int main( void )
{
      LoadDisplays();

      vSemaphoreCreateBinary( xUpdate);
```

```
	xDisplay = xQueueCreate( 2, sizeof(DISPLAY_MESSAGE));

	xOff = xQueueCreate( 1, sizeof(DISPLAY_MESSAGE));


	xTaskCreate( vDisplayHandler, "Display", configMINIMAL_STACK_SIZE,
	NULL, (tskIDLE_PRIORITY + 3), NULL );

	xTaskCreate( vKeyboardHandler, "Keyboard",
	configMINIMAL_STACK_SIZE,     NULL, (tskIDLE_PRIORITY + 3), NULL );

	xTaskCreate( vAlarmOff, "AlarmOff", configMINIMAL_STACK_SIZE, NULL,
	(tskIDLE_PRIORITY + 2), NULL );


	vTaskStartScheduler();

	ClearScreen(DEFAULT_SCREEN_COLOR);

	return 0;

}

static void vKeyboardHandler( void *pvParameters )

{

	static char ch = 0;

	static OFF_MESSAGE off;

	static DISPLAY_MESSAGE message;


	( void ) pvParameters;


	for( ;; )

	{

		xSemaphoreTake( xUpdate, portMAX_DELAY );

		ch = getch();


		switch( ch )

		{

			case '1':   message.data = ch - '0';

			message.idMessage = IDM_UPDATE_DISPLAY;

			xQueueSend( xDisplay, (void*)&message,0);

			xTaskCreate( vLptHandler, "LPT",
			configMINIMAL_STACK_SIZE, NULL, (tskIDLE_PRIORITY + 2),
			&xHandle);
```

```
                break;

                case '2':   xQueueSend( xOff, (void*)&off,0);
                break;

                case 'q':  vTaskEndScheduler();

                default:;
            }
        }
}

static void vLptHandler(void *pvParameters)
{
      int a;
      static OFF_MESSAGE off;

      for( ;; )
      {
            a = inp(LPT_STATUS);


      if (((a^0xbf)&0x40)==0)
            {
                  outp(LPT_DATA, 0xaa);

                  if(((a^0x20)&0x20)==0)
                  {
                        if(((a^0x80)&0xc0)==0)
                        {
                              outp(LPT_DATA,0x00);
                              delay(100);
                              outp(LPT_DATA, 0xff);
                              delay(100);
                        }
```

```
                }
                else
                {
                      xQueueSend( xOff, (void*)&off,0);
                }
          }
          else
          {
           outp(LPT_DATA,0x00);
          }
    }
}

static void vAlarmOff( void *pvParameters )
{
    static OFF_MESSAGE off;

    for( ;; )
    {
          while( pdFALSE == xQueueReceive(xOff,&off, portMAX_DELAY));

          ClearScreen(SCREEN_COLOR);
          PutSAt(0,0,SCREEN_COLOR, display[0]);
          outp(LPT_DATA, 0x01);
          vTaskDelete( xHandle );
          xSemaphoreGive( xUpdate );
    }
}

static void vDisplayHandler( void *pvParameters )
{
    volatile  char *pChDisplay =0;
    int x = 0, y =  0;
     static DISPLAY_MESSAGE message;
```

```
( void ) pvParameters;


ClearScreen(SCREEN_COLOR);
PutSAt(0,0,SCREEN_COLOR, display[0]);


for( ;; )
  {
      while( pdFALSE == xQueueReceive(xDisplay,&message,
      portMAX_DELAY));


            switch( message.idMessage)
            {
            case IDM_UPDATE_DISPLAY:
                  if( message.data )
                  {
                        pChDisplay = display[message.data];
                        SetCursorPos(0,0);
                  }
                  else continue;


                  while(*pChDisplay != 0)
                  {
                        if( *pChDisplay !='\n')
                        {
                        PutChAt(x,y,SCREEN_COLOR,*pChDisplay);
                        x++;
                        }
                        else
                        {
                              x=0;
                              y++;
                        }
                        if (x > 79)
                              {
                                    x=0;
```

```
                                        y++;
                                        if(y>24)
                                        y=0;
                                    }
                                pChDisplay++;
                            }
                        default:;
                        }
                }
}

//END OF MAIN.C-------------------------------------------------------------
```