Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Marko Mustonen

Pianosimulaattorin suunnittelu ja toteutus

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatio

Insinöörityö

1.3.2019

Tekijä Otsikko Sivumäärä Aika	Marko Mustonen Pianosimulaattorin suunnittelu ja toteutus 24 sivua + 3 liitettä 1.3.2019
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Sähkö- ja automaatiotekniikka
Ammatillinen pääaine	Automaatiotekniikka
Ohjaajat	suunnittelija Kari Aro lehtori Markku Inkinen

Insinöörityö tarkoitus oli suunnitella ja toteuttaa pianosimulaattorin. Pianosimulaattorin tarkoitus on soittaa sävelmä Minuet in G. Yksinkertainen soitettava kappale helpottaa simulaattorin suunnittelua ja toteutusta.

Pianosimulaattorilla tavoite on saada ledit ohjaamaan pianon painikeiden painalluksia oikeassa järjestystä ja oikean pituisilla painalluksilla. Pianosimulaattorin ledien kytkennän perusteella suunniteltiin insinöörityö. Ledit olivat ohjauksen kannalta hyvin tärkeässä roolissa. Ledikytkennän perusteella pystyttiin paremmin suunnittelemaan ohjauksen tarpeet. Ledikytkentä toteutettiin yksinkertaisimmalla menetelmällä.

Kytkentä tehtiin ledin ominaisuuksia käyttäen. Ledin toiminnalle ominaista on päästää virtaa läpi vain yhteen suuntaan. Virta kulkee ainoastaan positiivisesta navasta negatiiviseen napaan. Ledit sarjoitettiin kahdeksan sarjaan. Sarjoja valmistettiin kahdeksan peräkkäin. Näytönohjain pystyi vain ohjamaan tätä määrää maksimissaan. Pianosimulaattorissa käytettiin kahta näytönohjainta.

Pianosimulaattori toiminta onnistui täysin suunnitellusti. Pianosimulaattori sytyttää ledejä ohjelman mukaisesti mitä ohjelmaan on ohjelmoitu. Ledien täydellisellä toiminnalla saadaan ohjattua lopullista pianosimulaattoria täysin.

Avainsanat	automaatio, ohjelmointi, leditekniikka

Author Title Number of Pages Date	Marko Mustonen Design and implementation of a piano simulator 24 pages + 3 appendices 1 March 2019
Degree	Bachelor of Engineering
Degree Programmer	Electrical and automation engineering
Professional Major	Automation technology
Instructors	Kari Aro, Designer Markku Inkinen, Senior Lecturer

The purpose of this Bachelor's thesis was to design and build a simulator that is capable of playing the composition "Minuet in G major" on a piano. This fairly easy-to-play composition was chosen because of the engineering work itself was rather challenging.

The goal of the piano simulator was to control the order and length of piano keystrokes with LED lights. The engineering work was designed around LED light wiring. LED lights were critical to the operation of the simulator. Wiring was implemented in the simplest way possible and programming was done according to this.

Wiring was done based on LED characteristics: LEDs allow electrical current to pass only in one direction from the positive side (anode) to the negative side (cathode). Eight LEDs were connected in a series and a total of eight series were used. This was the largest number of LEDs that the simulator's GPU could handle. Two GPUs were used in the simulator.

The piano simulator works as designed: it lights LEDs according to what has been programmed in the simulator, and it controls the piano completely.

Keywords	Automation Technology, Microcontroller programming, LED technology

Sisällys

Liitteet

Liite 1. Sähköpiirustus

Liite 2. Arduino-ohjelmointiosuus pianosimulaattorissa

Liite 3. Led-kytkentä-Excel

Lyhenteet

Arduino	Avoimeen laitteistoon perustuva mikro-ohjain-/elektroniikka-alusta ja ohjelmointiympäristö.
I/O	Input/Output. Tiedon siirtämistä tietokonelaitteiston ja komponenttien välillä.
ORM	Object-relational mapping. Oliomallin mukaisen esityksen kuvaus relaatiomallin mukaiseksi esitykseksi.
PIC	Mikro-ohjaus eli mikrokontrolleri (MCU) on mikropiiri eli IC-piiri, jossa on mikroprosessori ja joitain muisti- ja liityntälohkoja.
TKHJ	Tietokannan hallintajärjestelmä. Ohjelmisto, jonka avulla hallinnoidaan tietokantoja.

1 Johdanto

Insinöörityö tarkoitus on suunnitella ja toteuttaa pianosimulaattorin. Pianosimulaattorin tarkoitus on soittaa kappaleen Minuet in G, jonka sävelsi Johann Sebastian Bach. Sävelmä valittiin työhön sen helpon soitettavuuden takia.

Pianosimulaattoreita ja syntetisaattoreita on tehty aikaisemminkin, mutta tässä insinöörityössä toteutetaan pianosimulaattori mahdollisimman yksinkertaisesti. Pianosimulaattorin tarkoituksena on musiikin ja pianonäppäimistön käytön opetus. Tällöin opitaan nopeasti soittamaan musiikkia pianolla, jonka avulla voidaan myös tallentaa erilaisia sointuja sekä kappaleita.

Tavoite pianosimulaattorilla on saada ledit ohjaamaan pianon painikkeiden painalluksia oikeassa järjestyksessä ja oikean pituisilla painalluksilla.

Ledit ovat ohjauksen kannalta tärkeässä roolissa. Ledikytkennän perusteella pystytään paremmin suunnittelemaan ohjauksen tarpeet. Ledikytkentä toteutetaan mahdollisimman yksinkertaisella menetelmällä. Kytkentä tehdään ledin ominaisuuksia käyttäen. Ledin toiminnalle ominaista on päästää virtaa läpi vain toiseen suuntaan. Virta kulkee ainoastaan positiivisesta navasta negatiiviseen napaan. Ledit sarjoitettiin kahdeksan sarjaan. Sarjoja valmistettiin kahdeksan peräkkäin. Näytönohjain pystyi vain ohjamaan tätä määrää maksimissaan. Pianosimulaattorissa käytettiin kahta näytönohjainta.

Insinöörityössä toteutetaan pianosimulaattori Arduinolla, jonka avulla MAX7219 -näytönohjain ohjataan. Pianosimulaattori valmistettiin ensiksi ledinauha, joka kytkentä perustuu leditekniikkaan. Näytönohjaimeen ja Arduinon tutustuminen tehtiin kaikkien kytkentöjen jälkeen, jolloin ohjelmalla päästiin testaamaan erityylisiä ohjelmia. Näytönohjaimen toimintaa testattiin samaan aikaan Arduinon ohjelmoinnin kanssa. Insinöörityö valmistui testauksen yhteydessä, jolloin korjattiin puutteelliset kytkennät paremmiksi ja varmemmiksi. Pianosimulaattori valmistui, kun sillä saatiin haluttu kappaleesta osa soitettua oikein.

2 Pianosimulaattori

Työssä suunniteltiin pianosimulaattori, joka on erilainen kuin valmiit tai muiden valmistamat pianosimulaattorit. Pianosimulaattoriin kerättiin muiden pianosimulaattorin ratkaisuista parhaimmat ominaisuudet, jotka ovat ledien värit pianon näppäimiin verrattuna ja ohjelmointitapa. Tarkoitus on toteuttaa pidemmälle itsestään soittava pianosimulaattoriversio. Pianosimulaattori on perusperiaatteeltaan vastaava kuin opettavat vastaavat valmiit laitteistot. Pianosimulaattori opettaa pianon soittamista koskettimien yläpuolelle asennetuilla ledeillä.

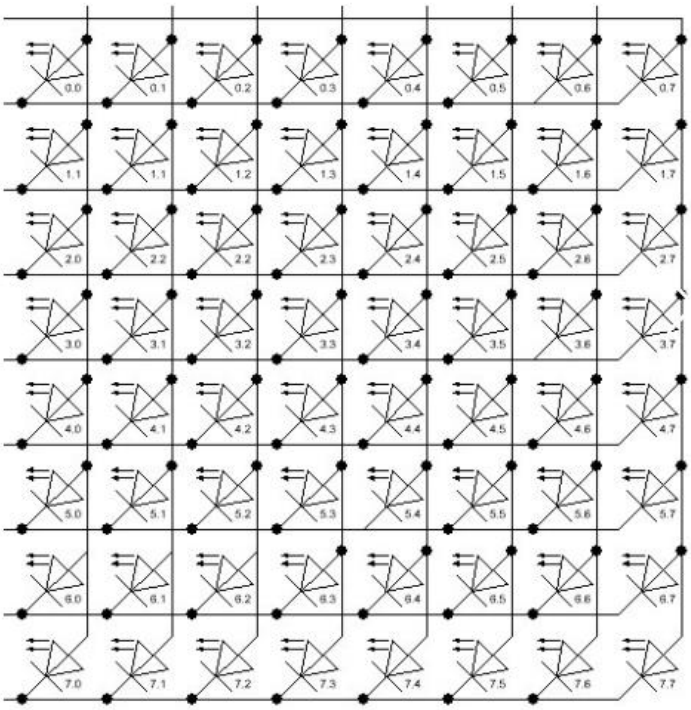
Alkuvaiheessa tehtiin helppoja sähkökytkentäpiirustuksia käsin paperille, että saisi paremman kuvan tulevasta pianosimulaattorista. Tämän ansiosta ensimmäisen version napaisuuden käännöskytkentä saatiin piirrettyä paperille ja toteutettua. Sen jälkeen huomattiin aiheen olevan liian vaikea toteuttaa tässä insinöörityössä.

Pianosimulaattoria yksinkertaistettiin leditekniikaltaan, mikä toteutettiin Arduino -nimisellä järjestelmällä. Insinöörityötä tutkittiin uudesta näkökulmasta paremmin, josta löytyi huomattavan paljon tietoa internetistä. Leditekniikan muutettiin yksinkertaisempaan kytkentään, joka on nähtävissä liitteenä 1 ja kuva 1. Arduinon avulla saatiin mahdollisuuksia ymmärtää paremmin ohjaustapojen käyttöä. Arduinon ohjelmoinnin ymmärtäminen parani tällöin paljon. Valmiita järjestelmiä on saatavilla esimerkiksi internetissä sekä Arduinon omista kirjastoista.

Insinöörityössä opiskellaan uusi käyttöympäristö Arduino. Arduinosta saa hyvin tietoa jopa suomeksi ja englanniksikin ohjelmoinnista, mikä helpottaa insinöörityön toteuttamista. Arduinon verkkosivuilta saa hyvän aloituksen ohjeistuksen pianosimulaattoriani varten.

Pianosimulaattori soittaa kappaleen Minuet in G. Kappale on valmiina Synthesia-kirjastossa. Synthesia-ohjelmisto on ohjelma, joka ohjaa pianosimulaattoria seuraavassa pianosimulaattoriversiossa. Pianosimulaattoria ei vielä ohjelmoitu toimimaan Synthesia-ohjelmiston kanssa. Pianosimulaattori toistaa kappaletta vain Arduinolle tehdyn ohjelman mukaan.

Insinöörityössä ei tarvinnut kuin toteuttaa pianosimulaattori alustavien suunnitelmien mukaisesti. Kuvassa 1 on lopullisesta ledikytkennästä piirustus.



Kuva 1. Pianosimulaattorin ledikytkentä.

3 Työn kulku

Työssä tutustutaan ensin eri komponentteihin ja niiden käyttömahdollisuuksiin. Tämän jälkeen tutustaan leditekniikoihin. Tämän jälkeen tutustutaan matriisimenetelmiin, jotka ovat yksi leditekniikan perusideoista, jota oli käytetty 7-segmenttinäytöissä (kuva 2). Laitteiston hankinta ja rahoitus on omakustanteinen. Asennus ja testaus suoritetaan laitteiston valmistuttua, sillä laitteiston sähköjärjestelmä ei ole sähköturvallisuuden piirissä, vaikka kyse on piensähköstä. Sähköjärjestelmää pidetään ohjaussähkön puolella alhaisien jännitteiden ansiosta (5 V). Järjestelmää valmisteltiin heti, kun komponentit saatiin tavarantoimittajilta.

Järjestelmä todetaan toimivaksi vasta, kun se on testattu. Testauksen hyväksyy työn valvoja ja sen valmistaja. Järjestelmää laajennetaan työn valmistuttua. Työhön lisätään

insinöörityön valmistumisen jälkeen solenoideja, jotka pystyvät soittamaan akustista pianolla itsenäisesti määriteltyjä kappaleita tietokoneen kautta.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Kuva 2. Järjestelmän yleinen matriisikaavio.

Järjestelmä etenee normaalisti samalla kun se valmistuu, aikataulua ei ole muuten määritelty kuin insinöörityön mukaisesti. Järjestelmä testataan ja parannellaan testauksen aikana paljon, sillä järjestelmän valmistuksessa tulee vaikeuksia saada järjestelmä toimimaan. Piirilevykytkennän sekä ledikytkennän on oltava valmiit järjestelmän käyttöönottoon mennessä.

Kuvassa 2 kaaviossa esitellään matriisimenetelmää. Kuvassa on vai yksi ledijärjestelmä. Insinöörityössä on kaksi sarjaan kytkettyä ledikytkentää, jotka ovat liitetty kahteen näytön ohjaimeen. Sarjakytkennällä saadaan suurempi binäärilukusarja aikaiseksi.

Ledikytkennässä virrankulkumahdollisuuksia on hyvin paljon kytkennän ansiosta. Väärin virrankulkua mahdollisuuksia pyritään hallitsemaan näytönohjaimella MAX7219. MAX7219 -piirin sisällä oleva kello tulee ottaa käyttöön, jolloin ledit eivät ole jatkuvasti ON- asennossa. Ledit vilkkuvat niin nopeasti, että ihmissilmä ei havaitse niitä. Voidaan helposti ohjata ledinauhaa halutulla tavalla.

Ledinauhat on kytketty ikään kuin kaksi neliömäistä matriisia. Ensimmäinen sarja on 8 x 8 ja toinen sarja on 8 x 8, joka lisää 128 lediä Arduinoon ohjattavaksi kahden matriisin 16 x 16-matriisi. Rivillä käytetään ainoastaan vain neljää lediriviä, jolloin ledien määrä laskee 88 kappaleeseen.

Kaikki tämä tehdään binäärilukuja apuna käyttäen Arduinon kehittämiä apukoodeja. Tämän ansiosta kappaleen ohjelmointi Arduinoon on helppoa. Arduino ohjelmointikielellä playKey kutsutaan haluttua lediä ledinauhasta. Ohjelma käyttää valittua lediä ledinauhassa näytönohjaimen jälkeen. Komento tulee Arduinoon ohjelmasta avaimen numeron

(playKey) ja kääntää sen niin, että näytönohjain tunnistaa tämän tekemällä seuraavaan playKey käännöksen rivin- ja sarakemääräksi, joka käännetään rivinumeroon ja binaariseen numeroon, joka edustaa sitä lediä siinä paikassa. Kutsutaan esimerkiksi playKey (13) ledinauhasta, tällöin ohjelma kääntää funktiot rivin 2 sarakkeen 5, sitten että 2 ^ 5 on 32. Tämä arvon Arduino muuttaa näytönohjaimelle. Lukemalla vasemmalta, 32 on 00000100, joka edustaa toisen rivin tilaa. Sama konsepti pätee, kun käsittään useiden ledien käyttämistä samassa binääriluvussa.

4 Pianosimulaattori toteutettiin Arduino Unolla

Arduinon on avoimeen laitteistoon ja ohjelmistoon perustuva mikrokontrolleri-elektroniikka-alusta ja ohjelmointiympäristö. Arduinon avulla toteutetussa laitteessa on kaksi kokonaisuutta: Arduinoon liitetty ulkoinen kytkentä ja Arduinoon syötetty ohjelma.

Arduino sisääntulonapoihin voi liittää erilaisia antureita, säätimiä tai kytkimiä, jotka ohjaavat ulostulonapoihin kytkettyjä ledejä, releitä, servoja tai moottoreita. Laitteen toiminnot määritetään laitteeseen syötetyllä ohjelmalla.

Arduinoja on avoin konsepti ja erilaisia Arduino-tuotteita myydään verkkokauppojen välityksellä halpaan hintaan. Euroopassa ”alkuperäiset” Arduino-tuotteet myydään Genuino-nimen alla. [1 s. 5.]

Arduino on eri malleja, jotka eroavat toisistaan mm. EEPROMin, keskusmuistin ja Flash-muistin sekä digitaalisten ja analogisten pinnien määrissä. Alkuperäisissä laitteissa ohjelmointi tehtiin sarjaportin kautta, nykyisin käytettävissä on usein USB tai Bluetooth.

Arduinoon on saatavilla myös lisälaitteita, jotka liitetään suoritinkortin päälle. Lisälaitteiden avulla Arduinossa voi käyttää mm. Internetiä, GPS, WLAN, sensorikortteja ja kosketusnäyttöjä. [5.]

Arduinossa on vain kaksi pääfunktiota:
setup() -funktio alustaa laitteen asetukset.
loop() -funktiota toistetaan virran sammuttamiseen asti.

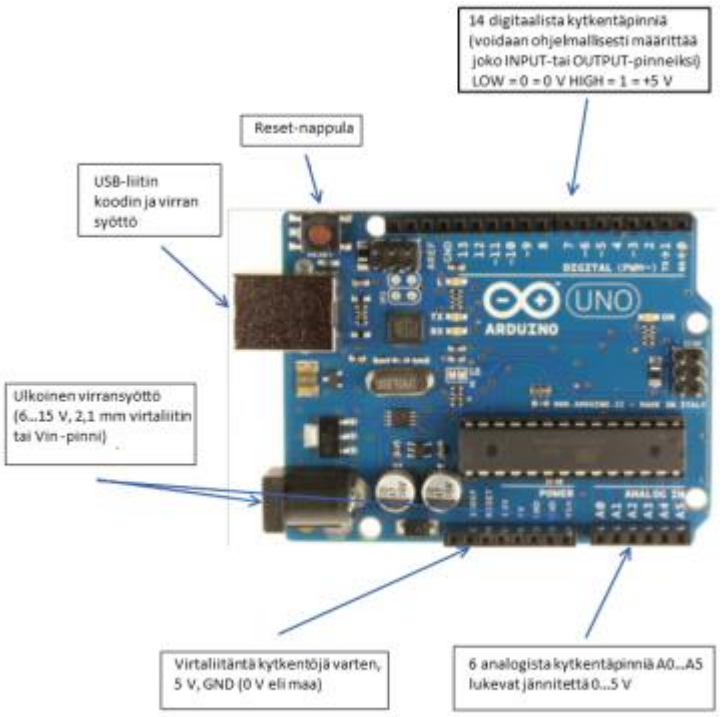
Yleisin ensimmäinen mikrokontrollerille siirrettävä ohjelma on Blink (suom. "Vilkku"), joka vilkuttaa yhtä lediä, koodiesimerkki 1.

```
#define LED_PIN 13
void setup () {
  pinMode (LED_PIN, OUTPUT);    // pinni 13 asetetaan tulostustilaan.
}
void loop () {
  digitalWrite (LED_PIN, HIGH); // kytkee LEDin päälle; HIGH on Arduinon vakio, joka antaa LEDille 5 volttia
  delay (1000);                 // odottaa sekunnin (1000 millisekuntia)
  digitalWrite (LED_PIN, LOW);  // sammuttaa LEDin
  delay (1000);                 // odottaa sekunnin
}

Kohteesta <https://fi.wikipedia.org/wiki/Arduino>
```

Koodiesimerkki 1. Arduino-esimerkki ohjelmakirjastosta. [5.]

Kuvassa 3 on nähtävillä Arduinon Unon osat.



Kuva 3 Arduinon osat. [3, s. 5.]

Arduinon ohjelmoimiseksi on ensin ladattava sen kehitysympäristö eli IDE seuraavasta osoitteesta: www.arduino.cc/en/Main/Software. Sieltä valitaan tietokoneen käyttöjärjestelmän mukainen versio. Käyttöjärjestelmät, jossa Arduino-ohjelmisto toimii, ovat Windows XP ja Windows 8 tai10, Mac OS X 10,7 Lion tai uudempi ja Linux 32 bittinen ja 64 bittinen ja Linux ARM. [2, s. 20.]

Arduino-ohjelmiston käyttö

Windows ja Linuxissa ladatut tiedosto puretaan hakemistoon, jonka nimi on muotoa Arduino[versio], esimerkiksi arduino-1.0. Kansio määritellään mieleiseen paikkaan, kuten työpöydälle tai Program Files -kansioon (Windowsissa). Kun halutaan käynnistää Arduinon IDE, tarvitsee vain avata Windows Arduino-kansio ja napsauttaa kuvaketta kahdesti. Tätä ennen on syytä tarkistaa ajuri tietokoneen USB-portilta ja tarvittaessa päivittää se. [2, s. 20.]

Kun laite on löytynyt, yhdistetään Arduino Uno tietokoneeseen. Tietokone ilmoittaa uudesta kytketystä laitteesta, jolloin Windows yrittää löytää ajurille oman päivityksensä.

Windows kysyy seuraavaksi, halutaanko etsiä ajuria Windowsin päivitysten kautta. Jos halutaan käyttää Windowsin päivitystä, valitaan ”Ei tällä kertaa/No not at this time, ja sen jälkeen Jatka/Next.

Seuraavaksi valitaan ”Asenna luettelosta tai määrätystä sijainnista/Install From a list or specific locotion” ja Jatka/Next.

Siirrytään Drivers-alihakemistoon Arduinon ohjelmiston tai lataushakemistossa, jossa Uno-ajuri sijaitsee, ja valitaan ajuriksi ArduinoUNO.inf (ei käytetä GTDI USB Drivers-alihakemistoa). Windows viimeistelee ajurin asetuksen tästä eteenpäin.

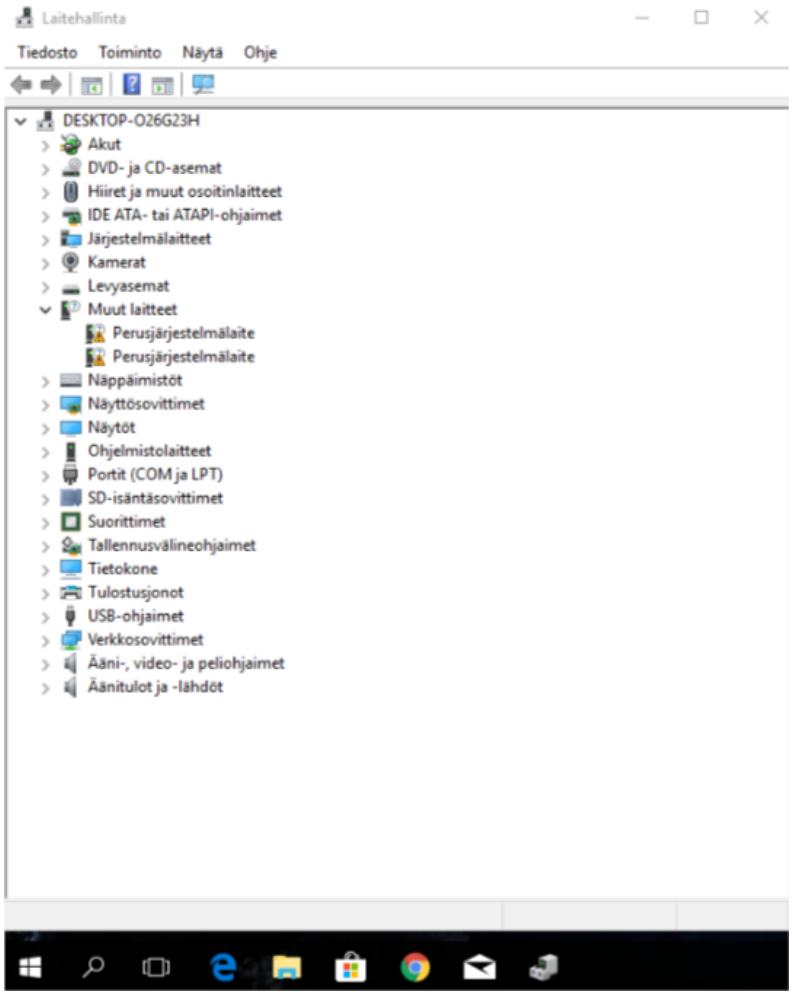
Kun ajuri on asennettu, voidaan käynnistää Arduino IDE ja aloittaa Arduinon käyttö. [2, s. 22.]

Porttien tunnistus; Windows

Vaihtoehtovalikko saadaan Windows-näppäimen kohdalla painamalla hiiren oikealla ja valitsemalla Laitehallinta/Device Manageria-vaihtoehdon, jolloin aukeaa suoraan tietokoneeseen liitetyt laitteet. Eri Windows-versioissa laitehallintaan mennään eri tavoin, mutta hakutoiminnon avulla löytää varmasti oikean paikan.

Etsitään Arduino Portit/Ports (COM & LPT) -kohdan alta. Arduino näkyy nimellä Arduino Uno, jonka perässä on suluissa porttia osoittava tarkennus, kuten esimerkiksi COM5 (kuva 5). Windows-laitehallinta näyttää kaikki saatavilla olevat portit.

Windows-laitehallinta näyttää kuvassa 5.



Kuva 5. [2, s. 24.]

Arduinon ohjelmointi

Arduino on avoimeen laitteistoon perustuva mikro-ohjain-/elektroniikka-alusta ja ohjelmointiympäristö. Laitteisto perustuu 8-bittiseen Atmel AVR -mikro-ohjaimeen, jonka pinneihin voi kytkeä erilaisia antureita, moottoreita, ledivaloja ja muita komponentteja. Laitteistoa ohjelmoidaan C++:aan perustuvalla Arduino-ohjelmointikielellä. ATMega328 on etenkin harrastelijapiireissä yleisesti käytetty mikrokontrolleri, jossa on 32 KB Flash-muistia, 2 KB SRAM -käyttömuistia ja 1 KB EEPROM -muistia datan pidempiaikaiseen säilytykseen, Flash-muisti esilataajan (bootloaderin) ja ohjelmakoodin käytössä. SRAM toimii ohjelman väliaikaisena käyttömuistina. EEPROM on ikään kuin kovalevy, jolle voi tallentaa dataa pitkäkestoisesti.

Insinöörityön ohjelmointi on määritelty tarkasti tarvittavat apuohjelmistot ohjelmoinnissa. Ohjelma toimii näiden avulla, jolloin ohjelmistoon lisätään halutut parametrimuuttujat. Parametrimuuttujat ovat ledien toimintaa ohjaavia toimintoja, joilla ohjataan ledien syttymistä ja sammumista ajallisesti. Parametrimuuttuja on HEX-koodi, jolla voidaan ohjata yhtä tai useampaa MAX7219 -näytönohjainta helposti. Ohjelmaan on määriteltävä mistä ohjelma käy lukemassa HEX-koodit ja minne tämä tieto lähetetään. Ohjelmistossa määritellään myös tulevat ja lähtevät signaalipaikat.

Arduino toimii 16 MHz:n taajuudella. I/O-pinnejä on 20 kappaletta, ja ne ovat vapaasti ohjelmoitavissa. Arduinossa on muitakin hyödyllisiä ominaisuuksia, kuten kolme ajastinta, 6-kanavainen PWM, kaksi AD-muunninta, USART-, SPI- ja 2-Wire -väylät, vahtikoira-ajastin ja keskeytykset. [5; 1.]

Kaikki Arduinolla ohjelmoimiseen tarvittava tulee olla Arduino-kehitysympäristön (Arduino-IDE) mukana. Se sisältää mm. AVR C-kääntäjän, joka kääntää C-koodin konekielelle ja poistaa turhat osat. Arduino-ohjelmointilevy hoitaa kommunikoinnin mikrokontrollerin ja tietokoneen välillä. Se siirtää koodin sarjaväylän kautta mikrokontrolleriin. Lisäksi on mm. koodieditori, kirjastonhallintatyökalu ja työkalu sarjaportin kuunteluun. [1, s. 6.]

Taulukko 1. C-ohjelmoinnin peruskielien lyhenteet. [1, s.10.]

+	summa	a == b	yhtä suuri
-	erotus	a != b	erisuuri
*	tulo	a < b	pienempi
/	osamäärä	a > b	suurempi
%	jakojäännös	a <= b	pienempi tai yhtä suuri
&	bittikohtainen ja	a >= b	suurempi tai yhtä suuri
|	bittikohtainen tai	a --	vähennä muuttujan arvoa yhdellä
^	bittikohtainen or	a ++	kasvata muuttujan arvoa yhdellä
=	asettaa muuttujan arvon		... ja monia muita

Ohjelmointikielen perus muuttujat esitetty yllä olevassa taulukossa 1. C-ohjelmointikielen muuttujia voidaan verrata matemaattisiin muuttujiin suoraan. Taulukossa verrataan muuttujan symbolia suoraan laskennallisiin symboleihin.

Taulukko 2. C-kielien Boolen logiikan Arduino-ohjelmointi kieleen. [1, s.11.]

a&&b	JA	If (true && false) digitalWrite (ledPin, HIGH)
a || b	TAI	If (true || false) digitalWrite (ledPin, HIGH)
! a	EI	If (!false) digitalWrite (ledPin, HIGH);

Digitaalisen ohjelmointiin tarvitaan lisäksi taulukon 2 muuttujia lisäksi, joka parantaa ohjelmointia ja mahdollistamaan nykyaikaisen ohjelmoinnin. Ohjelmointikielessä käytetään nykyisin JA-, TAI- ja EI-muuttujia ohjelmoinnissa.

Ohjelmointikielessä kannattaa hyödyntää, mikäli samankaltaisia muuttujia tarvitaan paljon, esimerkiksi ledipinnien alustamiseen tai vaikkapa ledianimaation vaiheiden tallentamiseen. Käytetään ensimmäisen ledin käyttöä myös kolmannelle ledille, eli ne samanaikaisesti, mikä on nähtävissä esimerkkikoodissa 2.

```
int led1 = 3, led2 = 5, led3 = 6;
Int ledit[3] = {3, 5, 6};
```

Esimerkkikoodi 2. Arduino-esimerkki ohjelmakirjastosta. [1, s. 16.]

Ohjelma kielessä jäsenet ovat alkioita, ja niihin pääsee käsiksi [n]- operaattorilla. Indeksointi alkaa nollasta. Tällöin muuttujia voidaan määritellä tietylle alkiolle, jonka voidaan aktivoida raja-arvojen sisäpuolelle tai ulkopuolelle, joka on nähtävissä alla olevassa esimerkkikoodissa 3. Tätä toimintaa käytetään myös insinöörityössä liitteessä 2 sivulla 4.

```
For (int i=0; i<3; i++)
{
pinMode(ledit[i], OUTPUT);
}
```

Esimerkkikoodi 3. Arduino-esimerkki ohjelmakirjastosta. [1, s. 16.]

Merkkijonot ovat char-taulukoita. Annetulla merkillä kerrotaan merkkijonon loppuvan, sillä muuttujaan ledin tallennetaan vain merkkijonon ensimmäisen alkion osoite, joka on nähtävissä esimerkkikoodissa 4.

```
const char* hedelma = ”omppo”;
// kyseinen merkkijono näyttää tältä
// {'o', 'm', 'p', 'p', 'o', '\0'}
'\0' -merkki kertoo, merkkijonon loppuvan, sillä muuttujaan ledin
tallennetaan vain merkkijonon ensimmäisen alkion muistiosoite.
Hyödyllisiä funktioita merkkijonon käsittelyyn ja muunnoksiin ovat
mm. strcpy, strcat, strcmp, itoa ja atoi.
```

Esimerkkikoodi 4. Arduino-esimerkki ohjelmakirjastosta. [1, s. 17.]

Arduino-koodia olisi hyvä heti jakamaan funktioihin, jotta koodin lukeminen helpottuisi ja eri koodina osia olisi helpompi käyttää uudelleen. Tämän insinöörityön käytössä on kaksi samanlaista mikrokontrolleria, joille tämä funktio on toteutettu. Funktio on nähtävissä esimerkkikoodissa 5.

```
void loop()
{
If (isHotDay(sensor1, sensor2)) {
 digitalWrite(hotLed, HIGH);
}
}
int isHotDay(int temp, int light) // return 1 if it is, 0 otherwise
{
if (temp >= 30 && light >= 50) return 1;
else return 0;
}
```

Esimerkkikoodi 5. Arduino-esimerkki ohjelmakirjastosta. [1, s. 18.]

Ylimääräisiä Arduino-esimerkki ohjelmakirjasta.

Asetetaan pinnin arvo joko LOW tai HIGH -tasoon (Arduino UNO, siis 0 V tai 5 V). Tämän ohjelman avulla voidaan todeta esimerkiksi ledin toimintakunto ja Arduinon kellotoiminta, joka on nähtävissä esimerkkikoodissa 6

```
/*
  Blink

*/

int ledPin = 13                       // LED yhdistettynä
                                      // digitaaliseen liittimeen 13

void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);       // asetetaan digitaalisen
                                      // liitinnastan output-tila
}
void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);    // sytyttää LED: in
  delay(1000);                        // odotta sekunnin
  digitalWrite(LED_BUILTIN, LOW);     // sammuttaa LED: in
  delay(1000);                        // odottaa sekunnin
}
```

Esimerkkikoodi 6. Arduino-esimerkki ohjelmakirjastosta. [1, s. 20.]

Arduino-kirjasto sisältää myös joukon muita hyödyllisiä funktioita, joista on apua myös muissa projekteissa. Tässä on Arduinoon lisä komentokäskyjä insinöörityön jatkoa varten. Näitä käskyjä voisi käyttää ohjelmoinnissa hyväksi, joka on nähtävissä esimerkkikoodissa 7.

```
millis()    // aika Arduinon käynnistyksestä millisekunneissa
map()       // muuttujan arvovälin mappaaminen toiselle
// esim. [0, 1023] → [0, 256]
min(), max(), abs(), sin(), cos()...
random()    // satunnaislukuja
Serial      // kirjasto sarjakommunikaatiolle
Servo       // kirjasto helpompaan Servomoottorin liikutteluun
```

Esimerkkikoodi 7. Arduino-esimerkki ohjelmakirjastosta. [1, s. 23.]

Lisäksi tarjolla on satoja ulkopuolisia kirjastoja toimintojen helpottamiseksi ja oheislaitteiden käyttöön.

5 Näytönohjain MAX7219/MAX7221

5.1 Pianosimulaattorin 8-numeroinen ledinäytön ajuri

MAX7219/MAX7221 on pienikokoinen näytönohjain, sarjaliitäntä / -lähtö yleisesti-katodin näytönohjaimet, jotka liittyvät mikroprosessorit 7-segmenttiseen numeeriseen LED näyttöihin enintään 8 numeroa, pylväsdiagrammi tai 64 yksilöä. BCD-koodi B dekooderi ollessa käytössä, niin skannattavaa on moninkertaiseksi vähemmän piiriin, segmentti ja numero ohjaimet ja 8x8 staattinen RAM muistiin, joka tallentaa jokaisen numeron omalle muistipaikalle. Segmentin asettamiseksi tarvitaan vain yksi ulkoinen vastus kaikille ledeille. MAX7221 on yhteensopiva SPI:n, QSPI:n ja MICROWIRE:n kanssa, ja se on rajoitettu segmenttiohjaimia EMI vähentämiseksi.

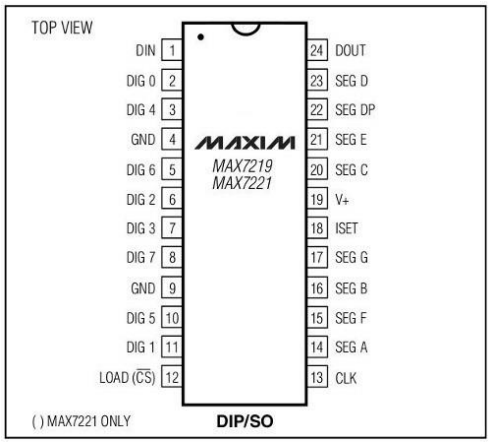
Käytetyn 4-johtiminen sarjaliitäntä yhdistyy kaikkiin yhteiset liitäntöihin näytönohjaimessa. Yksittäisiä numeroita voidaan käsitellä ja päivittää ilman koko näytön

uudelleen kirjoittamista. MAX7219/MAX7221 antaa käyttäjälle myös mahdollisuuden valita koodin B dekoodauksen tai ei-dekoodauksen, jota valittiin insinöörityössä B dekoodauksen. Tämä valittiin kahden näytönohjaimen ohjauksen takia. Laitteisiin sisältyy 150 µA:n virransäästötila, analoginen ja digitaalinen kirkkauden säätö, skannausraja rekisteri, jonka avulla käyttäjä voi näyttää numerot 1–8 ja testitila, joka pakottaa kaikki ledit päälle (kuva 6 ja 7 ja taulukko 3).

Taulukko 3. Pinnien kuvaus ja määritys. [4, s. 5.]

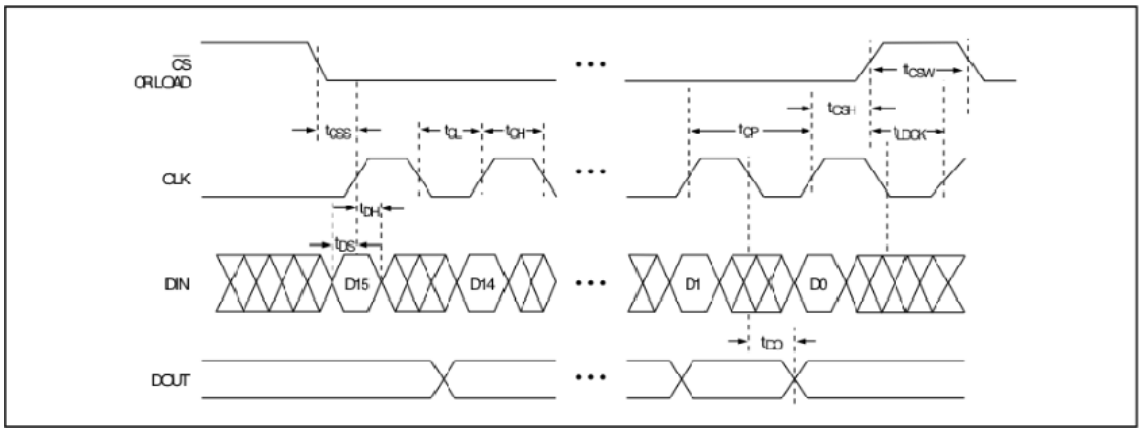
Pinni	Nimi	Toiminto
1	DIN	Sisään tuleva ohjaus. Ohjaus ladataan sisään 16-bittiseen siitorekisteriin CLK; n nousevalla reunalla.
2, 3, 5-8, 10, 11	DIG 0-DIG 7	Kahdeksan digitaali LED näyttön ohjaus, joka poistaa edellisen ohjauksen toiminnon. MAX7219 muuttaa numeron lähtöt V +: n sammus tilaan.
4, 9	GND	Maa (molemmat GND-nastat on kytkettävä)
12	Lataus (MAX7219)	Lataus tiedon tulo. Viimeiset 16 bittiä sarjatiedoista ovat kiinni latauksen nousevalla reunalla.
13	CLK	Sarjakellotulo 10MHz maksimikorkeudella. CLK: n nousevalla reunalla tieto siirretään sisäiseen muistiin.
14-17, 20-23	LED lähdöt	Normaalissa seitsemän segmenttiaalisessa ohjauksessa nämä asetataan lähdöt normaalisti toimintaan. Segmentti ohjain on pois päältä, kun tämä kytketään GND: n.
18	ISET	Liitä VDD: n vastuksen (RESET) kautta parantamiseksi virra arvon asettamimista (RESET-vastus ja ulkoisten ohjainten häiriöiden välttämiseksi.)
19	V+	Positiivinen syöttöjännite liitäntä + 5V.
24	DOUT	Sarja lähtö kellokaksolle. Tätä pinä käytetään daisy-ketjun useita MAX7219 / MAX7221: n ja ei koskaan korkea-impedanssi.

MAX7219 -näytönohjaimen pinnijärjestys on kuvassa 6.



Kuva 6. [4, s. 1.]

MAX7219 näytönohjaimen kellokaavio on kuvassa 7.



Kuva 7. Kellokaavio [4, s. 6.]

Kytkentämahdollisuutena on 10 MHz:n sarjaliitäntä. Yksilöllinen LED-segmenttiohjaus mahdollistaa paremman toistotarkkuuden pianosimulaattorissa. Digitaalinen ja analoginen kirkkauden säätö on optiona ohjelmoinnissa. Näyttö vilkkuu virran kytkemisen yhteydessä merkiksi laitteen toimintavalmiudesta. [4, s. 1.]

5.2 Numero- ja valvontaluettelot

Taulukossa 4 on lueteltu 14 kappaletta osoitettavia numero- ja valvontarekistereitä. Digitaaliset rekisterit toteutetaan sirulla, joka voi ohjata 8 x 8 lediportteja (SRAM) näytönohjaimen digitaalisen -lähtöjä. Ne on osoitettu suoraan niin, että yksittäisiä numeroita voidaan päivittää ja säilyttää tietoja, kun V+ ylittää tyypillisesti 2 V. Ohjausrekisterit koostuvat tulkintatilasta ja näyttötehosta sekä skannausrajasta (skannattujen numeroiden määrä) ja sammutuksesta ja näyttötestistä (kaikki ledit ovat päällä esimerkiksi).

Taulukko 4. Rekisteriosoite [4, s. 7.]

Toiminta	Osoitteet					HEX koodi
	D15-D12	D11	D10	D9	D8	
No-Op	X	0	0	0	0	0xX0
Digi 0	X	0	0	0	1	0xX1
Digi 1	X	0	0	1	0	0xX2
Digi 2	X	0	0	1	1	0xX3
Digi 3	X	0	1	0	0	0xX4
Digi 4	X	0	1	0	1	0xX5
Digi 5	X	0	1	1	0	0xX6
Digi 6	X	0	0	1	1	0xX7
Digi 7	X	1	0	0	0	0xX8
Tulkinta tila	X	1	0	0	1	0xXA
Intensiteetti	X	1	0	1	0	0xXB
Skannaus-raja	X	1	0	1	1	0xXC
Sammutus	X	1	1	0	0	0xXE
Näyttö tes-taus	X	1	1	1	1	0xXF

5.3 Sammutustila

Kun MAX7219 on sammutustilassa, skannausoskillaattori pysäytettynä, kaikki segmentin virtalähteet kytketään maadoitukseen, jolloin kaikki ledit sammuvat (nollaus). Tiedot ledien järjestyksestä ja valvontarekisterit pysyvät muuttumattomina. Sammutus voi olla käytännössä virran säästämistä tai vikatilan tullessa menemällä sammutustilaan, tämän

sijaan minimisyöttövirran sammutustilassa on oltava ohjeistusarvon mukainen (3 V), logiikkatulot pitäisi olla käytössä tai V+ (CMOS-logiikka-tasolla).

Tyypillisesti MAX7219 kestää alle 250 µs lähdettäessä sammutustilaan. MAX7219 voidaan ohjelmoida sammutustilassa ja seisontatilassa voidaan ohittaa leditestin toiminto. [4, s. 8.]

5.4 Alkuvalmistelut

Kun virta kytketään laitteen kaikki rekisterit palautuvat ja näyttö tyhjenee, jolloin MAX719 siirtyy odotustilaan. Ohjelma ohjaimessa (MAX7219) käynnistyy ennen ledien käyttöä. Muussa tapauksessa ohjain jää odottamaan logiikalta seuraavaa käskyä, jolloin se ei purkaa tietoja rekistereistä, ja intensiteettirekisteri asetetaan sen vähimmäisarvoon. [4, s. 7.]

5.5 Skaalausrajarekisteri

Skaalausrajarekisteri määrää, kuinka monta ledivaloa palaa. Ne näytetään tavallisimmilla näytön tarkistusnopeuksilla 800 Hz ja kaikki ensimmäiset kahdeksan lediä palavat. Kun ledejä on vähemmän käytössä, skaalausnopeus on 8 FOSC / N, jossa N on numeroiden määrä skannattuna. Skannattujen numeroiden määrä vaikuttaa näytön kirkkauteen, tällöin skaalausrajarekisteri ei saisi olla tyhjiin ledeihin kytkettyinä.

Skaalausrajoituksen sarjan arvo on kolme tai vähemmän, jolloin yksittäisten numeroiden kuljettajat kuluttavat liiallisia määriä muistia. Näin ollen Reset-vastuksen arvo on säädettävä ledien mukaan, tämä rajoittaa yksittäisen ledien käytetyn tehohäviöitä. [4, s. 9.]

5.6 Näyttötestirekisteri

Näyttötestirekisteri toimii kahdessa tilassa: normaali ja näyttötesti. Näyttötestaustila kytkee kaikki merkkivalot päälle ohittamalla, mutta ei muuttamalla kaikkia valvonta- ja

numerokirjaimia (mukaan lukien seisontarekisteri). Näyttökoetilassa kahdeksan lediä skannataan ja käyttöjakso on 31/32 (15/16 MAX7221:lle). [4, s. 10.]

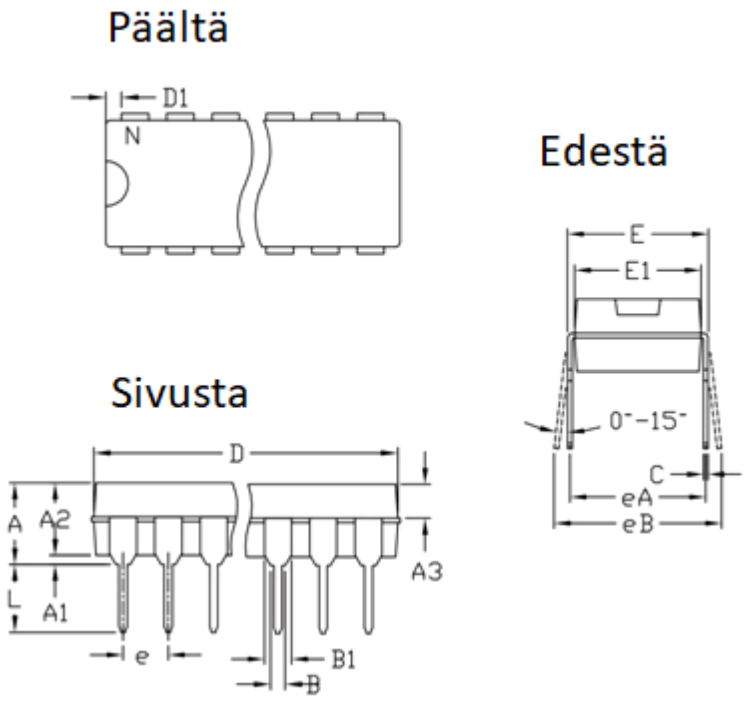
5.7 Ei-rekisteri

Ei-rekisteriä käytetään MAX7219-näytönohjaimen kokoamisen yhteydessä. Kaikkien laitteiden LOAD/CS-tulot yhdistetään ja kytketään DOUT DIN liitäntään vierekkäin laitteet. DOUT on CMOS-logiikan tason tulo, joka helposti ajaa peräkkäin peräkkäisiä osia DIN -liittimeen.

Esimerkiksi neljä MAX7219 on sarjoitettu toisiinsa, kirjoitetaan neljäs siru ja siirretään haluttuun 16-bittisen sanan ohjelmassa, jota seuraa kolme ei-op-koodia (hex 0xXX0X). Kun LOAD/CS menee korkealle, data on lukittu kaikissa laitteissa. Ensimmäiset kolme ohjelmarivin merkkiä saavat ei-op-komentoja ja neljäs vastaanottaa ohjelmoidut tiedot. [4, s. 10.]

5.8 Näytönohjaimen mitat

Näytönohjaimen mitat on havainnoitu kuvassa 8. Näytönohjaimen perusmitoitus on esitetty päältä, edestä ja sivulta.



	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	---	0.180	---	4.572
A1	0.015	---	0.38	---
A2	0.125	0.175	3.18	4.45
A3	0.055	0.080	1.40	2.03
B	0.015	0.022	0.381	0.56
B1	0.045	0.065	1.14	1.65
C	0.008	0.014	0.2	0.355
D1	0.005	0.080	0.13	2.03
E	0.300	0.325	7.62	8.26
E1	0.240	0.310	6.10	7.87
e	0.100 BSC.		2.54 BSC.	
eA	0.300 BSC.		7.62 BSC.	
eB	0.400 BSC.		10.16 BSC.	
L	0.115	0.150	2.921	3.81

Kuva 8. [4, s. 16.]

6 Toteutus ja valmistus

6.1 Pianosimulaattorin ledipiirilevyn valmistus

Pianosimulaattorin ledipiirilevy valmistettiin täysin käsin piirretyn sähköpiirustuksen perusteella juottamalla. Piirilevy muotoiltiin oikeaa leveyteen sähkökourun kiinnitys uraan nähden, josta saariin ledinauhalle hyvä runko aikaiseksi. Tässä vaiheessa testattiin ledien toiminta ennen juottamista. Värit järjesteltiin samalla oikein myös piirilevylle. Oranssi väri on pianon soittimien valkoisien koskettimien väri, ja punaiset ledit ovat mustien koskettimien väri. Nämä määräytyivät täysin kaupasta saatujen ledien ansiosta. Kytkennän johtojen määrä on valtaisa, sillä jokaisen kahdeksan ledin sarjasta plus(+) -johtimen oli tultava jokaisen kahdeksan sarjan ensimmäisen plus(+) -johtimen kautta loppuun riviliittimen yhteen pinniin. Kaikki miinus(-) -johtimet saivat tulla kaikkien kahdeksan miinus(-) -johtimien kautta loppuun riviliittimelle yksittäin. Plus(+) -johtimet saivat tulla jokaisen ensimmäisen kahdeksan ledin sarjan kautta loppuun riviliittimelle saakka.

Seuraavaksi täytyi tehdä koekytkentälevylle tarvittava kytkentä, johon voitaisiin Arduinosta ohjata MAX7219-näytönohjaimiin. Vapaan harkinnan mukaan toteutettiin kytkentä koekytkentälevylle. Kytkentä ei ollut monimutkainen, mutta hankala koekytkentälevyn puutteellisten tietojen takia. Koekytkentälevy oli jaettu kahteen piiriin, mikä ilmeni signaalin katkeamisena. Seuraavaksi kytkettiin ledit riviliittimeen koekytkentälevyyn. Tässä käytettiin apu Excel-taulukoita, pystyi helposti saamaan yksittäiset ledit palamaan (eli yksilöimään yksittäisen ledin ledinauhalta). Tällä saadaan oikeat johdot kytkettyä MAX7219-näytönohjaimista lähteviin pinneihin oikeassa järjestyksessä.

6.2 Ohjelmointi

Arduinon ohjelmointi on haastavaa. Ohjelmasta pianosimulaattorin ympärille ei kannata ruveta tekemään liian hienoa. Saatavilla on suoraan Arduinon valmiita ohjelmapohjia HEX-muuntimella, jolla pääsee heti testaamaan toimivuutta. HEX-muuntimen ansiosta pääsee hyvin helposti MAX7219-näytönohjaimen toimintaan perille, mutta on silti syytä tutustua lähemmin 8 x 8 LED-matriisin toimintaan. Kannattavaa on myös hankia tarvittavat komponentit tutustumista varten, mistä on suuri apu nykyaikaisten LED-televisioista.

HEX-muuntimella oli helppo saada lediyhdistelmät palamaan halutulla tavalla ja halutussa järjestyksessä. Pianosimulaattori saa rytmin aikaiseksi helposti ohjelman kellotaajuuteen laitetun arvon mukaan. Määrätty määrä painalluksia yhden syklin aikana, joka on pianosimulaattorin kappaleessa Minuet in G yksi sekunti. Yhden pitkän painalluksen aikana on seitsemän sykliä, jotka kestävät seitsemän sekuntia. Tässä tilanteessa tämä on järkevin tapa lähestyä kappaleesta tullutta rytmiä, koska vasemmalla kädellä on niin yksinkertaisempi rytmi. Annetun HEX-muuntimen avulla saadaan haluttujen pianopainikkeiden kohdalla ledit syttymään. Arduino-ohjelmistossa on kuitenkin tehtävä ohjelmalliset koodaukset ensiksi.

Ohjelmointia tehdessä on ymmärrettävä ledikytkentä täysin, että pystyy järjestämään oikeisiin paikkoihin ledin syttymisen. Tarvittiin ylimääräin Excel-taulukko ledikytkennästä, josta pystyy toteamaan ledikytkennän järjestyksen helpommin. Taulukosta on kuvakaappaus liitteenä 3. Taulukon avulla voin helposti selvittää led kytkennän lähtöpinnit, josta saa halutut ledit palamaan. Annettuihin liitäntänumeroihin pitää vain syöttää oikein miinus(-) -ja plus(+) -napoihin 3 V, niin saadaan halutut ledit palamaan. MAX7219-näytönohjain toimii yksinkertaisesti loppujen lopuksi. Se toimii täysin Arduinolta tulevan HEX-koodin mukaan pitämällä vain lähtöpinnien virransyöttöä päällä oikealla napaisuudella.

Pianosimulaattorissa käytetään kahta MAX7219-piiriä, sillä yhden piirin ohjaus ei olisi riittänyt ohjaamaan kaikkia 88 lediä. HEX-koodin ansiosta tämä on mahdollista, sillä MAX7219-näytönohjain pystyy tiedonsiirtonastoistaan jatkamaan HEX-koodin seuraavaan MAX7219-näytönohjaimeen. Tämän ansiosta voidaan helposti määritellä uudet 8 x 8 ledin ohjaus HEX-koodin avulla. Tärkeää on kuitenkin huomioida jälkimmäinen MAX7219-näytönohjaimen seuraavalla askelluksella. Jos jälkimmäistä piiriä ei ohjata seuraavassa jaksonohjauksessa laisinkaan, tämä muistaa viimeisimmän tulleen HEX-koodin, eikä muutu ennen kuin se resetoidaan tai annetaan uusi siirretty HEX-ohjauskoodi.

6.3 Kytkentä

Kytkentä on yksinkertainen, minkä huomaa harjoituslevyn kytkennästä sekä sähköpiirustuksesta, joka on liitteessä 1 sähköpiirustus. Kytkentä on tehty suoraan MAX7219-näytönohjaimen annettujen pinnitietojen perusteella. Tässä tapauksessa kaikille pinneille on oma tarkoitus. Testikytkennän avulla saadaan 8 x 8 -ledinäytöllä toiminta hyvin selville, mistä saa helpotusta ymmärtää MAX7219-näytönohjaimen toimintaa. Periaate perustui täysin nykypäivän LED-televisioiden toimintaperiaatteeseen. Tämän ansiosta suuretkin ledinäytöt saadaan toimimaan täysin valmistamalla pienistä paloista toteutettuja ratkaisuja.

6.4 Testaus

Oikein kytkemisen jälkeen käyttäminen ja testaus sujuivat käsi kädessä, eli mitä enemmän pianosimulaattoria käytti, sitä selvemmäksi sen toiminta tuli ja toimintavarmuus parantui huomattavasti. Pianosimulaattori alkoi toimimaan täysin vasta työn loppuvaiheessa. Testauksen aikana kuitenkin huomattiin harjoituskytkentälevyn huonot puolet, jolloin huomattiin pianosimulaattorin toimivan sattumanvaraisesti eri tapauksissa testausvaiheessa huomattujen kosketushäiriöiden yhteydessä piuhaliitoksissa. Työn edetessä oli tärkeää tehdä kaikki liitokset piirilevyille ja vielä juottamalla kaikki kytkennät toisiinsa, mikä on tämmöisien projektien suurin edellytys onnistumisen kannalta.

Testaustulokset molemmissa testauksissa, eli pianosimulaattorin ja 8 x 8 -matriisinäytön osalta saatiin tehtyä loppuun, joiksi havaittiin näiden kohdalla sovellus mahdollisuuksien määrän olevan valtaisa. Testauksesta saatujen tuloksien perusteella insinöörityön onnistuminen oli saatu haluttuun vaiheeseen ohjauksen osalta, kun ohjaus toimi täysin oikein testauksien jälkeen.

Testauksen yhteydessä testattiin Arduinon mahdollisuuksia 8 x 8 -matriisinäytöllä. Matriisimoduulin MAX7219 8 x 8 ledeillä, jonka kasattiin myös juottamalla. Tämän ansiosta päästiin paremmin perehtymään MAX7219-näytönohjaimen pinnijärjestyksen ja niiden erityylisiin käyttömahdollisuuksiin. Testaus tehtiin valmiiden ohjelmoitujen ohjelmien avulla ja tutustuen samalla ohjelmatoteutuksiin.

7 Yhteenveto

Pianosimulaattorin valmistusvaiheet alkoivat ledinauhan valmistuksella, jonka yhteydessä kytkentä suunniteltiin harjoituslevylle. Samaan aikaan insinöörityössä tutustuttiin MAX7219-näytönohjaimen toimintaan tarkemmin testaamisen avulla.

Seuraavaksi insinöörityössä testattiin Arduinolla ohjelman toimintaa, ja tutustuttiin ohjelmointikieleen. Arduinon mahdollisuuksien ohjelmoinnin takia insinöörityö tehtiin yksinkertaisella ohjelmoinnilla. Ohjelmoinnin yhteydessä päästiin HEX-muunnoksiin tutustumaan, sillä näytönohjain toimi HEX-arvoilla.

Pianosimulaattoria testattiin hyvin paljon ja tehtiin paljon erityyppisiä Ecxel-taulukoita ymmärtämisen helpottamiseksi. Näiden ansiosta ohjauspuolen ominaisuuksista saatiin selkeä kokonaisuus pelkälle ohjelmaosuudelle.

Pianosimulaattori ei vielä kuitenkaan soita pianoa, koska pianosimulaattorityössä ollaan vasta ohjauksen kohdalla. Seuraavaksi olisi tarkoitus valmistaa pianon koskettimien päälle solenoidipainikkeet, joiden olisi tarkoitus painella koskettimia ohjauksen mukaan oikeassa järjestyksessä. Insinöörityössä ei enempää keskitytty loppuu viemiseen, sillä oli tarkoitus saada ainoastaan ohjaus toimimaan.

Tällä ohjausmenetelmällä saataisiin ohjattua nykyajan konserttien valojärjestelmiä tai muita konserttien efektejä liittämällä järjestelmään sähköpianon koskettimien seurantaan, jolloin saataisiin oikea-aikaisia ohjauksia aikaiseksi.

Tarkempaa tarkastelua on syytä tehdä vielä Arduinon Uno-piirikortille, sillä siinä on vielä paljon käyttämätöntä kapasiteettiä käyttämättä. Taajuuden muuttamisella tulisivat käyttöön päävärit (vihreä, keltainen, punainen ja sininen). Pianosimulaattorista on myös mahdollista opiskella pianon soittoa. Pianosimulaattorissa on vielä olemassa paljon erilaisia käyttötarkoituksia ja kohteita.

Lähteet

1 Kronström Peter. 2014. ELEC-A4010 Sähköpaja Arduino ohjelmointi. Aalto-yliopisto Helsinki. Verkkoaineistoa. <https://docplayer.fi/7353013-Elec-a4010-sahkopaja-arduinon-ohjelmointi-peter-kronstrom.html>. Luettu 17.8.2018.

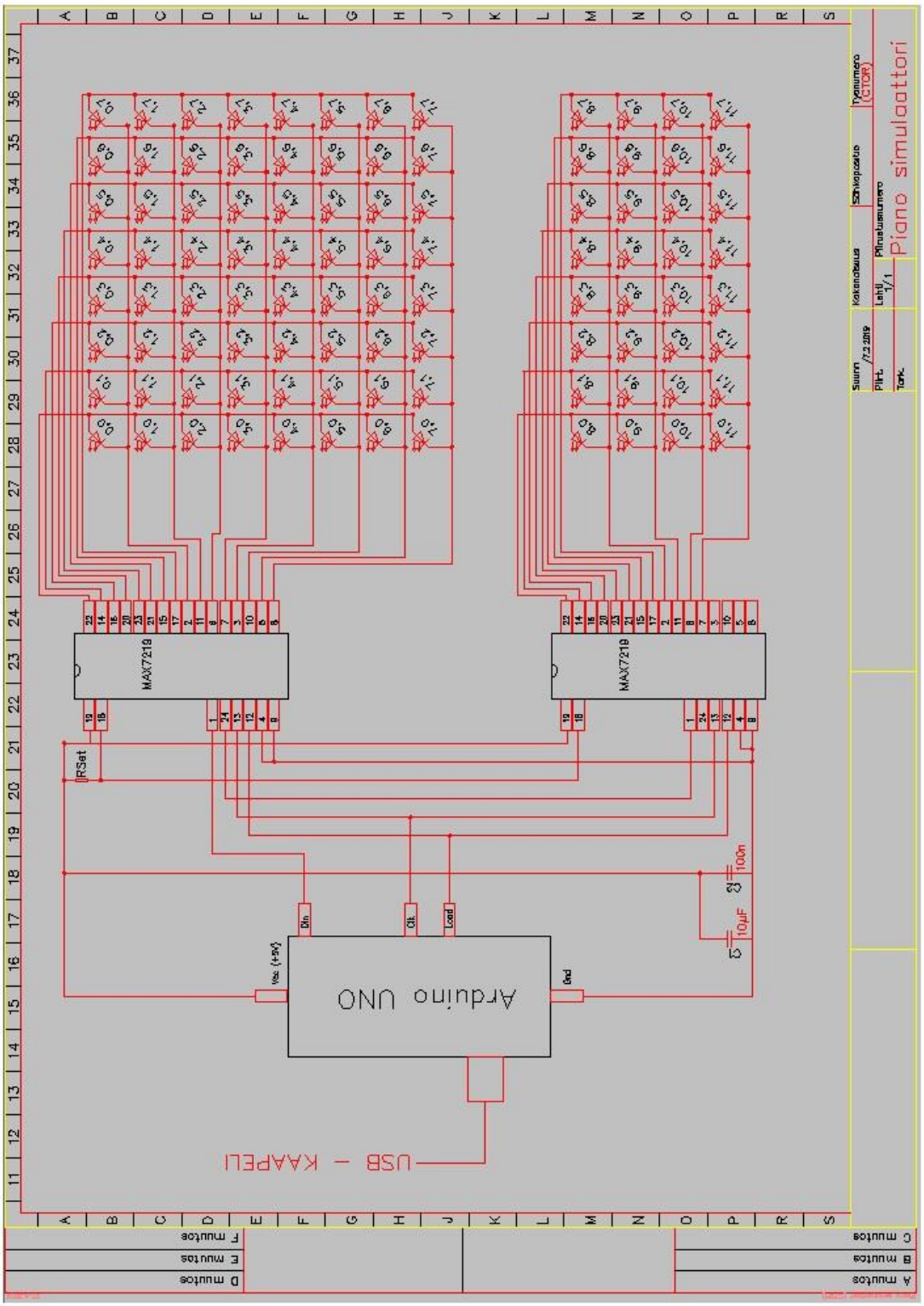
2 Massimo Banzi. 2011. Arduino perusteita hallintaan. Painos.BoD – Boks ON Demand, Norderstedt, Saksa.

3 Blinnikka Kyösti. 2016. Arduino tutuksi. Opiskelu materiaali. Verkkoaineistoa. Opetus- ja kulttuuri- ministeriö ja Luna-keskus Suomi. <https://docplayer.fi/67739197-Arduino-tutuksi-kyosti-blinnikka.html>. Luettu 9.12.2017.

4 Rio Roble and San Jose (Komponentti manuaali). 2003. Maxim Integrared 160. Ca 95134 USA.

5 Arduino. Verkkoaineisto. Wikipedia. <https://fi.wikipedia.org/wiki/Arduino>. Luettu 9.12.2017.

Sähköpiirustus



Arduino-ohjelmointiosuus pianosimulaattorissa

```
/*
Piano simulaattori (CTOR)
by Marko Mustonen Cone
Ohjelmointi osuus

*/
unsigned char i;
unsigned char j;

int Max7219_pinCLK = 11;
int Max7219_pinCS = 10;
int Max7219_pinDIN = 12;

unsigned char disp1[128][8]={
  0x00, 0x00, 0x40, 0x00, 0x00, 0x02, 0x00, 0x00, //0;01
0x00, 0x00, 0x40, 0x00, 0x00, 0x02, 0x00, 0x00, //0;02
0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, //0;03
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;04
0x00, 0x00, 0x00, 0x08, 0x00, 0x02, 0x00, 0x00, //0;05
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;06
0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, //0;07
0x00, 0x00, 0x40, 0x00, 0x20, 0x00, 0x00, 0x00, //0;08
0x00, 0x00, 0x40, 0x00, 0x20, 0x00, 0x00, 0x00, //0;09
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;10
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;11
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;12
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;13
0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, //0;14
0x00, 0x00, 0x10, 0x00, 0x10, 0x00, 0x00, 0x00, //0;15
0x00, 0x00, 0x10, 0x00, 0x10, 0x00, 0x00, 0x00, //0;16
0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, //0;17
0x00, 0x00, 0x00, 0x01, 0x10, 0x00, 0x00, 0x00, //0;18
0x00, 0x00, 0x40, 0x00, 0x10, 0x00, 0x00, 0x00, //0;19
0x00, 0x00, 0x10, 0x00, 0x10, 0x00, 0x00, 0x00, //0;20
0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0x00, //0;21
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;22
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;23
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;24
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;25
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;26
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;27
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;28
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;29
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;30
0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, //0;31
0x00, 0x00, 0x40, 0x00, 0x80, 0x00, 0x00, 0x00, //0;32
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;33
0x00, 0x00, 0x00, 0x02, 0x80, 0x00, 0x00, 0x00, //0;34
0x00, 0x00, 0x00, 0x08, 0x80, 0x00, 0x00, 0x00, //0;35
0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, //0;36
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;37
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;38
0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, //0;39
0x00, 0x00, 0x00, 0x01, 0x00, 0x02, 0x00, 0x00, //0;40
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;41
0x00, 0x00, 0x00, 0x08, 0x00, 0x02, 0x00, 0x00, //0;42
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;43
0x00, 0x00, 0x00, 0x40, 0x04, 0x00, 0x00, 0x00, //0;44
0x00, 0x00, 0x00, 0x40, 0x04, 0x00, 0x00, 0x00, //0;45
0x00, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, //0;46
```

```
0x00, 0x00, 0x00, 0x20, 0x04, 0x00, 0x00, 0x00, //0;47
0x00, 0x00, 0x00, 0x08, 0x04, 0x00, 0x00, 0x00, //0;48
0x00, 0x00, 0x00, 0x02, 0x04, 0x00, 0x00, 0x00, //0;49
0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, //0;50
0x00, 0x00, 0x00, 0x02, 0x04, 0x00, 0x00, 0x00, //0;51
0x00, 0x00, 0x00, 0x02, 0x04, 0x00, 0x00, 0x00, //0;52
0x00, 0x00, 0x00, 0x02, 0x04, 0x00, 0x00, 0x00, //0;53
0x00, 0x00, 0x00, 0x08, 0x10, 0x00, 0x00, 0x00, //0;54
0x00, 0x00, 0x00, 0x08, 0x20, 0x00, 0x00, 0x00, //0;55
0x00, 0x00, 0x00, 0x08, 0x80, 0x00, 0x00, 0x00, //0;56
0x00, 0x00, 0x40, 0x00, 0x20, 0x00, 0x00, 0x00, //0;57
0x00, 0x00, 0x40, 0x00, 0x20, 0x00, 0x00, 0x00, //0;58
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;59
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;60
0x00, 0x00, 0x00, 0x10, 0x20, 0x00, 0x00, 0x00, //0;61
0x00, 0x00, 0x00, 0x02, 0x20, 0x00, 0x00, 0x00, //0;62
0x00, 0x00, 0x00, 0x01, 0x20, 0x00, 0x00, 0x00, //0;63
0x00, 0x00, 0x40, 0x00, 0x00, 0x02, 0x00, 0x00, //0;64
0x00, 0x00, 0x40, 0x00, 0x00, 0x02, 0x00, 0x00, //0;65
0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, //0;66
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;67
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;68
0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, //0;69
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;70
0x00, 0x00, 0x00, 0x20, 0x00, 0x02, 0x00, 0x00, //0;71
0x00, 0x00, 0x00, 0x20, 0x10, 0x00, 0x00, 0x00, //0;72
0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, //0;73
0x00, 0x00, 0x00, 0x01, 0x10, 0x00, 0x00, 0x00, //0;74
0x00, 0x00, 0x40, 0x00, 0x10, 0x00, 0x00, 0x00, //0;75
0x00, 0x00, 0x10, 0x00, 0x10, 0x00, 0x00, 0x00, //0;76
0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00, //0;77
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;78
0x00, 0x00, 0x02, 0x00, 0x20, 0x00, 0x00, 0x00, //0;79
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;80
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;81
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;82
0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, //0;83
0x00, 0x00, 0x00, 0x20, 0x20, 0x00, 0x00, 0x00, //0;84
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;85
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;86
0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, //0;87
0x00, 0x00, 0x40, 0x00, 0x80, 0x00, 0x00, 0x00, //0;88
0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, //0;89
0x00, 0x00, 0x00, 0x02, 0x80, 0x00, 0x00, 0x00, //0;90
0x00, 0x00, 0x00, 0x08, 0x80, 0x00, 0x00, 0x00, //0;91
0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, //0;92
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;93
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;94
0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, //0;95
0x00, 0x00, 0x00, 0x01, 0x00, 0x02, 0x00, 0x00, //0;96
0x00, 0x00, 0x00, 0x02, 0x00, 0x02, 0x00, 0x00, //0;97
0x00, 0x00, 0x00, 0x08, 0x00, 0x02, 0x00, 0x00, //0;98
0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, //0;99
0x00, 0x00, 0x00, 0x08, 0x10, 0x00, 0x00, 0x00, //0;100
0x00, 0x00, 0x00, 0x08, 0x10, 0x00, 0x00, 0x00, //0;101
0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, //0;102
0x00, 0x00, 0x00, 0x02, 0x10, 0x00, 0x00, 0x00, //0;103
0x00, 0x00, 0x00, 0x08, 0x10, 0x00, 0x00, 0x00, //0;104
0x00, 0x00, 0x00, 0x20, 0x10, 0x00, 0x00, 0x00, //0;105
0x00, 0x00, 0x00, 0x40, 0x10, 0x00, 0x00, 0x00, //0;106
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;107
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;108
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;109
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;110
```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;111
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;112
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;113
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;114
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;115
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;116
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;117
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;118
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;119
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;120
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;121
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;122
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;123
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;124
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;125
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;126
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;127
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //0;128
};

void Write_Max7219_byte(unsigned char DATA)
{
  unsigned char i;
  digitalWrite(Max7219_pinCS,LOW);
  for(i=8;i>=1;i--)
  {
    digitalWrite(Max7219_pinCLK,LOW);
    digitalWrite(Max7219_pinDIN,DATA&0x80);
    DATA = DATA<<1;
    digitalWrite(Max7219_pinCLK,HIGH);
  }
}

void Write_Max7219(unsigned char address,unsigned char dat)
{
  digitalWrite(Max7219_pinCS,LOW);
  Write_Max7219_byte(address);
  Write_Max7219_byte(dat);
  digitalWrite(Max7219_pinCS,HIGH);
}

void Init_MAX7219(void)
{
  Write_Max7219(0x09, 0x00);
  Write_Max7219(0x0a, 0x03);
  Write_Max7219(0x0b, 0x07);
  Write_Max7219(0x0c, 0x01);
  Write_Max7219(0x0f, 0x00);
}

void setup()
{

  pinMode(Max7219_pinCLK,OUTPUT);
  pinMode(Max7219_pinCS,OUTPUT);
  pinMode(Max7219_pinDIN,OUTPUT);
  delay(201);
  Init_MAX7219();
}

void loop()
{
  for(j=0;j<128;j++)
  {
    for(i=1;i<9;i++)
```

```
      Write_Max7219(i,disp1[j][i-1]);
    delay(201);
  }
}
```

Led-kytkentä-Excel