Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Janne Kinnunen

Arduino muusikon työkaluna

MIDI-koskettimiston prototyypin rakentaminen

Metropolia Ammattikorkeakoulu

Muusikko (AMK)

Musiikin tutkinto

Opinnäytetyö

04.12.2020

Tekijä Otsikko Sivumäärä Aika	Janne Kinnunen Opinnäytetyön otsikko 25 sivua + 2 liitettä 04.12.2020
Tutkinto	Muusikko (AMK)
Tutkinto-ohjelma	Musiikin tutkinto
Suuntautumisvaihtoehto	Musiikin esittäjä, basso
Ohjaaja Arviointi	Lehtori Jukka Väisänen Lehtori Julius Mauranen

Opinnäytetyöni avartaa Arduino-kehitysalustan mahdollisuuksia muusikon näkökulmasta ja esittelen sulautetut järjestelmät. Sulautettu järjestelmä on laite, jonka toimintaa ohjaa tietokone ja se on suunniteltu suorittamaan tiettyä ennalta määrättyä tehtävää

Esimerkkiprojektina oleva MIDI-koskettimisto antaa esimerkin siitä, millaista on rakentaa prototyyppi Arduino-alustalle. Työssäni yhdistyy elektroniikka, ohjelmointi ja ohjelmiston käyttö niin, että esimerkkityön laitteisto voi lähettää MIDI-viestejä haluttuun tietokoneohjelmistoon. Työlläni haluan osoittaa, että oman sulautetun järjestelmän rakentaminen on alan harrastajalle toteutettavissa melko pienellä kynnyksellä.

Tärkeä lähde on yksi harvoista Arduino-alustaa suomeksi käsittelevä kirjoista ”Sulautetut: Opi rakentamaan sulautettuja järjestelmiä”. Suurin osa rakentamiseen liittyvistä tiedoista on löytynyt Arduino-harrastajien keskustelupalstoilta ja esimerkiksi YouTube-videoista. Arduinosta löytyy paljon opinnäytetöitä, mutta ei juurikaan muusikon näkökulmasta tehtyjä töitä.

Päädyin valmistamaan yksinkertaisen MIDI-koskettimiston prototyypin, koska siitä on hyvä jatkokehittää monipuolisempia laitteita. Työtavaksi voisi sanoa ”kokeilu ja erehdys”, jossa tutustun koodin ja laitteiden toimintaperiaatteeseen, ja etsin tietoa, kunnes löydän tilanteeseeni sopivan ratkaisun.

Lopputuloksena syntyi laite, jota voi käyttää kahden oktaavin MIDI-koskettimistona. Puutteena laitteessa on ilmaisuvoiman puute, koska se lähettää viestin vain, kun nuotti menee päälle ja vapautetaan pois. Tämä antaa kuitenkin hyvän pohjan tehdä tulevaisuudessa monipuolisempi Arduino-alustaan perustuva MIDI-kontrolleri.

Projektin rakentamisen alustukseksi pohdin rakentamisen filosofiaa, joka toimi ohjenuorana prototyypin tekemiseen.

Avainsanat	Arduino, DIY, koskettimisto, MIDI, sulautettu järjestelmä

Author	Janne Kinnunen
Title	Building an Arduino Based MIDI Keyboard
Number of Pages	25 pages + 2 appendices
Date	04 Dec. 2020
Degree	Bachelor of Music
Degree Programme	Music
Specialisation Option	Music Performance, Electric Bass
Supervisors	Jukka Väisänen, MMus Julius Mauranen, MMus

This final project examines the Arduino development environment from a musician's standpoint. This report introduces the hardware and software elements of Arduino development and how musicians can build and use embedded systems in music performance. It also explains the theoretical foundation of the MIDI protocol. The latter part includes a step-by-step report on a DIY project of building an Arduino based MIDI keyboard.

A lot of information about Arduino was found in online community sources, such as YouTube and the Arduino internet forum. Another key source was *Make: Arduino Bots and Gadgets* (2011), where Tero and Kimmo Karvinen presented some philosophical guidelines for the building process of an embedded system. In the prototyping phase, I followed the PDCA cycle - Plan-Do-Check-Act – also known as the trial and error method.

During the process, I learned how MIDI messages are formed. I gained insights into debugging and problem-solving in projects that combine electronics, software and programming. I used reverse engineering as I deconstructed a toy keyboard that I found in a flea market and used the parts to build the MIDI keyboard. This process resulted in interesting observations about electronics. The result of this project is a functional MIDI keyboard prototype, which will serve as a foundation to improve upon in future projects.

There is only a little literature on music related Arduino projects in the Finnish language. This final project helps to fill this gap. This report can be useful for tech-savvy musicians, who want to learn more about Arduino, or engineers, who want to explore musical embedded systems. I hope this project will encourage other DIY-minded people to experiment with the Arduino platform.

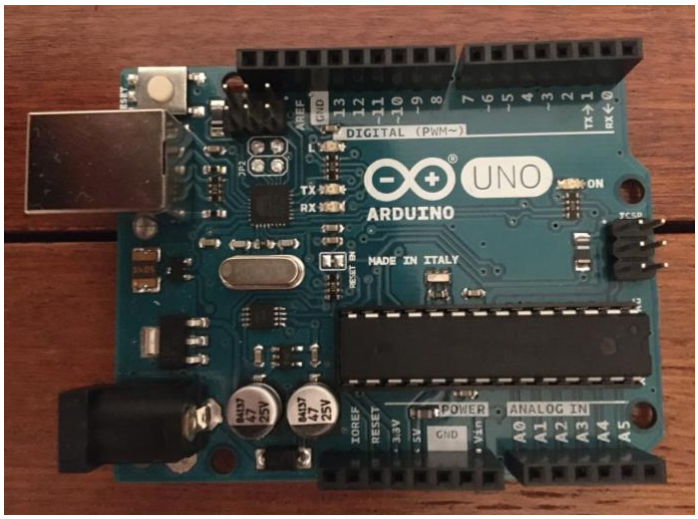
Keywords	Arduino, DIY, MIDI, Embedded System

Sisällys

Liite 2. MIDI-koskettimiston koodi

1 Johdanto

Opinnäytetyöni tarkoituksena on tutkia Arduino-elektroniikka-alustan (kuvio 1) mahdollisuuksia teknologiasta kiinnostuneen muusikon näkökulmasta. Työni päättyy esimerkkiprojektiin, jossa valmistan Arduino-pohjaisen MIDI-koskettimistoprototyypin (luku 5). Esittelen Arduino-alustan ja esimerkkiprojektini perusteet lyhyesti niiltä osin, miltä se työni luettavuuden ja toistettavuuden kannalta on olennaista. Opinnäytetyöni siis ei itsessään ole aloittelijan opas Arduinon ja sulautettujen järjestelmien maailmaan. Kuitenkin listaamieni lähteiden ja työprojektini avulla tavoitteeni on antaa lukijalle hyvä käsitys siitä, miten ja millaista on hyödyntää Arduino-alustaa musiikkisovelluksissa. Muusikolle kiinnostavia mahdollisuuksia Arduino tarjoaa esimerkiksi MIDIn ja digitaalisen signaaliprosessoinnin kautta. Tässä työssä keskityn Arduinon hyödyntämiseen MIDI-kontrollerina. Tavoitteenani on saada käyttöön yksinkertainen, mutta toimiva midikoskettimiston prototyyppi, josta voi myöhemmin kehitellä ja monipuolisemman laitteen.



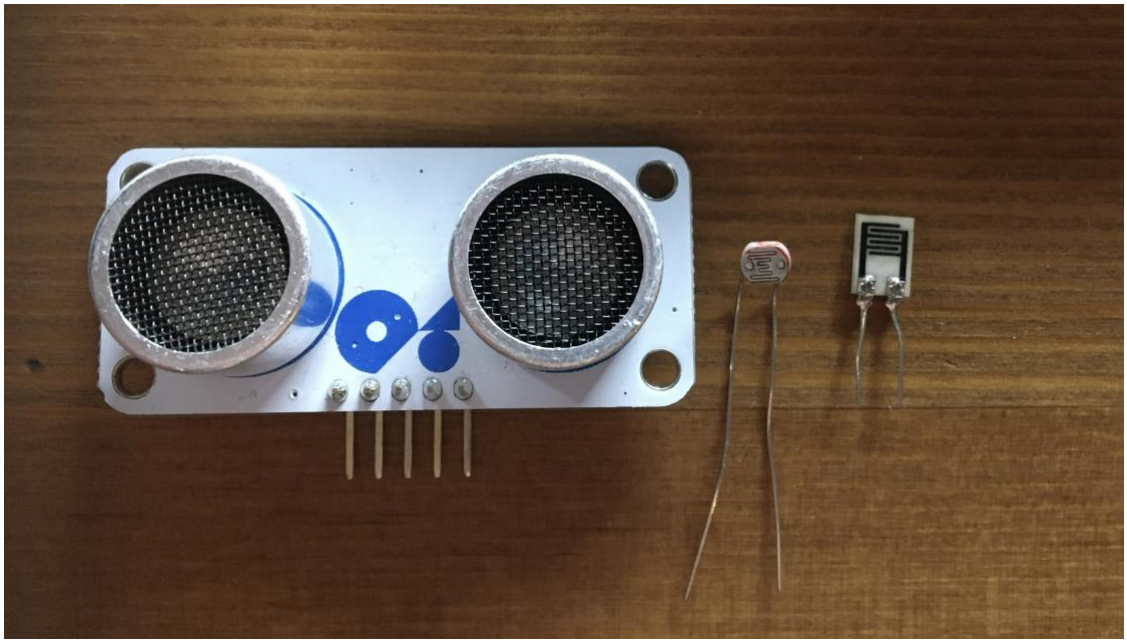
Kuvio 1. Arduino-kehitysalusta

Digitaaliset efektipedaalit sekä syntetisaattorit ja erilaiset midikontrollerit ovat sulautettuja järjestelmiä, joita muusikot käyttävät työssään päivittäin. Sulautettu järjestelmä on laite, jonka toimintaa ohjaa tietokone ja se on suunniteltu suorittamaan tiettyä ennalta määrättyä tehtävää (ks. luku 2). Sulautetun järjestelmän sisältämää tietokonesirua kutsutaan mikrokontrolleriksi.

Digitaaliteknologian kehitys on mahdollistanut sen, että enää sulautetun järjestelmän luomiseen ei ole välttämätöntä omata insinöörin tutkintoa tai vastaavaa tietotaitoa. Lisäksi erilaisia mikrokontrollereja on mahdollista hankkia helposti ja kuluttajaystävällisin hinnoin. Internetissä on paljon tee-se-itse-yhteisöjä, joissa jaetaan tietoa ja apua mikrokontrollerien parissa työskenteleville.

Arduinon kaltaiset mikrokontrollerit toimivat hyvin musiikkisovelluksissa, koska ne muodostavat usein reaktiivisia järjestelmiä, jotka reagoivat ulkomaailman impulsseihin nopeasti. (Karvinen ja Karvinen 2019, 9) Tämä mahdollistaa mahdollisimman pienen latenssin[1] esimerkiksi koskettimen painalluksen ja toivotun tehtävän, vaikkapa äänen toistamisen välillä.

Mikrokontrolleriin liitettävät ympäristöä mittaavat laitteet ovat nimeltään antureita (kuvio 2). Anturi voi olla esimerkiksi etäisyyttä mittaava ultra-äänisensori tai paineen tai valon määrään reagoiva vastus (kuvio 2). Kaiken tämän voi muuttaa esimerkiksi mididataksi tai sillä voi ohjata muita elektronisia laitteita. Tämä mahdollistaa suuren määrän erilaisia käytännön sovellutuksia.



Kuvio 2. Anturi-esimerkki: ultra-äänisensori ja valo- ja painevastus

[1] Latenssilla tarkoitetaan digitaalisessa musiikkiohjelmistossa tai – laitteistossa annetun komennon ja toiminnan suorittamisen välistä viivettä.

1.1 Opinnäytetyön tavoite

Esimerkkiprojektia tehdessä minulla oli selkeä tähtäin: haluan tehdä USB-MIDI-koskettimiston, jolla voin soittaa MIDI-instrumentteja tietokoneella. Tarkoituksena ei ole tehdä realistista ja tarkkaa painotettua pianon koskettimistoa, vaan muodostaa yhteys laitteiston ja ohjelmiston välille. Tässä työssä en varsinaisesti rakenna mitään, vaan yhdistelen olemassa olevia laitteita, ja ohjelmoin Arduinon toimimaan välikappaleena.

Käytin työni pohjana pianon koskettimistoa, koska se on tuttu, ja selkeästi musiikkiin sovellettavissa oleva laite. Käyttämäni koskettimisto ei ole itsessään kovin monimutkainen laite, kysehän on lopulta vain painikkeista, jotka on aseteltu 12 sävelen toistuvaan sarjaan. Kun painikkeen, jonka olotila on joko ”päällä” tai ”pois päältä” saa toimimaan haluamallaan tavalla, on sen jälkeen luontevaa kokeilla monipuolisempia tapoja luoda ääntä ja MIDI-viestejä.

1.2 Prosessin työtavat

Olennaista projektissani on tiedonhaku Arduino-foorumeilta, Youtube-videoista ja alan kirjallisuudesta. Internet-foorumeilta voi löytää rakentelijoita, jotka ovat työskennelleet vastaavien ongelmien parissa. Löysinkin ratkaisevan tärkeää aineistoa Arduino-foorumilta. Osien hankinta ja luova hyödyntäminen vaatii ongelmanratkaisutaitoja, ja kykyä lähestyä asioita järjestelmällisesti ja systemaattisesti.

Isoin osa työskentelystä on selvittää, miksi jokin asia ei toimi. Haastetta ja muuttujia lisää, kun mahdollinen vika voi piillä laitteistossa, omassa koodissa tai kolmannen osapuolen ohjelman väärin käyttämisessä. Suurin osa koodista on suoraan Arduino-forumeilta kopioitua, testattua ja omaan käyttöön sovellettua. Koska en rakenna kaikkia laitteita itse, vaan hyödynnän kierrätettyä laitteistoa ja lainattua koodia, haasteena on takaisinmallinnus (Engl. Reverse Engineering) (Luku 5.3.1).

Työssäni toteutan PDCA-sykliä (Plan, Do, Check, Act). Jaan projektin osa-alueet pienempiin osatavoitteisiin. Luonnostelen suunnitelman, jonka pohjalta teen prototyypin. Tarkistan toiminnan, jonka jälkeen teen tarvittavat korjaukset, tai siirryn seuraavaan vaiheeseen. Projektityötäni voi ajatella myös yhtenä PDCA-syklinä, jossa kehitetään ensimmäinen prototyyppi, sitten tutkitaan sen toimivuus ja pohditaan tulevaisuudessa

paranneltavat osa-alueet. Lopullisena tavoitteena on saada toimiva MIDI-koskettimisto. (Helsingin Kaupunki. n.d.)

1.3 Rakentamisen filosofia

Tekemisen ohjenuorana käytän Sulautetut-kirjan kappaletta ”Rakentamisen Filosofia” (Karvinen & Karvinen 2019, 15)

> Prototyyppi - ensin valmista sitten kunnollista - - Valmiiseen tulokseen pääsee karsimalla turhia ominaisuuksia ja menemällä yli siitä, mistä aita on matalin. Käytä huoleti kuminauhaa ja teippiä, jos se nopeuttaa testaamista. Älä yritä optimoida ensimmäisen version koodia. (Karvinen & Karvinen 2019, 15)

Tämä ei tietenkään tarkoita sitä, että tarkoituksena olisi valmistaa huono laite, vaan että aluksi keskitytään ulkoisten asioiden hiomisen sijaan laitteen toimintaan liittyvien ongelmien ratkaisemiseen. Toimivaa prototyyppiä voi myöhemmin hioa kestävämmäksi ja hienommaksi, sitä ei kannata kuitenkaan tehdä ennen kuin laite toimii. Rakentamisen filosofiasta inspiroituneina asetin itselleni ohjenuoria:

- Paras laite tai esine on se, joka minulta löytyy omasta takaa. Uuden johdon tai mutterin ostaminen ei ole välttämätöntä, jos sen voi korvata tai kiertää olemassa olevalla asialla. Voi olla, että uusi osa ei toimikaan lopullisessa prototyypissä, eli kannattaa aluksi käyttää sitä mitä on. On myös ympäristöystävällistä kierrättää osia.

- Rakennan laitteen niin, että sen saa purettua ja uudelleenkasattua helposti. Näin vianetsintä helpottuu, ja aiempaan toimivaan vaiheeseen palaaminen nopeutuu. Voin käyttää samoja osia helpommin myös muihin projekteihin.

- Suunnittelen laitteen pieni askel kerrallaan. Aina voi kokeilla yksinkertaisempaa koodia tai kytkentää, jotta vianetsintä helpottuu. Jos vika voi piillä koodissa, laitteistossa tai kolmannen osapuolen ohjelman käyttöönotossa, on hyvä eristää jokin näistä. Pienet askeleet on helpompi dokumentoida, ja niihin voi palata helposti myös tauon jälkeen.

- Maltilliset odotukset. Tämä on kolmas tai neljäs rakentamani prototyyppi, eli en välttämättä saa aikaan juuri sellaista laitetta, minkä prosessin alussa toivoin saavani. Tärkeintä on tehdä toimiva ja valmis laite, ja iloita sen onnistumisesta.

Kuten on havaittavissa, iso osa ohjeista perustuu siihen oletukseen, että laite ei tule heti toimimaan toivotulla tavalla. Jos heti ensimmäinen koodi ja kytkentä toimisi, tämä olisikin varsin lyhyt projekti. Usein eniten aikaa vievin osuus prototyyppien rakentamisesta ja koodaamisesta on selvittää, miksi jokin ei toimi.

2 Sulautetut järjestelmät

Sulautettu järjestelmä on rajattuun tarkoitukseen tehty laite, jonka toimintaa ohjaa pieni tietokone (Karvinen ja Karvinen 2009, 9). Esimerkiksi varhaisimmat matkapuhelimet voidaan laskea sulautetuiksi järjestelmiksi, mutta nykyaikaiset älypuhelinmallit ovat jo hyvin yleiskäyttöisiä tietokoneita, eli määritelmän ulkopuolella. Rumpupadi, joka muuttaa kapulan tai käden iskun MIDI-dataksi on hyvä esimerkki muusikoiden käyttämästä sulautetusta järjestelmästä. (Kuvio 3)

Kuvio 3. Rumpupadi, joka muuttaa iskut MIDI-dataksi. (https://www.keithmcmillen.com/products/boppad/)

Sulautetut järjestelmät ovat niin olennainen osa arkeamme, ettemme välttämättä aina edes huomaa niiden olemassaoloa.

2.1 Esimerkkejä sulautetuista soittimista

Muusikoille tuttuja sulautettuja järjestelmiä ovat esimerkiksi digitaaliset syntetisaattorit ja MIDI-kontrollerit. Näitä järjestelmiä ohjataan perinteisesti erilaisilla näppäimillä, koskettimilla, liu'uilla ja pedaalikytkimillä. Tästä löytyy myös poikkeuksia, joissa teknologian mahdollisuuksia on viety pidemmälle. Kaupallinen esimerkki tästä on Roland-syntetisaattoreista löytyvä *D-Beam* (kuvio 4). D-Beam on teknologia, jossa infrapunasensori muuttaa käden liikkeet dataksi, jolla voi muuttaa äänen virettä tai muita parametreja.

Kuvio 4. D-Beam tulkitsee käden liikkeet ja kontrolloi äänilähteen asetuksia. (https://www.thomann.de/fi/onlineexpert_page_controllers_types_of.html)

Jazzmuusikko Pat Methenyn albumi *Orchestrion Project* on toteutettu käyttämällä mm. MIDI-ohjattuja robotteja. Robotit soittavat akustisia intrumentteja, kuten perkussioita, kielisoittimia ja ksylofonia. Myös Arduinon voi ohjelmoida soittamaan akustisia soittimia yhdistämällä siihen solenoideja tai servomoottoreita (kuvio 5). Solenoidi on ikään kuin sähkömagneettinen iskuri, jota voi käyttää työntämiseen tai vetämiseen. Servo on taas sähkömoottori, jonka liikerata ja nopeus on ohjelmoitavissa. Molempien avulla voidaan rakentaa laite, joka iskee esimerkiksi kellopelin kieliä. Koska Arduinoa voidaan ohjata myös MIDIllä, on mahdollista rakentaa MIDI-ohjattuja akustisia soittimia.

Kuvio 5. Kellopeli, jota soitetaan siihen kiinnitetyillä solenoideilla. (https://create.arduino.cc/projecthub/Netcamprojects/arduino-powered-player-pianos-and-player-xylophones-d4e36d)

Uusia ilmaisutapoja on tutkinut myös projekti *Cave of Sounds.* Se on näyttely, johon on rakennettu instrumentteja, joita soitetaan esimerkiksi kehon liikkeillä. 3D-kamera tallentaa liikkeet komennoiksi, joilla luodaan ääntä. Instrumentit ovat rakentaneet *Music Hackspace*-ryhmän jäsenistä koostuva ryhmä. (Cave of Sounds, 2017)

Kokeilunhaluisia instrumentteja ja kontrollereita suunnitellessa vain mielikuvitus ja tekninen osaaminen on rajana; Arduinoon voidaan yhdistää lähes mitä tahansa määrettä mittaavia antureita. Muutokset ilman tai maan kosteudessa, lämpötilassa ja valon määrässä on mitattavissa sensoreilla. Kaikki mainitut vaihtelut voidaan muuttaa musiikiksi.

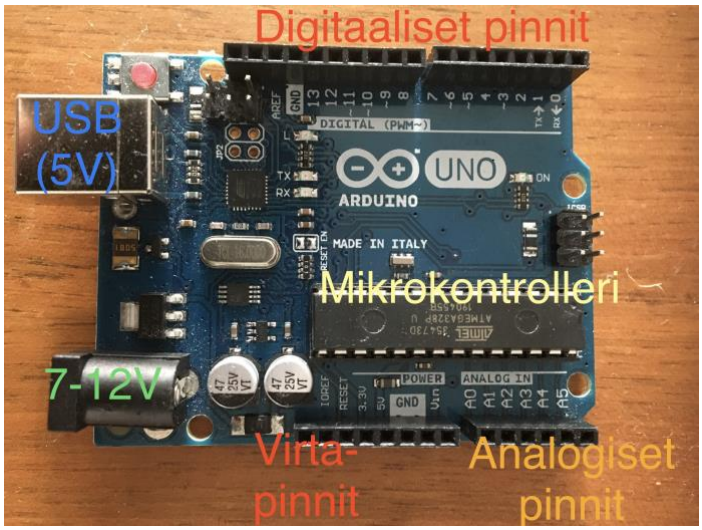
3 Arduino-ympäristö

Arduino on avoimeen lähdekoodiin perustuva kehitysympäristö. Työn suorittamiseen valitsin Arduinon, koska sille luotu ohjelmointiympäristö *Arduino IDE* on verrattaen aloittelijaystävällinen. Arduino on myös melko edullinen ja helposti saatavilla oleva, ja sille löytyy lukuisia valmiiksi kasattuja aloituspaketteja (starter kit) (kuvio 6). Arduinon ohjelmointiin tarvitsee vain tietokoneen, jossa on USB-väylä ja ohjelmointiympäristön, esimerkiksi ilmaisen Arduino IDE (ks. luku 5).

Kuvio 6. Virallinen Arduino Starter Kit sisältöineen. Sisältää osia ja ohjekirjan, jossa on esimerkkiprojekteja aloittelevalle Arduinon käyttäjälle. Myös kolmannen osapuolen settejä on saatavilla. Vaikka esimerkkiprojektit eivät välttämättä ole kiinnostavia kaikille käyttäjille, valmis setti tarjoaa kätevästi valikoiman osia yhdessä paketissa. Arduino-settejä voi löytää myös nettikirpputoreilta. (https://store.arduino.cc/genuino-starter-kit)

3.1 Arduino UNO R3

Arduino Uno on ”Arduinoperheen” referenssimalli. Arduinosta on tehty myös muita malleja erilaisiin sovellutuksiin. Niissä voi olla esimerkiksi sisäänrakennettu bluetooth-yhteys, tai ne voivat olla suunniteltu pienikokoisemmaksi. Arduino Uno (versio R3) valikoitui tähän projektiin, koska se on verrattain edullinen ja helppokäyttöinen, ja siitä löytyy kaikki tarvittavat ominaisuudet midikoskettimiston prototyypin tekemiseen.



Kuvio 7. Arduino UNO R3-kehitysalusta ja sen tärkeimmät liitännät.

Arduino voi saada virtansa suoraan USB-väylän kautta, tai siihen voidaan liittää DC-virtalähde 7-12 V jännitteellä (esimerkiksi paristo). UNO:ssa on 13 digitaalista pinniä, jotka lähettävät ja vastaanottavat jännitteitä 0-5 V välillä. Näistä lähdöistä kuusi toimii PWM-ulostulona. UNO:ssa on viisi analogista sisääntuloa, jotka toimivat myös enimmillään 5 V jännitteellä. Analogiset sisääntulot toimivat 10-bitin tarkkuudella, eli palauttavat arvot 0-1024 välillä. UNO R3:n ”aivoina” toimii ATmega 328-mikrokontrolleri. (Arduino. 2020)

3.2 Arduinon ohjelmointi

Ohjelmointi tarkoittaa ohjeiden antamista tietokoneelle. Ohjeet kirjoitetaan käyttäen ohjelmointikieltä, jonka jälkeen se kasataan ja tulkitaan tietokoneen ymmärtämälle kielelle. Ohjelmointityökaluna käytetään usein ohjelmaa, jota kutsutaan IDE:ksi (Integrated Developement Environment). Tässä alaluvussa esittelen lyhyesti Arduinon ohjelmoinnin periaatteet esimerkkikoodin avulla.

3.2.1 Arduino IDE

Arduinolle on luotu oma, ilmaiseksi ladattava ohjelmointiympäristö Arduino IDE. Ohjelmointi perustuu C++-ohjelmointikieleen. Arduino IDE toimii tekstinkäsittelyohjelmana, jolla voi tarkistaa ja koota kirjoitetun koodin ja lähettää valmis ohjelma Arduino-mikrokontrolleriin. Koska ohjelmointikieli on korkeatasoinen, sen kieliasu muistuttaa englanninkieltä ja on melko selkeästi luettavissa. Arduino IDE:n on

ladattavissa erilaisia kirjastoja (library), jotka helpottavat esimerkiksi erilaisia sensoreita ja MIDI-dataa käsittelevien ohjelmien tekemistä. Kirjastoja voi ajatella ”apukoodina” jossa tiettyjen toimintojen käyttöönottoa on suoraviivaistettu. Esimerkiksi koskettimiston MIDI-koodissa (luku 5.4) ei tarvinnut MIDI-kirjaston ansiosta syöttää MIDI-nuotteja binäärinä (luku 4), koska MIDI-nuotit oli ennalta määritetty. (Edstorm, 2016.)

IDE:ssä on myös koodin kirjoittamista ja lukemista helpottavia toimintoja. Koodin varatut avainsanat on väritetty. Lisäksi koodiin voidaan lisätä kommentteja käyttämällä // tai /* */ merkkejä (kuvio 8). Kommenteissa ohjelmoija voi omin sanoin selostaa mitä koodissa tapahtuu. Kommentit eivät vaikuta ohjelman toimintaan ja ovatkin ikään kuin ”näkymättömiä” Arduino IDE:lle. Ne ovat erittäin hyödyllisiä tilanteessa, jossa ohjelmaa kirjoittaa useampi henkilö ja ne myös toimivat muistiona kirjoittajalle itselleen. Jatkossa tässä työssä selostan käyttämäni koodien toimintaperiaatetta koodin sisällä käyttämällä Arduino IDE:n kommentointiominaisuutta.

3.2.2 Sketch

Arduino IDE:ssä ohjelmoituja ohjelmia kutsutaan nimellä ”*Sketch*”. Sketch:ssä on kaksi perusfunktiota; setup ja loop. Setup-funktio käydään läpi vain kerran Arduinon käynnistyessä, eli siinä alustetaan ohjelma kertaluontoisesti. Loop funktiota Arduino toistaa jatkuvasti, kunnes se sammutetaan. (Karvinen & Karvinen 2006)

Selkeä esimerkki Sketch on Blink-koodi (kuvio 8). Se on tyypillinen ensimmäinen ohjelma, jolla voi tarkistaa, että käytössä oleva Arduino ja tiedonsiirto sen ja tietokoneen välillä toimii.

Kuvio 8. Blink-Sketch, jossa Arduinoon sisäänrakennettua LEDiä välkytetään yhden sekunnin välein. Tämä testikoodi tulee Arduino IDE:n mukana.

Blink-koodissa (kuvio 6.) ensin määritetään ulostuloksi Arduinon sisäänrakennettu LEDi, joka on kytketty digitaaliseen pinniin 13. Tämä määritelmä tarvitsee tehdä vain kerran ohjelman alussa, eli se laitetaan setup-funktioon. Loop-funktiossa Arduino komennetaan sytyttämään sisäänrakennettu LED. Sen jälkeen Arduino odottaa tuhannen millisekunnin eli yhden sekunnin ajan. Sitten sammutetaan LED jälleen yhden sekunnin ajaksi. Tämän jälkeen loop alkaa uudestaan ja se jatkuu, kunnes Arduino sammutetaan.

4 MIDI

MIDI on initialismi sanoista Musical Instrument Digital Interface. Se on 1980-luvulla kehitetty protokolla, jolla musiikkiesitys voidaan välittää ja tallentaa digitaalisessa muodossa. MIDI-data ei sisällä itsessään ääntä, vaan tiedon musiikin piirteistä, kuten sävelen, nuottien väliset aika-arvot, äänenvoimakkuuden ja vireen. MIDI-datalla voidaan myös ohjata yhteensopivan laitteiston toimintaa, esimerkiksi syntetisaattorin alipäästösuotimen asetuksia. (Edstrom. 2016.) MIDI-dataa voidaan siirtää esimerkiksi 5-pinnisellä MIDI-kaapelilla, USB- tai Bluetooth-yhteyden kautta. (Kuvio 9).

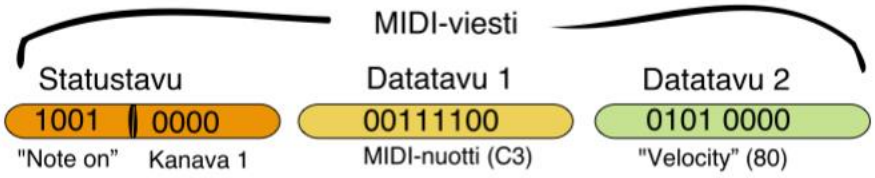
Kuvio 9. MIDI-liitäntöjä. Etualalla 5-pinninen MIDI-kaapeli.

Lisäksi haluan huomauttaa, että empiirisen kokemukseni pohjalta MIDI-laitteistot ja ohjelmistot käyttävät aina vakiintuneita englanninkielisiä nimityksiä eri komennoista. Sen vuoksi pyrin näyttämään komennot siinä muodossa kuin ne ovat, enkä välttämättä tarjoa suomennettua vastinetta. Ymmärrettävyyden takaamiseksi uutta termiä esitellessä selitän, mitä mikäkin komento tarkoittaa. Näin kirjoittamaani on suoraan sovellettavissa MIDI-laitteiden kanssa työskenneltäessä.

4.1 MIDI-komennot

MIDI on muodoltaan 8-bittisiä binääritavuja. 8-bittisessä binääritavussa kahdeksan peräkkäistä numeroa (bittiä) muodostaa lukujonon, joista jokainen luku on joko nolla tai yksi. Yksi 8-bitin sarja muodostaa tavun, ja tavuja MIDI-viestissä voi olla yhdestä kolmeen.

MIDI-viestit voi jakaa karkeasti kahteen ryhmään; kanavakohtaiset tai järjestelmää ohjelmoivat viestit. Koska projektityössäni käsittelen kanavakohtaisia viestejä, kuvailen niiden toimintaperiaatteen tarkemmin. MIDI-viestin ensimmäinen tavu on statustavu (statusbyte), joka määrittää millainen midiviesti on kyseessä. Statustavun neljä ensimmäistä numeroa määrittää viestin pääkomennon. Se voi olla esimerkiksi ”Note-on” tai ”Note-Off”, jolla soitetaan ääni tai lopetetaan sen soittaminen. Statustavun viimeiset neljä numeroa määrittävät käytössä olevan kanavan.



Kuvio 10. Esimerkki MIDI-viestistä, jossa kanavalta yksi soitetaan C3-nuotti ”velocity”-arvolla 80.

Datatavuja on viestistä riippuen yksi tai kaksi. ”Note on”-viestissä datatavu 1 määrittää mikä nuotti soitetaan. Datatavu 2 taas määrittää ”Velocity”-arvon, eli miten millä voimakkuudella ääni toistetaan.

Lisäksi on hyvä huomioida, että statustavun ensimmäinen merkki on aina 1. Datatavujen ensimmäinen numero taas on 0. Tästä voidaan laskea, että statustavun ensimmäisellä puolikkaalla voi olla 2^3 eli kahdeksan eri arvoa. Tavun jälkimäisellä puoliskolla on taas 2^4 arvoa, eli MIDI-kanavia on käytettävissä 16. MIDI-nuotteja ja ”Velocity”-arvoja on 2^7 eli 128 (0-127). (Earlham College, n.d) Arduinon voi ohjelmoida lähettämään sarjaportin kautta binääriä, jonka voi tulkita MIDI-muotoon siihen soveltuvalla ohjelmalla. (Luku 6.)

5 Midikoskettimiston rakentaminen

Tässä luvussa kuvailen projektityöni rakentamisprosessia. Käsittelen koskettimiston laitteiston toimintaperiaatetta ja ohjelmoin Arduinon lähettämään sarjadataa tietokoneelle.

5.1 Laitteisto ja työkalut

Työhön tarvittavat välineet:

- Arduino kehitysalusta
- Kaapeleita (hyppykaapelit)
- Yleismittari
- Juotin ja vapaavalintaiset tarvikkeet (mm. tinaa, kärjenpuhdistin)
- Koskettimisto (itse rakennettu tai valmis)
- Tietokone, jolla ohjelmoida
- USB-kaapeli
- Tarkoitukseen sopivat kotelointitarvikkeet

5.1.1 Koskettimisto

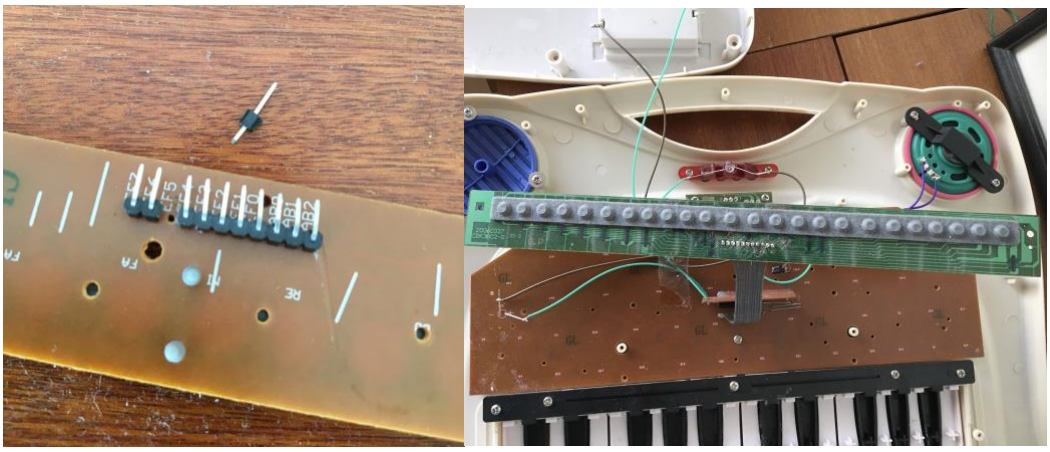
Ennen kuin on aika käyttää Arduinoa, täytyy suunnitella, rakentaa tai hankkia koskettimisto. Koska tarkoituksenani on rakentaa midikoskettimisto, jossa on pianon koskettimet, päätin etsiä sopivan laitteen kirpputorilta. Sopiva laitteen pohja löytyikin leluosastolta. Laite oli toimiva, ja kokeilin, että jokaista kosketinta painamalla laite toisti eri äänen kahden oktaavin alueelta. Tästä oli pääteltävissä, että laite on kytketty niin, että jokainen kosketin on kytketty virtapiiriin erikseen. Näin ollen pystyn myös lähettämään Arduinolle oman ”viestin” jokaiselta koskettimelta.

Kuvio 11. Kosketinsoitinlelu. Kirpputorit ja kierrätyskeskukset ovat hyviä paikkoja löytää elektroniikkaleluja, joista saa edullisia osia Arduino-projekteihin.

Paristojen irrottamisen jälkeen on aika avata laite ja tutkia kuinka se toimii.

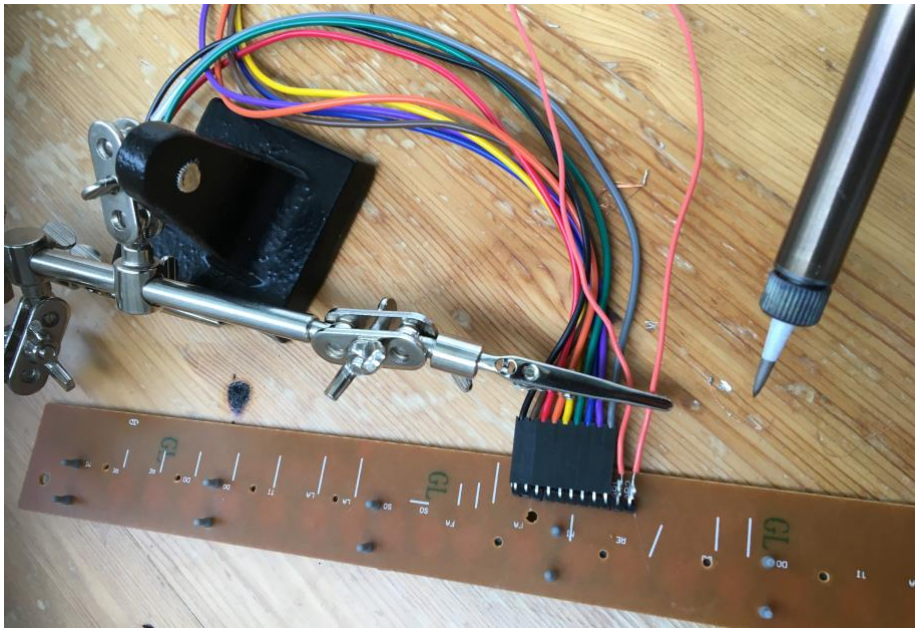
5.1.2 Koskettimiston uudelleenkytkentä

Koskettimiston irrotettua alta paljastui piirilevy, jossa on rivissä 24 painettavaa kytkintä (kuvio 7.). Jokaisen muovisen ”nyppylän” alla on sähköä johtavaa materiaalia, joka sulkee virtapiirin ollessaan kontaktissa piirilevyn kanssa. Koskettimisto on kiinnitetty lelun mikropiiriin 11 jäykähköllä johdolla, joita on vaikeaa hyödyntää sellaisenaan. Siksi irrotin vanhan kaapelin koskettimen piirilevystä juottimen avulla. Kaapeleiden jättämien reikien tilalle juotin 11-rivisen piikkiriman (Kuvio 8).



Kuvio 12. Alkuperäiset kaapelit korvattu piikkirimalla (vasemmalla). Piikkirimoihin voi juottaa jatkokaapelin, tai liittää naaras-hyppykaapelin.

Omasta takaa minulta löytyi yhdeksän yhdistettyä naaras-naaras-hyppykaapelia, joten juottaa tarvitsi vain kaksi jäljelle jäävää kaapelia.

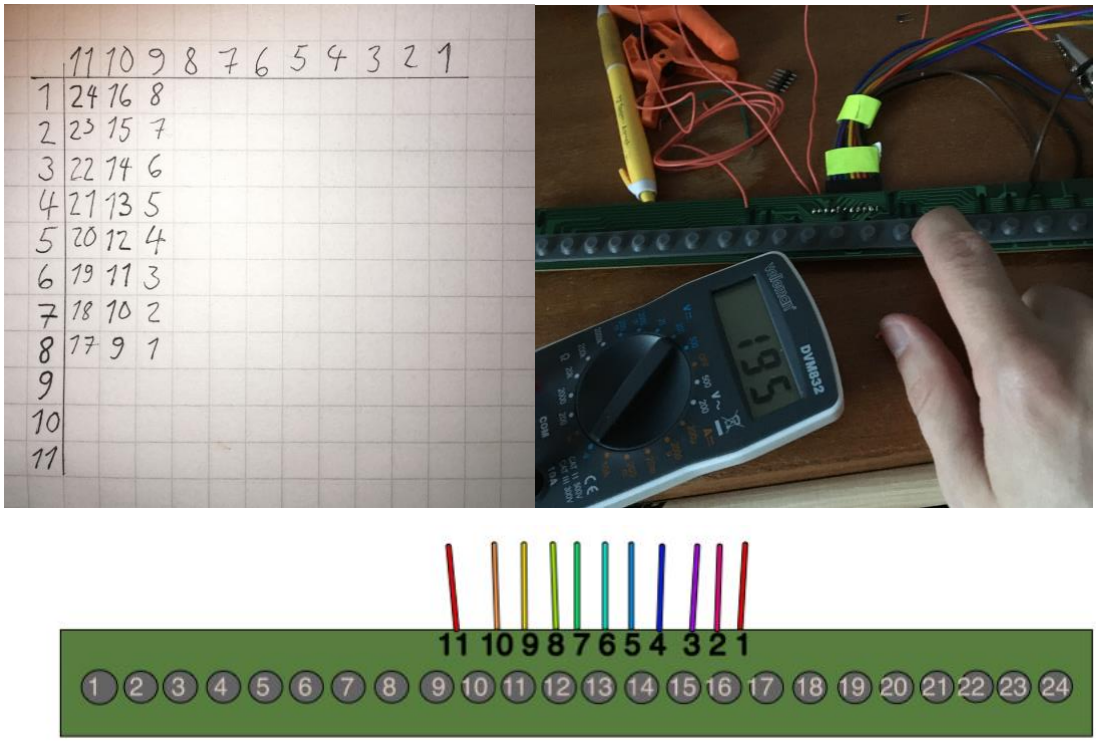


Kuvio 13. Koskettimiston uusi kytkentä. Kytkeminen ja viantesintä helpottuu, kun lähes kaikki kaapelit ovat kahta viimeistä lukuun ottamatta eri väriä. Naaras-hyppykaapelit pysyvät kiinni tukevasti, mutta ovat helposti irrotettavissa ilman juotinta.

5.1.3 Takaisinmallinnus ja Matriisi

24 koskettimen koskettimisto on liitetty piirilevyyn 11 johdolla. On siis selvitettävä, kuinka kytkentä toimii. Tämän voi tehdä yleismittarin avulla. Yleismittari asetetaan johtavuutta ilmaisevaan asetukseen. Valitaan kaksi johtoa, ja koskettimia painamalla selvitetään,

sulkeeko jokin painallus virtapiirin. Tulos on hyvä kirjoittaa ylös, jotta mahdollinen kaava paljastuu. (EvanKale n. d.)



Kuvio 14. Matriisi. Vaaka- ja pystyrivit näyttävät mitkä kaapelit on yhdistetty (1-11). Sisällä oleva numero 1-24 kertoo, mikä kosketin sulkee virtapiirin. Hauenleukapäät yleismittarin johdoissa vapauttaa kädet koskettimien painamiseen. Esimerkiksi, johdot numero kahdeksan ja yhdeksän on kytketty niin, että virtapiirin sulkee kosketin numero 1. Muutaman johtoyhdistelmän löydyttyä kaava selvisi nopeasti, eikä jokaista mahdollista kombinaatiota tarvinnut kokeilla.

Kyseinen koskettimisto on kytketty 3x8 matriisiin. Matriisissa siis on kahdeksan riviä ja kolme saraketta. Matriisia käyttämällä saadaan aikaan kytkentä, joka mahdollistaa tässä tapauksessa 24 koskettimen liittämisen vain 11 kaapelilla.

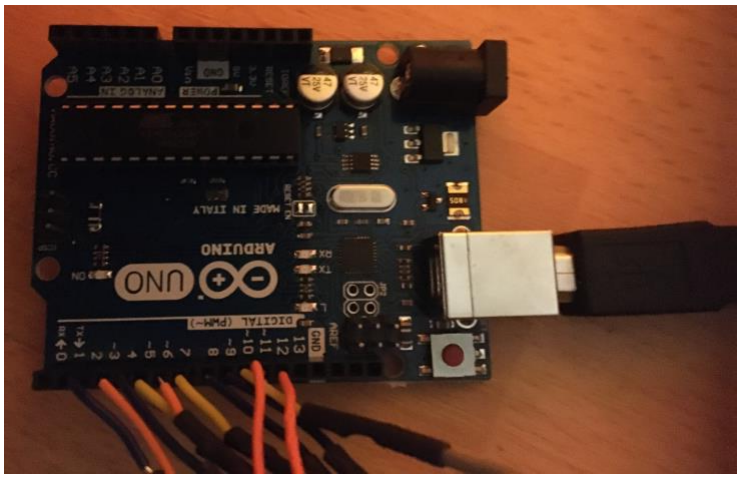
5.2 Sarjaportti ja -monitori

Arduino IDE:ssä on käytettävissä sarjamonitori. Jokaisessa Arduinossa on vähintään yksi sarjaportti, jonka kautta Arduinon voi ohjelmoida lähettämään viestejä esimerkiksi tietokoneelle USB-yhteyden kautta. Sarjamonitorin käyttö voi auttaa esimerkiksi tilanteissa, joissa Arduino IDE lataa koodin Arduinoon ongelmitta, mutta laite ei toimi toivotulla tavalla. Hyödynnän sarjamonitoria ensin prototyyppiohjelmassa (kappale 5.3.) jossa tutkin, että Arduinoon liittämäni koskettimet toimivat halutulla tavalla. Seuraavaksi

hyödynnän sarjaporttia lopullisessa laitteessa, jossa lähetän sarjamuotoista dataa Arduinon sarjaportista MIDI-sarjakoodi ”tulkille”. (kappale 6.) Sarjamonitorin käyttö on siis hyödyllistä virheiden etsinnässä (engl. debugging) ja sarjaportin kautta Arduino voi lähettää ja vastaanottaa haluttuja komentoja. On hyvä ottaa huomioon, että sarjaportti on käytettävissä kerrallaan vain yhteen toimintoon; jos sarjaportti on yhdistetty johonkin toiseen ohjelmaan, uusien Sketch:ien lataaminen Arduinoon ei onnistu. Arduino UNO:n kaksi ensimmäistä digitaalista pinniä ovat myös sarjaportin käytössä, joten niidenkin on oltava vapaana uutta ohjelmaa ladattaessa. (Arduino Reference 2019).

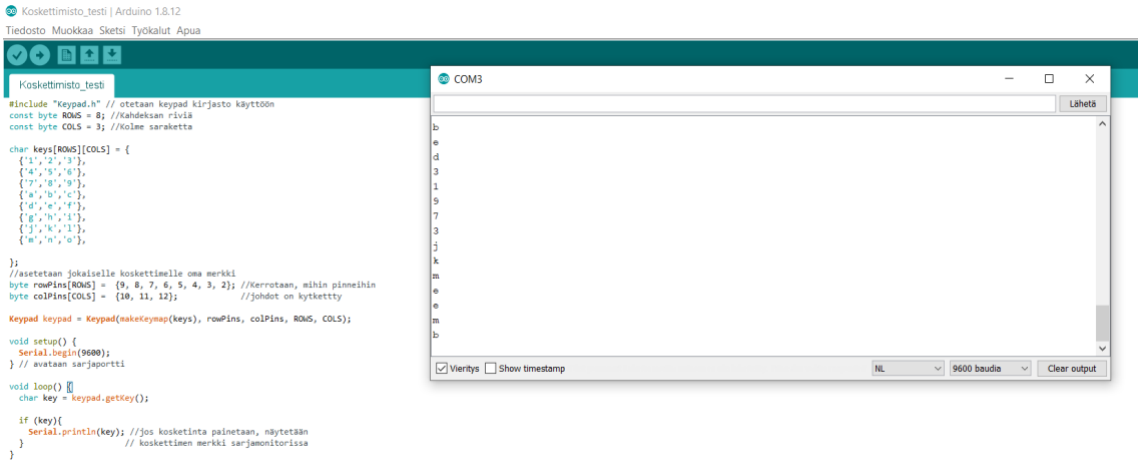
5.3 Prototyyppikoodi

Koskettimiston piirilevy on nyt valmis liitettäväksi Arduinoon. Kytkin piirilevystä lähtevät 11 kaapelia Arduinon digitaalisiin pinneihin 2-12 (kuvio 13).



Kuvio 15. Koskettimistosta lähtevät kaapelit kytkettynä Arduinon digitaalisiin pinneihin.

Tein ohjelman, jonka avulla voi tarkistaa toimiiko koskettimiston kytkentä, ja saako Arduino lähetettyä dataa tietokoneelle sarjaportin kautta. Ohjelma on tehty Arduino IDE:llä hyödyntäen Keypad-kirjastoa. Koodin pohja löytyi Arduino-foorumilta.



Kuvio 16. Koodi, jossa jokainen kosketin lähettää sarjaporttiin oman merkkinsä. Arduino IDE:n sarjamonitorin (oikeanpuoleinen ikkuna) avaamalla voi tarkistaa, että jokainen kosketin lähettää oman merkin.

Nyt jokainen kosketin lähettää oman merkkinsä sarjamonitoriin. Tällä tavalla on hyvä tarkistaa, että laitteen elektroniikka on kytketty oikein. Tähän ohjelmaan voi aina tarvittaessa palata, jos myöhemmissä versioissa on ongelmia ohjelmiston tai laitteiston kanssa.

5.4 MIDI-koodi

MIDI-koodissa oli hyödynnettävissä keypad-kirjasto (luku 5.3.). Käytännössä valmis koodi löytyi Arduino-foorumilta (Kuvio 9). (Grumpy Mike, 2019)

Midikoodi_kosketin §

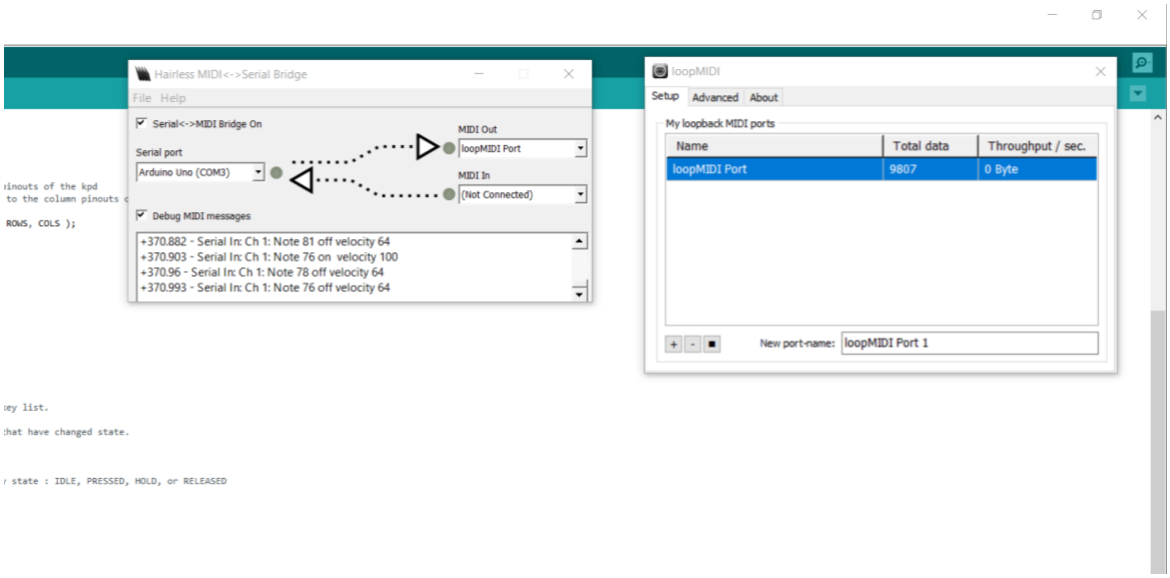
```
#include <MIDI.h> //otetaan käyttöön MIDI-kirjasto
#include <Keypad.h> //otetaan käyttöön Keypad-matriisikirjasto

const byte ROWS = 3; //kolme riviä
const byte COLS = 8; //kahdeksan saraketta


char keys[COLS][ROWS] = {

  { 71, 70, 69 },
  { 68, 67, 66 },
  { 65, 64, 63 },
  { 62, 61, 60 },
  { 59, 58, 57 },
  { 56, 55, 54 },
  { 53, 52, 51 },
  { 50, 49, 48 }
};                  //määritetään matriisi


byte rowPins[ROWS] = {12, 11, 10};
byte colPins[COLS] = {2, 3, 4, 5, 6, 7, 8, 9}; //määritetään kytketkyt digi-pinnit

Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

byte pressed = 32;
byte chanel = 0; //Midikanava

void setup() {
 Serial.begin(115200); // Baudi-arvo (koodin "näytteenottotaajuus", asetetaan samaksi Hairless-ohjelmassa)
}


void loop() {
 // Tämä koodi toistuu jatkuvasti (loop)
 // loop funktio tarkkailee muutoksia koskettimien tilassa ja lähettää viestejä sarjaporttiin
 if (kpd.getKeys())
 {
   for (int i = 0; i < LIST_MAX; i++) // Käydään läpi kaikki koskettimet
   {
     if ( kpd.key[i].stateChanged )   // Etsitään koskettimia, joiden olotila on muuttunut
     {

       pressed = kpd.key[i].kchar + 12;
       switch (kpd.key[i].kstate) {  // Ilmoitetaan Koskettimen olotila; painettu tai vapautettu)
         case PRESSED:
           sendMIDI(chanel | 0x90, pressed, 100);
           break;
          case RELEASED:
           sendMIDI(chanel | 0x80, pressed, 64);
           break;
       }
     }
   }
 }
}  // loopin loppu

void sendMIDI(byte type, byte note, byte velocity){
  Serial.write(type);
  Serial.write(note & 0x7F);
  Serial.write(velocity);  //lähetetään midiviesti sarjaportin kautta
}
```

Kuvio 17. IDE-koodi

Koodissa (kuvio 15.) määritellään näppäinten matriisi, ja olen antanut jokaiselle koskettimelle oman MIDI-nuottinsa. Sarjaportti aktivoidaan, jotta sarjakoodin lähettäminen Arduinosta eteenpäin on mahdollista. Koodin jatkuvasti toistuva osio (loop) tarkkailee koskettimien tilaa. Jos koskettimen status muuttuu, siitä lähetetään sarjakoodilla viesti, jossa kerrotaan nykyinen tila, mikä kosketin on kyseessä ja millä voimakkuudella (velocity) se toistetaan. Koskettimien midinuotit on annettu lukuina MIDI-protokollan mukaan (Inspired Acoustics. 2020).

6 Sarjakoodi midiksi

Koska Arduino UNO R3 ei sellaisenaan lähetä dataa MIDI-formaatissa, tarvitaan ohjelma, joka kääntää Arduinon viestit midisignaaliksi. Tähän on olemassa ilmainen ohjelma ”*Hairless Midi<->Serial Bridge*”. Lisäksi Windows-käyttöjärjestelmälle tarvitaan ohjelma, jonka avulla Hairless-ohjelma nähdään midiporttina DAW:ssa. Tähänkin löytyy ilmainen ratkaisu, *Loopmidiport*. Näiden ohjelmien avulla MIDI-viesti välittyy ja on tulkittavassa muodossa MIDI-instrumentille. Loopmidiport:in voi ajatella olevan ”virtuaalinen MIDI-kaapeli” (Switch & Lever, 2019).

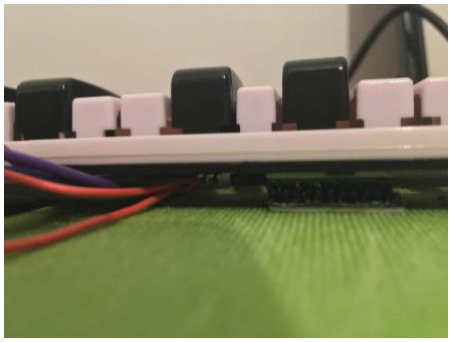


Kuvio 18. Hairless MIDI<->Serial Bridge tulkitsee sarjakoodin MIDI-muotoon ja lähettää sen loopMidi ohjelmalle. LoopMIDI:n voi nähdä MIDI-sisääntulona esimerkiksi sekvensseriohjelmassa.

Edellä mainitut ohjelmat on testattu Windows 10-käyttöjärjestelmällä.

7 Valmis prototyyppi

Kun koodi ja laitteisto on saatu toimimaan oikein, on aika suunnitella laitteen ulkoasu. Koska kyseessä ensimmäinen prototyyppi, asensin osat niin, että osat ja niiden kytkennät ovat näkyvillä, mutta tarvittaessa helppo poistaa. Prototyypin ei tarvitse olla visuaalisesti näyttävä, vaan toimiva, käytännöllinen ja helposti muokattavissa oleva. Pohjaksi valikoitui vanha leikkuulauta, jossa oli riittävästi tilaa laitteistolle ja varaa laajentamiseen tulevaisuudessa. Laitteiden kiinnittämiseen käytin Dual Lock-tarranauhaa, joka on muusikoille tuttu pedaalilaudan kiinnitysmateriaalina. Dual Lock pitää osat tiukasti paikallaan, mutta se on myös helposti irrotettavissa tarpeen vaatiessa.



Kuvio 19. Dual Lock kiinnitti koskettimiston leipälautaan tukevasti. Tarranauha nosti koskettimistoa juuri sopivasti, jotta piirilevyn johdot mahtuivat kulkemaan sen alapuolelta. Tämä kiinnitys ei välttämättä kestä vuosia kovaa käyttöä, mutta antaa mahdollisuuden muokata ja testata laitetta nopeasti.

Koska johtojen pituutta ei ole sovitettu juuri tätä sovellutusta varten, ne tekevät hieman epäergonomisen mutkan. (Kuva 8.) Kyseinen seikka ei kuitenkaan vaaranna laitteen toimintaa, ja on helposti muokattavissa (luku 1.3).

Kuvio 20. Valmis laite. Vaikka leikkuulaudan integroitu käsikahva mahdollistaa laitteen kuljetuksen, en vielä sellaisenaan ottaisi koskettimistoa mukaan keikalle. Laite kuitenkin pysyy kasassa ja on siirreltävissä sellaisenaan.

Laitteesta tuli käyttökelpoinen; koskettimiston tuntuma on mekaanisesti hyvä. Laite rekisteröi painallukset luotettavasti ainakin yksi ääni kerrallaan soitettuna.

8 Yhteenveto ja pohdinta

Työni tavoite oli esimerkkiprojektin avulla osoittaa, että oman elektronisen instrumentin tai musiikkikäyttöön tarkoitetun sulautetun järjestelmän rakentaakseen ei tarvitse olla insinööri, vaan kuka tahansa teknologiasta kiinnostunut voi rakentaa oman laitteen. Tee-se-itse-yhteisöjä tutkimalla niin YouTube-videoiden kuin foorumeiden avulla löytää paljon tietoa ja apua projekteihin. Tietoa on itseasiassa niin paljon, että haasteena on löytää oikea hakusana. Tämän projektin rakensin soveltamalla tietoa, jonka löysin hakemalla Google-palvelusta hakusanalla ”arduino midi keyboard matrix”.

Rakentamani kontrolleri täytti sille asettamani vähimmäisvaatimukset; pystyn soittamaan MIDI-ohjelmistosyntetisaattoreita. Sain aikaan toimivan prototyypin, josta voi jatkossa kehittää ilmaisuvoimaisemman MIDI-kontrollerin.

Heikkouksiakin laitteellani on. Koska jokaisella koskettimella on vain yksi virtapiirin sulkeva ”nappi”, voin tällä hetkellä viestiä Arduinolle, että onko kosketin ”päällä” vai ”pois päältä”. Jos yksi kosketin sulkisi painettaessa kaksi erillistä virtapiiriä, niiden sulkeutumisen välisen ajan voisi muuttaa äänen voimakkuuden (velocity) arvoksi. Jotta saisin lisää dynamiikkaa nykyiseen koskettimistooni, voisin kytkeä Arduinoon esimerkiksi etäisyyttä mittaavan sensorin, jonka antaman arvon perusteella voisin kontrolloida MIDI-viestin velocity-arvoa. Voisin säätää velocity-arvoa myös passiivisella potentiometrillä.

Lisäksi haluaisin tuoda ilmi, että itserakennettuihin sähkövirralla toimiviin laitteisiin kannattaa suhtautua varovaisuudella. Katkaisen virran rakentamastani prototyypistä aina, kun en pysty tarkkailemaan sen toimintaa. Vaikka Arduino-projekteissa yleensä käytetään verkkovirrasta huomattavasti pienempiä jännitteitä, on osien kärähtäminen hyvinkin mahdollista elektroniikan väärinkytkennän vuoksi. Jos Arduino on kytketty tietokoneeseen vian tapahtuessa, teoriassa myös tietokone voi vahingoittua. Tämän vuoksi en ole kokeillut prototyyppejä pääasiallisella työkoneellani, vaan päivittäisestä käytöstä vapautuneella vanhemmalla tietokoneella.

Miksi valmistaa MIDI-koskettimisto itse, kun käytännöllisempiä, sähköturvallisempia ja monipuolisempia massatuotettuja laitteita on saatavilla jopa alle käyttämieni osien ja työkalujen hinnalla? Tähän yhtenä vastauksena on kiinnostus teknologiaa kohtaan. Kun yrittää rakentaa tarvitsemansa laitteen itse, saa samalla syvällisen käsityksen siitä, miten se toimii. Ekologisesta näkökulmasta katsottuna, etenkin jos hankkii osia käytettynä, on mahdollista säästää luonnonvaroja. Laitteen voi myös suunnitella niin, että siihen käytetään mahdollisimman vähän ylimääräisiä osia. Itse suunniteltuihin prototyyppeihin ei ole välttämätöntä asentaa ominaisuuksia, joita ei tarvitse. Rakentamani prototyyppi onkin melkein yksinkertaisin mahdollinen USB-MIDI-koskettimisto, johon voi tulevaisuudessa lisätä ominaisuuksia tarpeen mukaan.

Projektityön aikana sain paljon uutta tietoa elektroniikasta ja digitaalisesta teknologiasta. Vaikka minulla on vuosien käyttökokemus MIDI-laitteista, vasta työn teoriasuutta kirjoittaessa aloin ymmärtää MIDI-protokollan toimintaperiaatteen perusteita. Nyt kun

ymmärrän enemmän MIDI-viestien koostumuksesta, voin suunnitella jatkossa monipuolisempia laitteita ja soveltaa oppimaani.

Huomionarvoista on, että harrastajan lähtökohdasta kykenin käyttämieni lähteiden avulla rakentamaan toimivan laitteen, vaikka teoreettisessa osaamisessani olikin ja onkin vielä syvennettävää. Tämän toivoisin rohkaisevan myös lukijan tarttumaan kolviin, ja tekemään projekteja. Itse MIDI-kontrollerin rakentaminen ja ohjelmointi onnistui yllättävän nopeasti oikeiden lähteiden löydyttyä, kun taas toimintaperiaatteiden sanallistaminen ja ymmärtäminen vaati aikaa ja opiskelua.

Rakentaessani toteutin ”rakentamisen filosofiaa” (luku 1.3.) jota itselleni määrittelin. Lähestyin ongelmia osa kerrallaan. Esimerkiksi en heti ensimmäiseksi yrittänyt ohjelmoida Arduinoa lähettämään MIDI-viestejä, vaan integroin kytkettyä laitteistoa ja tiekonetta vähitellen testikoodin avulla. Rakentamani koskettimiston kehikko, johon kiinnitin osat Dual Lock-tarranauhalla ei ole visuaalisesti näyttävä, mutta se on helposti muokattava ja pitää laitteen kasassa. Toteutin projektin hyödyntämällä materiaaleja, joita olen ostanut sekä netti- että kivijalkakirpputoreilta.

Varsinaisestihan en rakentanut mitään, oikeampi termi olisi ehkä ”kasata”. Yhdistin olemassa olevia laitteita Arduinoin avulla niin, että sain laitteen, jolla voi soittaa Midi-instrumentteja tietokoneelta. Se onkin ehkä Arduinon suurin vahvuus; siihen voi yhdistää lähes mitä tahansa elektronisia laitteita, jotka se saa kommunikoimaan keskenään reaaliajassa. Elektronisen musiikin esittämiseen itse rakennetut laitteet mahdollistavat uusia ilmaisun tapoja, jotka eivät ole välttämättä saavutettavissa kaupallisilla laitteilla.

Lähteet

Arduino Reference. 2019. Serial. Verkkosivu. Viitattu 19.11.2020 https://www.arduino.cc/reference/en/language/functions/communication/serial/.

Arduino Uno R3 FAQ. 2020. Verkkosivu. Viitattu 20.11.2020 https://store.arduino.cc/arduino-uno-rev3.

Cave of Sound. 2017. Verkkosivu. Viitattu 01.12.2020 https://caveofsounds.com

Earlham College. n. d. About MIDI and binary numbers. Viitattu 21.11.2020 http://legacy.earlham.edu/~tobeyfo/musictechnology/1b_BInaryNumbers_edit.html.

Edstrom, Brent. 2016. Arduino™ for Musicians: A Complete Guide to Arduino and Teensy Microcontrollers.

EvanKale (nimimerkki) Figuring Out a Key Matrix (Scan Matrix) (n. d.) Ohje. Viitattu 21.11.2020 https://www.instructables.com/Figuring-out-a-Key-Matrix-Scan-Matrix/#discuss.

Grumpy Mike (nimimerkki) 2019. The project is done (Old 49 keys casio) to midi using Arduino MEGA. Puheenvuoro Arduino-keskusteluyhteisössä 29.5.2019 https://forum.arduino.cc/index.php?topic=615932.60

Helsingin Kaupunki. n. d. Verkkosivu. Viitattu 22.11.2020 https://kehmet.hel.fi/menetelmalaari/pdca-sykli/.

Inspired Acoustics. MIDI Note Numbers and Center Frequencies. https://www.inspiredacoustics.com/en/MIDI_note_numbers_and_center_frequencies Viitattu 20.11.2020

Karvinen, Tero & Karvinen, Kimmo. 2009. Sulautetut: Opi rakentamaan robotteja ja muita sulautettuja järjestelmiä.

Smith, Warvick. 2016. C programming with Arduino.

Switch & Lever (nimimerkki). 2019. Building a MIDI controller using Arduino. YouTube-video, 10:46, julkaistu 3.12.2019 https://www.youtube.com/watch?v=JZ5yPdoPooU&t=675s

Painalluskytkin matriisikoodi

```
#include "Keypad.h" // otetaan keypad kirjasto käyttöön

const byte ROWS = 8; //Kahdeksan riviä

const byte COLS = 3; //Kolme saraketta



char keys[ROWS][COLS] = {

 {'1','2','3'},

 {'4','5','6'},

 {'7','8','9'},

 {'a','b','c'},

 {'d','e','f'},

 {'g','h','i'},

 {'j','k','l'},

 {'m','n','o'},



};

//asetetaan jokaiselle koskettimelle oma merkki

byte rowPins[ROWS] =  {9, 8, 7, 6, 5, 4, 3, 2}; //Kerrotaan, mihin pinneihin

byte colPins[COLS] =  {10, 11, 12};            //johdot on kytkettty
```

```
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);



void setup() {
  Serial.begin(9600);
} // avataan sarjaportti



void loop() {
  char key = keypad.getKey();



  if (key){
    Serial.println(key); //jos kosketinta painetaan, näytetään
  }                    // koskettimen merkki sarjamonitorissa
}
```

MIDI-koskettimiston koodi

```
#include <MIDI.h> //otetaan käyttöön MIDI-kirjasto

#include <Keypad.h> //otetaan käyttöön Keypad-matriisikirjasto



const byte ROWS = 3; //3 Riviä

const byte COLS = 8; //8 saraketta




char keys[COLS][ROWS] = {


  { 71, 70, 69 },

  { 68, 67, 66 },

  { 65, 64, 63 },

  { 62, 61, 60 },

  { 59, 58, 57 },

  { 56, 55, 54 },

  { 53, 52, 51 },

  { 50, 49, 48 }

};                  //määritetään matriisi
```

```
byte rowPins[ROWS] = {12, 11, 10}; //connect to the row pinouts of the kpd

byte colPins[COLS] = {2, 3, 4, 5, 6, 7, 8, 9}; //connect to the column pinouts of the kpd



Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );



byte pressed = 32;

byte chanel = 0; // MIDI-kanava



void setup() {

 Serial.begin(115200); // Baudi-arvo (koodin "näytteenottotaajuus", asetetaan samaksi Hairless-ohjelmassa)

}



void loop() {

 // Tämä koodi toistuu jatkuvasti (loop)

 // Koodi etsii muutoksia koskettimien tilassa
```

```
if (kpd.getKeys())

{

  for (int i = 0; i < LIST_MAX; i++) // Käydään läpi kaikki koskettimet

  {

    if ( kpd.key[i].stateChanged )   // Etsitään vain koskettimia, joiden olotila on muuttunut

    {



      pressed = kpd.key[i].kchar + 12;

      switch (kpd.key[i].kstate) {  // Ilmoitetaan Koskettimen olotila; pressed, released, idle,
held)

        case PRESSED:

          sendMIDI(chanel | 0x90, pressed, 100);

          break;

        case RELEASED:

          sendMIDI(chanel | 0x80, pressed, 64);

          break;

      }

    }

  }
```

```
 }

}  // End loop



void sendMIDI(byte type, byte note, byte velocity){

  Serial.write(type);

  //Serial.println(note);

  Serial.write(note & 0x7F);

  Serial.write(velocity);  //lähetetään koskettimen äänen olotila, velocity ja nuotti

}
```