STM32-mikrokontrolleri robotiikka-alustana

Joni Niemi

Haaga-Helia ammattikorkeakoulu

AMK-opinnäytetyö

2021

Tietojenkäsittelyn tutkinto

Tekijä(t)
Joni Niemi

Tutkinto
Tradenomi

Raportin/opinnäytetyön nimi
STM32-mikrokontrolleri robotiikka-alustana

Sivu- ja liitesivumäärä
41 + 5

Opinnäytetyön tavoitteena oli tutkia mikrokontrollereita, sulautettuja järjestelmiä ja varsinkin kehityspiirejä, ja niiden ohjelmointia. Näiden tutkimusten perusteella suunniteltiin etäohjattava robotti.

Työn teoriaosuudessa käsiteltiin ensin nykyaikaisia toimintaympäristöjä, jakamistaloutta, avoimen lähdekoodin sekä c++-ohjelmointikielen merkitystä niissä. Tämän lisäksi tutkittiin myös yhden piirin tietokoneita, järjestelmäpiirejä ja sulautettuja järjestelmiä.

Seuraavassa osassa tutkittiin tarkemmin kehityspiirejä ja -alustoja. Ensin vertaillaan kolmea suosittua kehitysalustaa, jotta ymmärretään niiden käyttötarkoituksia, käyttöä, suoritustehoa, hintaa sekä aloittelijaystävällisyyttä. Alustat ovat Arduino, ESP32 ja STM32.

Viimeinen osio koostuu robotin toteutusta. Osiossa käsitellään suunnittelun haasteet ja rajoitteet, robotin 3D-mallinnus ja tulostus, robotin kokoaminen ja ohjelmointi toteutettiin.

Opinnäytetyön lopputuloksena syntyi näkemys sulautettuihin järjestelmiin, kuinka niitä ohjelmoidaan ja miten ne yhdistetään robotiikkaan.

Produktina opinnäytetyölle syntyi enimmäkseen itse suunniteltu, mallinnettu, 3D-tulostettu, koottu ja ohjelmoitu robotti. Tuotoksen 3D-malliin voi perehtyä Thingiverse sivustolla osoitteessa https://www.thingiverse.com/thing:5138756 ja lähdekoodiin pääsee perehtymään osoitteessa https://github.com/muvox/tiny_tyger_robot

Asiasanat
mikropiirit, ohjelmointi, robotit, kehitysalustat, 3D-tulostus

Sisällys

1 Johdanto....1
2 Käsitteet....2
3 Nykyaikaiset toimintaympäristöt....4
3.1 C++ ohjelmointikieli....4
3.2 Avoin lähdekoodi....5
3.3 Yhden piirin tietokoneet, järjestelmäpiirit ja sulautetut järjestelmät....6
3.4 Jakamistalous....7
4 Kehityspiirit ja alustat....9
4.1 Arduino....9
4.2 ESP32....11
4.3 STM32....12
4.4 Vertailu....13
4.4.1 Käyttötarkoitus....13
4.4.2 Suoritusteho....14
4.4.3 Hinta ja saatavuus....15
5 Robotin toteutus....16
5.1 Robotin suunnittelu....16
5.2 Muut komponentit....16
5.2.1 Virtalähde....16
5.2.2 DC DC-alasmuunnin....17
5.2.3 Latausmoduuli....17
5.2.4 Moottoriohjain....18
5.2.5 Moottorit....19
5.2.6 BLE-moduuli....19
5.3 Robotin rakennus....20
5.3.1 3D-malli....20
5.3.2 3D-tulostus....23
5.3.3 Kokoaminen....25
5.4 Ohjelmointi....27
5.4.1 Etäohjaus sovellus....27
5.4.2 Kehitysympäristön pystyttäminen....29
5.4.3 Pinnien määrittely, setup funktio ja loop funktio....30
5.4.4 BLE-yhteys....32
5.4.5 Moottoriohjaus....32

5.4.6 Lopullinen ohjelmaa....33
6 Pohdinta....35
6.1 Robotin jatkokehitys....36
Lähteet....38
Liitteet....42
Liite 1. STM32F4x1xC pinnikaavio....42
Liite 2. Robotin lähdekoodi – Main.cpp....43
Liite 3. Robotin komponenttien kytkentäkaavio....46

1 Johdanto

Opinnäytetyön tavoitteena on tutkia mikrokontrollereita, kehityspiirejä, kehitysalustoja ja niiden käyttöä robotiikassa. Tämän lisäksi vertaillaan kolmea yleistä kehitysalustaa: Arduino, ESP32 ja STM32. Samalla selvitetään kehityspiirien GPIO-pinnien käyttöä, lisämoduuleita kuten Bluetooth-, moottoriohjain- ja akun latausmoduulia. Tämän tutkimuksen tuotoksena tulee olemaan itse suunniteltu, ohjelmoitu ja rakennettu robotti jota voi ohjata puhelimella.

Robotin määritelmä on laaja ja riippuu lähteestä sekä mielipiteestä.(Wikipedia) Useimmiten ihmisten mieleen tulee tieteiselokuvien metalliset ihmisen muotoiset mekaaniset koneet täynnä vilkkuvia valoja, tönkösti liikkuvat nivelet ja metallinen ääni. Itse määrittelisin robotin koneeksi jonka voi ohjelmoida tekemään erilaisia toimintoja, joko automaattisesti tai syötteiden välityksellä.

Opinnäytetyössä vertaillaan erilaisia kehitysalustoja, niiden toimintoja, tehoa, hintaa ja kelpoisuutta robotiikkaan. Kehitysalustoiksi on valittu kolme erilaista vaihtoehtoa, jotka ovat yleisiä ja helposti saatavia lähes kaikkialla maailmassa. Itse piirien lisäksi moduulit joita robotissa käytetään, ovat myös edullisia ja helposti hankittavia.

Vertailujen lisäksi työssä suunnitellaan robotti, johon sopii valitsemani moduulit, STM32-kehityspiiri ja muut komponentit. Haasteen vuoksi työssä robotin maksimikooksi on asetettu10cm^3. Opinnäyteyössä käydään myös läpi ohjelmoinnin käsitteitä, joita tarvitaan piirien ohjelmointiin ja robotin toteuttamiseen sekä tutustutaan PlatformIO-työkaluun, C++ ohjelmointikieleen ja sulautettujen järjestelmien ohjelmointiin.

Valitsin aiheen koska sulautetut järjestelmät, kehityspiirit ja 3D-tulostus ovat olleet nousevia aiheita maailmalla jo usean vuoden ja olen ollut kiinnostunut robotiikasta. (Stack Overflow Trends & 3D Printing Industry) Halusin selvittää mitä tarvitsee päästäkseen soveltamaan robotiikkaa, sulautettuja järjestelmiä ja 3D-tulostusta. Tämän lisäksi se yhdistää harrastuksiani, mielenkiinnon kohteitani, ammattini ja tutkintoni yhdeksi kattavaksi kokoelmaksi jossa pääsin opettelemaan jotain uutta.

2 Käsitteet

lähdekoodi	ihmisluettava tietokoneohjelman koodi
avoin lähdekoodi	vapaasti käytettävää ja jaettavaa lähdekoodia
sulautetut järjestelmät	ohjelmoitava piiri, osana suurempaa järjestelmää
SoC	System on a chip, järjestelmäpiiri, sisältää useamman piirin tai sulautetun järjestelmän
ohjelmointikieli	ohjeita joilla tietokone suorittaa algoritmeja
Slicer	tietokoneohjelma jolla käännetään 3D-malli käskyiksi 3D-tulostimelle
CPU	Central Processing Unit; mikroprosessori; elektronin komponentti joka suorittaa tietokone algoritmeja
I/O	Input/Output – liittimet, pinnit tai portit
GPIO	General Purpose Input/Output; yleiskäyttöiset I/O pinnit
PWM	Pulse Width Modulation; pulssinleveysmodulaatio
IoT	Internet of Things; esineiden internet
Git	versionhallinta työkalu
Linux	Linux-ydintä käyttävä käyttöjärjestelmä
DIY	Do It Yourself – tee-se-itse
Koekytkentälevy	levy johon voi kytkeä elektroniikkakomponentteja ilman kolvausta
IDE	Integrated development environment - ohjelmointiympäristö. Ohjelmisto joka on kehitetty alustaa tai ohjelmointikieltä varten.
RAM	Random Access Memory – hajasaantimuisti, esim. mikrokontrollerin muisti johon voi kirjoittaa ja josta voi lukea
SRAM	Static RAM – staattinen RAM-muisti
EEPROM	Electronically Erasable Programmable Read-Only Memory – uudelleenkirjoitettavaa pysyvää muistia
Flash-muisti	muistia joka säilyttää tietoa vaikka virta olisi pois päältä
TCP/IP	tietoliikenne protokolla
BLE	Bluetooth Low Energy – pienen virrankulutuksen bluetooth

DMIPS	Dhrystone Million Instructions Per Second – Tietokoneprosessorien laskentatehon mittayksikkö
ST-Link	STMicroelectronicsin mikrokontrollerien virheen korjaus- ja ohjelmointityökalu
USART	Universal Synchronous/Asynchronous Receiver-Transmitter – sarjaliikenteen lähetys- ja vastaan otto
baudinopeus	(engl. baudrate) Tiedonsiirron nopeus, merkkiä sekunnissa
Assembly, BCPL, B, FORTRAN	Vanhoja, lähes muinaisia ohjelmointikieliä

3 Nykyaikaiset toimintaympäristöt

3.1 C++ ohjelmointikieli

Jotta voisimme puhua C++-ohjelmointikielestä tule meidän ensin tutustua kieleen C. Vuonna 1972 mies nimeltä Dennis Ritchie loi C-kielen Unix-käyttöjärjestelmää varten. Ennen Dennisiä, Ken Thompson oli yrittänyt luoda työkaluja ja apuohjelmia Unix-käyttöjärjestelmälle. Ken Thompson kävi läpi FORTRAN, BCPL- ja itse kehittämänsä B-kielen. Näiden kielien rajoitteiden vuoksi Dennis päätti yhdistää kielien hyvät puolet ja lisätä siihen vielä tarpeellisia konsepteja. Näin syntyi C, kieli joka on yhtä tehokas ja voimakas kuin Assembly. Assembly kielellä kirjoitettu koodi oli kuitenkin prosessorikohtaista, joten ohjelmien siirrettävyys oli todella huonoa. Dennis Ritchie kirjoitti uudelleen suurimman osan Unix-käyttöjärjestelmän ytimestä C-kielellä. C-kielen suurimpia vahvuuksia oli luettavuus ja siirrettävyys. Pian ohjelmoijat kaikkialla halusivat päästä käsiksi C-kieleen(freeCodeCamp). Hieman yli kymmenen vuotta myöhemmin, myös C++-kieli syntyi.

C++ ohjelmointikieli on yleiskäyttöinen ohjelmointikieli, joka kehitettiin laajennukseksi C-ohjelmointikielelle. Luojansa Bjarne Stroustrup:in mukaan hän kehitti C++:an jotta olisi ohjelmointikieli jossa C-kielen tehokkuus ja joustavuus voidaan yhdistää korkeamman tason ohjelmoinnin ominaisuuksiin. Käytännössä tämä tarkoittaa sitä että C-kielestä tehtiin paranneltu versio, jolla on helpompi ja tehokkaampi ohjelmoida. C++-kielellä ei ole tiettyä käyttötarkoitusta, vaan se suunniteltiin yleiseksi ohjelmointikieleksi. (guru99)

C++ on käännettävä kieli, eli sen koodi käännetään konekielelle ennen kuin tietokone voi suorittaa käskyjä. C++ käyttää staattista tyypitystä (statically-typed). Tällä tarkoitetaan sitä että kaikki ohjelmiston muuttujat ja niiden tyypit on määritelty ennen kuin se käännetään ohjelmistoksi. (Technopedia)

C++ on niin sanottu keskitason ohjelmointikieli, koska se sisältää sekä korkean tason (high-level) että matalan tason (low-level) toimintoja. (tutorialspoint) Kun ohjelmointikielen taso on matala, se tarkoittaa sitä että ohjelmaa ajetaan suoraan laitteistolla. Ohjelman ja laitteiston välissä ei siis ole tulkkia (kuten esim. Python:lla) tai virtuaalikonetta (Java). Käytännössä tämä mahdollistaa parhaan mahdollisen suorituskyvyn laitteistosta. Tästä johtuen C++ ohjelmointikieltä käytetään käyttöjärjestelmien ohjelmoinnissa, videopelien luonnissa ja juuri sulautetuissa järjestelmissä. (KO2)

Sulautetuissa järjestelmissä on usein hyvin rajallinen määrä laskentatehoa ja muistia. Tämän vuoksi on erittäin tärkeää että käytetään ohjelmia jotka käyttävät näitä resursseja tehokkaasti.(KO2)

C++ on loistava kieli opetella, jos haluaa ohjelmoida lähempänä rautaa tai puristaa tietokoneesta kaiken tehon irti. Kirjoitushetkellä pelkästään Helsingissä on tarjolla yli 200 työpaikkaa hakusanoilla ”c++ developer”. (LinkedIn) Eli myös työllistymismahdollisuudet ovat loistavat.

3.2 Avoin lähdekoodi

Laitteiden ja tietokoneiden ohjelmien toiminta perustuu lähdekoodiin. Lähdekoodi on ihmisluettavaa koodia, joka käännetään muotoon jota tietokone ymmärtää. (Search Adpp Architectire) Esimerkiksi C, C++ ja Rust ovat ohjelmointikieliä jotka pitää kääntää lähdekoodista ohjelmaksi kääntimellä. On olemassa myös ohjelmointikieliä joita ei tarvitse kääntää. Näitä kutsutaan tulkatuiksi ohjelmointikieliksi. Esimerkkejä tulkatuista ohjelmointikielistä ovat Python, JavaScript ja Ruby.

Avoimella lähdekoodilla tarkoitetaan lähdekoodia, joka on kehitetty, tuotettu tai lisensoitu vapaaksi käyttää, muokata ja jakaa ilmaiseksi. Tämä mahdollistaa koodin muokkaamisen laajentaakseen ohjelmistoa omiin tarkoituksiin, vikojen korjaamisen tai oppimisen lähdekoodista täysin ilmaiseksi.

Avoimen lähdekoodin tarkemmaksi määritelmäksi voidaan luetella OSI:n (Open Source Initiative, 2007) 10 kriteeriä:

1. Ilmainen ja vapaa jakelu - lisenssi ei voi rajoittaa ketään levittämästä tai myymästä ohjelmistoa osana isompaa ohjelmistokokonaisuutta kun ohjelmistoja on useammasta eri lähteestä. Lisenssi ei saa vaatia maksuja myynnistä.

2. Lähdekoodi - lähdekoodin tulee olla saatavilla mahdollisten käännettyjen ohjelmien lisäksi. Lähdekoodi on oltava saatavilla kohtuulliseen hintaan jos lähdekoodia ei jaeta ohjelman mukana. Lähdekoodin tulee olla ohjelmoitavassa muodossa.

3. Johdetut teokset - lisenssin tulee sallia johdetut teokset ja niiden jakamisen samoilla ehdoilla kuin alkuperäisen ohjelmiston lisenssi.

4. Lähdekoodin eheys - lisenssi ei saa rajoittaa muokatun lähdekoodin levittämistä mikäli se ei salli korjauspäivitysten ja niiden lähdekoodin levittämistä. Lisenssillä saa estää johdettujen teoksien jakamisen samalla nimellä tai versiolla kuin alkuperäinen ohjelmisto.

5. Ei ihmisten tai ihmisryhmien syrjintää. Lisenssi ei saa syrjiä ihmistä tai ryhmää ihmisiä.

6. Ei käyttöalojen syrjintää. Lisenssi ei saa estää ohjelmiston käyttötarkoituksia. Esimerkiksi lisenssi ei saa estää ohjelmiston käyttöä liiketoimintaan tai geneettisen tutkimuksiin.

7. Lisenssin jakaminen. Lisenssi tulee olla samoilla oikeuksilla ja rajoituksilla tarjolla kaikille käyttäjille.

8. Lisenssi ei saa olla tuotekohtainen. Ohjelmiston oikeudet eivät saa olla riippuvaisia tietyistä ohjelmistoista tai ohjelmistokokonaisuuksista. Ohjelmiston käyttäjillä pitää olla samat oikeudet kuin alkuperäisen kokonaisuuden käyttäjillä.

9. Lisenssi ei saa rajoittaa muita ohjelmistoja. Lisenssi ei saa asettaa rajoituksia muihin ohjelmiin, joita sen lisenssin kanssa jaetaan. Esimerkiksi, lisenssi ei saa vaatia että kaikki muut sen mukana jaettavat ohjelmistot ovat avoimen lähdekoodin ohjelmistoja.

10. Lisenssin tulee olla teknologianeutraali. Lisenssi ei saa ottaa kantaa tekniseen toteutukseen tai asettaa ehtoja yksittäisten teknologioiden tai käyttöliittymien kohdalle.

Git:in yleistymisen myötä avoimen lähdekoodin jakelu on helpompaa kuin koskaan. Git mahdollistaa useiden ihmisten tai jopa järjestöjen yhteistyön mitä monimutkaisimmissa projekteissa. Suosittujen git-palveluiden kuten GitHub ja GitLab ansiosta kuka vain voi luoda, ladata, muokata ja päivittää avoimen lähdekoodin ohjelmistoja.

3.3 Yhden piirin tietokoneet, järjestelmäpiirit ja sulautetut järjestelmät

Moni saattaa sekoittaa järjestelmäpiirit (system on a chip tai SOC) ja sulautetut järjestelmät keskenään. Tämä on ymmärrettävää, sillä sulautetuilla järjestelmillä yleensä tarkoitetaan piiriä tai piirejä jotka sisältävät mikrokontrollerin ja yleensä myös joitain muita liittymiä

kuten A/D-muuntimia (analogisesta signaalista digitaaliseksi arvoksi) tai jännitesäätimiä. (omni-sci)

Sulautetuilla järjestelmillä tarkoitetaan ohjelmoitavaa laitteistoa, joka toimii osana suurempaa järjestelmää. Esimerkkeinä puhelimet (koostuu useammasta sulautetusta järjestelmästä), anturit jotka lähettävät dataa (ilman laadun mittaus) tai vaikkapa kodin älylaitteet kuten Amazon Echo ja Google Nest. (IoT Agenda)

Järjestelmäpiireillä tarkoitetaan mikropiiriä, joka koostuu useammasta piiristä ja/tai sulautetusta järjestelmästä yhdessä alustassa. Integroimalla kaikki järjestelmät yhteen piiriin saadaan aikaan pienempi koko, pienempi hinta ja parempi virransäästö. Laitteet jotka normaalisti tarvitsisivat useamman piirin voidaan tällä tavalla toteuttaa yhtenä pienempänä kokonaisuutena. Kaikki toiminnallisuus saadaan yhteen pieneen pakettiin ilman kalliita ja monimutkaisia prosessoreita. (asysilico)

Yhden piirin tietokoneet taas ovat nimensä mukaisesti kokonaisia tietokoneita, yhdessä piirissä. Tämä tarkoittaa sitä että tietokoneeseen tarvitsee asentaa käyttöjärjestelmän, kytkeä vain syöttölaite kuten näppäimistö tai hiiri ja ulostulo laite kuten näyttö. Yleisimpänä esimerkkinä yhden piirin tietokoneesta on Raspberry Pi. Yhden piirin tietokoneet ovat olleet osana valtavirtaa jo useamman vuoden. Raspberry Pi:tä ihmiset käyttävät mediapalvelimina, videopelipalvelimina tai jopa päivittäisinä käyttökoneina. Raspberry Pi:n käyttöjärjestelmänä toimii usein Linux-pohjaiset käyttöjärjestelmät. Yhden piirin tietokoneita voidaan myös käyttää osana sulautettua järjestelmää.

3.4 Jakamistalous

Jakamistaloudella tarkoitetaan melko uutta ilmiötä tai liikettä jolla tarkoitetaan yhteisöllistä lähestymistapaa asioiden ja esineiden kuluttamiseen, käyttöön, tuotantoon ja talouteen. Jakamistalous on noussut suosioon monista syistä. Esimerkiksi teknologian kehittyminen oli yksi suurimmista tekijöistä. Internetin saatavuus ympäri maailmaa mitä vakaammilla ja nopeammilla yhteyksillä ja sosiaalisten verkostojen yleistyminen sen myötä ovat olleet avaintekijänä jakamistalouden nousuun. Ihmiset ovat yhä enemmän ja enemmän yhteydessä toisiinsa. Myös ihmisten ymmärrys teknologian ekologisesta vaikutuksesta ympäristöön on ollut suuri ajava tekijä. (Jakamistalous)

Internet-verkkokauppojen nousun myötä ilmestyi myös useita yksityisten ihmisten käytettyjen esineiden kauppapaikkoja. Ihmiset pystyivät myymään ja ostamaan käytettyjä esinei-

tä helposti. Esine saattoi olla hyödytön alkuperäiselle omistajalle ja arvokkaiden asioiden pois heittäminen on nykyajan ihmisten mielestä sekä rahan että luonnon haaskausta. Tämä on siis täydellinen puolitie. Vanhasta pääsee eroon, se tulee käyttöön ja siitä sai osan alkuperäisestä hinnasta takaisin. Myös tavaroiden ilmaiseksi lahjoittaminen on yleistynyt.

Ajatusten, ideoiden ja esimerkiksi lähdekoodin jakaminen on yleinen tapa jakaa ja uudelleen käyttää teknologiaa. Tämän ansiosta ihmiset voivat enemmän uudelleen käyttää esimerkiksi vanhentuneita laitteita toisissa käyttötarkoituksissa ja ennen vanhaksi jäänyt laite voi saada täysin uuden tarkoituksen. Tämä on paljon parempi vaihtoehto kuin loputon laitteiden vieminen kaatopaikalle. Toinen esimerkki jakamistalouden yleistymisestä ovat asuntojen vuokrauspalvelu AirBnB ja ajoneuvojen vuokrauspalvelu BloxCar. AirBnB mahdollistaa yksityisten ihmisen asunnon laittamisen vuokralle, kun on esimerkiksi itse matkoilla. Samalla tavalla BloxCar helpottaa ihmisiä vuokraamaan ajoneuvojaan kun heillä ei sille itse ole käyttöä. (Jakamistalous)

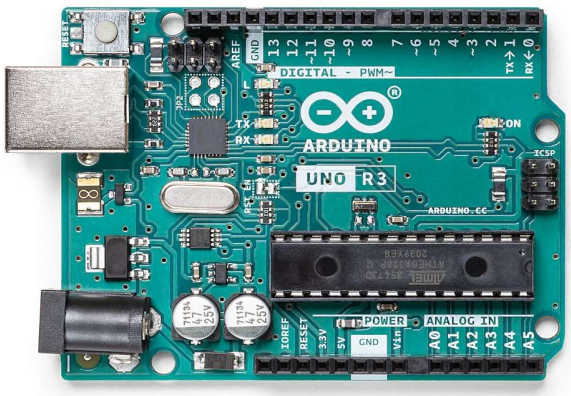
4 Kehityspiirit ja alustat

Kehityspiirien ja alustojen yleistyttyä ne ovat myös kehittyneet valtavasti. Ensimmäinen suuren suosion saanut kehitysalusta oli vuonna 2006 julkaistu Arduino. Mullistavinta Arduinossa oli se että se oli erittäin edullinen ja perustuu täysin avoimeen laitteistoon ja lähdekoodiin. Kuka vain pystyi ostamaan tai rakentaa sen itse, ja ohjelmoida sitä täysin vapaasti. Tämä synnytti valtavan suosion nousun teknologisissa DIY-piireissä ja kouluissa. Ihmiset alkoivat rakentaa ja keksiä innovatiivisia asioita kuten älypeilejä, käden liikkeellä ohjattavia autoja tai sääasemia. Kun Arduinon potentiaali oli huomattu, myös monet muut valmistajat astuivat markkinoille.

Loppupeleissä tulee kuitenkin muistaa että kehityspiirit ovat nimensä mukaisesti tarkoitettu tuotteiden tai palveluiden kehittämiseen. Piireissä on usein enemmän GPIO-pinnejä, laskentatehoa ja ominaisuuksia kuin lopputulos välttämättä tarvitsee. Kun kehittäjä on päässyt kehityksessään lopputulokseen, johon hän on tyytyväinen, seuraava vaihe on usein uuden mikropiirin suunnittelu. Näihin piirilevyihin tulee yleensä vain se määrä pinnejä, liittymiä ja toimintoja mitä tuotos tarvitsee. Näin säästetään sekä kokoa että rahaa lopputuotteessa. Nykypäivänä on olemassa paljon palveluita joista voi tilata itse suunnittelemia piirilevyjä, usein myös haluamillaan komponenteilla ja valmiiksi kolvattuna. Tämän myötä myös pienten toimijoiden on helppo tuottaa satoja, ellei jopa tuhansia piirilevyjä ja näin ollen tuotteita edullisesti. Mitä enemmän tilaat kerralla, sitä vähemmän yksittäinen kappale maksaa. (Nextpcb)

4.1 Arduino

Arduino on ilmainen, joustava ja helppokäyttöinen laitteisto- ja ohjelmistopohjainen avoimen lähdekoodin elektroniikan luontialusta luojille ja kehittäjille. Alustan avulla voit luoda erilaisia yhden piirin mikrotietokoneita, joille voi antaa erilaisia käyttötarkoituksia. (Circuitschools) Arduino ja sen yhteisöt ovat tehnet kehityspiirien ohjelmoinnista erittäin helppoa. Arduino julkaisi myös oman IDE:nsä, Arduino IDE. Tämä oli jälleen yksi helpotus askel



Kuva 1. Arduino Uno (Arduino Store)

monille kehityspiirien ohjelmointiin. Arduino IDE:en on sisäänrakennettu koodi editori, Arduino piirien ohjelmointi (ohjelmiston syöttö kehityspiirille) sekä sarjamonitori.

Suuren suosion saavuttanut Arduino on saatavilla eri kokoisina ja muotoisina kokonaisuuksina. Arduinon tarjontaan kuuluu esimerkiksi Arduino Mega, kehityspiiri jossa on sekä laskentatehoa että kattava määrä eri pinnejä, portteja ja käyttöliittymiä. Toinen suosittu kehityspiiri heiltä on Arduino Uno(kuvassa 1.). Tämän voisi todeta niin sanotuksi keskimalliksi. Kolmas todella suosittu on Arduino Nano. Nämä ovat erittäin pieniä, noin 2x5cm kokoisia kehityspiirejä. Vertailukohteeksi on valittu uusin Arduino Nano v3 kokonsa ja hintansa vuoksi. Arduino Nano on suosittu pienikokoinen kehityspiirisarja, joka sopii pinnien sijoittelun kannalta hyvin lähes kaikkiin koekytkentälevyihin. Arduino Nano on klassisen Arduino Nano:n jalanjäljissä seuraava pienikokoinen ja edullinen kehityspiiri.

Taulukko 1. Arduino Nano tekniset tiedot

Mikrokontrolleri	ATmega328	
	Kellotaajuus	16 MHz
Pinnit	Digitaaliset I/O	22
	Analogiset I/O	8
	PWM	6
Virta	Piirin käyttöjännite	5V
	Tulojännite	7-12V
	3.3V pinnien vaihtovirta	n/a
	5V pinnien vaihtovirta	40mA per pinni
Muisti	SRAM	2kB
	flash	32kB

Arduinon merkittävimpiä etuja on yhteisöllisyys ja helppokäyttöisyys. Arduinon helposti lähestyttävä IDE, kattavat palstat täynnä ihmisiä jotka ovat innokkaita auttamaan ja internet on pullollaan tutoriaaleja ja aloituspakkauksia. Arduinolle löytyy myös paljon niin sanottuja Shield-lisämoduleja. Shield-lisämodulien viehätys johtuu niiden helppokäyttöisyydestä, sillä nämä lisämoduulit on suunniteltu juuri Arduinon muodon ja pinnien sijoittelun perusteella. Näin ollen shieldit voidaan vain lisätä Arduinon päälle ja moduuli on käyttövalmis ohjelmoitavaksi.

Kaikkien Arduinon etujen lisäksi alustasta löytyy myös muutama suuri haitta. Vaikka se on erittäin lähestyttävä, siitä ei ole suurta hyötyä jos haluat työllistyä sulautettujen järjestelmien parissa. Sulautettuihin järjestelmiin erikoistuvat yritykset tuskin tulevat käyttämään arduinoja oikeassa tuotekehityksessä kun Arduinon perus lisenssi vaatii että lähdekoodi on saatavilla julkisesti. Tämän lisäksi piirit ovat melko kalliita verrattuna muuhun tarjontaan markkinoilla. Arduinojen hinta suoritustehoon ja ominaisuuksiin nähden on usein todella korkea. (embedded.fm)

4.2 ESP32

Espressif Systems on vuonna 2008 perustettu yritys, joka alkoi valmistaa mikropiirejä, joiden pääpaino oli suunnattu langattomiin yhteyksiin kuten Wi-Fi:n ja Bluetooth:iin. Vuonna 2014 läntisille markkinoille ilmestynyt ESP8266 mikropiiri oli edullinen ja tehokas Wi-Fi mikropiiri jossa oli sisäänrakennettu Wi-Fi ja se oli TCP/IP yhteensopiva. Tämän piirin dokumentointi oli lähes kokonaan kiinaksi mutta alhainen viiden dollarin hinta ja edellä mainitut toiminnot takasivat sen menestyksen. Pien ESP8266 menestyksen jälkeen, 2016 Espressif julkaisi ESP32-mikropiirisarjan. Näiden suosio ylitti edeltäjänsä.

ESP32-sarja oli suunnattu IoT-sovelluksiin ja laitteisiin. ESP32-sarjan piireissä oli edeltäjänsä mukaisesti Wi-Fi-toiminnallisuus. Uutena ominaisuutena tuli kuitenkin Bluetooth- ja BLE-toiminnallisuudet. Wi-Fi toimintoa parannettiin myös turvallisuuden kannalta sillä siinä mahdollisuus käyttää myös AES- tai SSL-salausprotokollaa. Piireissä oli myös mahdollisuus suurempiin määriin pinnejä, portteja ja käyttöliittymiä. Uuden sarjan myötä piirien suorituskyky nousi huomattavasti. Vertailukohteeksi on valittu Ai-Thinkerin tuottama NodeMCU-32S koska se on kooltaan ja hinnaltaan samaa luokkaa kuin muut vertailukohteet.

Taulukko 2. NodeMCU-32S tekniset tiedot

Mikrokontrolleri	ESP-WROOM-32s	
	Kellotaajuus	240 MHz
Pinnit	Digitaaliset I/O	32
	Analogiset I/O	16
	PWM	32*
Virta	Piirin käyttöjännite	2.2-3.6V tai 5V
	Tulojännite	7-12V
	3.3V pinnien vaihtovirta	12mA
	5V pinnien vaihtovirta	N/A

Muisti	SRAM	320kB
	flash	4Mb

NodeMCU-kehityspiirien suurin etulyöntiasema on sisäänrakennettu langattomuus, WiFi ja bluetooth. Tämän ratkaisee nopeasti ongelmia yhdistettävyyden kanssa juuri esimerkiksi robotiikassa ja erityisesti IoT-sovelluksissa. Toinen suuri etu varsinkin aloittelijoille on piirin kyky ajaa JavaScript-sovelluksia. JavaScript on helppo ja nopeasti opittava ohjelmointikieli ja monella saattaa olla sen osaamista jo entuudestaan.
ESP32-mikrokontrollereitä käytetään myös ammatillisesti ympäri maailmaa. On olemassa yrityksiä kuten u-blox joka tarjoaa valmiiksi sertifioituja langattomia moduuleita jotka käyttävät ESP32-mikrokontrollereita. (u-blox) ESP32-pohjaiset kehityspiirit ovat siis loistava lähtökohta aloittelevalle kehittäjälle joka on kiinnostunut sulautetuista järjestelmistä.

4.3 STM32

Vuonna 2007 STMicroelectronics julkisti ensimmäisen STM32-mikrokontollerisarjansa nimellä F1. F1-sarjan kehityspiirejä ja sarjoja valmisti vain muutama valmistaja, joten saatavuus rajallista. STM32-pohjaisten kehitysalustojen suosio nousi kuitenkin vasta ST:n itse kehittämänsä kehityspiirin, STM32VLDISCOVERY:yn myötä. Usein pelkällä ’discovery’ nimellä mainittu kehityspiiri oli suunnattu aloittelijoille. Piiri sisältää STM32F1-sarjan mikrokontrollerin ja piiriin integroidun ST-LINK vianetsintä ja ohjelmointi työkalut. Discovery pystyi ottamaan käyttövirtansa suoraan piirin USB-portin kautta. Nämä asiat tekivät piiristä helpon ja lähestyttävän sekä aloittelijoille että kokeneemmille kehittäjille. Tämän myötä STM32-sarjan piirit saivat suurta suosiota tee-se-itse-yhteisöissä.

Kolmannet osapuolet ovat alkaneet valmistamaan valtavia määriä F1- ja F4-sarjan piireillä varustettuja kehityspiirejä. Yksi erittäin suosittu F1-sarjan piiri joka sai kutsumanimensä sinisestä juottomaskistaan, oli bluepill. Se on klooni Leaflabs:in valmistamasta Maple Mini-kehityspiiristä. Bluebill kehityspiiriä seurasi pian askeleen tehokkaampi F4-piirillä varustettu blackpill. Kuten edeltäjänsä, nimitys tuli juottomaskin väristä. Vertailukohteeksi on valittu STM32F401CCU6-kehityspiiri koska se on pienikokoinen ja edullinen. Sekä mikropiirit että kehityspiirit STMicroelectronicsilta löytyvät erittäin edullisesti. Tämä mahdollistaa kehittäjää kokeilemaan järjestelmien kokoamista kaikenlaisilla moduuleilla pienelläkin budjetilla.

STM on myös kehittänyt oman IDE:nsä, joka kulkee nimellä STM32CubeIDE. Tässä IDE:ssä on kattavat käyttöliittymän mikrokontrollerin säätämiseen. Tämä mahdollistaa kattavan valikoiman erilaisia asetuksia kuten esimerkiksi pinnien konfigurointi ja mikropiirin suorittimen asetuksien säätö. STM32CubeIDE:ä saatavilla kaikilla yleisimmillä käyttöjärjestelmillä.

Taulukko 3. STM32F401CCU6 tekniset tiedot

Mikrokontrolleri	STM32F401CCU6	
	Kellotaajuus	84 MHz
Pinnit	Digitaaliset I/O	34
	Analogiset I/O	10
	PWM	25
Virta	Piirin käyttöjännite	3.3V tai 5V
	Tulojännite	2.0-3.6V
	3.3V pinnien vaihtovirta	25mA
	5V pinnien vaihtovirta	N/A
Muisti	SRAM	64K
	flash	256K

STM-mikrokontrollerien käyttämisen suurin etu sekä aloittelijoille että kokeneemmille harrastajille on se, että niitä käytetään ammattimaisesti ympäri maailmaa. Esimerkkejä oikeista tuotteista joissa käytetään STM:n tuottamia mikrokontrollereita, ovat MacBook Pro 15”, Xiaomi MiJia QiCycle sähköpyörä ja Google Pixel C android tabletti. (iFixit) Jos todella haluaa työllistyä sulautettujen järjestelmien ja mikrokontrollerien parissa, STM32-piirit ovat loistava valinta.

4.4 Vertailu

Oikeanlaisen kehityspiirin valinta voi olla hankala tehtävä, varsinkin aloittelijoille. Kehityspiirejä löytyy kaikista mahdollisista hintaluokista ja jokaisella alustalla sekä piirivalmistajalla on vannoutuneet käyttäjäkuntansa. Kun valitsee kehityspiiriä, tärkeitä katselmointikohtia ovat käyttötarkoitus, suoritusteho ja liitettävyys.

4.4.1 Käyttötarkoitus

Arduinon kohdalla tarkoitus on hyvin selkeä. Arduino kehitettiin sitä varten että ihmisillä olisi mahdollisimman helppo työkalu prototyyppaamiseen ilman aiempaa kokemusta ohjelmoinnista tai elektroniikasta.(arduino) Heidän piireistään löytyy helposti käytettävät liittimet, he tarjoavat kattavan IDE-kokonaisuuden ja paljon käyttäjiä jotka auttavat toisiaan luomaan mitä loistavampia tuotoksia.
ESP32-pohjaisten kehityspiirien ominaisuudet selkeästi pyörivät yhdistettävyyden ympärillä. Kun jokainen piiri sisältää valmiiksi WiFi- ja Bluetooth-yhteyden mahdollisuuden, on langattomien tuotoksien kehittäminen helppoa.

STM32-piireillä suurin houkutus on varmasti oikeat tuotteet ja työllistyminen sulautettujen järjestelmien parissa. STMicroelectronicsin valmistavia mikropiirejä, antureita, mikrosysteemeitä ja sulautettuja järjestelmiä löytyy monelta toimialalta. Heillä on asiakkaina suuria nimiä kuten esimerkiksi Apple, Sony ja Samsung. (applied materials)

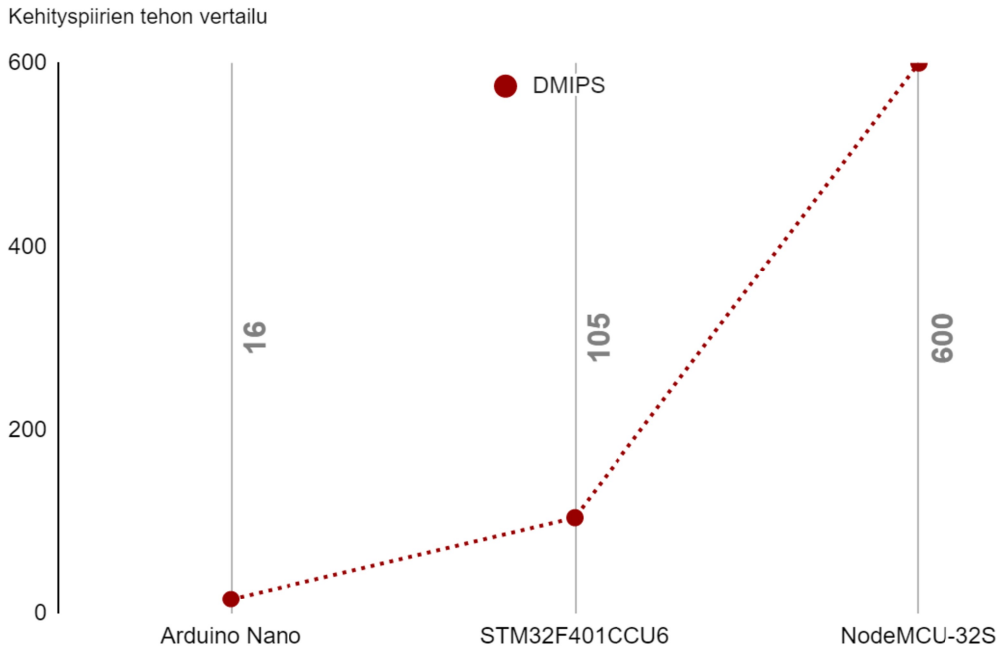
4.4.2 Suoritusteho

Yksi avaintekijöitä monilla eri toimialoilla on suoritusteho. Tästä syystä arduino usein laahaa perässä. Tämä ei kuitenkaan tarkoita sitä että se olisi huono vaihtoehto opettelemaan esimerkiksi robotiikkaa. STM32-piirit ovat todettu toimiviksi ja vakaiksi valinnoiksi kautta toimialojen. ESP32-piirien suoritusteho on korkeaa luokkaa, mutta niiden käyttöönotto on ollut melko suppeaa.

Yksi tapa laskea mikropiirien suoritustehoa on Dhrystone Million Instructions Per Second, lyhennettynä DMIPS. DMIPS:llä mitataan kuinka monta miljoonaa kokonaisluku ohjeistusta prosessoria suorittaa sekunnissa. Tällä tavalla saadaan numero, jolla voidaan mitata prosessorin tehokkuutta muihin prosessoreihin.

Vaikka Arduinon Nano näyttää tässä vertailussa melko heikolta (kuvio 1.), on hyvä pitää mielessä että DMIPS on miljoonaa ohjeistusta sekunnissa. Tämä on silti erittäin kykenevä määrä käskyjä, joita piiri pystyy suorittamaan ja tällä teholla pystyy tekemään paljon erikoisempiakin tuotoksia. Toiseksi tehokkaimman paikan sai tällä mittarilla STM32-piiri. Yli viisi kertaa tehokkaampana piirinä STM32F401 pystyy jo paljon rankempaan suoritustehoon. Tehokkaimpana tehovertailussa nousi ESP32-mikrokontrolleria käyttävä piiri. Kyseinen ESP32-piirin prosessori sisältää kaksi erillistä ydintä, joista toinen on vakiona käytössä langattomien yhteyksien suorittamiseen. (Explore Embedded) Vaikka DMIPS olisi tuolloin puolet yllä mainitusta lukemasta, se on silti lähes kolme kertaa tehokkaampi kuin

STM32F401. Tässä on myös tärkeää tietää että kaksi ytimiset ESP32-piirit voidaan myös ohjelmoida käyttämään molempia ytimiä haluamallaan tavalla. (Circuit Digest)



Kuvio 1. DMIP vertailukaavio

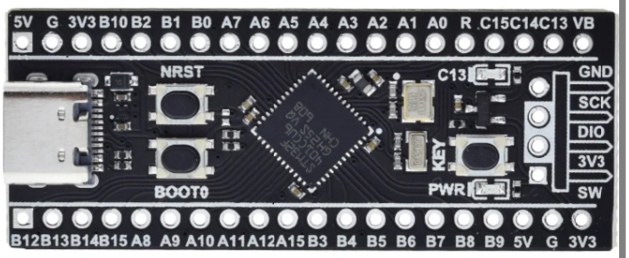
4.4.3 Hinta ja saatavuus

Arduino Nanoa löytyy virallisista lähteistä melko suurella hintaerolla. Arduinon virallisesta kaupasta 18 euron hintaan ja Suomesta 24-25 euron luokkaa. Hinta ei kuulosta suurelta ennen kuin sitä verrataan muihin vertailukohteisiin. Samasta kaupasta Suomessa (triopak) löytyy Arduino Nano hintaan 24,00 ja NodeMCU-32s hintaan 17,50. Kun ottaa huomioon ESP32-piirien suoritustehon ja sisäänrakennetun langattomuuden on NodeMCU-32s selkeästi houkuttelevampi vaihtoehto. Jos Arduinon piirin haluaisi samalla toiminnallisuudella täytyy ostaa erikseen Wi-Fi- ja/tai bluetooth-moduuli.

Pienten STM32-kehityspiirien löytäminen Euroopan alueelta on yllättävän hankalaa. Jos kuitenkin haluaa erittäin edullisia kehityspiirejä, paras suunta on silloin Kiina. Sivustot kuten aliexpress, taobao ja banggood ovat täynnä edullisia kehityspiirejä. Esimerkiksi Arduino Nanoa ja sen toimintoja täysin vastaavia tuotteita löytyy 5,49 eurolla. NodeMCU-32s piirien hinta Kiinan suunnalta vastaavasti 4,39 euroa. Tässä kohtaa STM32-kehityspiirit loistavat, sillä niitä löytyy jopa 2,56 eurolla. Tilasin oman STM32F401CCU6-kehityspiirini AliExpressistä.

5 Robotin toteutus

Vaikka työssä suunniteltu laite ei robotin kuvauksen mukaan korvaa mitään ihmisen tekevää työtä, keksin robotille kutsumanimeksi Tiny Tyger. Robotin toteutusta varten kehityspiiriksi on valittu STM32F401CCU6 (kuvassa 2.) piirin koska se oli todella edullinen ja sille löytyy paljon yhteisöllistä tukea. Halusin haastaa itseäni sillä että kehityspiiri ei itsessään sisällä langattomien yhteyksien komponentteja, eli ne tulee lisätä itse. Samoin myös moottoriohjain, moottorit ja latauspiiri pitää kytkeä itse. Tämän dokumentin liitteenä on käyttämäni kytkentäkaavio. (liite 3)



Kuva 2. STM32F401CCU6 (AliExpress)

5.1 Robotin suunnittelu

Asetin itselleni haasteet joiden mukaan haluaisin robotin toteutuvan. Robotin tulee olla mahdollisimman pieni, mieluiten alle 10cm³. Se tulisi olla ohjattavissa puhelimella, bluetoothin välityksellä. Halusin myös että robottiin on mahdollista lisätä muita komponentteja kuten antureita, näyttöjä tai kaiuttimia jälkikäteen. Tämä tulee ottaa huomioon mallintaessa. Silloin voi tulostaa muokatun version osasta johon mahdollinen lisäkomponentti tulee. Robotin tulisi olla sisältää oma virtalähteensä, ja se tulisi olla ladattavissa ilman että robottia tarvitsee purkaa tai avata. Kun aikomukseni on 3D-tulostaa mahdollisimman suuri osa robotista, tulee pitää mielessä tulostettavuus ja tulostuksen helppous mallia tehdessä.

5.2 Muut komponentit

5.2.1 Virtalähde

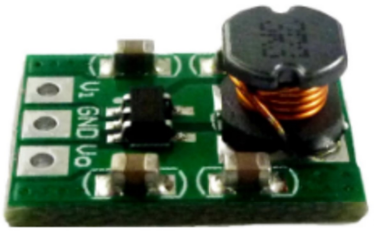
Aluksi asettamani haasteet tuntuivat erityisen haastavilta. Virtalähteeksi kelpaa jos monenlainen paristo tai akku mitä löytyy suoraan kaupan hyllyiltä. Nämä uudelleenladattavat virtalähteet kuitenkin usein vaativat virtalähteen poistoa tai vähintäänkin virtalähteen johtojen irrottamisen laitteesta jotta sen voi kytkeä laturiin. Toinen ylitettävä este oli jännite ja virta. Käyttämäni kehityspiirin minimi käyttöjännite on 2.0V ja maksimi käyttöjännite on 3.6V. Tämä tarkoittaa sitä että tavallisia AA-paristoja tulisi olla vähintään kaksi kun yhden AA-pariston jännite on 1.5V. Tämän lisäksi AA-paristoissa on purkuvirta todella pieni, yleensä noin 2A. En pitänyt ajatuksesta käyttää kahta AA-paristoa joten löysin 18650-

akut. Ne ovat hieman pyöreitä, suurempia, uudelleenladattavia ja monilla valmistajilla on tarjolla malleja jotka voivat tuottaa korkean purkausvirran, jopa 25A. (proakku) Tämä mahdollistaa jatkossa useamman moottorin, näytön tai muiden lisämoduulien lisääminen ilman pelkoa siitä että akusta loppuisi voima. Tällä hetkellä tyydyin kuitenkin Camelion ICR18650-2600 akkuun, jonka jatkuvaksi purkuvirraksi on ilmoitettu 5A.

Virtalähteen lisäksi robotin sisälle tulee sijoittaa akkukotelo, josta akku on helppo vaihtaa, mikäli akku vioittuu tai tarvitsen tehokkaampaa akkua.

5.2.2 DC DC-alasmuunnin

Seuraava haaste oli 18650-akkujen purkujännite. Se voi olla jopa 4.2V. Tämä jännite rikkoisi kehityspiiristä komponentteja. Tähän ongelmaan löysin ratkaisuksi DC DC-alas muuntimen. Kyseessä on komponentti joka voi ottaa vastaan X-Y jännitteen, mutta tarjoilee aina Z. Muuntimessa jonka hankin, oli sisääntuleva jännite 3.3-5.5V ja ulos tuleva jännite 3.3V.



Kuva 3. DC DC-alasmuunnin (AliExpress)

Moduulin kytkentä on helppoa. Virtalähteestä tuleva positiivinen johto liitetään Vi pinniin (kuva 3.) ja Vo pinnistä liitetään positiivinen johto kehityspiirin VIN liitäntään. Sekä virtalähteen että vastaanottavan laitteen negatiiviset johdot kytketään moduulin keskellä sijaitsevaan GND-pinniin

5.2.3 Latausmoduuli

Kuva 4. TP4056 latausmoduuli (AliExpress)

Toinen akkuihin liittyvä ongelma oli uudelleenladattavuus. 18650-akuille löytyy kyllä paljon latureita, mutta ne ovat yleensä seinään kytkettäviä koteloita, joihin akku tai akut asetetaan ladattavaksi. Löysin kuitenkin latausmoduulin joka oli tehty juuri 18650-akuille.

Siinä on vihreä LED joka ilmoittaa kun akku on latautunut ja punainen LED joka kertoo kun akkua ladataan. Moduuli sisältää myös muutaman suojaominaisuuden. Moduuli estää akkujen käyttämisen alle 2.5V jännitteen. Tämä ehkäisee akun purkautumista liian tyhjäksi. Moduulissa on myös ylilataussuoja, jotta akku ei ylikuumenisi latauksen yhteydessä.

Latausmoduulia käytetään kytkemällä akun positiivinen napa moduulissa olevaan B+ pinniin ja akun negatiivinen napa B- pinniin. Virtaa tarvitsevan laitteen positiivinen napa kytketään OUT+ pinniin ja negatiivinen napa OUT- pinniin. (kuvassa 4.) Akun lataaminen onnistuu kytkennän jälkeen asettamalla USB-C-kaapeli latausmoduulin USB-porttiin.

5.2.4 Moottoriohjain

Kun haluaa ohjata moottoreita kehityspiireillä, tarvitaan usein moottoriohjain. Tämä johtuu siitä että kehityspiirit pystyvät yleensä antamaan verrattain rajallisen määrän purkuvirtaa. Tämä tarkoittaa sitä että kehityspiirin virta ei riitä pyörittämään moottoreita. Moottoriohjainta hallitaan kehityspiirin PWM-pinnien välityksellä, ja moottoriohjain saa virran suoraan virtalähteestä. Tämä mahdollistaa koko purkuvirran käytön moottorien pyörittämiseen.

Kuva 5. L298N-moottoriohjain (AliExpress)

Tässä toteutuksessa käytetään L298N-moottoriohjainta. Moottoriohjaimen kytkentä on hieman monimutkaisempi kuin aikaisemmat moduulit. Kehityspiiristä tulee valita neljä PWM-pinniä, jotka kytketään moottoriohjaimen A1, A2, B1 ja B2 pinneihin. A1 ja A2 ovat A-moottorin ohjausta varten myötäpäivään ja vastapäivään. Samoin B1 ja B2 pinnit B-moottorille. Moottorit kytketään kuvan oikeassa reunassa sijaitseviin MotorA ja MotorB pinneihin.(kuvassa 5.) Moottorin kytkentöjen valinta riippuu moottorin pyörimissuunnasta. Tämän lisäksi moottoriohjaimeen tulee kytkeä virtalähde. Virtalähteen positiivinen johto kytketään plus pinniin ja negatiivinen johto miinus pinniin.

5.2.5 Moottorit

Suunnitteluvaiheessa halusin robotin olevan mahdollisimman pieni, jotta en tarvitsisi suuria moottoreita tai suurta käyttöjännitettä. Valitsin siis moottoreiksi kaksi pientä 3V DC-vaihteistomoottoria. Nämä pienet moottorit pystyvät kahteensataan kierrokseen minuutissa, joka ei ole erityisen hurja vauhti. Moottoreissa on sisäänrakennettu ratasvaihteisto joka muuntaa moottorin pyörimisvoimaa sen pyörimisnopeudesta väännöksi. Tästä johtuen moottori pyörii hitaammin mutta siitä löytyy enemmän vääntöä ja se pystyy pyörittämään raskaampia kuormia. Tämä helpottaa robotin jatkokehitystä, jos siihen asennetaan paljon lisämoduuleita tai muita osia.

Moottorien kytkentä on todella helppoa. Molemmissa moottoreissa on kaksi kontaktia, joista ne voidaan kytkeä johdoilla suoraan moottoriohjaimen MotorA- ja MotorB-pinneihin. Toisen moottorin navat kytketään MotorA-pinneihin ja toinen MotorB-pinneihin. Sillä miten päin moottorien johdot kytketään, ei periaatteessa ole mitään väliä sillä moottorien pyörimissuunnan voi ohjelmoida kehityspiiriin.

5.2.6 BLE-moduuli

Kun STM32F401CCU6-kehityspiirissä ei ole minkään näköistä langatonta yhteyttä, se pitää lisätä itse. Koska kyseessä on robotti ja sitä ohjataan langattomasti, suurien määrien tiedonsiirto ei ole tarpeellista. Bluetooth low energy, eli BLE on langaton teknologia, joka pyrkii minimoimaan virrankulutuksen. Vaihtokauppana virrankulutuksesta tulee hitaampi tiedonsiirto.(link labs) Tämä ei kuitenkaan haittaa esimerkiksi kaukosäätimien tai peliohjainten tapauksessa, ja juuri siksi se on sopiva valinta robotin etäohjaukseen.

Kuva 6. AT-09 BLE-moduuli(AliExpress)

Valitsemani BLE-moduuli (kuvassa 6.) on sekä android- että IOS-yhteensopiva. Tämä tarkoittaa sitä että jos puhelimelle löytyy sopiva sovellus, sitä voi käyttää sekä android- että IOS-laitteilla.

BLE-moduulin kytkentään robotin tapauksessa tarvitaan vain neljä pinniä. RX, TX, VIN ja GND. RX-portilla tarkoitetaan receive, eli vastaanottavaa ja TX-portilla transmit eli lähettävää pinniä.VIN tarkoittaa sisääntulevaa jännitettä ja GND maadoitus, eli moduulin negatiivinen pinni. Lisähuomiona RX- ja TX-kytkennästä on

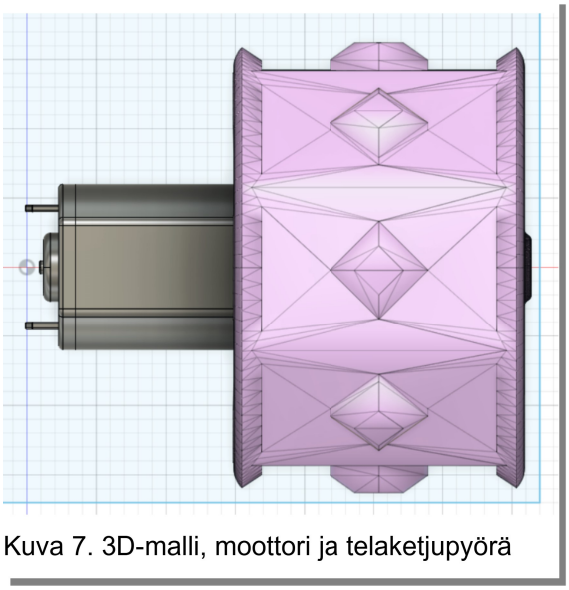
se että kun näitä pinnejä käytettäviä laitteita kytketään yhteen, tulee ne kytkeä niin sanotusti ristiin. RX-pinni BLE-moduulista kytketään kehityspiirin TX-pinniin ja sama myös toisin päin. TX- ja RX- pinnien välityksellä kehityspiirit ja BLE-moduuli keskustelevat keskenään. Tämä mahdollistaa käskyjen vastaanottamisen bluetooth yhteyden välityksellä

5.3 Robotin rakennus

Kun olin löytänyt haluamani osat, seuraava vaihe on robotin rakentaminen. Tämä projekti on jaettu useampaan osioon. 3D-malli osiossa käydään läpi, miten 3D-mallin luominen tapahtuu Autodesk:in Fusion 360-sovelluksella. 3D-tulosus osiossa käsitellään 3D-mallin siirtämistä slicer-ohjelmistoon ja miten 3D-tulostus tapahtuu. Kokoamis osiossa selvitetään miten robotin tulostetut osat, kehityspiiri ja muut elektroniset komponentit kolvataan yhteen ja asennetaan paikoilleen.

5.3.1 3D-malli

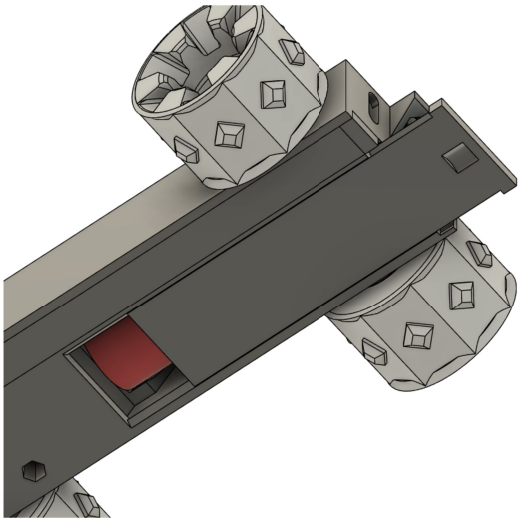
Vaikka olen käyttänyt Fusion 360-sovellusta jo melkein kaksi vuotta, on 3D-mallien luominen työläs ja haastava prosessi. Halusin robottini kulkevan telaketjuilla ja näyttää edes etäisesti Tiger I-panssariajoneuvolta. Koska pelkästään robotin ulkokuoren mallintamiseen kului paljon aikaa, päätin käyttää jakamistaloutta apunani. Thingiverse on internet sivusto, jossa ihmiset voivat jakaa luomiaan 3D-malleja ilmaiseksi. Sivustoa pääsääntöisesti käyttää 3D-tulost yhteisöt. Yritin aluksi katsoa viittauksia ja esimerkkejä miten 3D-tulostettu telaketju kannattaisi toteuttaa. Hetken tutkittuani sivustolta löytyi muutama toimivaksi todettu malli. Aikarajoitteiden puitteissa päätin käyttää oman robottini telaketjuina ja pyörinä SMARS modular robot-mallia pohjana. (thingiverse) Käytin edellä mainitun mallin telaketjun, pyörien ja pyörin kiinnitystä esimerkkinä omalle 3D-mallilleni.



Kuva 7. 3D-malli, moottori ja telaketjupyörä

Jotta robotin rungon mallintaminen olisi mahdollisimman helppoa, otin mitat kaikista elektronisista komponenteista, ja tein niistä mallit joiden mukaan voin mitoittaa runkoa. Kun olin saanut karkean luonnoksen koosta ja muodosta aloitin asettelemalla telaketjun palasia kunnes niistä tuli telaketju. Tämän jälkeen asettelin pyörät paikoil-

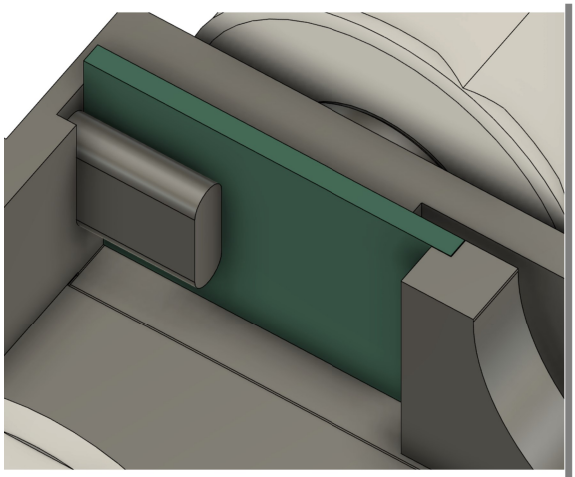
leen. Kun pyörät olivat oikeassa kohdassa, asettelin takapyörien kiinnityskohdat runkoon. Etupyörien sijoittelu oli huomattavasti haastavampaa.



Kuva 8. 3D-mallin pohjassa akkukotelon luukku

Kun halusin pitää robotin koon mahdollisimman pienenä, tuli suurin haaste vastaan moottoreissa ja niiden koossa. Jos moottorit laittaa vierekkäin, ja niiden jatkoksi pyörät, on robotilla jo huomattavasti leveyttä. Ensiksi ajattelin käyttäväni kartiohammaspyörää. Tällöin olisin voinut sijoittaa moottorit rinnakkain ja käyttää hammaspyöriä voimansiirrossa telaketjun pyörille. Tämäkin osoittautui ongelmalliseksi koska tulostimellani ei voi tulostaa niin pientä kartiohammaspyörää ja niin tarkasti kuin tarvitsisi. Lopulta huomasin että telaketjun pyörä ovat sen verran isot että voin sijoittaa moottorit pyörien sisään. (kuvassa 8.) Moottorit siis sijoitetaan runkoon ja niiden ympärille tulee kuori. Kuoren ympärille asetetaan telaketjupyörä. Tämä toimii hyvin sekä tilan säästämiseksi robotin rungon sisällä, että myös voimansiirron kannalta. Ei ylimääräisiä rattaita tai hihnoja. Jotta moottorit istuisivat tarpeeksi varmasti rungon sisällä, moottorien väliin tulee pieni palikka. Tämä estää moottoreita irtoamasta kuoristaan.

Seuraava haaste oli akun ja varsinkin akkukotelon sijoittelu runkoon. Jotta akku olisi helposti vaihdettavissa, tarvitsee runko siis luukun. Luukun optimaalinen koko olisi sellainen että akkukotelo ei mahdu siitä läpi mutta akku mahtuu. Päätin käyttää yksinkertaista mekanismia, jossa akkukotelo asetetaan rungon sisään ja akkukotelon kohdalla on pohjassa aukko. Tämä aukko peitetään tarvittaessa liukuvalla kannella, joka pysyy paikallaan kitkalla. Tämä osoittautui toimivaksi ratkaisuksi, sillä se ei vaadi liimaa tai ruuveja.

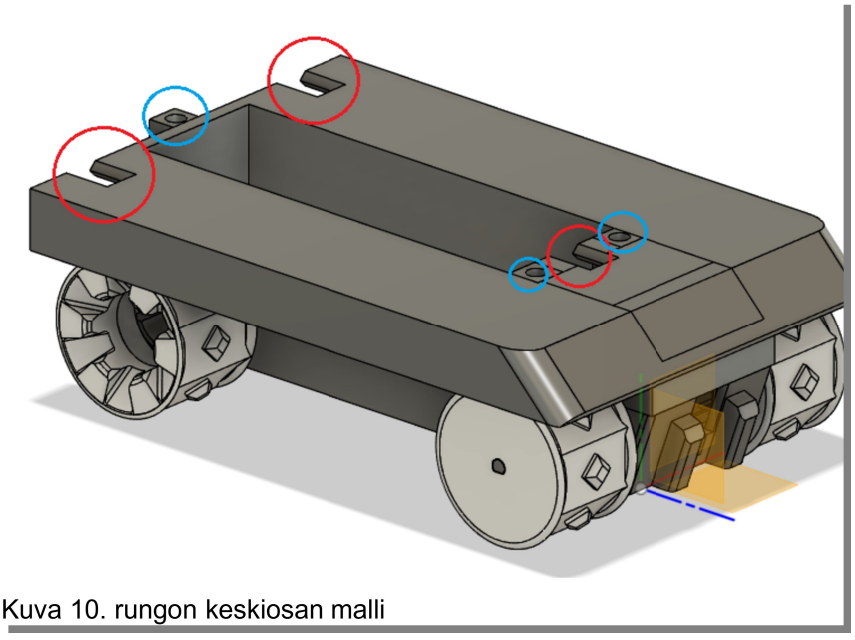


Kuva 9. latausmoduuli rungossa

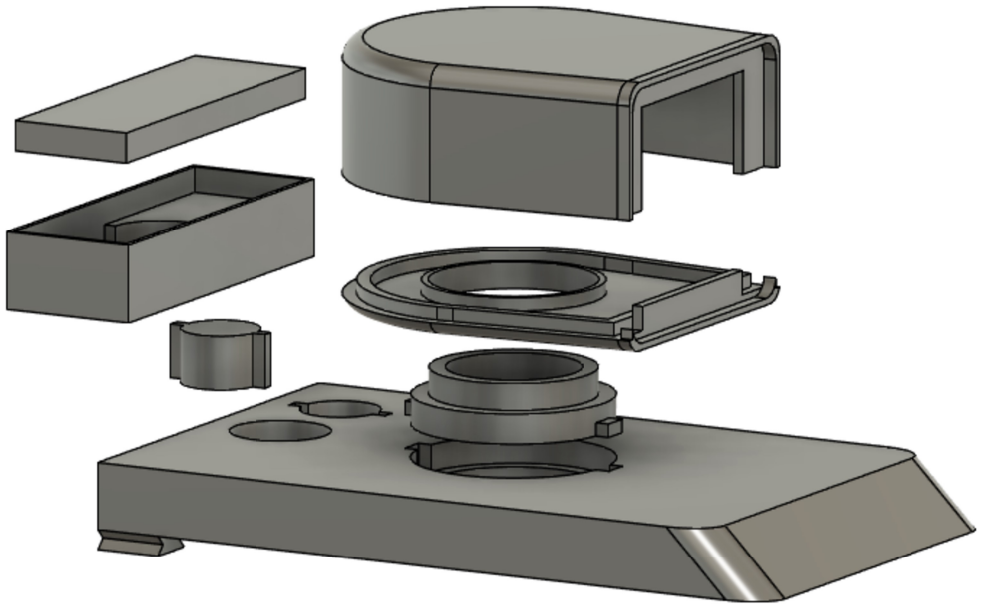
Seuraava erittäin tärkeä kohta mallia on latausmoduulin sijainti. Moduuli tulisi sijoitella

se siten että se pysyy paikalla vaikka siihen liitetään USB-C johto. Eli latausmoduulin taakse tarvitaan jotain mikä pitää sen paikallaan. Tämän lisäksi latausmoduulin tulisi sijaita paikassa jossa siihen voi kytkeä kyseisen johdon. Kuvassa kahdeksan näkee miten sijoittelun latausmoduulin. Se on kyljellään takarenkaiden takana siten että USB-portti sijaitsee robotin takaosassa. Moduulin toiselle puolelle sijoittelin vastakappaleen joka pitää moduulin paikallaan mutta siinä on myös tilaa johdoille, jotka latausmoduuliin kytketään. Kuvassa vihreä osa on latausmoduuli. (kuva 9)

Tämän jälkeen suunnittelin rungon keskiosan. Keskiosan suunnittelu oli suhteellisen suorasukaista, sillä keskiosan tarkoituksena on antaa rungon sisäosaan lisää tilaa komponenteille ja pitää telaketjujen pyörät kiinni moottoreissa. Keskiosa kiinnitetään alempaan runkoon kolmella ruuvilla, kaksi edessä ja yksi takana. Tämän jälkeen sijoittelin keskiosaan kolme rakoa, jotka ovat yläosalle tarkoitetut kiinnityskohdat. Nämä kiinnityskohdat toimivat kitkalla. Kuvassa merkkasin punaisella kolot ja sinisellä ruuvien reiät. (kuva 10)
Rungon yläosan mallintaminen oli jo hieman haastavampaa. Tässä vaiheessa rupesin miettimään BLE-moduulin sijoittelua. Alun perin ajattelin että olisi hienoa jos moduulin saisi antennin kaltaisesti robotin takaosaan tai mahdollisesti robotin torniin. Kävin läpi muutaman iteraation kunnes huomasin ajattelevani asiaa liian monimutkaisesti. Päätin sitten että moduuli kiinnitetään omassa pienessä kotelossa robotin päälle, takaosaan. Tällöin rungon yläosa tarvitsee reiän sekä tornille että BLE-moduulin johdoille. Yläosaan tuli vielä kolo johon sijoitetaan moduulin kotelon ja rungon yläosan parittava pala. Tämä parituspala oli yksinkertainen sylinteri, jonka molemmin puolin on väkänen. Tämä estää kiinnitettyä BLE-moduulin koteloa pyörimästä. Halusin myös rungon yläosaan kiinnittyvän tornin pystyä pyörimään. Silloin jos haluan jatkokehittää robottiin moottoroidun tornin tai muita lisäosia, ei minun tarvitsisi välttämättä muokata runkoa ollenkaan. Voin toteuttaa tornin siten että komponentit tai mahdolliset moottorit voi sijoittaa itse torniin.



Kuva 10. rungon keskiosan malli



Kuva 11. rungon yläosa, BLE-moduulin kotelo ja torni

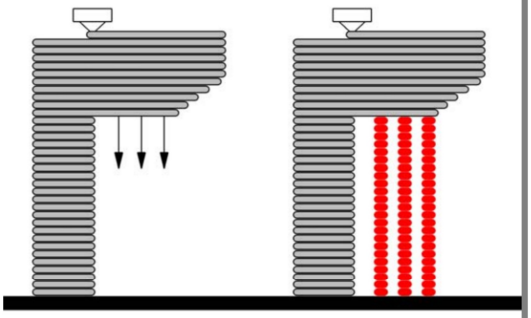
Kun robotin rungon yläosa oli valmis, aloin mallintamaan tornia. Edellä mainittua jatkokehitystä ajatellen suunnittelin tornin ja rungon yläosan väliin parituspalan joka oli samanlainen kuin BLE-moduulin kotelon ja rungon välissä. Tornin ja rungon väliin tuleva parituspala on kuitenkin paljon suurempi, ja siinä on reikä keskellä mahdollisia johtoja varten. Jotta torni olisi mahdollisimman helppo tulostaa, jaoin sen kahteen osaan. Tällöin valmistellessani robotin osia tulostettavaksi minun ei tarvitsisi tehdä erityistoimenpiteitä tulostamisen helpottamiseksi.

5.3.2 3D-tulostus

Käytin 3D-tulostimeen omistamaani Artillery Sidewinder X1-filamentti tulostinta. Filamenttitulostimella tarkoitetaan 3D-tulostinta joka käyttää muovilankaa, eli filamenttia, tulosteen toteuttamiseksi. Tämä tulostin sulattaa muovia ja siirtää tulostuspäätä tulostusalustalla, eli pedillä, siten että se pursottaa tulostettavan esineen sulatetusta muovista. Tämä toistetaan asetuksista ja tulostimesta riippuen 0.20 millimetrin välein. (MedicalDesign & outsourcing)

Tulostuksen aloittamiseksi tarvitsee kuitenkin 3D-mallin, ja se malli pitää kääntää tulostimen ymmärrettäviksi käskyiksi. Oman tulostimeni kohdalla kyseiset käskyt tapahtuvan g-

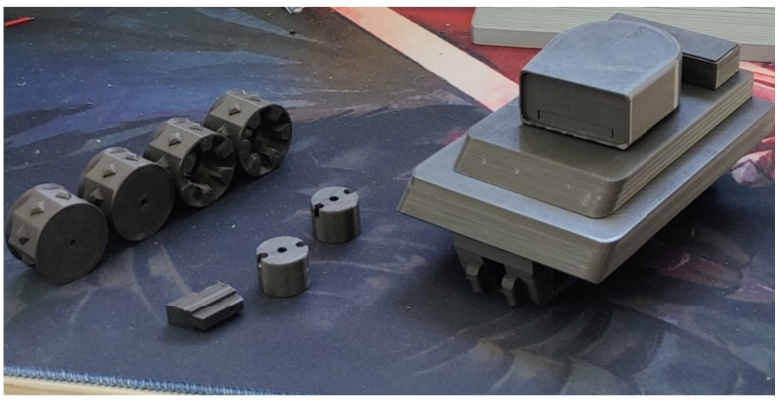
code-tiedostoilla. Jotta mallin saisi g-code:ksi, tarvitaan tulostuksen valmisteluohjelmaa eli slicer:iä. Slicer:in tarkoitus on englanninkielisen nimensä mukaisesti pilkkoa 3D-malli kerroksiin, ja kerroksista sitten käskyiksi eli g-code-komennoiksi. Kun 3D-malli on asetettu slicer:ssä haluamaansa kohtaan ja tarvittavat asetukset ovat säädetty, slicer tuottaa g-code-tiedoston. Kun tämän tiedoston tarjoilee 3D-tulostimelle, tulostin seuraa tiedoston sisältämiä ohjeita ja alkaa suorittamaan käskyjä. Jos malli ja asetukset olivat kohdillaan, on tuotoksena odottelun jälkeen valmis fyysinen esine, 3D-tuloste.



Kuva 12. 3D-tuloste ja sen tukeminen (Jipa, Bernhard ja Dillenburger, 2017)

Olen käyttänyt omia 3D-tulostimia aktiivisesti lähes kaksi vuotta. Tästä johtuen olen oppinut paljon periaatteita, joiden mukaan 3D-malleja on hyvä tehdä tulostusta varten. Mallinnusvaiheessa käytin paljon aikaa ja ajattelua siihen, että malli olisi mahdollisimman helppo tulostaa. Tämä tarkoittaa käytännössä sitä että malleissa on mahdollisimman vähän tuettavia tulostuskohtia. Tuettavalla tulostuskohdalla tarkoitetaan usein kohtia mallista jotka roikkuvat ilmassa sillä hetkellä kun tulostin on pääsyt siihen kerrokseen, jota tulostetaan. Kuvassa 12. on vasemmalla esimerkki, jossa osa mallista roikkuu ilmassa. Tästä johtuen tulostus saattaa epämuodostua tai epäonnistua. Oikealla on esimerkki punaisella, jossa käytetään tukena minimaalinen määrä materiaalia, joka poistetaan tulostuksen jälkeen.

Robotin osien tulostus tapahtui neljässä osassa. Ensimmäiseen osaan sain sijoiteltua rungon alaosan, keskiosan, yläosan sekä moottorien kuoret. Toisessa osassa tulostin tornin yläosan, alaosan ja liitoskappaleen. Kolmanteen osaan sain mahdutettua kaikki 48 telaketjun palaa, joita robottiin tarvitsi. Viimeisessä osassa tulostin BLE-moduulin kotelon ja liitoskappaleen.



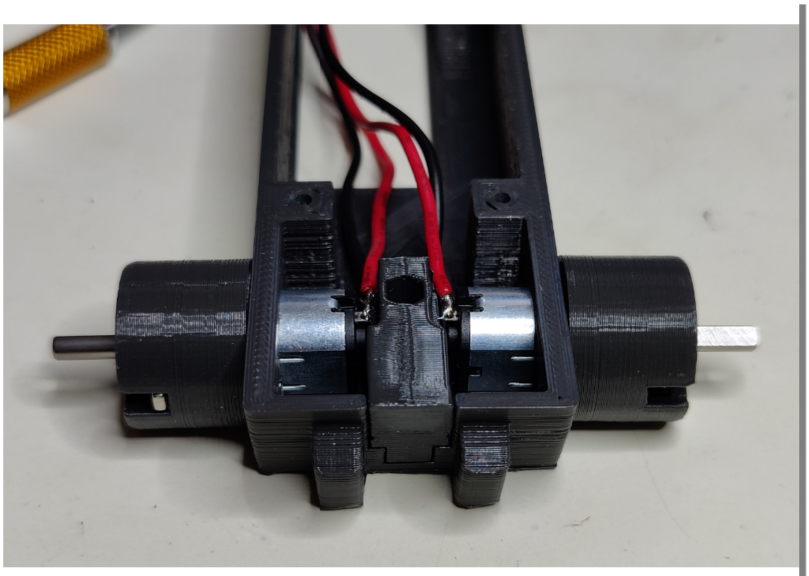
Kuva 13. 3D-tulostetut osat ja rungon testisovitus

Tulostus osoittautui yllättävän helpoksi. Kun mallintaessa otti huomioon tulostettavuuden, tuli tukia vain muutamaan pieneen koh-

taan. Niidenkin tukien määrä oli hyvin minimaalinen, ja ne oli helppo poistaa. Kun osat olivat tulostettu, tarvitsi vain sovittaa että kaikki osat sopivat paikalleen. Lähes kaikki osat istuivat paikoilleen ilman jälkikäsittelyä. Rungon keskiosa ja yläosa vaati pientä hiomista. Ainoa suuri virhe mallinnusvaiheessa tapahtui latausmoduulin portin sijoittelun kanssa. Latausmoduulia on mahdoton sijoittaa paikalleen ilman että rungosta leikkaa pienen palan pois. Kyseessä oli siis onnistunut tulostus, sillä mitään osia ei tarvinnut tulostaa toistamiseen. Tämä varmistui sovittamalla tulostettuja osia ilman elektronisia komponentteja. (kuva 13)

5.3.3 Kokoaminen

Kun robotin osat oli mallintanut itse, osoittautui kokoaminen todella helpoksi. Aivan ensiksi tulee sijoitella moottorit paikoilleen. Moottorien päälle asetetaan niiden kuoret ja kuoret kiinnitetään runkoon kahdella mutterilla ja ruuvilla. Moottorien johdot tulee kuitenkin kolvata ennen asennusta. Kun moottorit ovat paikoillaan, voidaan niiden väliin asettaa palikka joka pitää moottorit paikoillaan. Palikka kiinnitetään yhdellä ruuvilla ja mutterilla.



Kuva 14. moottorien asennus runkoon

Seuraavaksi laitetaan sekä taka- että eturenkaat paikoilleen. Takarenkaat asennetaan clip-on-tyylisellä kiinnityksellä. Eturenkaat asentuvat paikoilleen moottoreiden akseleihin. Kun rungon keskiosa kiinnitetään, se estää etupyöriä liukumasta pois akseleilta. Tässä vaiheessa on tarpeellista kolvata akkukotelo, moottoriohjain- ja latausmoduuli. Kun latausmoduuli on kolvattu, se asetetaan sille mallinnettuun paikkaan rungon alaosassa.

Rungon keskiosa kiinnitetään ruuveilla ja muttereilla. Ruuvien sijainti näkyy kuvissa 10. ja 14. Mutterit tiputetaan niille tehtyihin koloihin rungon alapuolelta. Kun rungon keskiosa on kiinnitetty, on seuraavaksi kolvattava DC DC-alasmuunnin ja itse kehityspiiri. BLE-moduulin pinneissä joihin kehityspiirin johdot kiinnitetään, on suositeltavaa käyttää esimerkiksi

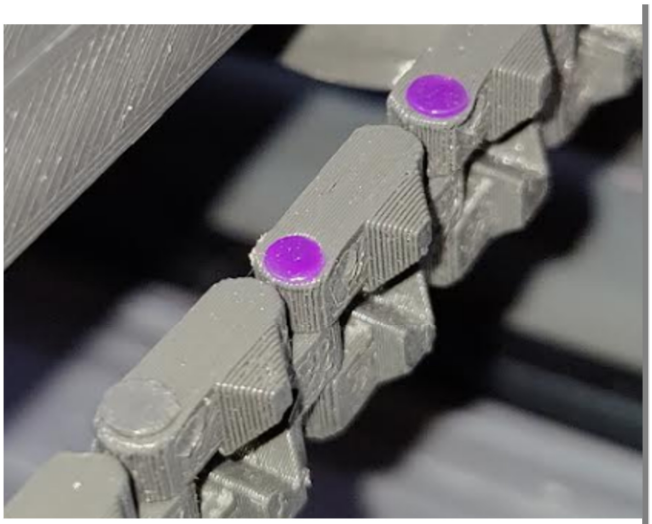
dupont-liittimiä. Dupont-liittimien avulla voit irrottaa ja asentaa BLE-moduulin uudestaan-jos robottia tarvitsee koskaan avata.

Kun kaikki tarvittava komponentit ovat kolvattu, asetetaan kehityspiiri, moottoriohjain ja alas muunnin rungon sisään. Tämän jälkeen voidaan asentaa rungon yläosa paikoilleen. Rungonyläosaa asennetaan, pitää ensin syöttää BLE-moduulin johdot yläosan takana olevasta reiästä. Tämän jälkeen rungon yläosan voi asettaa kohdilleen siten että liitoskohdat ovat vierekkäin. Rungon yläosa sitten työnnetään paikoilleen.

Kuva 15. BLE-moduuli kotelossa

Kun rungon yläosa on paikoillaan, BLE-moduulin kotelon voi asentaa paikoilleen liitospalalla. Kotelon voi liimata paikoilleen, mutta se tuntui pysyvän paikoillaan ilman kiinnikkeitä tai liimaa. Kehityspiiristä tulevat johdot voi kytkeä BLE-moduuliin ja BLE-moduulin voi asettaa paikalleen koteloon (kuvassa 15.). Kotelon päälle laitetaan vielä kansi.

Kun runko ja BLE-moduuli on paikoillaan, voidaan asettaa tornin sovituspala. Tornin alaosa asetetaan sovituspalan päälle. Tähän vielä tornin yläosa jonka pitäisi napakasti istua tornin alaosassa. Tornin pitäisi pyöriä vapaasti lukuun ottamatta BLE-moduulin koteloa. Viimeiseksi voidaan kiinnittää telaketjut. Telaketjujen kiinnittämiseksi suunnittelin alunperun käyttäväni ruuveja jotka sopivat nätisti telaketjujen reikiin. Tämä osoittautui kuitenkin ongelmalliseksi tavaksi, sillä se vaatii kaksi ruuvia per telaketjun pala. Ruuvit nostaisivat telaketjujen painoa hieman mutta vielä ongelmallisemmin se vaatisi 96 ruuvia. Olin nähnyt keskustelupalveluissa kuinka muut 3D-tulostusharrastajat ovat käyttäneet kolvia tulostusmuovin sulattamiseen. Tästä sain idean jonka avulla telaketjujen yhdistämisestä tuli helppoa. Telaketjujen reikään mahtui löysästi tulostuslankaa. Kun tämän langan asettaa telaketjun paloista läpi, ja leikkaa



Kuva 16. telaketjujen liitos tulostuslangalla

langan siten että siitä jää hieman yli, voi langan kuumentaa sytyttimellä pehmeäksi. Tämän jälkeen kun painaa telaketjun sulatettu lankapuoli vasten viileää ja tasaista pintaa, pehmennyt muovi täyttää telaketjun sivun reiän ja kovettuu paikalleen. Telaketjun toisen pään reikä jää löysästi tulostuslangan ympärille ja se pyörii vapaasti. Telaketjun viimeisen linkin kiinnitin kuitenkin yhdellä ruuvilla. Tällä tavoin telojen poisto onnistuu rikkomatta muita telaketjun palasia tai liitoksia. Kuvassa 16. näkyy kun violettia tulostuslankaa on käytetty harmaassa osassa.

Kun telaketju on valmis, se syötetään rungon keskiosan ja eturenkaiden välistä. Sitten se viedään takarenkaan ympäri ja yhdistetään ruuvilla. Näin se muodostaa ketjun molempien ketjupyörien ympäri. Tässä vaiheessa robotin fyysinen muoto on valmis.

5.4 Ohjelmointi

5.4.1 Etäohjaus sovellus

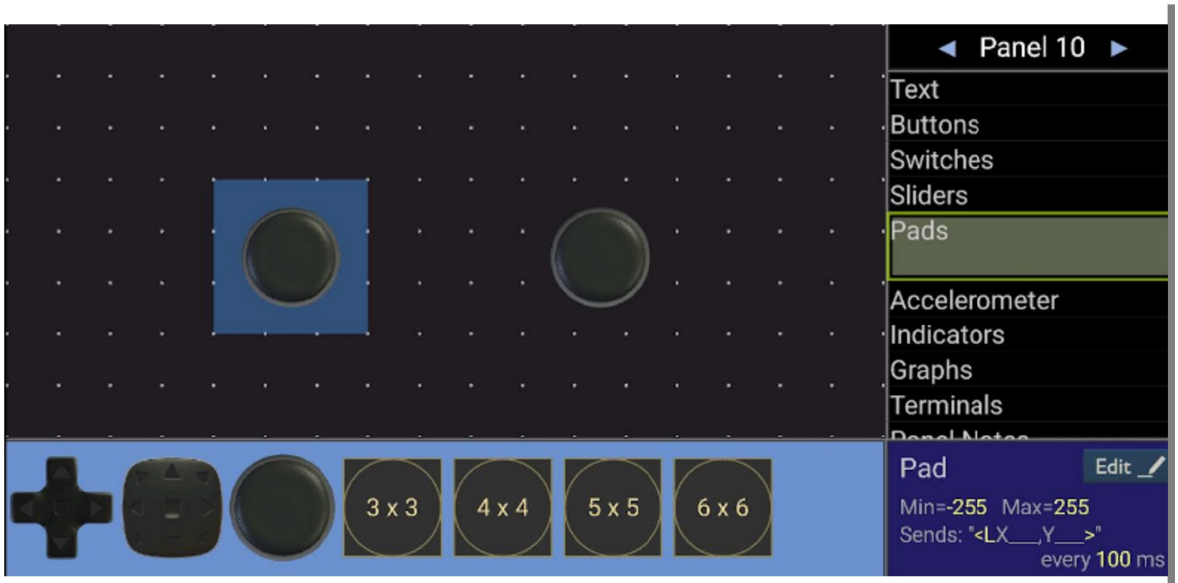
Koska halusin toteuttaa robotin etäohjauksen puhelimitse, sovelluksessa pitää pystyä määrittelemään lähetettävä komento. Sovelluksessa tulisi olla myös mahdollisuus joko liukusäädin- tai joystick-ohjaukseen. Mikäli sovelluksesta löytyy liukusäädin tai joystick, sen tulisi palautua keskelle, lähettää miinusarvoja yhdessä päässä ja positiivisia arvoja toisessa päässä. Onnekseni Google:n Play-kaupasta löytyi Bluetooth Electronis niminen sovellus.

Bluetooth Electronics sovellus tukee sekä bluetooth- että BLE-yhteyttä. Sovelluksessa on myös kattava työkalu erilaisten ohjauspaneelien luomiseen. Tästä työkalusta löytyy monia erilaisia nappuloita, liukusäätimiä ja jopa gyroskooppi. Tärkein oli kuitenkin joystick-ohjaus. Kun löytää nappulat tai säätimet joita tarvitsee, ne vain siirretään paneeliin. Säätimet saa itse sijoittaa paneelin minne haluaa. Tämän jälkeen säätimiä voidaan konfiguroida.

Joystick-säätimen konfigurointiin löytyi erittäin kattavat vaihtoehdot. Tärkeimpänä koin itselleni kuitenkin ”Start with” ja ”End with” kohdat. Näiden avulla voit päättää mitä merkkejä paneeli lisää arvoihin ennen kun se lähetetään ja ”End with” kohtaan . ”Start with” kohtaan määritellään merkit joita lisätään arvon alkuun ja ”End with” kohtaan määritellään mitä tulee arvon perään. Tämä helpottaa ohjelmoinnin kannalta siten, että voin ohjelmoida kehityspiirin odottamaan aloitusmerkkiä ”<L” tai ”<R” ja lopetusmerkkiä ”>”. Tämä mahdollistaa komentojen erittelyn siten että ”<L” merkeillä alkavat komennot ovat tarkoitettu vasem-

manpuoleiselle moottorille ja ”<R” merkeillä alkavat oikealle. ”<” merkki lisätään siksi että tiedämme koska kukin komento alkaa ja päättyy.

Toinen tärkeä asetus on ”Min value” ja ”Max value”. Näillä kentillä asetetaan lähetettävän arvon minimi- ja maksimiarvot. Koska käytän Arduino-ohjelmistokehystä, PWM-arvot joita kehityspiiri käsittelee, on 0-255. Tämä tarkoittaa sitä että kun PWM-pinniin lähetetään arvo 0, se on pois päältä. Kun taas lähetetään arvo 255, PWM-pinniä käskytetään olemaan koko ajan päällä. Asetan molempien joystick-säätimien minimiarvoksi -255 ja maksimi arvoksi 255. Tämä tarkoittaa käytännössä sitä että kun haluan että moottori pyörii eteenpäin täyttä vauhtia, sovellus lähettää arvon 255. Kun haluan moottorin pyörivän taaksepäin, se lähettää arvon -255. Vielä yksi tärkeä asetus on ”Send continuously”. Tämä mahdollistaa sen että niin kauan kuin joystick-ohjaimeen kosketaan sovelluksessa, se lähettää jatkuvasti arvoja robotille.



Kuva 17. kuvakaappaus ohjauspaneelin muokkaustyökalusta

Lopputuloksena ohjauspaneelissa on kaksi joystick-säädintä, joista toinen lähettää arvoja vasemmanpuoleista moottoria varten ja toinen oikeanpuoleista moottoria varten (kuvassa 17.). Tällä tavoin lähettää arvoja yhdelle moottorille tai molemmille moottoreille samanaikaisesti. Näin myös moottorien eri teho ja pyörimissuunta ovat mahdollista.

Täytyy myös huomioida että hallinta sovelluksesta tulee neljäakselisen joystick-säätimen arvot. Joystick-säädin lähettää aina kaksi arvoa, X- ja Y- arvoa. Tulen käyttämään anne-

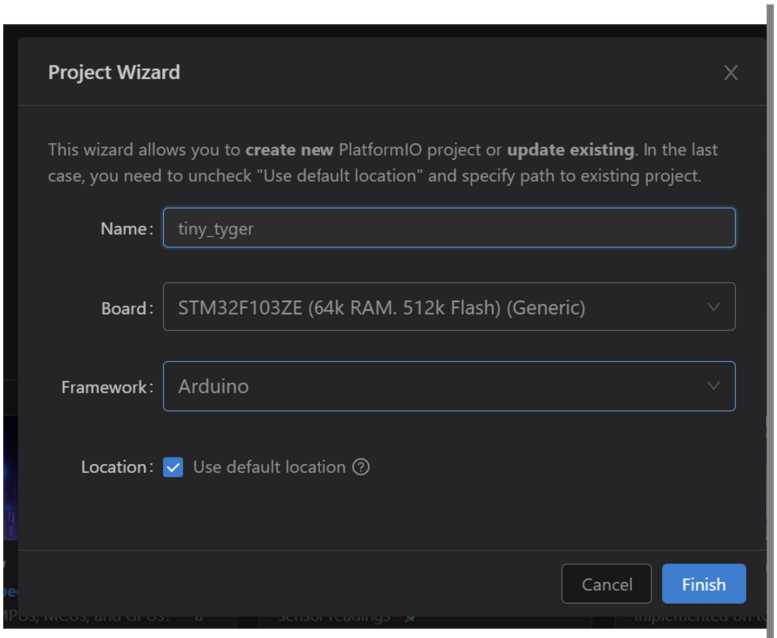
tuista arvoista ainoastaan ensimmäistä, eli X-arvoa. Y-arvo tullaan parsimaan pois ennen kuin arvot käsitellään ja lähetetään moottoriohjaimelle.

5.4.2 Kehitysympäristön pystyttäminen

Aloittelijoille kehityspiirien ohjelmoinnista on tehnyt todella helppoa PlatformIO niminen työkalu. PlatformIO on kokoelma moderneja työkaluja jotka toimivat useimmilla käyttöjärjestelmillä, useilla sulautetuilla järjestelmäalustoilla ja monella eri rajapinnalla. PlatformIO sisältää myös vianetsintä-, yksikkötestaus- ja koodin analysointityökaluja. Työkalun tarkoitus on helpottaa eri kirjastojen ja kehityspiirien käyttöä. (PlatformIO)

Itse käytän PlatformIO-työkalua Visual Studio Code:n laajennuksen muodossa. Visual Studio Code on ohjelmointiin kehitetty tekstieditori, jossa on korostettu paljon lisäosien tai laajennuksien käyttöä. PlatformIO-lisäosan asennus on todella helppoa. Visual Studio Coden laajennusvalikossa kirjoitetaan hakukenttään PlatformIO, ja haku tarjoaa lisäosaa heti ensimmäisenä nimellä PlatformIO IDE. Tämän asennettua kehittämisen aloittaminen on todella helppoa.

PlatformIO-lisäosan asennuttua Visual Studio Code:en ilmestyy uusi valikko, PlatformIO. Tämän valikon alta voi aloittaa uusia projekteja, selata kirjastoja, laitteita tai esimerkiksi kehitysalustoja. Kun aloittaa uuden PlatformIO-projektin aukeaa ohjattu asetusikkuna. Tässä ikkunassa voidaan valita projektin nimi, käytettävä kehityspiiri ja ohjelmistokehys. Valitsin itse ohjelmistokehykseksi Arduino:n, sillä se on helposti lähestyttävä ja aloittelijaystävällinen. Tämä tarkoittaa käytännössä sitä että Arduinoille kirjoitettua koodia voidaan soveltaa STM32F401CC-kehityspiireissä. Jos aikaa olisi ollut enemmän, olisin perehtynyt STMic-



Kuva 18. PlatformIO:n ohjattu asetusikkuna

roelectronics:in omaan ohjelmistokehykseen, eli STM32Cube:en. Kun painaa ”Finish” (kuvassa 18) ohjelma aloittaa projektin valmistelun.

Kun projekti on valmisteltu, PlatformIO asettelee kaikki tarvittavat tiedostot ja kansiorakenteen projektia varten. Se lataa myös tarvittavat työkalut ja kirjastot, jotta kirjoitettu koodi voidaan helposti kääntää ohjelmaksi kehityspiirille. Yksi tärkeimmistä tiedostoista on kuitenkin platformio.ini. Tämä on projektin asetustiedosto, missä määritellään käytetty kehi-

```
11  [env:genericSTM32F401CC]
12  platform = ststm32
13  board = genericSTM32F401CC
14  board_build.mcu = stm32f401ccu6
15  framework = arduino
16  upload_protocol = stlink
17  build_flags =
18      -D PIO_FRAMEWORK_ARDUINO_ENABLE_CDC
19      -D USBCON
20  monitor_dtr = 1
```

Kuva 19. platformio.ini tiedosto

tyspiiri, kehitysalusta ja ohjelmistokehys. Nämä ovat valmiiksi täytetty, mutta jotkin kehityspiirit saattavat tarvita lisäkomentoja. Esimerkiksi oman kehityspiirini kohdalla minun tuli asettaa muutama ylimääräinen asetus jotta PlatformIO alkoi kääntämään koodia oikein. (kuvassa 19) Rivillä 14. asetan tarkan mikrokontrollerin mallin koska STM32F401CC:stä on useampi variaatio. Rivillä 16. asetan koodin siirtämistavaksi ST-Link, koska käytän kyseistä laitetta koodin lataamiseen kehityspiirille. Riveillä 17-19 asetan muutaman asetuksen jotka mahdollistavat sarjaportin lukemisen USB-portin välityksellä. Tämä on tärkeää siinä vaiheessa kun aletaan ohjelmoimaan BLE-moduulia. Rivillä 20. asetetun asetuksen avulla ilmoitetaan kohdelaitteelle että sen tulee olla valmis myös vastaanottamaan komentoja USB-portin välityksellä. (PlatformIO) Tämän jälkeen kehitysympäristöni on valmis ja voin aloittaa kehityspiirin ohjelmoinnin.

5.4.3 Pinnien määrittely, setup funktio ja loop funktio

Kehitysympäristön määriteltyä, täytyy seuraavaksi määrittää pinnit. Pinnien määrittelyllä kerrotaan koodissa, mitä kehityspiirin pinnejä aiot käyttää. Pinnit on esimääritelty PlatformIO:n ansiosta, mutta niiden määrittely omassa koodissa helpottaa koodin lukua. Esimerkki:

```
1 #define LED_BUILTIN PC13
```

Robottia varten tarvitsen kaksi sarjayhteyspinniä ja neljä PWM-pinniä. Sarjayhteyspinnit ovat BLE-moduulin kanssa keskustelua varten ja PWM-pinneillä hallitaan moottoriohjainta. Määrittelin pinnikaavion mukaisesti moottorien pinnit PB0, PB1, PA6 ja PA7. Määrittelin ne moottoriohjaimen numerointia käyttäen, eli moottorin A pinnit olivat PA6 nimellä MTR_A1 ja PA7 nimellä MTR_A2. Samoin myös moottorille B, PB0 on MTR_B1 ja PB1 on MTR_B2. (liite 2)

Kun pinnit on määritelty, tulee seuraavaksi määritellä niin sanottu setup funktio. Setup on funktio, joka ajetaan joka kerta kun kehityspiiri käynnistetään. Tässä funktiossa määritellään, missä tilassa pinnejä käytetään (ADC, PWM, etc.) ja mahdolliset sarjayhteydet. Moottoriohjain tarvitsee neljä pinniä, ja kaikkien tulee olla PWM-tilassa. Pinnien tilan asettaminen onnistuu helposti käyttämällä pinMode funktiota. Funktio tarvitsee toimiakseen pinnin määritellyn nimen ja tilan jossa pinniä käytetään. Kun robotin tapauksessa tulee lähettää PWM-komentoja, käytetään tilaksi OUTPUT. Tällöin komennoksi koostuu pinMode(MTR_A1, OUTPUT). Tämä toistetaan kaikille moottoriohjaimeen meneville pinneille.

Rakentamani robotti tarvitsi vain yhden sarjayhteyden BLE-moduulia varten. Määrittelin kuitenkin vianetsinnän helpottamiseksi myös sarjayhteyden USB-viestintää varten. Sarjayhteys määritellään kehityspiirin USART-pinnien mukaisesti. Kehityspiirin pinnikaaviosta (Liite 1.) käy ilmi että PA3- ja PA2-pinnit ovat USART-pinnejä ja niiden toiminnot ovat RX2 ja TX2, eli vastaanottava ja lähettävä pinni sarjayhteydelle numero kaksi. Kun pinnit on määritelty ja yhdistetty oikein, voidaan setup funktiossa kutsua Serial2.begin(<baudinopeus>). Tämä ilmoittaa kehityspiirille että sarjayhteyttä numero kaksi aiotaan käyttää sulkuihin kirjoitetulla baudinopeudella. Sekä kehityspiiri että BLE-moduuli tukevat baudinopeutta 115200, joten komento sarjayhteyden aloittamiseksi on Serial2.begin(115200). USB-sarjayhteyden asettamiseksi tarvitsee vain kirjoittaa SerialUSB.begin(115200).

Kun pinnit on määritelty ja setup funktio toiminnassa, voidaan aloittaa itse ohjelman kirjoittaminen. Tämä tapahtuu loop funktiossa. Loop funktioon kirjoitetaan varsinainen ohjelma, eli mitä lähetetään ja vastaanotetaan mistäkin pinnistä. Loop funktiota nimensä mukaisesti pyöritetään jatkuvasti. Eli loop funktion sisällä olevia muuttujia ja muita funktioita ajetaan kunnes virta katkaistaan.

5.4.4 BLE-yhteys

BLE-yhteys osoittautui erittäin ongelmalliseksi, sillä hankkimani BLE-moduuli ei ollut aito ja alkuperäinen piiri. Tämä tarkoittaa sitä että dokumentaatio jota BLE-piirille aioin käyttää ei pätenyt oman piirini kohdalla. BLE- ja Bluetooth-moduulien konfigurointi tapahtuu AT-komennoilla. AT komennot ovat sarja merkkejä, jotka pitää lähettää moduulille sarjayhteyden välityksellä. Kun alkuperäisten suunnitelmieni mukaiset komennot eivät toimineet, jouduin kirjoittamaan pienen sovelluksen kehityspiirille joka vain lähettää ja vastaanottaa AT-komentoja BLE-moduulille.

Onnekseni BLE-moduuli kuitenkin vastasi lähes universaaliin AT-kemontoon: AT+HELP. Tällä komennolla moduuli vastasi sarjayhteyden välityksellä kaiken tiedon itsestään, ja pystyin aloittaa BLE-moduulin konfiguroinnin. Ensimmäisenä asetin BLE-moduulin PIN-koodin käyttäen komentoa AT+PASS<PIN-koodi>. Esimerkiksi AT+PASS1234, asettaisi moduulin PIN-koodiksi 1234. Seuraava komento jota käytin, oli AT+NAME. Tämä komento asettaa BLE-moduulin yhteyden nimen, eli nimen joka näkyy kun hakee bluetooth-laitteita puhelimella. Asetin oman BLE-moduulin nimeksi ”bestman” komennolla AT+NAME-bestman. Viimeisenä komentona asetin baudinopeudeksi 115200, komennolla AT+BAUD8. Tämän komennon sai selville ensin lähettämällä komennon AT+BAUD. BLE-moduulin vastaus kertoi kaikki baudinopeudet jota se tuki ja numeron jolla mitäkin baudinopeutta käytetään komennon yhteydessä. Baudinopeudella ei ole suurta merkitystä tämän sovelluksen kannalta, mutta halusin yhteyden toimivan niin nopeasti kun se pystyy. Nyt kun BLE-moduuli oli konfiguroitu, pystyin yhdistämään puhelimeni siihen. Pystyin myös tässä vaiheessa kokeilemaan arvojen syöttöä Bluetooth Electronics sovelluksesta.

```
40    analogWrite(MTR_A1, 255);
41    delay(500);
42    analogWrite(MTR_A1, 127);
43    delay(500);
44    analogWrite(MTR_A1, 0);
45    delay(500);
46
47
48    analogWrite(MTR_A2, 255);
49    delay(500);
50    analogWrite(MTR_A2, 127);
51    delay(500);
52    analogWrite(MTR_A2, 0);
53    delay(500);
```

Kuva 20. kuvakaappaus moottorin testausohjelmasta

5.4.5 Moottoriohjaus

Kun BLE-yhteys toimii ja järjestelmän komponentit ovat yhdistetty, pystyin kokeilla moottoriohjauspiirin toimintaa lähettämällä asetettuihin MTR_A ja MTR_B pinneihin. Tein pienen testiohjelman joka lähettää eri suuruisia PWM-arvoja moottoriohjaimen pinneihin. Jokaisen komennon jälkeen asetetaan pieni odotus aika. Kokeilin ohjelmassani jokaista moottoriohjaimen pinniä täyttä vauhtia, puoli vauhtia ja

pois päältä. Näin varmistuin että jokainen kontakti johtaa virtaa ja sekä moottorit että moottoriohjain toimii. Kuvassa 20 näemme, miten se tapahtuu. Ensin käytetään funktiota analogWrite. Tähän funktioon tarvitsee ilmoittaa PWM-pinnin, johon käsky lähetetään. Pinnin lisäksi funktio tarvitsee arvon kuinka usein pinniin lähetetään käsky. Rivillä 40. annetaan komento analogWrite. Komento kohdennetaan MTR_A1 pinniin ja sille annetaan arvoksi 255, eli maksimi. Tämän komennon jälkeen käytetään funktiota delay, joka käytännössä pyytää mikropiiriä odottamaan sille annetun määrän millisekunteja.
Kuvan 20. mukaisesti moottori-A pyöri ensin täyttä vauhtia eteenpäin puoli sekuntia, sitten hieman hitaammin toiset puoli sekuntia ja sitten pysähtyi. Kun samat komennot lähetettiin MTR_A2 pinnille, moottori pyöri taakse päin samaan tapaan. Toistin nämä komennot myös MTR_B1 ja MTR_B2 pinneille jotta pystin todeta myös toisen moottorin toimivuuden ja pyörimissuunnan. Näin pystyin testiohjelmalla toteamaan sekä laitteiston että ohjelmiston toimivaksi.

5.4.6 Lopullinen ohjelmaa

Aloitin uuden projektin PlatformIO:n ohjatulla työkalulla kuten aikaisemmin. Asetin mainitsemani arvot platformio.ini tiedostoon ja määrittelin pinnit ohjelmatiedoston alussa. Asetin setup funktiossa pinnien tilan sekä aloitin sarjayhteyden sekä USB:lle että BLE-moduulille. Tämän lisäksi ohjelmani tarvitsi muutaman globaalin muuttujan.

```
34  // maximum number of character that can be stored at once from serial
35  const byte numChars = 32;
36
37  // char array to store the chars
38  char receivedChars[numChars];
39
40  // Is new valid data incoming?
41  boolean newData = false;
```

Kuva 21. globaalit muuttujat

Ensiksi on const byte numChars. Tällä muuttujalla määritellään maksimi määrä merkkejä, jonka sarjayhteydestä voidaan vastaanottaa. Sitten luodaan lista, jonka pituus on numChars, eli tässä tapauksessa 32(kuvassa 21.). Ja kolmantena määritellään boolean newData. Tässä muuttujassa pidetään kirjaa siitä, onko luettavassa sarjaportissa uutta dataa.

Oma loop funktioni osoittautui erittäin pieneksi. Kaiken koodin työntäminen yhteen funktioon vaikeuttaa koodin lukemista ja vianetsintää. Tästä johtuen ohjelmani koostuu setup

ja loop funktion lisäksi kahdesta muusta funktiosta; recieveBLEValues ja controlMotors. Ensimmäisen funktion toiminto on lukea BLE-moduulin sarjayhteyttä. Kun kehityspiiri tietää että sarjayhteyden yli on tullut dataa, siihen pääsee käsiksi Serial2.available funktiolla. Tämä näkyy liitteessä 2, rivillä 60. Kun uutta dataa on tulossa ja datan ensimmäinen merkki on ”<”, eli aloitusmerkki, ohjelma alkaa tallentamaan merkkejä receivedChars muuttujaan. Tätä jatketaan niin kauan kunnes saapuu lopetusmerkki, eli ”>”. Kun lopetusmerkki on luettu, asetetaan newData muuttuja todeksi. Toinen vaihtoehto vastaanottamisen päättymiselle on se että merkkijono jota sarjayhteydeltä luetaan, on yli 32 merkkiä.

Kun recieveBLEValues funktio on saanut koko komennon aloitus ja lopetusmerkkien välistä muistiin, ja newData muuttuja on asetettu todeksi, controlMotors funktio voi aloittaa tehtävänsä. Ensin receivedChars muuttujasta parsiaan pois Y-arvo, koska sitä ei käytetä mihinkään. (liite 2, rivi 94) Seuraavaksi katsotaan alkaako tallennettu merkkisarja L- tai R-kirjaimella. Jos kyseessä on L, lähetetään merkkiä seuraava arvo vasemmalle moottorille, muuten oikealle moottorille. (liite 2, rivit 95 ja 112) Kun ohjelma on päättänyt kummalle moottorille komento menee, katsotaan onko arvo negatiivinen vai positiivinen. Jos arvo on suurempi kuin 0, lähetetään taakse päin pyörittävälle pinnille arvoksi 0 ja eteenpäin pyörittävälle pinnille luettu arvo. Suunta jolle ei anneta arvoa, on aina hyvä nollata ettei kehityspiiri yritä pyörittää samaa moottoria eteen- ja taaksepäin samanaikaisesti. Kun pinnille on asetettu PWM-arvo, controlMotors funktio asettaa newData muuttujan epätodeksi. Kun newData muuttuja on jälleen epätosi, recieveBLEValues funktio voi jälleen alkaa lukea uutta arvoa mitä BLE-moduulin sarjayhteydeltä tulee.

Tämän jälkeen lähdekoodi käännetään ja ohjelmoidaan kehityspiirille PlatformIO:n työkalujen avulla. Robotin lähdekoodi on tämän dokumentin liitteenä (liite 2) kommentoituna jotta sitä olisi helppo katselmoida.

6 Pohdinta

Työn tutkimusosuuteen kuului tutkia mikropiirejä, sulautettuja järjestelmiä sekä kehityspiirejä ja alustoja. Tämän työn tuloksena saatiin laaja näkemys siitä, mitä ne ovat ja missä niitä käytetään. Aloittelijoiden ja aiheesta kiinnostuneiden kannalta on loistava tilanne alkaa selvittämään miten mikropiirit, sulautetut järjestelmät ja niiden ohjelmointi toimii. Jos menemme muutaman vuosikymmenen taaksepäin, nämä suuret määrät tietoa olisivat olleet todella vaikea löytää. Mikrokontrollerien ohjelmointi ei ole koskaan ollut helpompaa. Tästä voidaan kiittää sekä Arduino:a että jatkuvasti kehittyvää jakamistaloutta.

Arduino:n menestyksen myötä, ihmiset ovat alkaneet luomaa mitä erikoisempia tuotoksia. Arduinon ja näiden tuotoksien ympärille on kasvanut valtavia yhteisöjä, jotka ovat jatkokehittäneet Arduino:n visiota. Arduino:n saavutuksien vuoksi, ovat monet valmistajat alkaneet tuottamaan omia kehityspiirejään ja alustojaan. Nämä muut alustat ovat myös kasvattaneet ympärilleen omat loistavat yhteisönsä. Ja näiden yhteisöjen myötä kehityspiirien ohjelmointi ei ole koskaan ollut helpompaa. Ennen kaikki tehtiin käyttäen C- tai C++-ohjelmointikieltä, kun taas nykyään voidaan tehdä omia tuotoksia käyttäen esimerkiksi JavaScriptiä. Eli pienellä opiskelulla ja muutamalla eurolla kuka vain voi aloittaa sulautettujen järjestelmien, IoT-sovellusten tai omien robottien ohjelmoinnin.

Vaikka Arduino ja ESP32 ovat erittäin suosittuja kehitysalustoja, on vaikea olla valitsematta STM32-pohjaisia kehityspiirejä. Vertailuissa ne eivät olleet kaikkein tehokkaimpia, mutta kaikkein halvimpia. Paljon painavampia syitä STM32:en puolesta olivat oikean maailman käyttö ja työllistymismahdollisuudet. STMicroelectronics on jo yli 30 vuotta tuottanut mikropiirejä, joita käytetään kaikkialla maailmalla. Tietokoneet, sähköpyörät, ajoneuvot ja jopa lääketeknologia ovat kaikki aloja, joissa käytetään STM32-piirejä. Kun sulautetuissa järjestelmissä on paljon töitä tarjolla jopa Suomessa, oli STM32 varma valinta.

Työn lopuksi halusin selvittää, miten STM32-pohjainen kehityspiiri soveltuu robotin luomiseen aloittelijalle. Vaikka minulla on aikaisempaa kokemusta 3D-mallinnuksesta ja ohjelmoinnista, en ollut koskaan tutustunut sulautettuihin järjestelmiin, robotteihin tai niiden kehittämiseen. Aloittelijoiden suureksi iloksi PlatformIO-työkalu tekee kehitysympäristön pystyttämisestä todella helppoa. Valitset vain laitteen, jota käytät ja millä ohjelmistokehyksellä haluat ohjelmoida. Jakamistalouden yleistymisen myötä, internet on täynnä ohjelmia ja ohjeita, joiden avulla voit sekä suunnitella että toteuttaa robotteja, usein vielä ilmaiseksi. PlatformIO:n lisäksi Arduino:n ohjelmistokehys on käännetty todella monelle kehityspiirille

ja mikrokontrollerille. Näistä syistä kehityspiirien ohjelmoinnin aloittaminen on todella helppoa.

Koen myös että lukuisten avointen työkalujen ja ohjelmistojen käytön jälkeen, haluan itse antaa takaisin jakamistalouden tavoin. Lisään luomani 3D-mallit ilmaiseen jakeluun thingiverse-sivustolle ja kirjoittamani lähdekoodin github:iin kaikkien nähtäväksi. Sain itse suurta intoa robotiikkaan ja sulautettuihin järjestelmiin tutkittuani aiheita. Toivon että jaetut tuotokseni loisivat samanlaista into muissa ihmisissä, sillä se olisi esimerkillinen tapa toteuttaa jakamistaloutta. Pyrin robotin mallinnusvaiheessa pitämään taka-ajatuksena sen, että joku (itseni mukaan lukien) saattaisi haluta tehdä malleista muokattuja versioita. Tästä johtuen aion jakaa myös alkuperäiset CAD-tiedostot, jotka mallinnuksen yhteydessä syntyivät. Tämä helpottaa muita ihmisiä muokkaamaan 3D-mallia.

Olen suuresti yllättynyt siitä, että projektin aikana tuli hyvin vähän vastoinkäymisiä. Suurin ongelmistani oli varmasti BLE-moduulin dokumentaation puutteellisuus. Tämäkin osoittautui mukavaksi oppimiskokemukseksi, josta kävi ilmi että joillain piireillä ja moduulityypeillä saattaa olla universaaleja yhtäläisyyksiä. Uskon tämänkin pienen tiedon tuovan joskus hyötyä urallani ohjelmoijana ja harrastajana.

Oman ammatillisen kehittymisen kannalta projekti oli loistava idea. Pääsin opettelemaan uusia työkaluja ja alustoja, ohjelmoimaan kielellä jota olen käyttänyt hyvin vähän sekä näyttämään että osaan suunnitella ja toteuttaa kokonaisuuksia. Projekti myös innoitti haluani oppia sulautetuista järjestelmistä enemmän sekä harraste- että ammattimielessä. Jos työnantajani tarvitsee sulautettua järjestelmää, pystyn ainakin konsultoimaan asian suhteen. Tarpeen vaatiessa uskon pystyväni suunnittelemaan ja ohjelmoimaan mahdollisen järjestelmän. Vaikka sulautettuihin järjestelmiin on helpompi päästä sisään kuin koskaan ennen, on C++:n ja sulautettujen järjestelmien mielenkiinto maailmalla ollut laskussa jo vuosia. (StackOverflow trends) Kehittääkseni omaa ammatillista osaamistani sulautettujen järjestelmien parissa, seuraava askel olisi varmasti PCB-suunnittelu ja tarkemmin elektroniikkakomponenttien tutkiminen.

6.1 Robotin jatkokehitys

Vaikka robotin valmistuminen tuntui mahtavalta saavutukselta, se on loppupeleissä hyvin yksinkertainen. Robottia ohjataan langattomasti bluetooth-yhteyden välityksellä. Robotti ajaa eteenpäin ja taaksepäin, se kääntyy sekä vasemmalle että oikealle ja näiden yhdistelmiä. Robottia voidaan ladata poistamatta virtalähdettä robotin sisältä.

Robotin jatkokehitysmahdollisuudet ovat kuitenkin lähes loputtomat. Minulla on 3D-tulostin ja kyky luoda 3D-malleja. Tällä yhdistelmällä pystyn luomaan robotille mitä tahansa lisärakenteita voin kuvitella. Tämän lisäksi olen opinnäytetyön myötä oppinut miten sarjayhteydet toimivat. Voin siis lisätä lähes mitä tahansa elektronisia komponentteja.

Yksi jatkokehitysidea mitä haudoin jo robotin suunnitteluvaiheessa oli näyttö, joka näyttäisi robotin ’ilmeen’. Jos tähän saisi lisäksi vielä kameran, voisi robotin ohjelmoida kääntymään suuntaan jossa kohde, esimerkiksi ihminen, on ja näyttämään jotain ilmettä. Nyt kun koneoppiminen ja koneäly ovat nousevia aloja, myös niitä voisi mahdollisesti hyödyntää vastaavissa sovelluksissa.

Robotin ohjelmaa on myös mahdollista muokata toteuttamaan sarjan käskyjä ilman ohjausta. Tämä käytännössä sitä että robotin voisi käskeä ajamaa esiohjelmoituja reittejä. Tätä voi vielä jatkokehittää ultraäänianturilla. Ne ovat yleistyneet robottiharrastajien parissa, sillä niiden avulla on helppo luoda itsestään kulkevia robotteja jotka osaavat vältellä esteitä.

Lähteet

3D Printing Industry. 3D HUBS AM TRENDS REPORT REVEALS 3D PRINTING GREW 21% DESPITE COVID-19. Luettavissa: https://3dprintingindustry.com/news/3d-hubs-am-trends-report-reveals-3d-printing-grew-21-despite-covid-19-189087/. Luettu: 21.09.2021

AliExpress. STM32F401CCU6. Luettavissa: https://www.aliexpress.com/item/4000138305460.html. Luettu: 16.11.2021

AliExpress. DC DC Buck Converter 3.3-5.5 to 3.3V. Luettavissa: https://www.aliexpress.com/item/4000491521885.html. Luettu: 16.11.2021

AliExpress. TP4056 Charging Board. Luettavissa: https://www.aliexpress.com/item/32649780468.html. Luettu: 16.11.2021

AliExpress. AT-09 BLE-module. Luettavissa: https://www.aliexpress.com/item/1005003160275028.html. Luettu: 16.11.2021

AliExpress. H-Bridge Motor Driver L298N. Luettavissa: https://www.aliexpress.com/item/4000781763273.html. Luettu: 16.11.2021

Applied Materials. STMICROELECTRONICS: ON TOP OF KEY MARKETS. Luettavissa: https://www.appliedmaterials.com/nanochip/nanochip-fab-solutions/december-2013/stmicroelectronics-on-top-of-key-markets. Luettu: 12.10.2021

Arduino. Introduction. Luettavissa: https://www.arduino.cc/en/guide/introduction. Luettu: 11.10.2021

Circuit Diges. ESP32 Dual Core Programming. Luettavissa: https://circuitdigest.com/microcontroller-projects/esp32-dual-core-programming-using-arduino-ide- Luettu: 13.10.2021

CircuitSchool. What is Arduino, how it works and what you can do with arduino. Luettavissa: https://www.circuitschools.com/what-is-arduino-how-it-works-and-what-you-can-do-with-arduino/. Luettu: 04.10.2021

Embedded.fm. DON'T USE ARDUINO (FOR PROFESSIONAL WORK). Luettavissa: https://embedded.fm/blog/2017/8/12/dont-use-arduino-for-professional-work. Luettu: 08.10.2021

Explore Embedded. Overwiev of ESP32 features. Luettavissa: https://www.exploreembedded.com/wiki/Overview_of_ESP32_features._What_do_they_practically_mean%3F. Luettu: 13.10.2021

freeCodeCamp. What is The C Programming Language? A Tutorial for Beginners. Luettavissa: https://www.freecodecamp.org/news/what-is-the-c-programming-language-beginner-tutorial/#history. Luettu: 16.11.2021

Guru++. What is C++. Luettavissa: https://www.guru99.com/cpp-tutorial.html. Luettu: 17.10.2021

iFixit. Google Pixel C Teardown. Luettavissa: https://www.ifixit.com/Teardown/Google+Pixel+C+Teardown/62277. Luettu: 09.10.2021

iFixit. MacBook Pro 15" Touch Bar Teardown. Luettavissa: https://www.ifixit.com/Teardown/MacBook+Pro+15-Inch+Touch+Bar+Teardown/73395. Luettu 09.10.2021

iFixit. Xiaomi MiJia QiCycle Folding Electric Bike Teardown. Luettavissa: https://www.ifixit.com/Teardown/Xiaomi+MiJia+QiCycle+Folding+Electric+Bike+Teardown/67654. Luettu: 09.10.2021

Interne of Things Agenda. Embedded Systems. Luettavissa: https://internetofthingsagenda.techtarget.com/definition/embedded-system. Luettu: 26.09.2021

Jakamistalous. Jakamistalous mitä se on? Luettavissa: https://jakamistalous.fi/mita-on-jakamistalous/. Luettu: 29.09.2021

Jipa, Bernhard ja Dillenburger. Submillimeter Formwork: 3D-Printed Plastic Formwork for Concrete Elements. Luettavissa: https://www.researchgate.net/publication/327792655_Submillimeter_Formwork_3D-Printed_Plastic_Formwork_for_Concrete_Elements. Luettu: 16.10.2021

KO2. What can you do with C++?. Luettavissa: https://www.ko2.co.uk/what-can-you-do-with-c-plus-plus/. Luettu 17.10.2021

Link Labs. Bluetooth 5.0 and the Industrial Revolution. Luettavissa: https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy. Luettu: 16.10.2021

LinkedIn. Työpaikkahaku: c++ developer. Luettavissa: https://www.linkedin.com/jobs/c%2B%2B-developer-jobs-helsinki/?originalSubdomain=fi. Luettu: 17.10.2021

MedicalDesign & outsourcing. What is an FDM printer?. Luettavissa: https://www.medicaldesignandoutsourcing.com/what-is-an-fdm-printer-and-how-does-it-work/. Luettu: 16.10.2021

Nextpcb Online Quote. Hintalaskuri. Luettavissa: https://www.nextpcb.com/pcb-quote. Luettu: 04.10.2021

Omni-sci. Embedded Systems. Luettavissa: https://www.omnisci.com/technical-glossary/embedded-systems. Luettu: 15.10.2021

Open Source Initiative. The Open Source Definition. Luettavissa: https://opensource.org/docs/osd. Luettu: 21.09.2021

PlatformIO. Documentation. Luettavissa: https://docs.platformio.org/en/latest/. Luettu: 16.10.2021

PlatformIO. What is PlatformIO. Luettavissa: https://docs.platformio.org/en/latest/what-is-platformio.html. Luettu 16.10.2021

ProAkku. Sony VTC5A. Luettavissa: https://proakku.fi/en/tuote/18650-akku-sony-vtc5a-2600mah-35a-us18650vtc5a/. Luettu 15.10.2021

SearchApp. Architecture. Luettavissa: https://searchapparchitecture.techtarget.com/definition/source-code. Luettu: 21.09.2021

Stack Overflow. Trends. Luettavissa: https://insights.stackoverflow.com/trends?tags=stm32%2Carduino. Luettu: 21.09.2021

StackOverflow Trends. Luettavissa: https://insights.stackoverflow.com/trends?tags=c%2B%2B%2Cembedded. Luettu: 01.11.2021

Technopedia. Statically Typed. Luettavissa: https://www.techopedia.com/definition/22321/statically-typed. Luettu: 17.10.2021

Thingiverse. SMARS modular robot. Luettavissa: https://www.thingiverse.com/thing:2662828. Luettu: 16.10.2021

tutorialspoint. What is C++ programming language?. Luettavissa: https://www.tutorialspoint.com/What-is-Cplusplus-programming-language. Luettu: 17.10.2021

u-blox. NINA-W15 series. Luettavissa: https://www.u-blox.com/en/product/nina-w15-series. Luettu: 11.10.2021

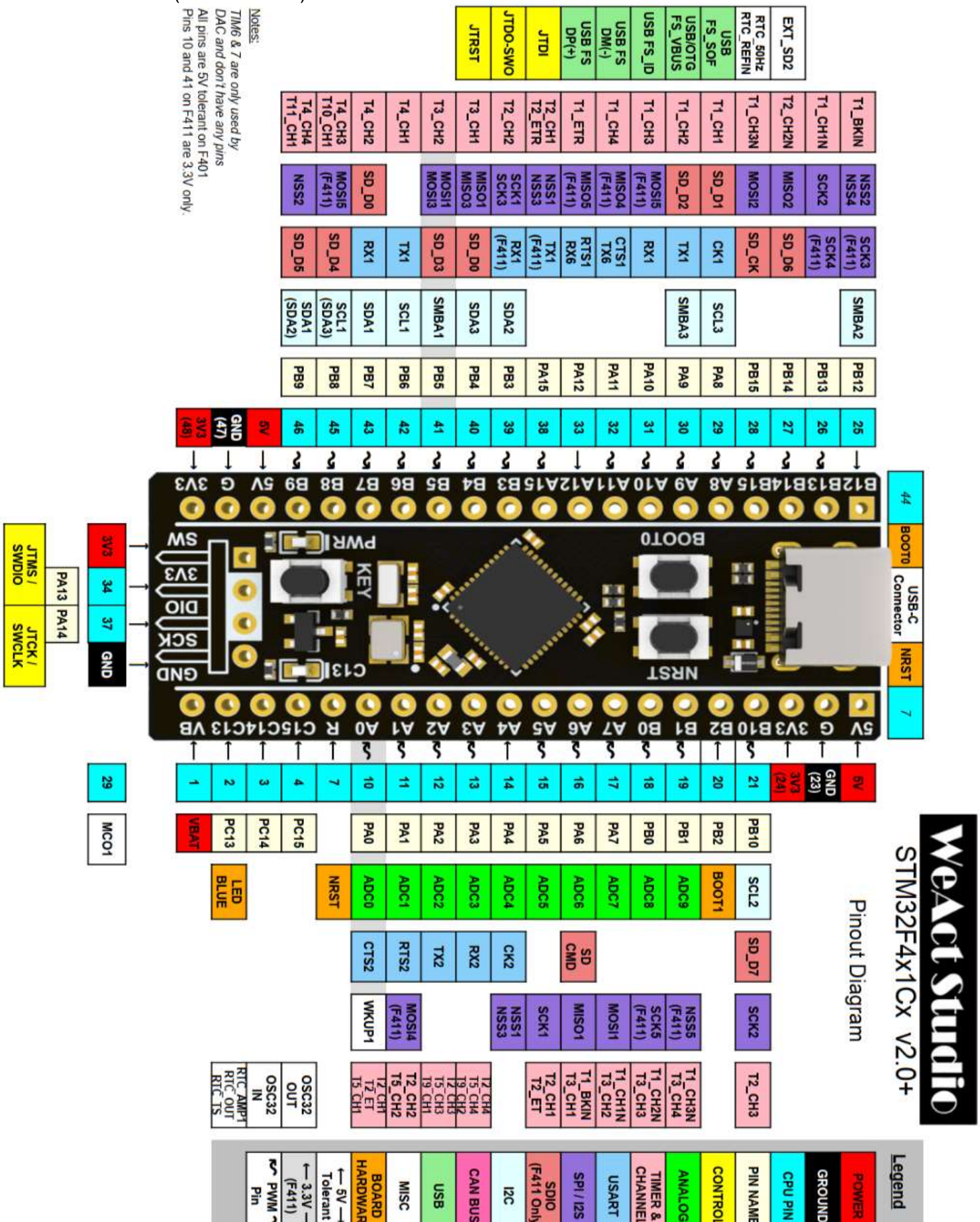
WeAct Github. MiniSTM32F4x1. Luettavissa: https://github.com/WeActTC/MiniSTM32F4x1. Luettu: 17.10.2021

Wikipedia Robot. Luettavissa: https://en.wikipedia.org/wiki/Robot. Luettu: 21.09.2021

Liitteet

Liite 1. STM32F4x1xC pinnikaavio

(WeAct Github)



Liite 2. Robotin lähdekoodi – Main.cpp

```
#include <Arduino.h>
#include <string.h>


// BLE Pins:
#define BLE_RX  PA3
#define BLE_TX  PA2

// Define motor controller pins
#define MTR_A1 PA6
#define MTR_A2 PA7
#define MTR_B1 PB0
#define MTR_B2 PB1

// Prototype functions
void recieveBLEValues();
void controlMotors();

// maximum number of character that can be stored at once from serial
const byte numChars = 32;

// char array to store the chars
char receivedChars[numChars];

// Is new valid data incoming?
boolean newData = false;

void setup()
{
  // Wait for the board and BLE module to properly power up
  delay(3000);

  //Setup motor pins
  pinMode(MTR_A1, OUTPUT);  //Left forward
  pinMode(MTR_A2, OUTPUT);  //Left backward
  pinMode(MTR_B1, OUTPUT);  //Right backward
  pinMode(MTR_B2, OUTPUT);  //Right foward

  //Initiate serial connection between board and the BLE module
  Serial2.begin(115200);
  delay(3000);
}
```

```
44
45 void loop() {
46   recieveBLEValues();
47   controlMotors();
48 }
49
50 // Scans the serial input for start marker (<)
51 // then starts storing data into receivedChars array until end marker (>)
52 // comes in via serial
53 void recieveBLEValues(){
54   static bool recvInProgress = false;
55   static byte ndx = 0;
56   char startMarker = '<';
57   char endMarker = '>';
58   char rc;
59
60   // Read serial
61   while (Serial2.available() > 0 && newData == false) {
62     rc = Serial2.read();
63
64     // if recieve in progress, start storing characters
65     // to receivedChars at index, unless the character
66     // is a endmarker
67     if (recvInProgress == true) {
68       if ( rc != endMarker) {
69         receivedChars[ndx] = rc;
70         ndx++;
71         if (ndx >= numChars) {
72           ndx = numChars -1;
73         }
74       } else {
75         // end marker recieved, slap end string null
76         receivedChars[ndx] = '\0';
77         recvInProgress = false;
78         ndx = 0;
79         newData = true;
80       }
81       // if recieved character is the start marker
82       // set recieve in progress to true
83     } else if (rc == startMarker) {
84       recvInProgress = true;
85     }
86   }
87 }
88
89
90 // Function for sending motor control commands to L298N
91 // All the SerialUSB.print/ln are for debuggin
92 void controlMotors() {
93   // If new data has arrived
```

```
  if (newData == true) {
    char * pch;
    pch = strtok(receivedChars,"Y"); // The app pushed X and Y values, parsing Y here.
    if (pch[0] == 'L') { // If first character of data is L, it is referring to left.
      char* command = pch+2; // Remove axis and side identifier
      int power;
      sscanf(command, "%d", &power); // Serial input parsed into a numbe value (-255 - 255)
      if (power > 0) {
        // forward
        analogWrite(MTR_A2, 0);
        analogWrite(MTR_A1, power);
      } else if (power < 0) {
        // reverse
        analogWrite(MTR_A1, 0);
        analogWrite(MTR_A2, abs(power));
      } else {
        // both off
        analogWrite(MTR_A1, 0);
        analogWrite(MTR_A2, 0);
      }
    } else {
      //right motor
      char* command = pch+2;
      int power;
      sscanf(command, "%d", &power); // Serial input parsed into a numbe value (-255 - 255)
      if (power > 0) {
        // forward
        analogWrite(MTR_B2, 0);
        analogWrite(MTR_B1, power);
      } else if (power < 0) {
        // reverse
        analogWrite(MTR_B1, 0);
        analogWrite(MTR_B2, abs(power));
      } else {
        // both off
        analogWrite(MTR_B1, 0);
        analogWrite(MTR_B2, 0);
      }
    }
    newData = false;
  }
}
```

Liite 3. Robotin komponenttien kytkentäkaavio