Olli Panula-Ontto

Nanomateriaalitulostimen ohjausyksikkö

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikan koulutusohjelma

Insinöörityö

3.5.2018

Metropolia

Tekijä Otsikko Sivumäärä	Olli Panula-Ontto Nanomateriaalitulostimen ohjausyksikkö 19 sivua 3.5.2018
Tutkinto	Insinööri (AMK)
Koulutusohjelma	tietotekniikka
Suuntautumisvaihtoehto	Sulautetut järjestelmät
Ohjaajat	Lehtori Sainio Sami

Insinöörityön tarkoituksena oli suunnitella ja toteuttaa nanomateriaalitulostimen ohjausyksikkö. Nanomateriaalit ovat kasvavassa osassa jokapäiväistä elämää. Näiden materiaalien tutkimus ja ymmärtäminen ovat välttämätön osa uusien ja kestävämpien laitteiden, vaatetuksen ja komponenttien valmistuksessa. Pinnoitettavat materiaalit vaihtelevat, ja pinnoitusprosessia halutaan automatisoida tarvittavan toistettavuuden saavuttamiseksi. Aalto-yliopisto tutkii näitä materiaaleja ja tarvitsi opiskelijakäyttöön soveltuvan helppokäyttöiseen laitteen.

Aiemmin näytteiden teko tapahtui käsin. Vapaalla kädellä ruiskutetut näytteet eivät ole toistettavissa riittävän tarkasti, sillä pinnoitteen paksuus vaihtelee suuttimen nopeudesta ja etäisyydestä riippuen. Lisäksi ruiskutusprosessi tuottaa nanomateriaaliliuossumua, joka on mahdollisesti terveydelle haitallista ja ruiskuttavan henkilön on pukeuduttava raskaaseen suojavarustukseen.

Näitä ongelmia lähdettiin ratkaisemaan suunnittelemalla tavalliseen 3D-tulostimeen ohjausyksikkö, joka mahdollistaa tulostimen käytön nanomateriaaliliuoksen levitykseen mekaanisesti. Mekaaninen levitys mahdollistaa toistettavissa olevien näytteiden teon ja vähentää näytteenottajan riskiä altistua nanomateriaaleille.

Ohjausyksikkö saatiin tuotua pisteeseen, jossa sitä voidaan käyttää nanomateriaalien levitykseen. Laite voidaan myös siirtää koekäytössä käytettyä tulostinta vastaavaan askelmoottorilla toimivaan 3D-tulostimeen. Tämän laitteen avulla nanomateriaalinäytteiden tulostaminen helpottuu huomattavasti ja sallii toistettavissa olevien näytteiden teon.

Avainsanat	FreeRTOS, reaaliaikaohjelmointi, sulautettu Linux

Author Title Number of Pages Date	Olli Panula-Ontto Control unit for a nanomaterial printer 19 pages 3 May 2018
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation Option	Embedded Systems
Instructors	Sainio Sami, Senior Lecturer

The goal of this thesis was to design and implement a control unit for a nanomaterial printer. Nanomaterials are a growing part of our everyday life. Studying and understanding these materials is a vital part of manufacturing new and more durable devices, clothing and components. Substrate materials vary, and coting process needs to be automated to guarantee samples that can be replicated. Aalto university is studying these materials and needed a device that is easy to use and suitable for students.

Before samples were made by hand. Samples sprayed with free hand cannot be replicated with required accuracy because of the thickness of the coating will vary due to speed and distance of the nozzle. In addition, coating process causes nanomaterial fog, that can be hazardous to humans and the person making the samples must wear protective clothing.

To solve these problems a control unit was designed. This control unit allows nanomaterials to be printed with regular 3D-printer. Mechanical dispersion of the nanomaterial solution allows creation of samples that can be replicated and lower the risk of exposure to hazardous materials.

The control unit was able to disperse nanomaterial. The device can also be transferred to a similar printer that uses stepper motors. With this device printing nanomaterials will be easier. and allows creation of samples that can be replicated.

Keywords	FreeRTOS, Realtime Programming, Embedded Linux

Sisällys

Lyhenteet

SCT	State Configurable Timer. Tilapohjainen ajastin LPC1549.
PID	Proportional–Integral–Derivative controller. Verrannollinen Integraali-Derivaatta-kontrolleri.
UART	Universal Asynchronous Receiver-Transmitter.
ADC	Analog-Digital-Converter, käytetään muuttamaan analoginen data digitaaliseksi.
CNC	Computer Numerical Control. Tietokoneen ohjaama laite, jota usein käytetään leikkamiseen tai tulostamiseen.
MSB	Most Significant Bit. Suuriarvoisin bittijonon bitti.
LSB	Least Significant Bit. Pieniarvoisin bittijonon bitti.
WLAN	Wireless Local Area Network. Langaton tietoverkko, jota käytetään laitteiden yhdistämiseen langattomasti.
ROM	Read Only Memory. Haihtumaton muisti, jota käytetään tietokoneissa ja muussa elektroniikassa.
SRAM	Static Random-Access Memory. Puolijohdemuisti.
EEPROM	Electrically Erasable Programmable Read-Only Memory. Haihtumattoman muistin tyyppi jota käytetään tietokoneissa, älykorteissa ja muissa elektronisissa laitteissa.
SPI	Serial Peripheral Interface. Synkroninen kommunikointimetodi, jota käytetään lyhyen matkan viestinnässä.

USB	Universal Serial Bus. Standardi kaapeleille, liittimille, protokollille ja virransyötölle.
SBC	Single Board Computer. Tietokone, joka on rakennettu yhdelle piirilevylle.
TFT	Thin-Film-Transistor.
DSI	Display Serial Interface. Spesifikaatio, joka on suunniteltu vähentämään näytön ohjauspiirien hintaa mobiililaitteissa.

1 Johdanto

Insinöörityönä tehtävän ohjausyksikön ja sen graafisen käyttöliittymän on tilannut Aalto-yliopisto helpottamaan nanomateriaalin tulostusprosessia ja muuntamaan kolmella akselilla toimiva tulostin graafisella käyttöliittymällä varustetuksi, helppokäyttöiseksi nanomateriaalitulostimeksi.

Nanomateriaaleille on käyttökohteita päällysteissä, komposiiteissa ja elektroniikassa. Käyttötarkoitukset vaihtelevat käyttöä kestävistä tekstiileistä nanomateriaaleilla päällystettyihin antureihin, joita voidaan soveltaa esimerkiksi hermovälittäjäaineiden havaitsemiseksi. [1.] Nanomateriaalien kehittäminen riippuu paljon uusista valmistus- ja prosessointiteknologioista, yhdessä näiden pinnoitteiden ymmärtämisen kanssa. Toistettavien pinnoitusten tekeminen on tästä syystä oleellista.

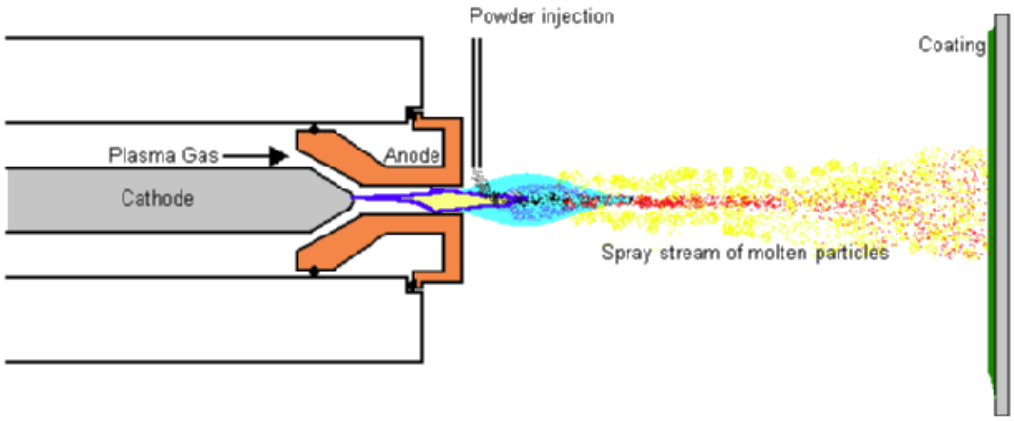
Nanomateriaalipinnoitteiden tutkimuksen kannalta on tärkeää pystyä tuottamaan toistettavia pinnoitteita. Nanomateriaalien kuljettamiseksi pinnoille on valittu korkeapaineruiskutus, joka on aikaisemmin suoritettu maaliruiskulla ihmisen toimesta. Tällä menetelmällä on pystytty tuottamaan ensimmäiset proof-of-concept-tulokset, mutta tämä menetelmä ei ole riittävän toistettava yhtenäisten pinnoitekerrosten valmistamiseksi. Lisäksi pinnoittajien välinen laatuvaihtelu on merkittävää. Näiden ongelmien lisäksi maaliruiskua operoivan on tullut pukeutua raskaisiin suojavarusteisiin estääkseen altistumisen nanopartikkeleille.

Insinöörityössä tehtävä laite mahdollistaa tulostusprofiilien tallentamisen ja lataamisen, ja se toteutetaan helposti integroitavaksi muihin laitteisiin uudelleen kirjoittamalla vain moottorien ohjausta hallitseva osuus. Työssä on tarkoitus valmistaa käyttövalmis ja integroitava laite prototyypin sijaan.

2 Ruiskutuspäällystys

Insinöörityössä toteutettu laite käyttää kylmäruiskutustekniikan kaltaista menetelmää nanomateriaalin levittämiseen kohdealustalle. Tämän tekniikan lisäksi nanomateriaalien levitykseen on myös muita tekniikoita, esimerkiksi: lämpöruiskutus ja plasmaruiskutus. Näihin tekniikoihin verrattuna kylmäruiskutus on helpompi toteuttaa, sillä se ei vaadi ruis-

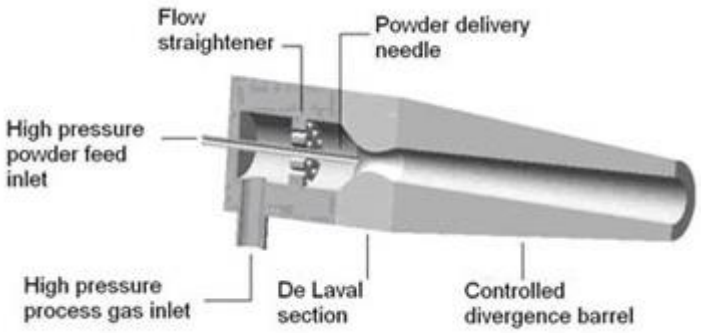
kutettavan materiaalin lämmittämistä. Lämpö- ja plasmaruiskutusmenetelmissä ruiskutettava materiaali lämmitetään yli sulamispisteen. Päällystysmateriaali syötetään erittäin kuumaan plasmaliekkiin, jossa se kuumennetaan nopeasti ja kiihdytetään suureen nopeuteen (kuva1). Kuuma materiaali osuu substraattiin ja jäähtyy nopeasti muodostaen pinnoitteen. [1.]



Kuva 1. Plasmaruiskutusmenetelmän toiminta [7].

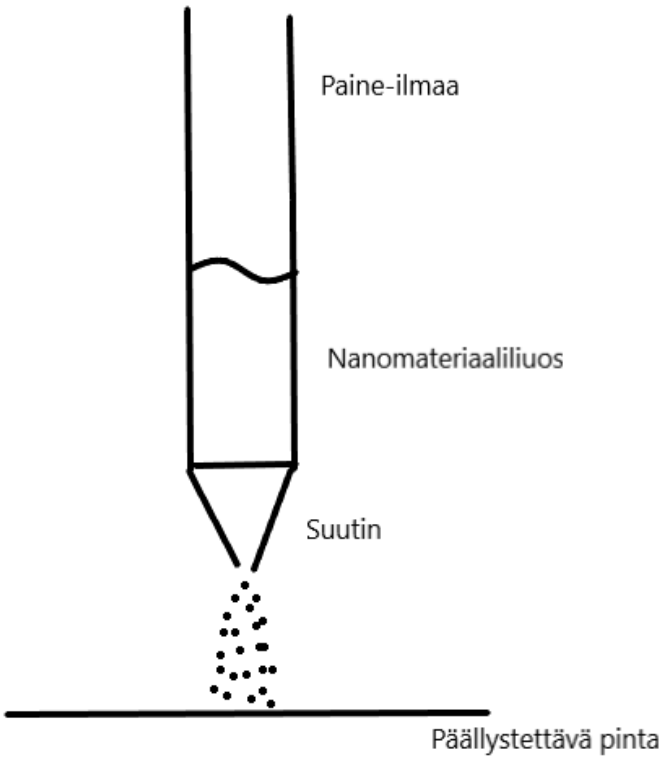
Kylmäruiskutuksen kehittivät alun perin 1980-luvun puolivälissä Anatolii Papyrin ja hänen kollegansa. He onnistuivat päällystämään useita eri materiaaleja puhtailla metalleilla, metalliseoksilla ja komposiiteillä ja osoittivat näin kylmäruiskutusprosessin soveltuvan useisiin käyttötarkoituksiin.

Kylmäruiskutus on prosessi, jossa päällystettävä pinta altistetaan suuren nopeuden omaavalle pienistä partikkeleista koostuvalle suihkulle. Partikkelit kiihdytetään suuriin nopeuksiin paineistetun kaasun avulla, jonka lämpötila on aina matalampi kuin ruiskutettavan materiaalin sulamispiste. Suuri nopeus saavutetaan suppenevan-erkanevan suuttimen avulla, jota kutsutaan myös deLaval-suuttimeksi (kuva 2). Tämä prosessi johtaa päällysteeseen, jossa ruiskutettu materiaali pysyy aina kiinteässä muodossa eikä tästä syystä muodosta kiteitä pinnoitteeseen. [2.]



Kuva 2. DeLaval-suuttimen geometria [3].

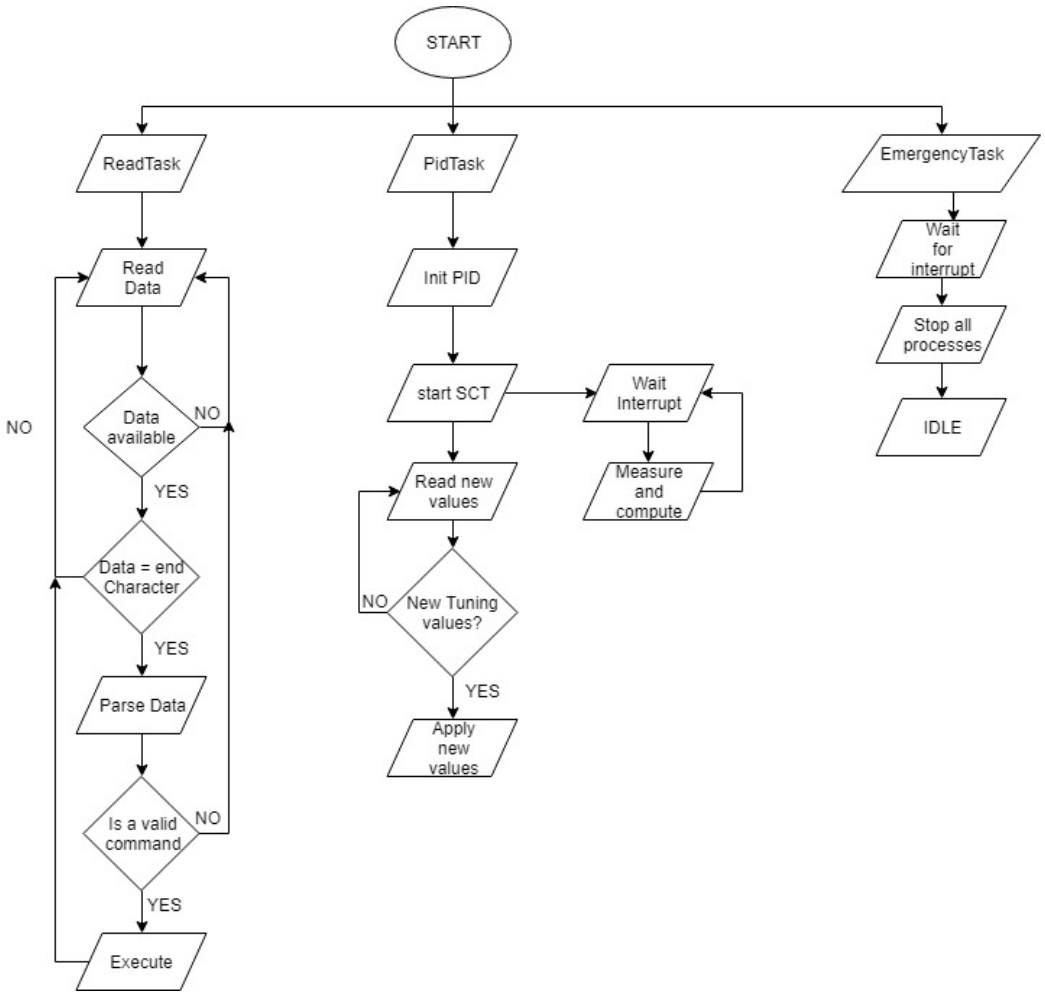
Työssä käytetyn tulostimen ruiskutusmenetelmä eroaa osin kylmäruiskutusmenetelmästä (kuva 3). Kuten kuvasta 2 nähdään, kylmäruiskutusprosessissa päällystysmateriaali syötetään suurinopeuksiseen kaasuvirtaan. Tämän sijasta tulostimessa suuttimeen johtava putki täytetään nanomateriaaliliuoksella, joka työnnetään suuttimen läpi paineilmalla. Ruiskutettava liuos koostuu alkoholista ja nanomateriaalista, ja laitteen lämpöpeti nopeuttaa alkoholin haihtumista.



Kuva 3. Projektissa käytetty ruiskutusjärjestelmä.

3 Käytetyt tekniikat ja teknologiat

Insinöörityössä toteutettu laite koostuu kahdesta osasta: ohjausyksikösta (kuva 4) ja käyttöliittymäyksiköstä. Näiden osien välinen kommunikointi toteutettiin UART-kommunikoinnilla. Käyttöliittymä laitteen tehtävänä on lähettää käyttäjän antamat käskyt ohjausyksikölle, joka käskyt saatuaan suorittaa tarvittavat toimenpiteet. Ohjausyksikkö on myös vastuussa tulostimen lämpöelementin kontrolloinnista. Se on toteutettu käyttäen PID-algoritmia.



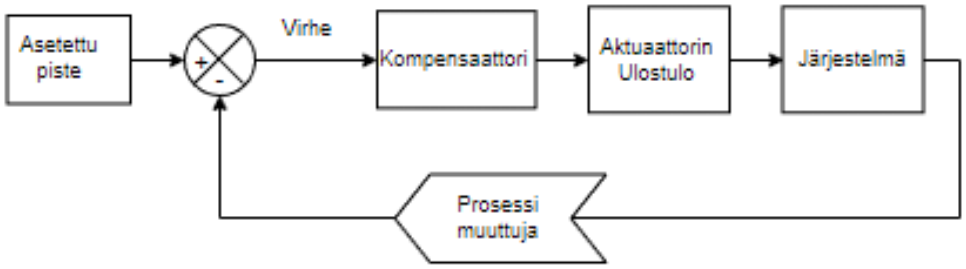
Kuva 4. Ohjausyksikön toiminta.

3.1 PID-kontrolleri

PID-kontrolleri on teollisuudessa yleisimmin käytetty kontrollointialgoritmi [4]. PID:n suosio johtuu osin sen vakaasta suorituskyvystä laajalla alueella ja yksinkertaisuudesta, joka mahdollistaa helpon operoinnin. Kuten nimestä voi päätellä, PID-algoritmi koostuu kolmesta osasta: verrannollisesta, integraalista ja derivaatasta, joita muuttamalla saadaan haluttu tulos. [4.] Käsittelen tässä työssä vain suljetun kierron PID-kontrolleria.

Ohjausjärjestelmä

Perusidea PID-kontrollerissa on lukea arvoja sensorista, minkä jälkeen lasketaan verrannollinen, integraali- ja derivaattavaste ja lisätään nämä kolme komponenttia. Tyypillisessä järjestelmässä prosessimuuttuja on parametri, jota halutaan kontrolloida, esimerkiksi paine tai lämpötila. Asetettu piste kuvaa haluttua prosessimuuttujan arvoa. Näiden kahden arvon eroa käytetään laskettaessa, millä arvolla järjestelmää on ajettava, jotta haluttu arvo saavutetaan. [4.] Tässä työssä PID-kontrolleria käytettiin lämpöelementin säätämiseen ja kyseessä oli suljetun kierron kontrolleri (kuva 4).



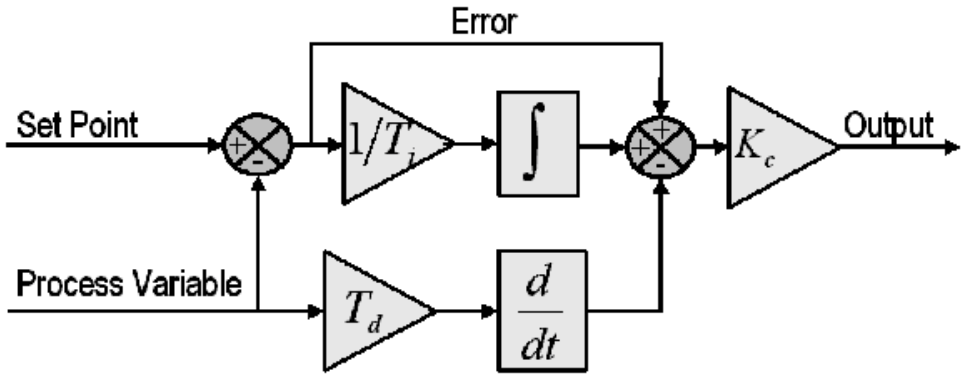
Kuva 5. PID-kontrollerin lohkodiagrammi.

PID

Verrannollinen osuus algoritmissa on riippuvainen vain asetetun arvon ja prosessimuuttujan erosta. Tätä eroa kutsutaan virhemuuttujaksi, ja se määrittää suhteen ulostulon vasteen ja virhemuuttujan välillä. Jos suhteellinen saanti nousee, nousee myös ulostulon vastenopeus. Jos virhemuuttuja on liian suuri, prosessimuuttuja alkaa aaltoilla ja järjestelmästä tulee epävakaa. [4.]

Integraaliosa laskee yhteen virheen ajan kuluessa. Tästä seuraa integraaliosan hidas kasvu, jos järjestelmässä on pienintäkään virhettä. Tarkoitus on saada vakaan tilan virhe nollaan. Vakaan tilan virhe on lopullinen ero asetetun pisteen ja prosessimuuttujan välillä.

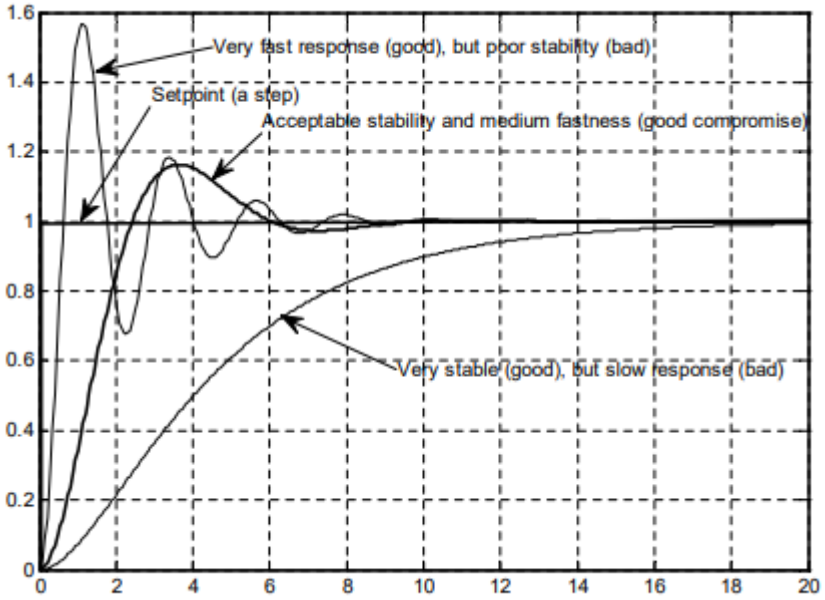
Derivaattakomponentti laskee ulostuloa, jos prosessimuuttuja kasvaa nopeasti. Derivaattakomponentin kasvattaminen johtaa vahvempaan reaktioon virhetermin muuttuessa (kuva 6). [4.]



Kuva 6. Yksinkertaisen kontrollerin lohkokuvaaja [4].

PID-kontrollerin hienosäätö

PID-kontrollerit tarvitsevat hienosäätöä toimiakseen. Hienosäätötavat voidaan jakaa kahteen pääkategoriaan: suljetun ja avoimen silmukan hienosäätöön. Suljetun silmukan metodia käytetään, jos säädettävä kohde operoi automaattisessa tilassa. Vastaavasti avoimen silmukan säätöä käytetään kohteen ollessa manuaalisessa tilassa. Molemmissa kategorioissa on useita eri tekniikoita hienosäätöön. Kaikkien tekniikoiden päämäärä kuitenkin on sama: saada nopea vaste ja vakaa järjestelmä. Nopea vaste ja vakaa järjestelmä vaikuttavat kuitenkin toisiinsa, kuten kuvasta 7 huomataan. Suuri vastenopeus johtaa epävakaaseen järjestelmään, ja erittäin vakaalla järjestelmällä on hidas vasteaika. [5.]



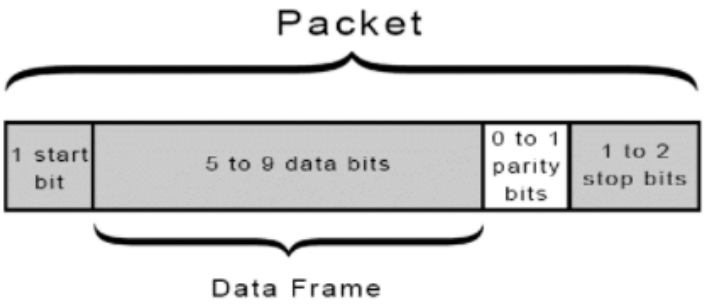
Kuva 7. Nopean vasteen ja vakaan järjestelmän omaavien järjestelmien suhde [5].

3.2 UART-protokolla

UART-kommunikoinnissa kaksi UART-laitetta kommunikoi suoraan keskenään. Lähettävä laite muuntaa ohjaavan laitteen rinnakkaisdatan sarjamuotoiseksi ja lähettää sen vastaanottavalle puolelle, joka muuttaa sarjamuotoisen datan rinnakkaiseksi.

UART siirtää dataa epäsynkronisesti, mikä tarkoittaa, että kahden laitteen välillä ei ole erillistä kellosignaalia tahdistamassa lähetettyjä bittejä. Tämän sijasta UART lisää aloitus- ja lopetusbitin lähetettävään datapakettiin. Nämä bitit kertovat nimensä mukaan paketin alun ja lopun, jotta vastaanottava UART tietää, milloin lukea bittejä. Kun vastaanottaja havaitsee aloitusbitin, se alkaa lukea bittejä taajuudella, jota kutsutaan baud-nopeudeksi. Baud-nopeus on yksikkö, joka kertoo, montako bittiä sekunnissa siirretään. Kommunikointi toimii vain, jos molemmat osapuolet toimivat samalla taajuudella. [6.]

UART-datapaketti koostuu aloitusbitistä, databiteistä, parillisuusbiteistä ja lopetusbiteistä (kuva 8).



Kuva 8. UART-datapaketti [6].

Yleisesti datalinjaa pidetään korkeassa jännitetasossa, kun liikennettä ei ole. Aloittaakseen lähetyksen lähettävä puoli laskee jännitetason korkeasta matalaksi yhden kellojakson ajaksi. Tämä muutos on aloitusbitti, jonka jälkeen vastaanottaja aloittaa bittien lukemisen. [6.]

Datakehys (kuvassa 8 Data Frame) on kooltaan parillisuudesta riippuen viidestä yhdeksään bittiä. Tämä osa paketista sisältää itse datan. Useimmiten paketin lukeminen aloitetaan vähiten merkitsevästä bitistä (LSB), mutta on myös mahdollista lukea bitit toisessa järjestyksessä merkitsevimmästä bitistä (MSB) aloittaen. [6.]

Datapaketin parillisuusosiota käytetään paketin eheyden tarkistamiseen. Kun paketti on vastaanotettu, siitä lasketaan bitit, joiden arvo on 1. Jos näiden bittien lukumäärä on parillinen ja parillisuusbitti on nolla, tiedetään, että siirto tapahtui ilman virheitä. Jos parillisuusbitti ja laskettujen bittien arvot eivät ole samat, on tapahtunut virhe [6.].

Lopetusbitin on määritelty olevan jännitemuutos matalasta korkeaksi vähintään kahden kellojakson ajaksi. [6.]

Näitä UART-asetuksia voidaan muokata suhteellisen vapaasti. Joissain toteutuksissa ei käytetä parillisuusbittejä ja lopetusbitin pituus on vain yhden kellojakson mittainen. [6.]

3.3 G-koodi

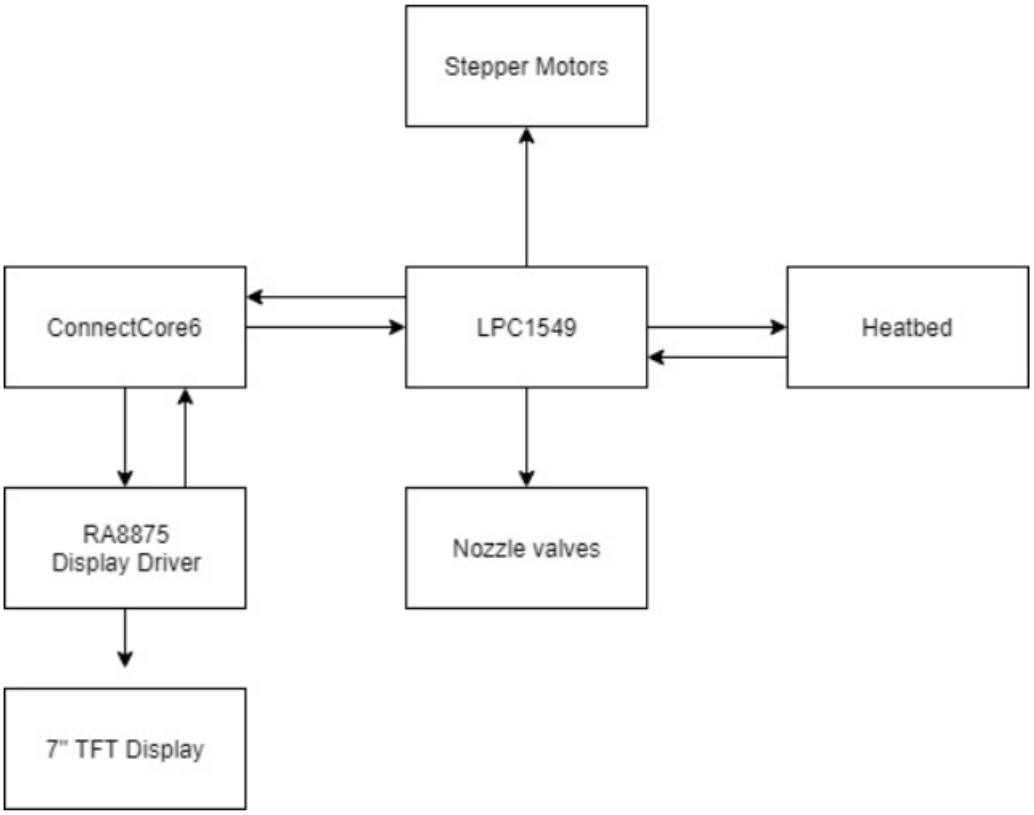
G-koodi, tunnettu myös nimellä RS-274, on ohjelmointikieli, jota käyttävät useat tulostimet, 3D-tulostimet ja muut CNC-tyyppiset laitteet.

G-koodi kertoo laiteelle, kuinka paljon ja millä nopeudella moottoreita on siirrettävä. Nämä koodit koostuvat useasta osasta. Esimerkin vuoksi tarkastellaan koodia ”G1 X130”. Ensimmäinen osio ”G1” kertoo laiteelle, että komento on valmisteleva lineaarine-ninterpolaatio komento. Seuraava osio ”X130” kertoo laitteelle, millä akselilla liikutaan, ja laitteen konfiguroinnista riippuen, kuinka paljon tai mihin koordinaattiin tulee liikkua. G-alkuisten käskyjen lisäksi usein käytettään myös M-kirjaimella alkavia. Näitä komentoja kutsutaan sekalaisiksi käskyiksi ja käytetään sovelluskohtaisissa käskyissä, esimerkiksi suuttimen asennon muokkaamiseen.

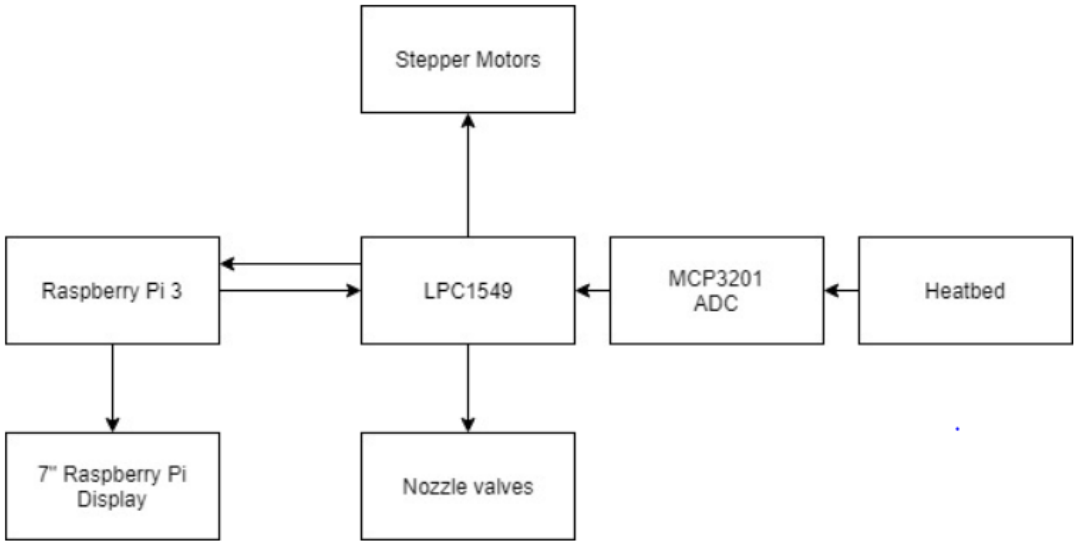
Tässä projektissa G-koodilla annettiin liikkumiskäskyjen lisäksi käskyt avata ja sulkea suutin, muuttaa nopeutta ja säätää lämpöelementin lämpötilaa.

4 Laitevalinnat

Tässä luvussa käydään läpi projektiin valittuja laitteita ja syitä, miksi juuri näihin laitteisiin päädyttiin. Laitevalinnat muuttuivat projektin aikana, kuten kuvista 9 ja 10 nähdään.



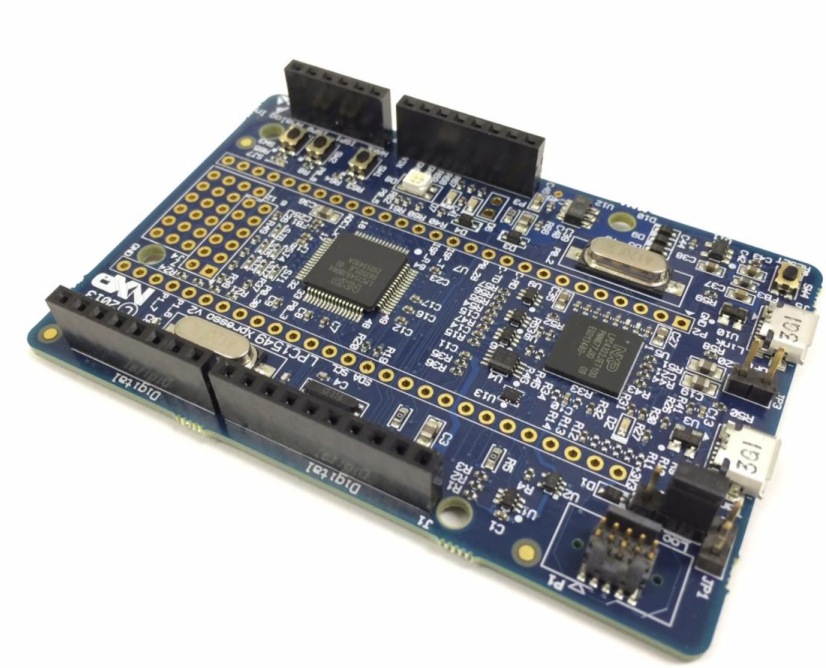
Kuva 9. Alkuperäiset laitevalinnat.



Kuva 10. Laitevalinnat projektin lopussa.

4.1 LPC1549-laite

LPC1549 (kuva 11) on LPCXpresso-tuoteperheeseen kuuluva alusta, jossa on ARM Cortex-M3:een pohjautuva 32-bittinen mikrokontrolleri. LPC1549 kykenee operoimaan 72 MHz:n taajuuteen asti. Corex-M3-prosessori käyttää Harvard-arkkitehtuuria.



Kuva 11. LPC1549-kehitysalusta [8].

Alustassa on 256 kilotavua flash-muistia, 32 kilotavua ROM-muistia, 4 kilotavun EEPROM ja 32 kilotavua SRAM-muistia. Oheisliitäntöihin sisältyy muun muassa yksi USB 2.0-portti, kaksi SPI-liittymää, SCT-ajastimia ja 12-bittisiä ADC-muuntimia. [7.] Mikrokontrolleri sisältää useita ajastimia ja mahdollisuuden käyttää reaaliaikakäyttöjärjestelmää. Tämä mikrokontrolleri valittiin projektiin aiemman kokemuksen perusteella.

Ongelmat

Aikaisemman käyttökokemuksen perusteella LPC oli havaittu luotettavaksi ja helppokäyttöiseksi laitteeksi. Projektia tehtäessä laitteesta kuitenkin löytyi useita ongelmia, joista osan epäiltiin johtuvan suunnitteluvirheestä.

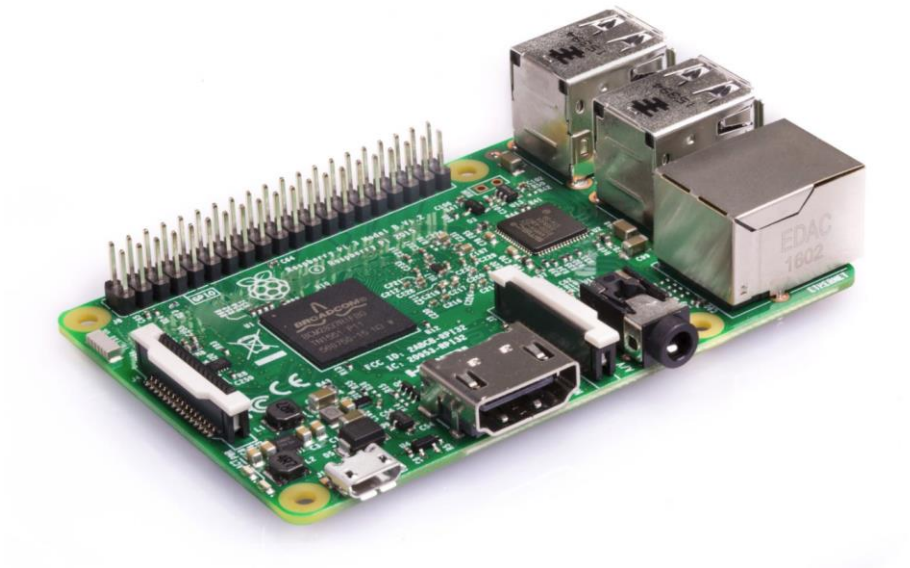
Ohjausyksikkö tarvitsee ACD-muuntimen pystyäkseen lukemaan lämpöelementtiin kytketyn sensorin arvoja, joita tarvitaan lämpötilan pitämiseen tarvittavissa rajoissa. LPC:n tapauksessa ADC on erittäin epätarkka. Useimmiten 12-bittisen ADC:n virhe on +-1

LSB:n luokkaa, mutta LPC:n kohdalla virhe on +-1 MSB. Käytännössä tämä tarkoittaa, että virheen sattuessa ADC:n virhe on noin 10 millivoltin sijasta noin 2,5 volttia. Tämä virhe tekee LPC:n omasta muuntimesta käyttökelvottoman.

Laitteiden välinen kommunikointi toteutettiin UART-protokollaa käyttäen. LPC tarjoaa useita mahdollisuuksia UART-kommunikointiin. UART-kommunikointitapa jouduttiin vaihtamaan projektin loppupuolella. Aikaisemmin käytetty tapa ei ole mahdollinen, sillä se on tarkoitettu toimivaksi vain debuggerin ollessa läsnä. Kommunikointitapaa vaihtaessa selvisi, että kytkennän kannalta helpoin vaihtoehto USB-UART toimii keskeytyskeskeisellä toteutuksella hyvin epäluotettavasti ja jumiutuu joissakin tapauksissa.

4.2 Raspberry Pi 3 Model B -laite

Raspberry Pi 3 (kuva 12) on kolmannen sukupolven Raspberry Pi SBC (Single Board Computer) -laite, jossa on ARMv8-pohjainen 64-bittinen, neljällä ytimellä varustettu keskusyksikkö. Laiteessa on WLAN ja Bluetooth-tuki, 1 Gb RAM-muistia sekä neljä USB-porttia.



Kuva 12. Raspberry Pi 3 [9].

Raspberry Pi -laitteet ovat erittäin suuressa suosiossa harrastelijoiden keskuudessa niiden edullisen hinnan ja helpon saatavuuden takia. Ammatilliseen käyttöön kyseiset laitteet soveltuvat korkeintaan prototyyppien kehittämiseen. Laitteeseen päädyttiin ajan puutteen takia.

Raspberry Pi 3:n suurin ongelma aktiivikäyttöön suunnitellussa laitteessa on Linux-käyttöjärjestelmän mahdollinen korruptoituminen, jos laitetta ei suljeta oikein virran poiston yhteydessä.

4.3 Kosketusnäyttö

Projektin alussa oli tarkoitus käyttää 40 pinnin TFT-kosketusnäyttöä yhdessä RA8875-ohjaussirun kanssa. Tästä yhdistelmästä kuitenkin luovuttiin, kun huomattiin kosketusominaisuuden toteuttamisen olevan äärimmäisen vaikeaa toteuttaa Raspberry PI 3:lle. Lisäksi tämän päivitystaajuuden huomattiin olevan liian hidas.

Aiemmin käytetty näyttö korvattiin Raspberry Pi:n valmistamalla näytöllä, joka on suunniteltu käytettäväksi Raspberry Pi -tuotteiden kanssa. Näyttö käyttää DSI-liuskaliitintä, ja siinä on kapasitiivinen kosketusnäyttö. Liitännän ja valmiin ohjauspiirin ansiosta näyttö on erittäin helppo ottaa käyttöön. Näyttö tukee kymmentä yhtäaikaista kosketusta ja suurta päivitystaajuutta. Näytön resoluutio on 800 x 480 ja koko seitsemän tuumaa.

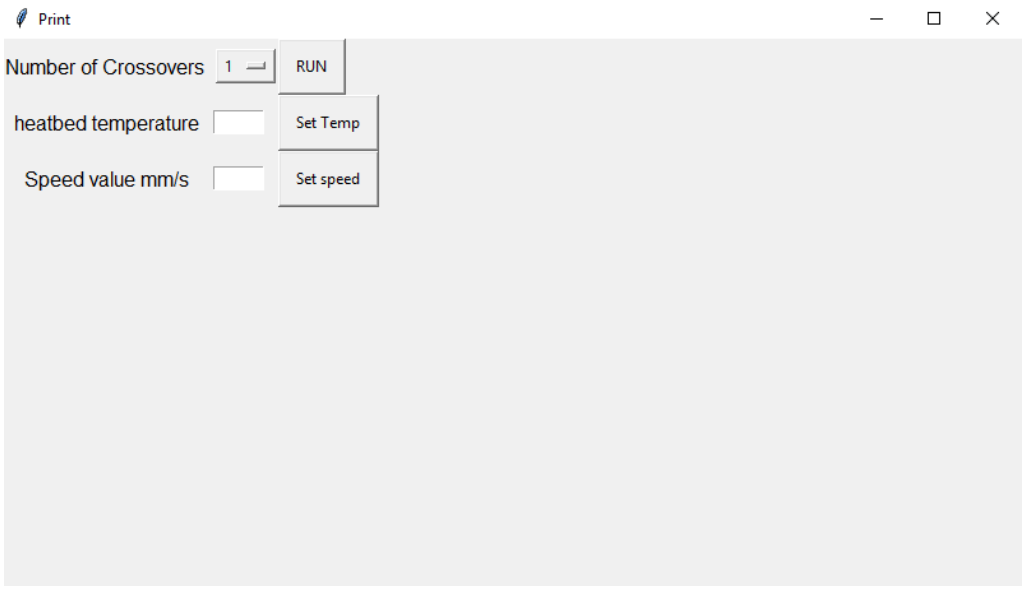
5 Laitekokonaisuuden toteutus

Ohjausyksikön suunnitteluvaiheessa harkitsin kahta vaihtoehtoa, Linux-pohjaista laitetta ja LPC1549-laitetta RTOS-käyttöjärjestelmällä. Päädyin toteuttamaan ohjausyksikön LPC-laitteelle muutaman keskustelun ja tutkimuksen tuloksena. Ongelma Linux-pohjaisessa laitteessa oli ajastus: Linux käyttöjärjestelmää on erittäin vaikeaa saada suorittamaan ohjelman säikeitä riittävällä tarkkuudella. Linux tarjoaa pehmeän reaaliaikakäyttäytymisen. Tämä tarkoittaa, että ydin koettaa ajoittaa ohjelmat asetettujen aikarajojen sisällä, mutta ei takaa sen onnistumista. [10.]

Laitteiden välinen kommunikointi toteutettiin UART-protokollaa ja G-koodimallia käyttäen. UART-kommunikointi toteutettiin keskeytyspohjaisella lähestymistavalla kehäpuskureita käyttäen. Tässä lähestymistavassa laite odottaa viestiä, jonka saavuttua tapahtuu keskeytys. Keskeytyksen tapahtuessa laite lukee vastaanotetun viestin ja palauttaa sen.

Käyttöliittymäyksikkö

Käyttöliittymä toteutettiin Python-ohjelmointikielellä Raspberry PI 3 alustalle. Laitteeseen liitettiin 7 tuuman kosketusnäyttö lisäämään helppokäyttöisyyttä. Käyttöliittymä koostuu kolmesta osasta: lämpötilan säädöstä, liikkumiskäskystä ja nopeudesta (kuva 13).



Kuva 13. Ohjausyksikön Python-käyttöliittymä.

Yhtä kuvan 13 napeista painettaessa käyttöliittymä lukee arvon napin viereisestä kentästä, muokkaa sen oikeaan muotoon ja lähettää tämän jälkeen ohjausyksikölle tulkittavaksi.

Käyttäytyminen

Ohjausyksikkö koostuu kolmesta päätehtävästä (Task). Käynnistyksen yhteydessä nämä tehtävät annetaan vuorottajalle, jonka tehtävänä on ajoittaa tehtävien suoritus. Kuten esimerkkikoodista 1 nähdään jokainen tehtävä saa oman prioriteettiarvon (tskIDLE_PRIORITY + nUL). Arvojärjestystulkinta on suuruusjärjestyksessä siten, että korkeimman arvon saanut tehtävä saa korkeimman prioriteetin.

```
/* Task 1 thread */
xTaskCreate(readTask, "readTask", configMINIMAL_STACK_SIZE * 2, NULL,
        (tskIDLE_PRIORITY + 2UL), (TaskHandle_t *) NULL);

/* Task 2 thread */
xTaskCreate(pidTask, "pidTask", configMINIMAL_STACK_SIZE * 2, NULL,
        (tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

/* Task 3 thread */
xTaskCreate(emergencyTask, "emergencyTask", configMINIMAL_STACK_SIZE * 2,
        NULL, (tskIDLE_PRIORITY + 3UL), (TaskHandle_t *) NULL);

/* Start the scheduler */
vTaskStartScheduler();
```

Esimerkkikoodi 1. Tehtävien luonti ja vuorottajan käynnistys.

Korkeimman prioriteetin tehtävä emergencyTask suunniteltiin keskeyttämään kaikki toiminta, jos yksikään laitteen päätykytkimistä on suljettu. Jos päätykytkin on suljettu, laite saa laitteistokeskeytyksen ja siirtyy toimettomaan tilaan. Tämä ehkäisee moottorien vaurioitumisen, jos laite koettaa saavuttaa koordinaation joka on fyysisesti laitteen saavuttamattomissa. Ainoa tapa palata tehtävästä on sammuttaa virta ja siirtää kiskot siten, että päätykytkimet ovat auki, ja käynnistää laite uudelleen.

Toiseksi korkein prioriteetti on tehtävällä readTask. Tämä tehtävä alustaa moottorit, lukee laitteelle lähetettyä tietoa, tulkitsee sen ja suorittaa vaaditun tehtävän. Tehtävällä on toiseksi korkein prioriteetti, jotta lähetetyt viestit saadaan kokonaisuudessaan luettua. Saadessaan käskyn ohjelma parsii komennon osiin ja suorittaa vaaditun tehtävän tai palauttaa virhearvon. Esimerkiksi laitteen saadessa käskyn liikkua X-akselilla käytetään komentoa ”G1 X130”. Ohjelma valitsee komennon tyypin (Esimerkkikoodi 2), G- tai M-tyyppi, etsii ja suorittaa oikean komennon (esimerkkikoodi 3), tässä tapauksessa komennon 1: liiku X-akselilla 130 millimetriä.

```
int GCodeParser::parse(char *command){
    char variable;
    int code;
    char parameters[50];
    sscanf(command, "%c %d %[^\t\n]", &variable, &code, parameters);
    switch(variable){
    case 'G':
        return parseG(code, parameters);
    case 'M':
        return parseM(code, parameters);
    default:
        return -1;
    }
    return 0;
}
```

Esimerkkikoodi 2. Kirjainkoodin tulkitsija.

```
int GCodeParser::parseG(int code, char *parameters){
    char x, y;
    float xVal, yVal;
    switch(code){
    case 1:
        //move
        sscanf(parameters, "%c %f %c %f", &x, &xVal, &y, &yVal);
        motorController.move(xVal, (int)yVal);
        putStrUart("ok\r\n");
        return 0;
    case 28:
        //move home
        Board_UARTPutSTR("OK \n");
        return 0;
    default:
        return -1;
    }
    return 0;
}
```

Esimerkkikoodi 3. G-koodin parsinta.

Kolmas tehtävä säätelee tulostimen lämpöelementtiä pitäen sen halutuissa rajoissa. Tehtävä käynnistää SCT-ajastimen, alustaa PID-kontrollerin ja tarvittaessa vastaanottaa ja asettaa uudet arvot PID-kontrollerille (esimerkkikoodi 4). SCT-ajastimen tehtävänä on mitata 100 millisekunnin välein arvoja lämpöelementistä ja laskea uudet arvot, jotta lämpötila pysyisi halutuissa rajoissa.

```
void pidTask(void *pvParameters) {
    double *input;
    double *output;
    double *setpoint;
    double i=1.9;
    double s = 5;
    input = output = &i; //DEBUGVALUES
    setpoint = &s;
    double kp, kd, ki;
    kp = kd = ki = 0.5;
    pid = new PID(input, output, setpoint, kp, ki, kd);
    PID_SCT();
    while (1) {
        if (xSemaphoreTake(taskMutex,portMAX_DELAY) == pdTRUE) {

            if(xQueueReceive( xPIDQueue, &( msg ), ( TickType_t ) 10 ) == pdPASS){
                kp = msg.kp;
                ki = msg.ki;
                kd = msg.kd;
                if(pid->getKp() != kp || pid->getKi() != ki || pid->getKd() != kd){
                    pid->setTuning(kp, ki, kd);
                }
            }
        }
        vTaskDelay(configTICK_RATE_HZ / 500);
    }
    while(1);
}
```

Esimerkkikoodi 4. pidTaskin toiminta.

PID-kontrollerin hienosäätöä helpottamaan ohjelmaan lisättiin osio joka mahdollistaa PID-termien muokkauksen ajon aikana.

6 Tulokset ja yhteenveto

Insinöörityöraportin kirjoituksen aikaan laitekokonaisuus oli toiminnallinen lämpöpetiä ja tulostimen Z-akselia lukuun ottamatta. Todettiin, että laite tulee helpottamaan näytteiden tekoa huomattavasti. Asiakkaan asettamiin tavoitteisiin päästiin osin ja laitetta jatkokehitetään, jotta tavoitteet saataisiin suurilta osin täytettyä.

Suurin vastaan tullut ongelma olivat projektin suunnitteluvaiheessa tehdyt erittäin huonot laitevalinnat. Näiden valintojen vuoksi tehtiin paljon turhaa työtä, eikä alun perin valituista laitteista käytetty kuin yhtä (LPC1549). Muita, pienempiä ongelmia aiheuttivat LPC1549:n ADC-muunnin, sekä USB-UART. En suosittelisi laitetta käytettäväksi projekteissa, jotka vaativat ADC-muuntimen käyttöä.

Jatkokehityksessä, käyttöliittymän ulkoasua ja toiminnallisuutta voitaisiin parantaa merkittävästi. Käyttäjäprofiilien tallennus ja lataus olisi yksi jatkokehitysaihe. Lisäksi laitteelle voitaisiin suunnitella kotelo ja piirilevy, johon kaikki liittimet voitaisiin kytkeä siististi ja varmasti.

Vaikka projektia ei saatu täysin vastaamaan vaatimuksia, siitä saatiin hyödyllistä kokemusta vastaavien projektien kehitykseen. Jos projekti toteutettaisiin uudelleen, valitsisin aluksi helppokäyttöiset laiteet prototyypin luomiseen, minkä jälkeen voisi tarvittaessa korvata osia vaikeammin lähestyttävillä ammattimaisemmilla laiteilla.

Lähteet

1 England, Gordon. Thermal Spray Coatings. Verkkoaineisto. <https://www.gordonengland.co.uk/ps.htm>. Luettu 21.3.2018

2 Ghelichi, Ramin. 2011. Cold spary coating aimed at nanocrystallization. Verkkoaineisto. Politesi. <https://www.politesi.polimi.it/bitstream/10589/65961/1/PhDThesis-Ghelichi-Polimi.pdf>. Luettu 21.3.2018.

3 Supersonic Laser Deposition. Verkkoaineisto. Wordpress. <https://testlft.wordpress.com/sld/>. Luettu 26.6.2018.

4 PID Theory Explained. 2011. Verkkoaineisto. National Instruments. <http://www.ni.com/white-paper/3782/en/>. Luettu 20.3.2018.

5 Tuning Of PID Controllers. Verkkoaineisto. <http://home.hit.no/~hansha/training/labview/controlandsimulation/documents/tuning_pid_controller.pdf>. Luettu 12.4.2018

6 Basics Of UART Communication. Verkkoaineisto. Circuit Basics. <http://www.circuitbasics.com/basics-uart-communication/>. Luettu 19.3.2018.

7 LPC15xx Product Data Sheet. 2015. Verkkoaineisto. NXP. <https://www.nxp.com/docs/en/data-sheet/LPC15XX.pdf>. Luettu 10.3.2018.

8 LPC1549. Verkkoaineisto. NXP. <https://www.nxp.com/assets/images/en/photography/OM13056_MAIN.jpg>. Luettu 13.4.2018.

9 Raspberry Pi 3. Verkkoaineisto. RaspberryPI. <https://www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-1-1619x1080.jpg>. Luettu 13.4.2018

10 Love, Robert. 2003. Linux Process Scheduler. Verkkoaineisto. InformIT. <http://www.informit.com/articles/article.aspx?p=101760&seqNum=4>. Luettu 23.3.2018.