SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Mikael Nykänen

3D-tulostimen ohjaaminen Beckhoffilla

Opinnäytetyö
Kevät 2023
Insinööri (AMK), Automaatiotekniikka

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Tutkinto-ohjelma: Insinööri (AMK), Automaatiotekniikka

Suuntautumisvaihtoehto: Sähköautomaatio

Tekijä: Mikael Nykänen

Työn nimi: 3D-tulostimen ohjaaminen Beckhoffilla

Ohjaaja: Jyri Lehto

Vuosi: 2023 Sivumäärä: 35 Liitteiden lukumäärä: 5

Opinnäytetyön aiheena oli suunnitella ja ohjelmoida yksinkertainen ohjelmisto, jolla voitaisiin ohjata yksinkertaista 3D-tulostinta. Tämä tehtiin paikalliselle teollisuusluokan 3D-tulostimia valmistavalle yritykselle nimeltä miniFactory Oy Ltd. Työn tavoitteena oli selvittää, voisiko Beckhoffin valmistamia ohjelmoitavia logiikoita mahdollisesti käyttää yrityksen omissa 3D-tulostimissa tai 3D-tulostustarpeissa tulevaisuudessa. Yritys käyttää tällä hetkellä usean eri valmistajan logiikkaohjaimia, ja tämä auttaisi heitä siirtymään käyttämään vain yhden valmistajan rautaa.

Työn pääohjelma tehtiin hyödyntäen Beckhoff TwinCAT 3 -ohjelmointiympäristöstä löytyviä MC2- ja NCI-ohjelmakirjastoja. Ohjelmointikielenä työssä käytettiin Structured Text -ohjelmointikieltä. Myös yksinkertainen HMI-käyttöliittymä luotiin, jotta ohjelmaa olisi helpompi ohjata. Muutama 3D-tulostuksessa käytetty M-funktio lisättiin ohjelmaan, ja lähdekoodista selviää, miten ne toimivat, että näitä funktioita olisi helpompi lisätä ja laajentaa tulevaisuudessa.

Työssä saatiin aikaiseksi yksinkertainen ohjelmisto, jota yritys voisi mahdollisesti käyttää omissa 3D-tulostustarpeissaan tai muokata ja laajentaa sitä tulevaisuudessa. Pääohjelma ja sen komponentit kommentoitiin, jotta koodia olisi helpompi seurata ja ymmärtää. Työn aikana selvisi, että NCI-kirjasto ei ole aivan ajan tasalla ja tulkki ei suoraan ymmärrä nykyisten viipalointiohjelmistojen generoimia G-koodeja. Nämä generoidut G-kooditiedostot vaativat tällä hetkellä melko paljon muokkaamista, että NCI-tulkki pystyy ajamaan niitä.

Beckhoffilta on tulossa parannuksia varsinkin 3D-tulostamiseen tulevaisuudessa, mutta harmillisesti niitä ei ollut vielä julkaistu tämän opinnäytetyön tekemisen yhteydessä.

[1] Asiasanat: Beckhoff, 3D-tulostus, PLC, ohjelmoitavat logiikat, G-koodi, viipalointiohjelmat, TwinCAT, NCI, MC2

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Degree programme: Automation Engineering

Specialisation: Electrical Automation

Author: Mikael Nykänen

Title of thesis: 3D-printing using Beckhoff PLC

Supervisor: Jyri Lehto

Year: 2023 Number of pages: 35 Number of appendices: 5

The subject of the thesis was to design and program a software for a Beckhoff programmable logic controller so that it could be used to control a simple 3D-printer. This was done to help a local manufacturer of industrial 3D-printers (miniFactory Oy Ltd) to consider perhaps switching over to Beckhoff PLCs as their main controller hardware in the future as today they use controllers from multiple manufacturers.

The main program was made using Beckhoff's MC2 and NCI programming libraries available at Beckhoff's TwinCAT 3 programming software. Structured Text was used as the programming language. A user interface was also designed and created using Beckhoff's HMI (Human Machine Interface) addon and libraries. Some 3D-printing specific G-code M-functions were also added to the program and they can be easily expanded in the future.

As the result of the thesis a simple software was created which the company can expand and/or modify for their own 3D-printing needs. The main program was commented so that it would be easier to follow and understand. Unfortunately, the currently available version of the TwinCAT NCI interpreter turned out to be insufficient when interpreting modern 3D-printing slicer software generated G-code. The generated G-code files had to be modified quite extensively so that the interpreter could read the code correctly.

Beckhoff will make updates and improvements to the NCI library that should help when creating programs for 3D-printing in the future. Unfortunately, these updates were not available at the time of this thesis project.

[1] Keywords: Beckhoff, 3D-printing, PLC, programmable logic controllers, G-code, slicer software, TwinCAT, NCI, MC2

SISÄLTÖ

Kuva-, kuvio- ja taulukkoluettelo

Käytetyt termit ja lyhenteet

CNC	Tietokoneistettu numeerinen ohjaus.
G-koodi	Numeerisesti ohjatuissa koneissa käytetty yleinen ohjelmointikieli.
G-funktio	G-koodissa käytetty komento liikkeille (niin kutsuttu ”Geometric”- tai ”General”-komento).
HMI	Ihminen-kone-rajapinta (Human Machine Interface) koneiden ohjaamiseksi.
M-funktio	Käytetään G-koodissa koneen muiden ominaisuuksien kuten tuulettimien, lämmityselementtien ym. ohjaamiseen (niin kutsuttu ”Miscellaneous”- tai ”Machine”-komento).
MC2	TwinCAT-ohjelmointiympäristön liikkeenohjauksen ohjelmointikirjasto.
NCI	TwinCAT-ohjelmointiympäristön numeerisen ohjauksen interpoloinnin ohjelmointikirjasto.
Pursotin	3D-tulostuksessa käytetty yleinen termi materiaalia pursottavasta tai lisäävästä laitteistosta.
Regex	Säännöllinen lauseke, jota voidaan käyttää tekstin vertaamiseen ja sen korvaamiseen tai muokkaamiseen.
Slicer	3D-tulostamisessa käytettävä ohjelmisto 3D kappaleiden viipalointiin kerroksittain. Näistä kerroksista generoidaan ohjelmakoodi (G-koodi) tulostimen akseleille.

1 JOHDANTO

1.1 Työn tausta

Työssä oli tarkoitus tutustua Beckhoffin TwinCAT 3 -ohjelmointiympäristöön ja ohjelmoida yksinkertainen ohjelma, jolla olisi mahdollista ohjata 3D-tulostinta. Tämän työn toimeksiantajana toimii miniFactory Oy Ltd. Tarkoituksena oli selvittää yritykselle, miten tämä ohjelma olisi järkevä alustaa niin, että yritys voisi mahdollisesti tulevaisuudessa vaihtaa nykyiset usean valmistajan logiikkaohjaimensa yhden valmistajan logiikkaohjaimen alle.

1.2 Työn tavoite

Työn tavoitteena on saada aikaiseksi yksinkertainen ohjelma, jolla voidaan teoreettisesti ohjata 3D-tulostinta. TwinCAT 3 -ohjelmistossa ohjelman luomiseen käytetään valmiita NC-ohjauksen ohjelmakirjastoja. Näihin kirjastoihin kuuluu Tc_MC2-liikkeenohjauskirjasto ja NCI-kirjasto. MC2-kirjastosta löytyy tarvittavat ohjaimet ja toiminnot NC-akselien liikuttamiseen ja NCI-kirjastosta löytyy ominaisuudet G-koodin ajamiseen ja tulkitsemiseen TwinCAT 3 -ympäristössä. Ohjelmistoon tulisi lisätä myös 3D-tulostamisessa yleisimmin käytetyt M-funktiot.

Tavoitteena on saada aikaiseksi yksinkertainen ohjelmisto, jota yritys voisi mahdollisesti myöhemmin hyödyntää omissa 3D-tulostimissaan, jos yritys päättää vaihtaa ohjaimensa Beckhoffin valmistamiin logiikkaohjaimiin. Ohjelmakoodi kommentoitiin, että tulevaisuudessa ohjelmaa olisi helpompi laajentaa ja muokata yrityksen omiin 3D-tulostustarpeisiin ja mahdollisesti jatkokehittää sitä toimimaan yrityksen omien 3D-tulostimien kanssa.

1.3 Työn rakenne

Työ alkaa johdannolla, jossa kerrotaan, mitä työssä tavoitellaan, miksi tämä työ tehtiin ja mille yritykselle työ tehtiin. Työn teoriaosassa on kerrottu 3D-tulostamisesta ja erilaisista 3D-tulostusmenetelmistä. Tässä osassa kerrotaan lukijalle myös 3D-tulostamisessa käytetyistä viipalointiohjelmista ja niillä generoiduista G-kooditiedostoista yleisesti. G-koodista kerrotaan myös hieman tarkemmin, jotta lukija ymmärtää, mitä se sisältää.

Teoriaosuuden jälkeen kerrotaan hieman Beckhoffista yrityksenä ja heidän kehittämistään automaatioratkaisuista. Tämän jälkeen syvennytään TwinCAT-ohjelmointiympäristöön ja työssä käytettyihin ohjelmointikirjastoihin tarkemmin. Ohjelmointikirjastoista käytettyjä toimintolohkoja käydään läpi tarkemmin, jotta liitteenä sisällytetty lähdekoodi olisi lukijalle helpommin ymmärrettävissä. Työosuuden lopuksi käydään vielä läpi työhön luotu yksinkertainen käyttöliittymä. Työosuuden jälkeen käydään vielä läpi työn yhteenveto ja pohdinta. Liitteenä on työhön luotu ohjelmiston lähdekoodi.

1.4 Yritysesittely

MiniFactory Oy Ltd on yritys, jota ohjaa halu valmistaa korkean suorituskyvyn ja parhaan tulostuslaadun omaavia teollisuusluokan 3D-tulostimia (Minifactory, i.a.-a). Yritys perustettiin vuonna 2012. Yrityksen pääkonttori sijaitsee Seinäjoella.

Yrityksen lippulaivatulostin on Ultra 2, jonka avainominaisuuksiin kuuluu mm. lämmitetty tulostus- ja materiaalikammio, lisenssivapaa uusien materiaalien validointi, nopea aloitusaika, automatisoitu huollon ajoitus ja korkealaatuiset servomoottorit (Minifactory, i.a.-b).

2 3D-TULOSTAMINEN YLEISESTI

3D-tulostus tai materiaalia lisäävä valmistus on esineiden tai kappaleiden valmistusta digitaalisista mallinnuksista (3DPrinting, i.a.). Materiaalia lisäävällä valmistustekniikalla tarkoitetaan jatkuvaa erillisten kerrosten lisäämistä päällekkäin, kunnes saadaan valmis kokonainen kappale, joka vastaa digitaalista mallinnusta. Jokainen näistä kerroksista on nähtävissä leikkauksina valmiin kappaleen pinnalla. 3D-tulostaminen voidaan ajatella vastakohtana materiaalia poistaville tekniikoille, kuten CNC-jyrsimiselle, joissa materiaalia leikataan pois kappaleista. 3D-tulostamalla on mahdollista valmistaa vaikeampia muotoja kuin normaalein valmistusmenetelmin samalla materiaalia säästäen.

2.1 Erilaisia 3D-tulostusmenetelmiä

International Standards Organization (ISO) -järjestö on jakanut 3D-tulostusmenetelmät seitsemään eri alueeseen riippuen, mitä tulostetaan ja mitä materiaalia menetelmät käyttävät tulostuksessa (All3DP, 2023).

2.1.1 Material Extrusion

Material Extrusion -menetelmässä materiaali pursotetaan suuttimen läpi (All3DP, 2023). Yleisesti materiaali on muovia, joka kuumennetaan lähelle materiaalin sulamispistettä samalla kun se pursotetaan suuttimen läpi. Tulostin pursottaa materiaalin tulostusalustalle seuraten tulostusohjelmistossa ennalta määrättyä rataa. Materiaali jäähtyy erittäin nopeasti ja jähmettyy muotoonsa poistuttuaan pursottimesta. Tämä menetelmä on yleisin 3D-tulostuksessa käytetty menetelmä.

2.1.2 Vat Polymerization

Vat Polymerization -menetelmässä käytetään hyödyksi ultraviolettivaloa, jolla kovetetaan nestemäistä fotopolymeerihartsia astiassa (All3DP, 2023). Valoa ohjataan tiettyihin pisteisiin astiassa olevaan telineeseen, jolloin nestemäinen hartsi kovettuu. Kun yksi kerros on saatu kovettuneeksi, voidaan esinettä nostaa vadista telineellä ja aloittaa seuraavan kerroksen tulostaminen. Tämä prosessi toistetaan niin monta kertaa, kunnes haluttu kappale on kokonaisuudessaan tulostettu.

Tulostuksen jälkeen valmis kappale vielä puhdistetaan ja jälki käsitellään joko ultraviolettikammiossa tai auringonvalossa, jotta tuloste vahvistuu ja kuivuu lopullisesti (All3DP, 2023).

2.1.3 Powder Bed Fusion

Powder Bed Fusion -menetelmässä sulatetaan haluttu muoto jauhemuodossa olevan materiaalin pintaan lämpölähdettä hyväksikäyttäen (All3DP, 2023). Jauhemuodossa oleva materiaali levitetään ohuesti koneessa tulostuskammioon yleensä hyödyntäen jonkinlaista pyyhintä, terää tai rumpua. Tämän jälkeen jauhe kuumennetaan esimerkiksi laserilla niin, että se sulaa haluttuun muotoon, ja prosessi toistetaan niin monta kertaa, kunnes haluttu fyysinen kappale on tulostettu.

2.1.4 Material Jetting

Material Jetting -menetelmässä materiaalia lisätään pienillä tipoilla, minkä jälkeen se kovetetaan (All3DP, 2023). Menetelmässä hyödynnetään fotopolymeerisiä nesteitä tai vahaa, joka kovetetaan ultraviolettivalolla. Menetelmä mahdollistaa erilaisten materiaalien, värien ja tekstuurien käytön samassa tulosteessa.

2.1.5 Binder Jetting

Binder Jetting -menetelmä yhdistää Powder Bed Fusion- ja Material Jetting -menetelmän keskenään (All3DP, 2023). Menetelmässä käytetään nestemäistä sidosainetta, jolla kovetetaan haluttu muoto jauhemuodossa olevaan materiaaliin. Kuten Powder Bed Fusion-tekniikassa, jauhe levitetään tulostuskammioon kerros kerrokselta.

2.1.6 Directed Energy Deposition

Directed Energy Deposition -menetelmässä syötetään metallimateriaalia, joka samalla sulatetaan erittäin voimakkaalla energialla (All3DP, 2023). Materiaalina voi toimia metallilanka tai metallijauhe. Lämpöenergian lähteenä voi toimia esimerkiksi laser, elektronisäde/-suihku, sähkökaari ym. Menetelmä muistuttaa hyvin paljon hitsaamista. Menetelmä helpottaa yksittäisten kappaleiden korjaamista ja pienten osaerien valmistamista, joka muutoin tulisi kalliiksi esimerkiksi metallivalulla.

2.1.7 Sheet Lamination

Sheet Lamination -menetelmässä pinotaan ja laminoidaan erittäin ohuita levyjä päällekkäin, minkä jälkeen pino leikataan lopulliseen muotoonsa käyttäen joko fyysistä leikkuria tai laseria (All3DP, 2023). Materiaalilevyt yhdistetään toisiinsa hyödyntäen esimerkiksi lämpöä tai ääntä riippuen siitä, mitä materiaalia ollaan tulostamassa. Menetelmä poikkeaa paljon muista 3D-tulostusmenetelmistä mutta on silti osa 3D-tulostamista.

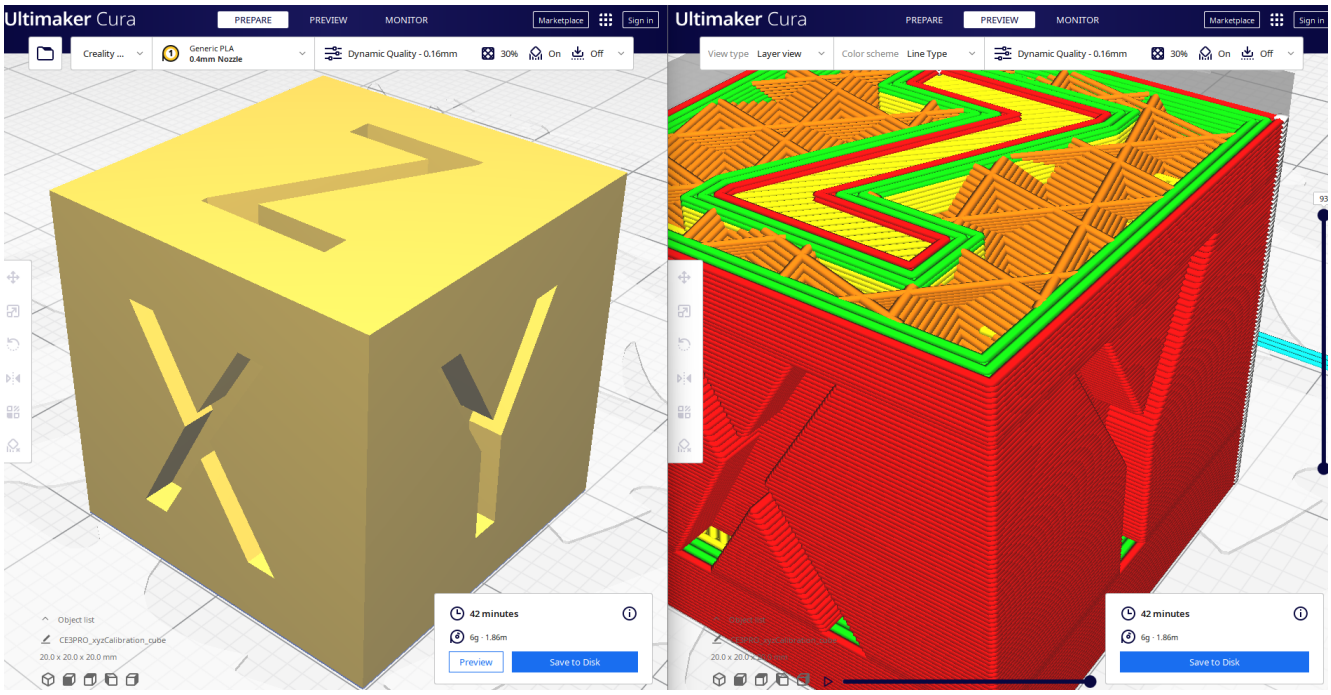
2.2 Viipalointiohjelmat

Viipalointiohjelmat (englanniksi *slicer*) ovat ohjelmistoja, jotka muuttavat digitaalisen 3D-mallin 3D-tulostimille sopivaan käskymuotoon niin, että tulostin pystyy tulostamaan kyseisen mallin fyysisesti (All3DP, 2022b). Käskyt sisältävät myös käyttäjän syöttämiä parametrejä, kuten esimerkiksi tulostuskerrosten paksuuden, tulostusnopeuden ja mallin tukirakennelmien asetukset.

Jokainen 3D-tulostusteknologia perustuu fyysisten tulosteiden luomiseen kerros kerrokselta, ja viipalointiohjelmat nimensä mukaisesti viipaloivat digitaaliset 3D-mallit kaksiulotteisiin kerroksiin, jotka tulostetaan yksi kerros kerrallaan (All3DP, 2022b).

3D-malleja voidaan luoda monen erilaisen 3D-mallinnusohjelman avulla. Ne voivat olla esimerkiksi avoimen lähdekoodin taiteellisia ohjelmistoja kuten Blender tai vaikkapa paljon teknisempiä ja ammattitason ohjelmia kuten SolidWorks (All3DP, 2022b).

Viipalointiohjelmat generoivat 3D-tulostukseen tarvittavat tiedostot, jotka sisältävät erittäin yksityiskohtaiset tulostukseen tarvittavat käskyt (All3DP, 2022b). Yleisimmin nämä käskyt ovat G-koodimuodossa.



Kuva 1. Havainnekuva 3D-mallista Ultimaker Cura -viipalointiohjelmassa. Vasemmalla alkuperäinen malli ja oikealla viipaloitu malli, jossa tulosteen kerrokset (punainen) ja tukirakenteet (oranssi) ovat hyvin näkyvillä.

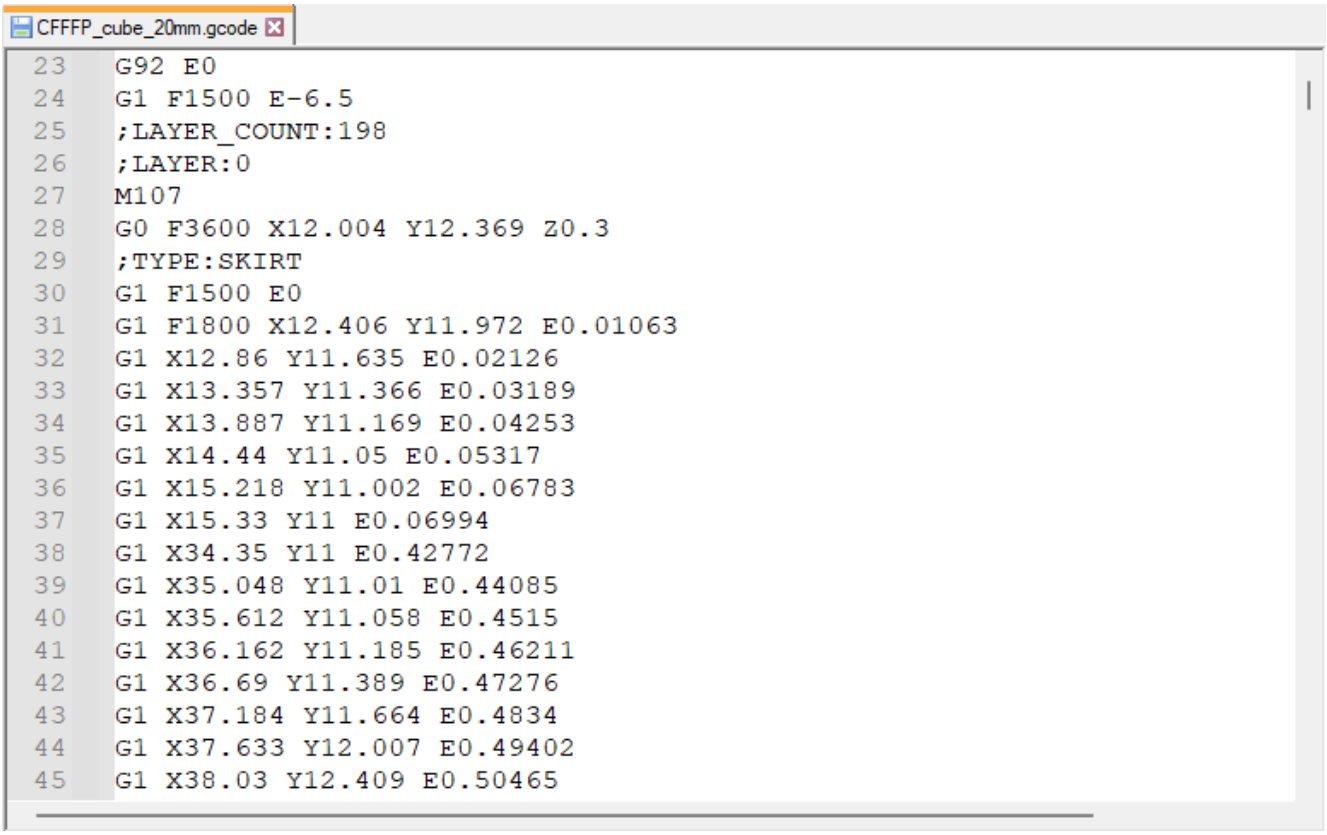
2.3 G-koodi

G-koodi (englanniksi *G-code*) on ohjelmointikieli, jota käytetään numeerisesti ohjattujen (CNC) koneiden ohjaamiseen (All3DP, 2022a). G-koodi sisältää listan rivejä, joista kone pystyy päättelemään seuraavat liikkeet tai mahdolliset muut toiminnot. Kone tekee nämä liikkeet ja toiminnot järjestyksessä, rivi riviltä, kunnes G-kooditiedoston loppu saavutetaan.

G-koodissa jokainen rivin komento aloitetaan kirjaimilla G (*general*) tai M (*miscellaneous*) (All3DP, 2022a). G-komentotyypillä ohjataan yleisesti koneen liikkeitä kuten X-, Y- ja Z-akseleiden liikettä ja myös lisäakseleiden kuten pursottimen tai useamman pursottimen liikettä. M-funktioilla voidaan taas tehdä kaikkia muita liikkeeseen kuulumattomia koneen toimintoja. Näitä toimintoja voisi olla esimerkiksi 3D-tulostimen pursottimen esilämmitys, tulostusalustan esilämmitys tai jäähdytystuulettimien ohjaaminen.

Koska jokainen G-koodin rivi noudattaa tietynlaista syntaksia ja jokainen rivi on koneelle yksi komento, tämä voi johtaa koodissa erittäin pitkiinkin riveihin (All3DP, 2022a). Ensimmäinen argumentti on aina komennon tyyppi eli G tai M, jonka jälkeen tulee numero, joka määrittelee

liikkeen tyypin tai koneen toiminnon. Jos rivi esimerkiksi alkaa G0 komennolla, se tarkoittaa, että koneen tulee tehdä suora, lineaarinen liike.

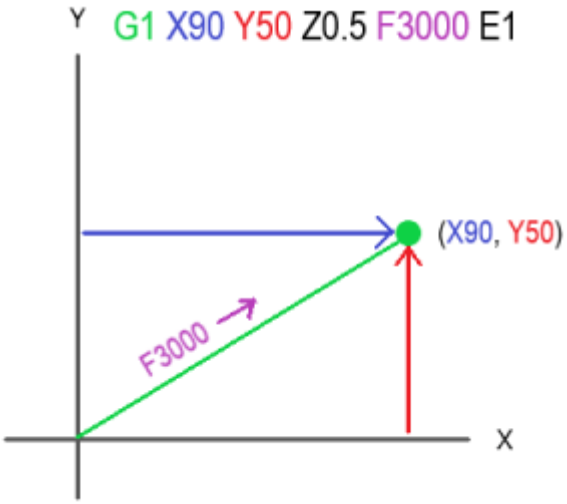


Kuva 2. Esimerkki G-koodista avattuna Notepad++ -ohjelmassa.

2.3.1 Yleisimmät G-funktiot 3D-tulostamisessa

Yleisimmät G-funktiot 3D-tulostamisessa ovat G0- ja G1-komennot, joita kumpaakin käytetään suorissa liikkeissä aloituspisteestä seuraavaan koordinaatistoon (All3DP, 2022a). G0-liikekomentoa käytetään yleensä silloin, kun tulostimen materiaalipursotin (englanniksi *extruder*) ei ole käytössä, kun taas G1-komennolla ohjataan myös lisäakseleita kuten pursotinta samanaikaisesti. Kummatkin komennot toimivat kuitenkin samalla tavalla.

Esimerkiksi ”G1 X90 Y50 Z0.5 F3000 E1” -komennon syntaksi liikuttaa pursottimen koordinaatteihin X90 mm, Y50 mm ja nostaa pursotinta tai laskee tulostusalustaa Z0.5 mm (All3DP, 2022a). F3000-syntaksi määrittelee akseleiden nopeuden, joka on yksiköissä mm/min. Samalla kun liike tehdään, pursotin lisää materiaalia matkan aikana yhden millimetrin E1-syntaksilla.



Kuva 3. Havainnekuva aikaisemmasta G1-liikekomennosta kaksiulotteisella tasolla.

G90- ja G91-komennot kertovat tulostimelle, miten koordinaatistoa tulisi tulkita (All3DP, 2022a). G90 määrittelee absoluuttisen liikkeen, kun taas G91 määrittelee relatiivisen liikkeen. Kumpikaan näistä komennoista ei tarvitse syntaksia, ja käyttämällä toista poistetaan siksi aikaa toinen käytöstä.

Jos esimerkiksi halutaan liikuttaa X-akselin asemaa 30 mm absoluuttisella liikkeellä, voitaisiin ensin valita G90-komento ja sitten antaa suora liikekomento G1 X30. Tämä komentaisi koneen liikuttamaan X-akselin koordinaatistossa suoraa kohtaan 30 mm positiiviseen suuntaan (All3DP, 2022a). Jos käytetään relatiivista liikettä G91 ja ajetaan perään sama liikekomento G1 X30, tarkoittaa se, että kone liikkuu nykyisestä koordinaatistosta +30 mm X-askelin suuntaisesti. Tällöin näiden liikekomentojen jälkeen olisi X-akseli positiossa X60 mm.

G28-komennolla voidaan ajaa tulostimen etsintäkomento (englanniksi *homing*), jolla etsitään tulostimen akselien rajamitat (All3DP, 2022a). Tällä komennolla liikutetaan akselit päästä päähän, kunnes koneen fyysiset lähestymiskytkimet antavat ohjelmistolle tiedon akseleiden minimi- ja/tai maksimiasennoista. Tämä komento tehdään yleisesti aina ennen tulostusta.

2.3.2 Yleisimmät M-funktiot 3D-tulostamisessa

M104- ja M109-komennoilla ohjataan 3D-tulostimen pursottimen lämpötiloja (Simplify3D, i.a.). M104 aloittaa pursottimen lämmittämisen haluttuun arvoon, mutta antaa tulostimen ajaa muita komentoja samanaikaisesti. Jos taas käytetään M109-komentoa, tulostin odottaa

tekemättä muuta niin kauan, kunnes pursottimelle annettu lämpötila saavutetaan. Esimerkiksi komento M109 S210 asettaa pursottimen kohdelämpötilan 210 °C asteeseen ja tulostin odottaa niin kauan, kunnes saa tiedon, että tähän lämpötilaan on päästy.

Samalla tavalla tulostusalustan lämmittämiseen käytetään komentoja M140 ja M190 (Simplify3D, i.a.). Nämä toimivat samalla lailla eli M140-komennolla voidaan lämmittää tulostusalustaa ja ajaa samalla muita komentoja, kun taas M190 odottaa, että tulostusalusta pääsee annettuun lämpötilaan ennen kuin tulostin voi jatkaa seuraavaan komentoon G-koodissa.

Ohjausta vaativien jäähdytystuulettimien ohjaaminen tapahtuu M106- ja M107-komennoilla (All3DP, 2022a). M106 käynnistää tuulettimen ja määrittelee sen pyörimisnopeuden. Pyörimisnopeus määritellään tässä komennossa arvon 0–255 välillä. Esimerkiksi M106 S128 P1 -syntaksi käynnistäisi tuulettimen numero yksi ja S128-syntaksi määrittelisi kyseisen tuulettimen pyörimisnopeudeksi 50 %. M107-komennolla voidaan sammuttaa haluttu tuuletin.

3 BECKHOFF JA TWINCAT 3

Yrityksen perustamisesta lähtien, vuodesta 1980 on Beckhoffin lähtökohta ollut kehittää innovoivia ratkaisuja PC-pohjaisiin ohjausjärjestelmiin, ja se on ollut yrityksen jatkuvan menestyksen mahdollistaja (Beckhoff, i.a.).

Beckhoff kehitti ensimmäisen PC-pohjaisen logiikkaohjaimensa (PLC) vuonna 1986 (Beckhoff, 2021, s. 2). Kymmenen vuotta myöhemmin kehitettiin TwinCAT-ohjelmiston ensimmäinen versio, joka yhdisti kaikki automaatiotoiminnat keskeiseen pakettiin.

Beckhoffin tuotevalikoima kattaa pääosin teollisuustietokoneet, I/O- ja kenttäväyläkomponentit, liikkeenohjaimet, automaatio-ohjelmistot, ohjauskaapittomat automaatiojärjestelmät ja konenäkölaitteistot (Beckhoff, i.a.). Kaikilta tuotevalikoimasektoreilta löytyy mahdollisuudet käyttää komponentteja erikseen tai integroida ja rakentaa komponenteista isompia ja toisiinsa yhteensopivia kattavia kokonaisuuksia.

Yrityksen pääkonttori sijaitsee Verlissä, Saksassa, jossa sijaitsevat yrityksen keskeiset yksiköt tuotekehitykselle, valmistukselle, johdolle, markkinoinnille, tuelle ja huollolle (Beckhoff, i.a.). Beckhoff toimii tai on edustettuna yhteistyökumppaneiden kautta 75 eri maassa maailmanlaajuisesti.

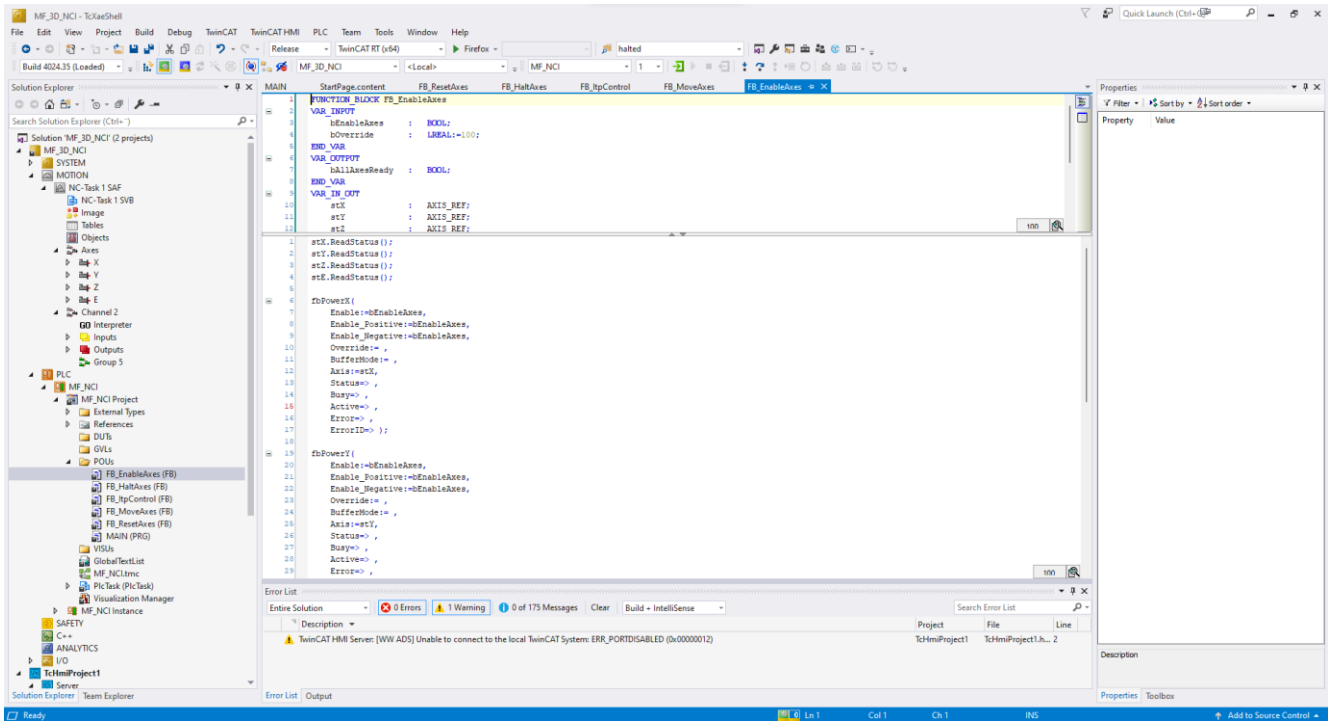
3.1 TwinCAT 3 -ohjelmisto

Nykyaikaisten koneiden monimutkaisuuden hallitsemiseksi ja samalla turhan insinöörityön vähentämiseksi on nykyinen trendi modulaariset ohjausohjelmistot (Beckhoff, 2022, s. 8). Yksittäisiä funktioita, kokoonpanoja tai koneyksiköitä pidetään moduuleina. Moduulien tulisi olla mahdollisimman itsenäisiä ja rakenteeltaan hierarkkisia. Rakenteellisen formaatin tulisi olla sellainen, että alimmat moduulit ovat helpoimpia ja uudelleenkäytettäviä peruselementtejä. Standardisoiduilla liitännöillä ohjelmistomoduulit korkeammilta moduuleilta voidaan yhdistää monimutkaisempiin koneyksikköihin, jopa kokonaiseksi koneeksi. Ihanteellisesti näitä yksittäisiä moduuleita voidaan ottaa käyttöön, laajentaa, skaalata ja käyttää uudelleen toisista riippumatta.

TwinCAT 3 -ohjelmiston arkkitehtuuri mahdollistaa juuri tämänkaltaisen ohjelmoinnin (Beckhoff, 2022, s. 8). Näiden ominaisuuksien vuoksi TwinCAT 3 -ohjelmistoa kutsutaan myös

nimellä eXtended Automation (XA). eXtended Automation on yhdistelmä uusimpia tietotekniikan teknologioita ja tiedemaailman ohjelmistotyökaluja automaatioteknologian kanssa.

Yksi TwinCAT 3 -ohjelmiston tärkeimmistä lähestymistavoista on yksinkertaistaa ohjelmistokehittämistä (Beckhoff, 2022, s. 8). Sen sijaan että kehitettäisiin omia yksittäisiä työkaluja, on järkevämpää integroitua valmiisiin ja yleisesti käytettyihin ohjelmistokehitysympäristöihin. TwinCAT 3:ssa tänä alustana toimii Microsoft Visual Studio -ohjelmointiympäristö. Integroimalla TwinCAT 3 -ohjelmisto osaksi Microsoft Visual Studiota tarjotaan käyttäjälle hyvin laajennettava ja käyttövarma kokonaisuus.



Kuva 4. Kuvankaappaus TwinCAT 3 -ohjelmointiympäristöstä, joka muistuttaa paljon Microsoft Visual Studio -kehitysympäristöä.

3.2 TwinCAT MC2 -liikkeenohjauskirjasto ja työssä käytetyt toimintolohkot

TwinCAT Motion Control PLC -kirjasto (MC2) sisältää toimintolohkoja (englanniksi *function block*), joita voidaan käyttää konesovellutusten kehittämiseen (Beckhoff, 2023b, s. 10). Kirjasto perustuu PLCopen yhdistyksen 2.0-version liikkeenohjaustoimintolohkoihin.

Riippuen ohjelmistovaatimuksista ja projektien määrityksistä joutuvat insinöörit käyttämään laajoja valinnaisia liikkeenohjauskokonaisuuksia (PLCopen, i.a.). Menneisyydessä tämä vaati

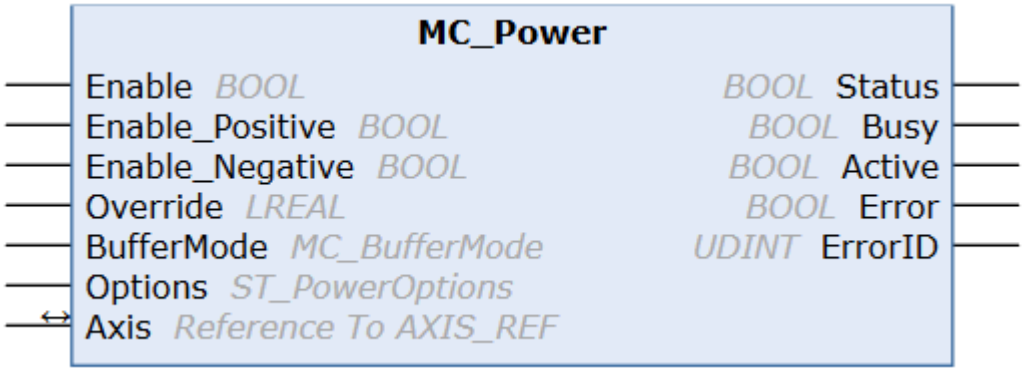
harvinaislaatuisien ohjelmistojen luomista jokaiseen käyttötarkoitukseen, vaikka toiminnot olivatkin samanlaisia. PLCopen liikkeenohjausstandardi tarjoaa tavan käyttää standardisoituja ohjelmistokirjastoja, joita voidaan käyttää uudelleen useissa erilaisissa laitteistoalustoissa.

Tämä laskee kehitys-, huolto- ja ylläpitokustannuksia samalla poistamalla kehittämisen epäselvyyksiä (PLCopen, i.a.). Lisäksi insinöörityöstä tulee helpompaa, koulutusmenot pienentyvät ja ohjelmistot ovat uudelleenkäytettävissä eri alustoilla. Käytännössä tämä tehdään määrittelemällä uudelleenkäytettävistä komponenteista kirjastot. Tällä tavalla ohjelmistot ovat vähemmän laitteistoriippuvaisia, ohjelmistojen uudelleenkäyttöaste kasvaa, koulutus ja kunnossapitokustannukset laskevat ja ohjelmistosta tulee laajennettava toimimaan monien eri ohjausjärjestelmien kanssa.

Liikkeenohjausmarkkinoilla on laaja valikoima yhteensopimattomia järjestelmiä ja ratkaisuja (PLCopen, 2011, s. 8). Yrityksissä, joissa käytetään paljon erilaisia järjestelmiä, nämä yhteensopimattomuudet aiheuttavat huomattavia lisäkustannuksia loppukäyttäjälle, oppiminen on vaikeaa, suunnittelu muuttuu vaikeaksi ja markkinoiden kasvu hidastuu. Standardisointi vähentää näitä negatiivisia tekijöitä. Standardisointi ei tarkoita vain ohjelmointikieliä vaan myös käyttöliittymien standardisointia kohti erilaisia liikkeenohjausratkaisuja. Tällä tavalla liikkeenohjausratkaisujen ohjelmointi on vähemmän laitteistoriippuvaista.

MC2-kirjasto sisältää kaikki tarvittavat toimintolohkon numeerisen ohjauksen järjestelmiin. Näillä toimintolohkoilla voidaan ohjata numeerisesti ohjatun koneen eri toimintoja. Kirjasto sisältää esimerkiksi valmiit ohjauslohkot absoluuttisille ja relatiivisille liikkeenohjauksille. Toimintolohkot on jaettu kahteen eri kategoriaan hallinnollisiin ja liikkeenohjauslohkoihin. Hallinnollisilla toimintolohkoilla voidaan hallita liikkeenohjauslohkoja ja muita mahdollisia laitteiston toimintoja.

3.2.1 MC_Power

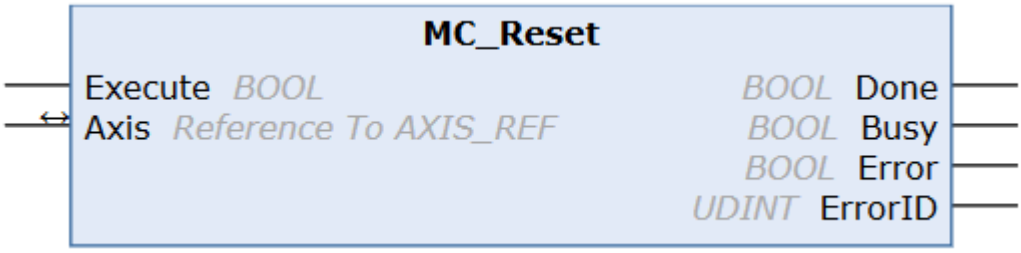


Kuva 5. MC_Power-toimintolohko havainnollistettuna. Vasemmalla puolella ovat lohkon tulot ja oikealla lähdöt (Beckhoff, 2023b, s. 17).

MC_Power-toimintolohkolla ohjataan eri liikeakselit päälle ohjelmistossa tulolla ”Enable” ja määritellään, onko kyseisellä akselilla lupa liikkua kumpaankin vai vain tiettyyn suuntaan (Enable_Positive- ja Enable_Negative-tulot). Näiden tulojen tulee pysyä totena, jotta akselia voidaan liikuttaa. Tämä toimintolohko kuuluu hallinnollisten lohkojen kategoriaan. Tuloihin kuuluu myös mahdollisuus määrittää akselin maksiminopeus (Override), jonka arvo syötetään 0–100 %:n välillä. Jokainen akseli tarvitsee MC_Power-toimintolohkon saadakseen luvan liikkumiselle ohjelmistopuolelta. Tämän vuoksi lohkosta löytyy tulo, jolla määritellään, mitä akselia lohko ohjaa (Axis). Tulolla ”BufferMode” voidaan määritellä, pysähtyykö akseli heti, jos ”Enable”-tulo nollataan, vai käytetäänkö puskuroitua moodia, jolloin suoritetaan nykyinen akselin käsky loppuun ennen kuin se pysäytetään (MC_Buffered-parametri). ”Options”-tulo ei ole käytössä tässä toimintolohkossa.

Lohkon lähdöistä voidaan lukea akselin nykyinen tila (Status) sekä mahdollinen virhetila (Error) ja virhetieto (ErrorID). Lähtö ”Busy” pysyy totena niin kauan, kun lohkon ”Enable” tulo pysyy totena. Lähtö ”Active” taas on tosi, jos akselissa suoritetaan liikekomentoja.

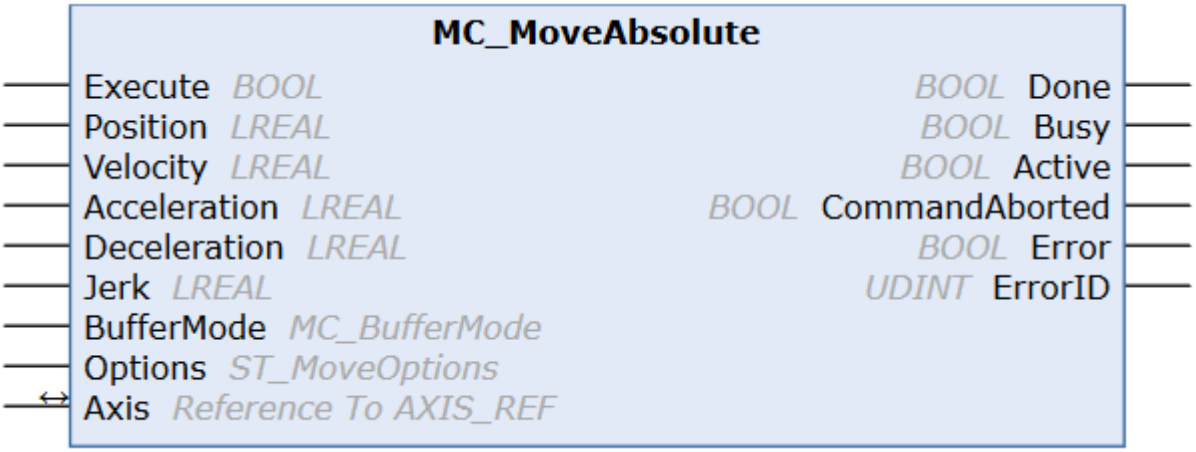
3.2.2 MC_Reset



Kuva 6. MC_Reset-toimintolohko havainnollistettuna (Beckhoff, 2023b, s. 18).

MC_Reset-toimintolohkolla voidaan nollata haluttu akseli käyttäen ”Execute”-tuloa. Tämä toimintolohko lukeutuu hallinnollisiin lohkoihin. Lohko vaaditaan erilaisten vikatilanteiden nollaamiseksi. Jokainen käytettävä akseli vaatii oman MC_Reset-toimintolohkonsa ja ohjattava akseli voidaan määritellä ”Axis”-tuloon. Lähdöistä voidaan lukea, onko nollaus tapahtunut onnistuneesti (Done), ja lisäksi lähdöistä voidaan lukea mahdollinen virhetila (Error) ja virhetieto (ErrorID).

3.2.3 MC_MoveAbsolute



Kuva 7. MC_MoveAbsolute-toimintolohko havainnollistettuna (Beckhoff, 2023b, s. 59).

MC_MoveAbsolute-toimintolohko aloittaa akselin ajamisen määriteltyyn absoluuttiseen pisteeseen ja tarkkailee akselin liikettä koko tämän ajon aikana (Beckhoff, 2023b, s. 59).

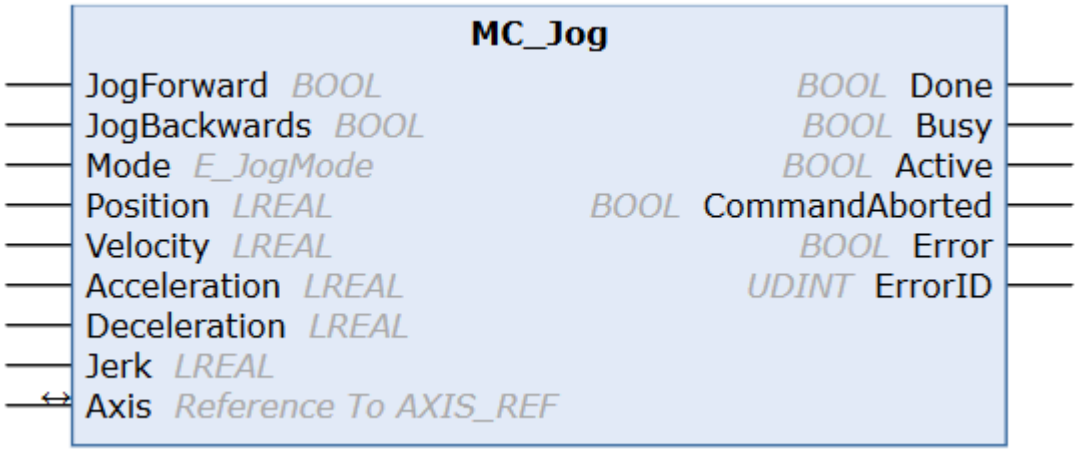
MC_MoveAbsolute kuuluu liikkeenohjauslohkojen kategoriaan. MC_MoveAbsolute-lohkolla voidaan ajaa akseleita absoluuttisesti pisteestä pisteeseen. Akselin ajaminen voidaan aloittaa

”Execute”-tulolla ja määritellä absoluuttinen piste, johon akseli ajetaan, antamalla joko positiviinen tai negatiivinen arvo tulolle ”Position”. Toimintolohkon tuloissa on mahdollista myös määritellä akselin nopeus (Velocity) millimetreissä, kiihtyvyys (Acceleration) ja hidastuvuus (Deceleration) yksiköissä mm/s^2 ja näiden nykäisyvoima (Jerk) yksiköissä mm/s^3. Tuloissa määritellään myös, mitä akselia toimintolohkolla ohjataan (Axis). Tulolla ”Buffermode” voidaan määritellä, miten peräkkäiset liikekomennot yhdistetään. Tulolla ”Options” voidaan määritellä harvemmin käytettyjä erikoisparametrejä lohkolle, mutta tämä voidaan yleensä jättää tyhjäksi.

Toimintolohkon lähtöihin lukeutuu mahdollisuus lukea tilatieto, joka ilmaisee, onko haluttuun pisteeseen määrätty ajo toteutunut (Done). Lähdöistä voidaan myös lukea, onko nykyinen ajo vielä tapahtumassa ja onko lohko varattuna lähdöllä ”Busy”, ja jos tämä lähtö on epätosi, voidaan aloittaa ajaminen seuraavaan määriteltyyn pisteeseen. Jos pisteeseen ajaminen ei jostain syystä onnistu, liikkuminen on pysäytetty, tai jos nykyinen liikekomento on yliajettu ennen sen loppumista, nousee ”CommandAborted”-lähtö todeksi. Tämänkin toimintolohkon lähtöihin kuuluu myös mahdollisuus lukea virhetila (Error) ja virhetiedot (ErrorID).

Tätä toimintolohkoa käytettiin työssä pääsääntöisesti vain X-, Y- ja Z-akseleiden kotipisteeseen ajossa, kun käyttäjä painaa nollauskomennon akseleille.

3.2.4 MC_Jog



Kuva 8. MC_Jog-toimintolohko havainnollistettuna (Beckhoff, 2023b, s. 92).

MC_Jog-toimintolohko kuuluu liikkeenohjauslohkojen kategoriaan, ja sitä käytetään akseleiden ajamiseen käsiohjauksella. Lohkon tuloihin kuuluu mahdollisuus ajaa akselia positiiviseen suuntaan (JogForward) ja negatiiviseen suuntaan (JogBackwards).

Tulossa ”Mode” voidaan määritellä, miten MC_Jog lohko käyttäytyy. Yleisesti tässä voidaan valita, käytetäänkö niin sanottua yhden askeleen liiketilaa (MC_JOGMODE_INCHING parametrillä), jolloin akseli liikkuu vain ”Position”, tuloon määritellyn matkan millimetreissä jokaisella käsiohjauksen painalluksella. Tätä ominaisuutta ei hyödynnetty tässä työssä.

Tässäkin liiketoimintolohkossa on mahdollisuus määritellä akselin nopeus (Velocity) millimetreissä, kiihtyvyys (Acceleration) ja hidastuvuus (Deceleration) yksiköissä mm/s^2 ja näiden nykäisyvoima (Jerk) yksiköissä mm/s^3. Tuloissa määritellään myös, mitä akselia toimintolohkolla ohjataan (Axis). Lohkon lähdöt ovat samat kuin MC_MoveAbsolute-toimintolohkossa.

3.3 TwinCAT NCI -kirjasto ja työssä käytetyt toimintolohkot

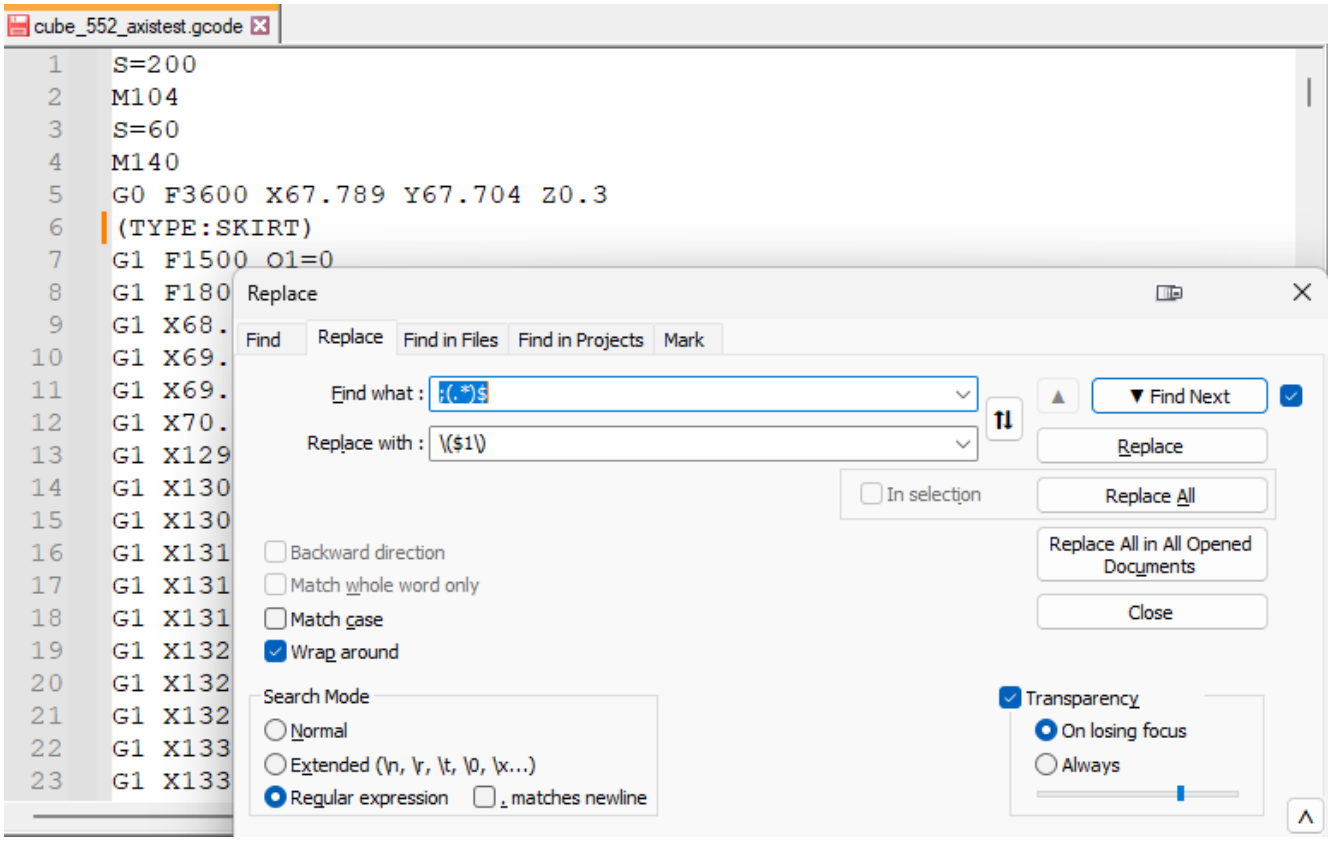
TwinCAT NCI tulee sanoista ’numeerinen kontrollin interpolointi’ (englanniksi *numerical control interpolation*) ja toimii NC-järjestelmänä interpoloitujen polkujen liikuttamiseen (Beckhoff, 2023a, s. 10). TwinCAT NCI tarjoaa 3D interpoloinnin (tulkitsija, asetuspisteen luonti, paikkaohjain), integroidun PLC:n, jossa on NC-liitäntä ja I/O-liitäntä akseleille kenttäväylän kautta. NCI-kirjastoa voidaan käyttää kolmen akselin ohjaamiseen ja maksimissaan viiden lisäakselin ohjaamiseen.

NCI-tulkkia (englanniksi *interpreter)* käytetään G-koodin kääntämiseen TwinCAT-ohjelmistolle. Tässä työssä ilmeni, että nykyinen tulkki ei osaa lukea suoraa viipalointiohjelmistojen generoimia G-kooditiedostoja. Viipalointiohjelmistojen tuottamia G-kooditiedostoja jouduttiin hieman muokkaamaan, että NCI-tulkki saatiin ajamaan tiedostoa.

Viipalointiohjelmistojen generoimissa tiedostoissa G-koodirivien kommentointi aloitetaan puolipisteellä, kun taas NCI-kirjaston tulkki haluaa kommentoinnit sulkujen sisälle. Generoiduissa G-koodeissa myös yleisesti materiaalipursottimen lisäakselin syntaksi merkitään muodossa E# (esim. E100), kun taas NCI-kirjaston tulkki haluaa lisäakselit merkittävän muotoon Q1=# ... Q5=# (esim. Q1=100).

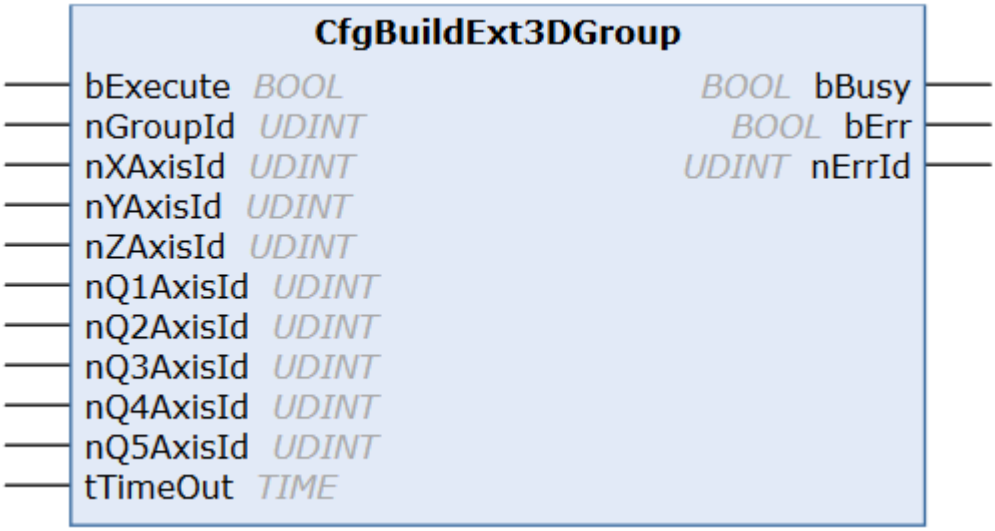
M-funktiot generoidaan viipalointiohjelmistoissa niin, että lisätiedot ovat lueteltuina suoraa samalle riville. Esimerkiksi M104-funktiolla pursotinpään kuumennus 200 ℃ asteeseen generoidaan muotoon M104 S200, jossa S-parametri on lämpötilatieto. NCI-kääntäjä ei osaa lukea tätä riviä suoraa, vaan on käytettävä erillistä S-parametriä erillisellä rivillä, joka tulee G-koodissa ennen M-funktiota, ja tämä tieto voidaan lukea koodissa käyttäen ”ItpGetSParam”-toimintolohkoa.

Työssä käytettiin hyväksi Notepad++ -ohjelmiston tekstin korvaus- ja muokkaamisominaisuutta, joka voidaan avata painamalla ohjelmistossa näppäinyhdistelmää ”Ctrl+H”. Koska viipalointiohjelmistojen generoimat G-kooditiedostot ovat erittäin pitkiä, olisi näiden rivien muokkaus manuaalisesti mahdotonta. Notepad++ -ohjelmistossa voidaan hyödyntää regex- (säännöllinen lauseke) sääntöjä, jotka helpottavat vain tiettyjen haluttujen rivien muokkaamista ja korvaamista.



Kuva 9. Kuvankaappaus Notepad++ -ohjelmistosta, jossa näkyy korvaustyökalu avattuna ja regex-lauseke puolipisteellä alkavien kommenttirivien korvaamiseksi, jotka muutetaan muotoon (rivi).

3.3.1 CfgBuildExt3DGroup



Kuva 10. CfgBuildExt3DGroup-toimintolohko havainnollistettuna. Vasemmalla puolella ovat lohkon tulot ja oikealla lähdöt (Beckhoff, 2023a, s. 198).

Tämä toimintolohko määrittelee 3D-ryhmän kolmesta mahdollisesta liikeakselista (X, Y ja Z) (Beckhoff, 2023a, s. 198). Tämän lisäksi voidaan määritellä viisi lisäakselia (Q1 … Q5).

Tätä toimintolohkoa käytettiin työssä 3D-ryhmän luomiseen liikeakseleille X, Y, Z ja lisäakselille E (Q1).

Toimintolohkosta löytyy tulo ”bExecute”, jolla luodaan haluttu 3D-liikeryhmä. ”nGroupId”-tulolle täytyy hakea tieto kääntäjälle luodusta akseliryhmästä (ItpGetGroupId). Tämä akseliryhmä luodaan automaattisesti, kun NCI kääntäjä luodaan TwinCAT ohjelmiston liikkeenohjaus (MOTION) projektipuun alle. Tuloille ”nAxisId” määritellään 3D-ryhmässä käytetyt akselit, ja käyttämättömät akselit voidaan merkitä tuloihin nollaksi. Tulolle ”tTimeOut” voidaan määritellä ADS-kommunikoinnin aikalisän viivästyminen. Tätä tuloa ei työssä käytetty missään NCI-kirjaston toimintolohkoissa, koska työssä ei käytetty ADS-kommunikointia ulkoisten ohjelmistojen kanssa.

Toimintolohkon lähtöihin kuuluu ”bBusy”, joka pysyy totena niin kauan kunnes lohko on tehnyt komentonsa loppuun tai niin kauan kunnes ”tTimeOut”-sisääntuloon määritelty viivästysaika on ajettu. Niin kauan, kun tämä lähtö on totena, ei kyseinen lohko voi ottaa uusia komentoja vastaan. Lähdöistä löytyy myös virhetilatieto (bErr) ja virhekoodin tieto (nErrId).

3.3.2 CfgReconfigGroup

Kuva 11. CfgReconfigGroup-toimintolohko havainnollistettuna (Beckhoff, 2023a, s. 200).

Tällä toimintolohkolla voidaan vapauttaa CfgBuildExt3DGroup-lohkolla luotu 3D-liikkeenohjausryhmän akselit takaisin normaalin tilaan ja poistaa luotu ryhmä. Jos 3D-akseliryhmä on luotu, aiheuttaa se ongelmia, kun akseleita yritetään ajaa perinteisillä liikkeenohjauskomennoilla.

Toimilohkoa voidaan ohjata tulolla ”bExecute”. Muut tulot ja lähdöt ovat samoja mitä aikaisemmassa CfgBuildExt3DGroup-toimintolohkoa koskevassa alaotsikossa käsiteltiin ja käyttäytyvät samalla tavalla. Nämä tulot ovat ”nGroupId” ja ”tTimeOut”. Lähdöt ovat samat ”bBusy”, ”bErr” ja ”nErrId”.

3.3.3 ItpLoadProgEx

Kuva 12. ItpLoadProgEx-toimintolohko havainnollistettuna (Beckhoff, 2023a, s. 227).

ItpLoadProgEx-toimintolohkoa käytetään G-kooditiedoston lataamiseen NCI-järjestelmälle.

Lohkon tuloihin kuuluu ”bExecute”, joka toteuttaa tiedoston lataamisen ohjelmaan. ”sPrg”-tuloon voidaan määritellä G-kooditiedoston nimi, jolloin TwinCAT hakee tämän tiedoston vakiona hakemistosta ”…TwinCAT\Mc\Nci\”. Tulolle ”nLength” syötetään G-kooditiedoston tiedostonimen pituus, jossa käytetään hyväksi muuttujatyypin muunnosparametriä

”INT_TO_UDINT”. Tuloon ”sNciToPlc” määritellään kommunikointikanava NCI-ohjelman ja PLC:n välille (NCTOPLC_NCICHANNEL_REF parametrillä). Muut tulot ja lähdöt ovat samoja mitä aikaisemmassa ”CfgBuildExt3DGroup”-toimintolohkoa koskevassa alaluvussa käsiteltiin ja käyttäytyvät samalla tavalla. Nämä tulot ovat ”nGroupId” ja ”tTimeOut”. Lähdöt ovat samat ”bBusy”, ”bErr” ja ”nErrId”.

3.3.4 ItpResetEx2

Kuva 13. ItpResetEx2-toimintolohko havainnollistettuna (Beckhoff, 2023a, s. 233).

ItpResetEx2-toimintolohko suorittaa NCI-kanavan nollauksen, ja tämä poistaa kaikki luodut ryhmät ja pysäyttää akseleiden liikkeen (Beckhoff, 2023a, s. 233).

Kuten MC2-kirjaston MC_Reset-toimintolohko, käytetään tätä toimintolohkoa samalla tavalla mahdollisten virhetilojen nollaamiseen NCI-puolella. Myös jos liike pysäytetään manuaalisesti ennen kuin G-koodin loppu on saavutettu, joudutaan NCI-kanava nollaamaan, jotta akseleiden ajaminen voidaan aloittaa uudelleen.

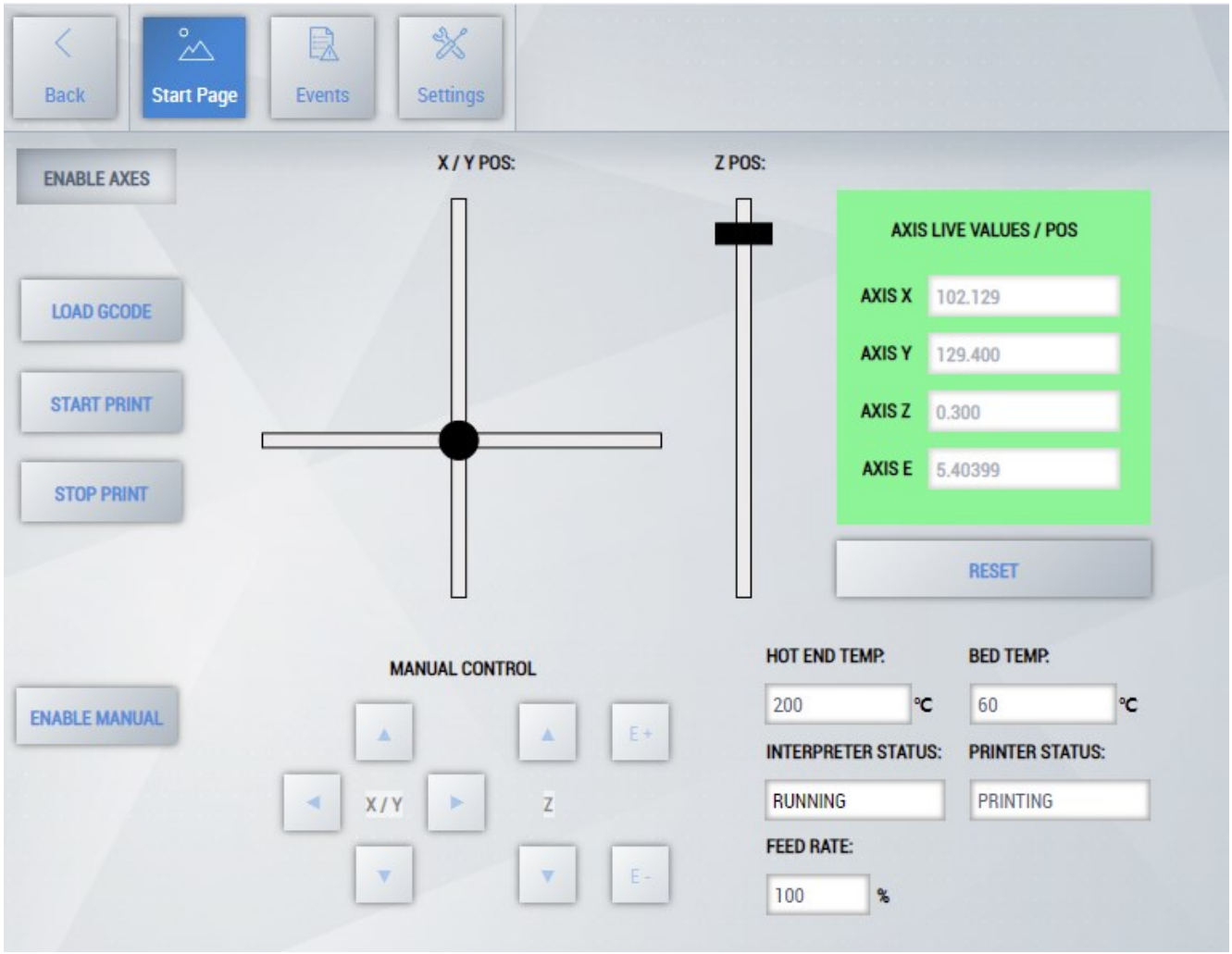
Tulolla ”bExecute” suoritetaan nollaus. Tuloon ”sNciToPlc” määritellään kommunikointikanava NCI-ohjelman ja PLC:n välille (NCTOPLC_NCICHANNEL_REF parametrillä). Muut tulot ja lähdöt ovat samoja mitä aikaisemmassa ”CfgBuildExt3DGroup”-toimintolohkoa koskevassa alaluvussa käsiteltiin ja käyttäytyvät samalla tavalla. Tämä tulo on ”tTimeOut”. Lähdöt ovat samat ”bBusy”, ”bErr” ja ”nErrId”.

3.3.5 ItpStartStopEx

Kuva 14. ItpStartStopEx-toimintolohko havainnollistettuna (Beckhoff, 2023a, s. 245).

ItpStartStopEx-toimintolohkolla käynnistää (bStart) tai lopettaa (bStop) G-kooditiedoston ajamisen NCI-kanavalle. Tuloon ”sNciToPlc” määritellään kommunikointikanava NCI-ohjelman ja PLC:n välille (NCTOPLC_NCICHANNEL_REF parametrillä). Muut tulot ja lähdöt ovat samoja mitä aikaisemmassa ”CfgBuildExt3DGroup”-toimintolohkoa koskevassa alaotsikossa käsiteltiin ja käyttäytyvät samalla tavalla. Tämä tulo on ”tTimeOut”. Lähdöt ovat samat ”bBusy”, ”bErr” ja ”nErrId”.

3.4 Työhön suunniteltu käyttöliittymä



Kuva 15. Kuvankaappaus työhön luodusta käyttöliittymästä.

Työhön luotiin yksinkertainen käyttöliittymä hyödyntäen TwinCAT-ohjelmiston lisäosaa ihminen-kone-rajapinta (HMI).

Käyttöliittymä sisältää mahdollisuuden aktivoida liikeakselit painamalla ”ENABLE AXES” painonappia. Alempana löytyvät NCI-ohjaukseen liittyvät painonapit, kuten G-koodin lataaminen (LOAD GCODE), tulostuksen aloittaminen (START PRINT), tulostuksen pysäyttäminen (STOP PRINT) ja käsiohjauksen aktivointi (ENABLE MANUAL).

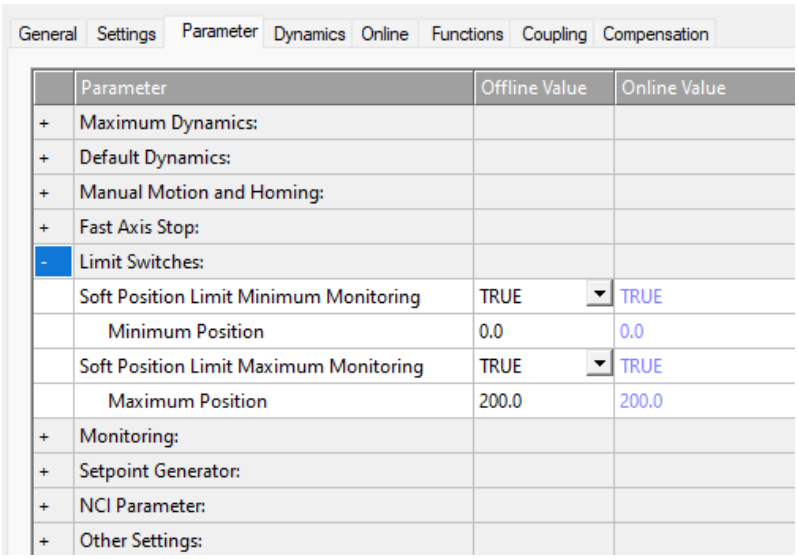
Käyttöliittymän keskelle luotiin myös graafiset elementit, joista voi tarkastella tulostuksen edistymistä ja liikeakseleiden liikettä (X, Y ja Z) suorana. Myös akseleiden ajamat arvot ovat näkyvillä käyttöliittymän vihreän laatikon sisällä (AXIS LIVE VALUES/POS). Tässä on esitettynä myös pursottimen (E) suora arvo ajon aikana.

Alempaa käyttöliittymästä löytyy käsiajon liikuttamiseen tarvittavat painonapit. Näillä voidaan liikuttaa X-, Y- ja Z-akseleita manuaalisesti. Myös pursottimen (E) käsiohjaus on mahdollista. Käsiohjauksen vierestä löytyy myös S-parametrejä koskevia tekstikenttiä, joissa näkyy G-koodissa asetetut pursottimen lämpötilatieto (HOT END TEMP) ja tulostusalustalle asetettu lämpötilatieto (BED TEMP).

Näiden alla löytyy myös NCI-kanavan tilatieto (INTREPETER STATUS) joissa yleisimmät tilat ovat ”IDLE”, ”READY” tai ”RUNNING”. Vierelle tehtiin myös M-funktioita esittävä tekstikenttä (PRINTER STATUS), jossa tällä hetkellä voi lukea M109 pursottimen lämmitysfunktion tapahtuessa ”HOT END HEATING” tai M190 tulostusalustan lämmitysfunktion tapahtuessa ”BED HEATING”. Näiden alle tehtiin vielä mahdollisuus muuttaa liikeakseleiden nopeutta (FEED RATE) arvolla 0–100 %.

3.5 Lähestymiskytkimet

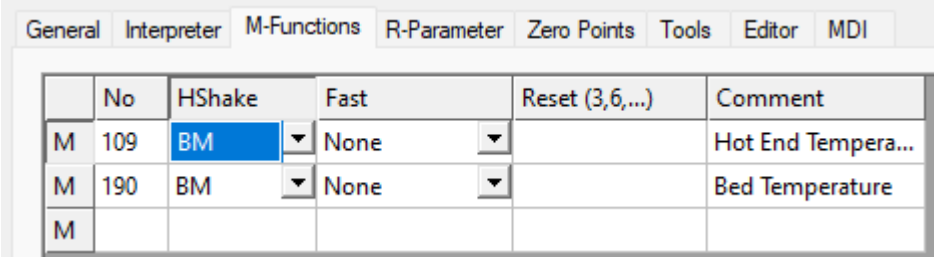
Koska työssä ei ollut käytössä fyysistä rautaa, piti TwinCAT-ohjelmistosta asettaa jokaiselle liikeakselille simuloidut ohjelmistopuolen rajakytkimet päälle, että akseleita ei voitaisi ajaa yli arvojen. Nämä rajat voidaan asettaa MC2-ohjaimessa jokaisen akselin kohdalla alivalikosta ”Parameter” ja valitsemalla listalta ”Limit Switches”. Työssä rajat asetettiin X-, Y- ja Z-akseleille välille 0–200 mm.



Kuva 16. Kuvankaappaus akselin "Parameter" alivalikosta löytyvästä ohjelmistopuolen lähestymiskytkimien (Limit Switches) rajojen asettamisesta välille 0–200 mm.

3.6 M-funktioiden parametrointi

TwinCAT-ohjelmassa on mahdollisuus parametroida jokainen M-funktio toimimaan eri tavalla. Nämä parametrisointiasetukset löytyvät NCI-tulkin asetuksista alivalikosta ”M-Functions”. Tässä alivalikossa voidaan määritellä, tapahtuvatko M-funktioiden toiminnot ennen seuraavaa liikekomentoa (Before Move) vai liikekomennon jälkeen (After Move) kohdassa ”HShake”. Asetuksissa voidaan myös valita, nollaako jokin muu M-funktio asetetun M-funktion (Reset).



Kuva 17. Kuvankaappaus M-funktioiden parametrointivalikosta. Kuvassa asetetut M-funktiot on määritelty toimimaan BM- eli ”Before Move” -tilassa.

4 TULOKSET, YHTEENVETO JA POHDINTAA

Työn alkuperäiseen tavoitteeseen päästiin ja saatiin luotua yksinkertainen ohjelma, jolla voitaisiin teoreettisesti ohjata 3D-tulostinta. Tätä ei kuitenkaan ole työssä todennettu, koska työ ei sisältänyt fyysistä rautapuolen osuutta. Opinnäytetyön tekijällä itsellään on aikaisempaa kokemusta pienen kokoluokan 3D-tulostimista, mikä helpotti työn edistymistä ja työn tavoitteiden ymmärtämistä.

Työssä tehty lähdekoodi voi olla hyödyllinen miniFactorylle tulevaisuudessa, jos he alkavat ohjelmoimaan omaa logiikkaohjainta Beckhoffin valmistamaa PLC:tä käyttäen. Työn M-funktion osa-aluetta olisi voitu tehdä hieman laajemmaksi, mutta lähdekoodista voidaan hyvin päätellä, miten näitä funktioita ohjelmoidaan ja ajetaan. Työssä luotua lähdekoodia pitäisi olla helppo tulkita ja muokata yrityksen omiin 3D-tulostustarpeisiin. Nykyinen versio NCI-kirjastosta osoittautui kuitenkin hieman ongelmalliseksi varsinkin viipalointiohjelmistoilla generoitujen G-kooditiedostojen käsittelyssä. Nykyiset viipalointiohjelmistojen generoimat G-koodit eivät ole yhteensopivia NCI-kääntäjän kanssa ja vaativat paljon muokkaamista koodiin, että kääntäjä osaa lukea näitä generoituja tiedostoja.

Työn tulevaisuutta ajatellen olisi voitu tehdä jonkinlainen kääntäjäohjelmisto, joka automaattisesti muuntaisi G-kooditiedostot NCI-tulkille ymmärrettävään muotoon. Ohjelma voisi käyttää regex-sääntöjä G-koodirivien muuntamiseen.

Beckhoffilta on lähitulevaisuudessa tulossa muutoksia ja päivityksiä kyseiseen kirjastoon, ja muutosten pitäisi helpottaa juuri 3D-tulostukseen tarkoitettujen laitteistojen ohjelmointia. Ikävästi kyseisiä päivityksiä ja parannuksia ei ollut vielä julkaistu tämän opinnäytetyön tekemiseen mennessä.

LÄHTEET

3DPrinting. (i.a.). *What is 3D Printing?* https://3dprinting.com/what-is-3d-printing/

All3DP. (21.5.2022a). *3D Printer G-code Commands: Main List & Quick Tutorial.* https://all3dp.com/2/3d-printer-g-code-commands-list-tutorial/

All3DP. (13.11.2022b). *What Is a 3D Slicer? – Simply Explained*. https://all3dp.com/2/what-is-a-3d-slicer-simply-explained/

All3DP. (3.2.2023). *The 7 Main Types of 3D Printing Technology/Additive Manufacturing.* https://all3dp.com/1/types-of-3d-printers-3d-printing-technology/

Beckhoff. (i.a.). *Beckhoff Automation*. https://www.beckhoff.com/en-en/company/

Beckhoff. (2021). *TwinCAT 3: The flexible software solution for PC-based control.* https://download.beckhoff.com/download/Document/Catalog/Beckhoff_TwinCAT3_e.pdf

Beckhoff. (21.11.2022). *TwinCAT 3: Product overview.* https://download.beckhoff.com/download/document/automation/twincat3/Product_overview_EN.pdf

Beckhoff. (31.3.2023a). *TF5100 TwinCAT 3: NC I.* https://download.beckhoff.com/download/Document/automation/twincat3/TF5100_TC3_NC_I_EN.pdf

Beckhoff. (5.4.2023b). *TE1000 TwinCAT 3 PLC Library: Tc2_MC2.* https://download.beckhoff.com/download/Document/automation/twincat3/TwinCAT_3_PLC_Lib_Tc2_MC2_EN.pdf

Minifactory. (i.a.-a). *Minifactory – Your trusted partner in additive manufacturing*. https://minifactory.fi/

Minifactory. (i.a.-b). *Ultra 2: Make it right. Every time.* https://minifactory.fi/industrial-3d-printer/ultra-2/

PLCopen. (i.a.). *Motion Control.* https://plcopen.org/technical-activities/motion-control

PLCopen. (17.3.2011). *Function blocks for motion control.* https://plcopen.org/system/files/downloads/plcopen_motion_control_part_1_version_2.0.pdf

Simplify3D. (i.a.). *3D Printing G-code Tutorial.* https://www.simplify3d.com/resources/articles/3d-printing-gcode-tutorial/

LIITTEET

Liite 1. TwinCAT 3 MAIN-ohjelmakoodi

```
PROGRAM MAIN
VAR
	//interpreter controls and user inputs
	bUserStartItp		: BOOL;
	bUserStopItp		: BOOL;
	bUserResetItp		: BOOL;
	bUserLoadProgram	: BOOL;

	//user defined interpreter feed rate (default = 100)
	nUserFeedRate		: INT:=100;

	//manual controls and user inputs
	bUserEnableAxes		: BOOL;
	bUserResetAxes		: BOOL;
	bUserManCtrl		: BOOL;

	bUserXJogPlus		: BOOL;
	bUserXJogMinus		: BOOL;
	bUserYJogPlus		: BOOL;
	bUserYJogMinus		: BOOL;
	bUserZJogPlus		: BOOL;
	bUserZJogMinus		: BOOL;
	bUserEJogPlus		: BOOL;
	bUserEJogMinus		: BOOL;

	//motion control definitions
	io_X			: AXIS_REF;
	io_Y			: AXIS_REF;
	io_Z			: AXIS_REF;
	io_E			: AXIS_REF;

	fbEnableAxes		: FB_EnableAxes;

	fbResetAxes		: FB_ResetAxes;
	bResetAxes		: BOOL;

	fbMoveAxes		: FB_MoveAxes;
	bXMove			: BOOL;
	bYMove			: BOOL;
	bZMove			: BOOL;
	bEMove			: BOOL;
	fXPos			: LREAL;
	fYPos			: LREAL;
	fZPos			: LREAL;
	fEPos			: LREAL;


	fbItp			: FB_ItpControl;
	itpStart		: BOOL;
```

```
        itpStop                     : BOOL;

        //m-function confirm
        fbConfirmHsk                : ItpConfirmHsk;
        bConfirmHsk                 : BOOL:=FALSE;

        nItpState                   : UDINT;

        //itp state as string for HMI
        sState                      : STRING(25);

        //read/write information from/to PLC
        in_stItpToPlc     AT %I*    : NcToPlc_NciChannel_Ref;
        out_stPlcToItp    AT %Q*    : PLCTONC_NCICHANNEL_REF;



        //machine/miscellaneous functions
        //simulate hot end temps
        bHotEndHeatEna              : BOOL;
        nHotEndTemp                 : INT;

        //simulate bed temps
        bBedHeatEna                 : BOOL;
        nBedTemp                    : INT;

        //s-parameter info
        nSParam                     : UINT;

        //m-function state as string for HMI
        sMfuncState                 : STRING(25);

        testTimer                   : TON;
END_VAR
```

```
//enable axes
fbEnableAxes(
          bEnableAxes:=bUserEnableAxes,
          stX:=io_X,
          stY:=io_Y,
          stZ:=io_Z,
          stE:=io_E,
          bOverride:= ,
          bAllAxesReady=> );

//reset axes
fbResetAxes(
          bResetAxes:= bResetAxes,
          bResetAxesDone=> ,
          stX:=io_X,
          stY:=io_Y,
          stZ:=io_Z,
          stE:=io_E);

//absolute move and jog controls
fbMoveAxes(
          fbXPos:=fXpos,
          fbYPos:=fYpos,
          fbZPos:=fZpos,
          fbEPos:=fEpos,
          fbXAxisMove:=bXMove,
          fbYAxisMove:=bYMove,
          fbZAxisMove:=bZMove,
          fbEAxisMove:=bEMove,
          fbXJogPlus:= ,
          fbXJogMinus:= ,
          fbYJogPlus:= ,
          fbYJogMinus:= ,
          fbZJogPlus:= ,
          fbZJogMinus:= ,
          fbEJogPlus:= ,
          fbEJogMinus:= ,
          stX:=io_X,
          stY:=io_Y,
          stZ:=io_Z,
          stE:=io_E);

//interpreter controls
fbItp(
          start:=itpStart,
          stop:=itpStop,
          reset:=bUserResetItp,
          load:=bUserLoadProgram,
          nFeedRate:=nUserFeedRate,
          stItpToPlc:=in_stItpToPlc,
```

```
          stPlcToItp=>out_stPlcToItp,
          stX:=io_X,
          stY:=io_Y,
          stZ:=io_Z,
          stE:=io_E);

//confirm M-function and continue g-code
fbConfirmHsk(
          bExecute:=bConfirmHsk ,
          sNciToPlc:=in_stItpToPlc ,
          sPlcToNci:=out_stPlcToItp ,
          bBusy=> ,
          bErr=> ,
          nErrId=> );

//get M-function S parameter information
nSParam:=ItpGetSParam(sNciToPlc:=in_stItpToPlc);

//get iterpreter state
nItpState:=ItpGetStateInterpreter(in_stItpToPlc);

//timer to simulate time passed
testTimer(IN:= , PT:= , Q=> , ET=> );

//reset variables to defaults
bResetAxes:=FALSE;
bXMove:=FALSE;
bYMove:=FALSE;
bZMove:=FALSE;
bEMove:=FALSE;
itpStart:=FALSE;
itpStop:=FALSE;
testTimer.IN:=FALSE;
```

```
//print start and stop (interpreter)
IF bUserEnableAxes AND bUserStartItp AND NOT bUserResetAxes AND NOT
bUserManCtrl THEN
         itpStart:=TRUE;
END_IF
IF bUserEnableAxes AND bUserStopItp AND NOT bUserResetAxes AND NOT
bUserManCtrl THEN
         itpStop:=TRUE;
END_IF

//enable jog controls in manual mode
IF bUserEnableAxes AND bUserManCtrl AND NOT bUserResetAxes THEN
         fbMoveAxes.fbXJogPlus:=bUserXJogPlus;
         fbMoveAxes.fbXJogMinus:=bUserXJogMinus;
         fbMoveAxes.fbYJogPlus:=bUserYJogPlus;
         fbMoveAxes.fbYJogMinus:=bUserYJogMinus;
         fbMoveAxes.fbZJogPlus:=bUserZJogPlus;
         fbMoveAxes.fbZJogMinus:=bUserZJogMinus;
         fbMoveAxes.fbEJogPlus:=bUserEJogPlus;
         fbMoveAxes.fbEJogMinus:=bUserEJogMinus;
END_IF

//reset and move axes to reference
IF bUserResetAxes THEN
         bResetAxes:=TRUE;

         fXpos:=0;
         fYpos:=0;
         fZpos:=0;
         //fEpos:=0;

         bXMove:=TRUE;
         bYMove:=TRUE;
         bZMove:=TRUE;
         //bEMove:=TRUE;

         bHotEndHeatEna:=FALSE;
         nHotEndTemp:=0;

         bBedHeatEna:=FALSE;
         nBedTemp:=0;

         sMfuncState:='RESET';
END_IF
```

```
//reads and confirms M-function
IF ItpIsHskMFunc(in_stItpToPlc) AND NOT fbConfirmHsk.bBusy THEN
ItpGetHskMFunc(in_stItpToPlc);

        IF in_stItpToPlc.HskMFuncNo=109 THEN
                sMfuncState:='HOT END HEATING';
                bHotEndHeatEna:=TRUE;
                nHotEndTemp:=UDINT_TO_INT(nSParam);

                //wait 10 seconds to simulate time passed
                testTimer.PT:=T#10S;
                testTimer.IN:=TRUE;

        IF testTimer.Q AND nHotEndTemp=UDINT_TO_INT(nSParam) THEN
                        sMfuncState:='PRINTING';
                        bConfirmHsk:=TRUE;
                END_IF
        END_IF

        IF in_stItpToPlc.HskMFuncNo=190 THEN
                sMfuncState:='BED HEATING';
                bBedHeatEna:=TRUE;
                nBedTemp:=UDINT_TO_INT(nSParam);

                //wait 10 seconds to simulate time passed
                testTimer.PT:=T#10S;
                testTimer.IN:=TRUE;

        IF testTimer.Q AND nBedTemp=UDINT_TO_INT(nSParam) THEN
                        sMfuncState:='PRINTING';
                        bConfirmHsk:=TRUE;
                END_IF
        END_IF

ELSE
        bConfirmHsk:=FALSE;
END_IF
```

```
//itp state to string
IF nItpState=1 THEN
         sState:='IDLE';
ELSIF nItpState=2 THEN
         sState:='READY';
ELSIF nItpState=5 THEN
         sState:='RUNNING';
ELSIF nItpState=13 THEN
         sState:='FAULT';
END_IF
//list of possible state codes
(*0 ITP_STATE_INITFAILED
1 ITP_STATE_IDLE
2 ITP_STATE_READY
3 ITP_STATE_STARTED
4 ITP_STATE_SCANNING
5 ITP_STATE_RUNNING
6 ITP_STATE_STAY_RUNNING
7 ITP_STATE_WRITETABLE
8 ITP_STATE_SEARCHLINE
9 ITP_STATE_END
10 ITP_STATE_SINGLESTOP
11 ITP_STATE_ABORTING
12 ITP_STATE_ABORTED
13 ITP_STATE_FAULT
14 ITP_STATE_RESET
15 ITP_STATE_STOP
16 ITP_STATE_WAITFUNC
17 ITP_STATE_FLUSHBUFFERS*)

//timer must be called at the end of program
testTimer();
```

Liite 2. TwinCAT 3 FB_EnableAxes-ohjelmakoodi

```
FUNCTION_BLOCK FB_EnableAxes
VAR_INPUT
         bEnableAxes        : BOOL;
         bOverride          : LREAL:=100;
END_VAR
VAR_OUTPUT
         bAllAxesReady      : BOOL;
END_VAR
VAR_IN_OUT
         stX                : AXIS_REF;
         stY                : AXIS_REF;
         stZ                : AXIS_REF;
         stE                : AXIS_REF;
END_VAR
VAR
         fbPowerX           : MC_Power;
         fbPowerY           : MC_Power;
         fbPowerZ           : MC_Power;
         fbPowerE           : MC_Power;
END_VAR
```

```
stX.ReadStatus();
stY.ReadStatus();
stZ.ReadStatus();
stE.ReadStatus();

fbPowerX(
         Enable:=bEnableAxes,
         Enable_Positive:=bEnableAxes,
         Enable_Negative:=bEnableAxes,
         Override:= ,
         BufferMode:= ,
         Axis:=stX,
         Status=> ,
         Busy=> ,
         Active=> ,
         Error=> ,
         ErrorID=> );

fbPowerY(
         Enable:=bEnableAxes,
         Enable_Positive:=bEnableAxes,
         Enable_Negative:=bEnableAxes,
         Override:= ,
         BufferMode:= ,
         Axis:=stY,
         Status=> ,
         Busy=> ,
         Active=> ,
         Error=> ,
         ErrorID=> );

fbPowerZ(
         Enable:=bEnableAxes,
         Enable_Positive:=bEnableAxes,
         Enable_Negative:=bEnableAxes,
         Override:= ,
         BufferMode:= ,
         Axis:=stZ,
         Status=> ,
         Busy=> ,
         Active=> ,
         Error=> ,
         ErrorID=> );
```

```
fbPowerE(
         Enable:=bEnableAxes,
         Enable_Positive:=bEnableAxes,
         Enable_Negative:=bEnableAxes,
         Override:= ,
         BufferMode:= ,
         Axis:=stE,
         Status=> ,
         Busy=> ,
         Active=> ,
         Error=> ,
         ErrorID=> );

bAllAxesReady:= fbPowerX.Status AND fbPowerY.Status AND
fbPowerZ.Status AND fbPowerE.Status;
bOverride:= fbPowerX.Override := fbPowerY.Override := fbPowerZ.Over-
ride := fbPowerE.Override;
```

Liite 3. TwinCAT 3 FB_ItpControl-ohjelmakoodi

```
FUNCTION_BLOCK FB_ItpControl
VAR_INPUT
        start                       : BOOL;
        stop                        : BOOL;
        reset                       : BOOL;
        load                        : BOOL;
        enaSingleBlock              : BOOL;
        trgSingleBlock              : BOOL;

        nFeedRate                   : LREAL;

        stItpToPlc                  : NCTOPLC_NCICHANNEL_REF;
END_VAR
VAR_OUTPUT
        stPlcToItp                  : PLCTONC_NCICHANNEL_REF;
END_VAR
VAR_IN_OUT
        stX                         : AXIS_REF;
        stY                         : AXIS_REF;
        stZ                         : AXIS_REF;
        stE                         : AXIS_REF;
END_VAR
VAR
        cGroup                      : CfgBuildExt3DGroup;
        createGroup                 : BOOL;

        dGroup                      : CfgReconfigGroup;
        deleteGroup                 : BOOL;

        loadProg                    : ItpLoadProgEx;
        loadProgram                 : BOOL;
        progName  : STRING(255):='cube_552_axistest.gcode';

        nInterpreterState           : UDINT:=0;

        resetChannel                : ItpResetEx2;

        startStop                   : ItpStartStopEx;
        itpStart                    : BOOL;
END_VAR
```

```
//create 3D interpolation group
cGroup(
         bExecute:=createGroup,
         nGroupId:=ItpGetGroupId(sNciToPlc:=stItpToPlc),
         nXAxisId:=stX.NcToPlc.AxisId,
         nYAxisId:=stY.NcToPlc.AxisId,
         nZAxisId:=stZ.NcToPlc.AxisId,
         nQ1AxisId:=stE.NcToPlc.AxisId,
         nQ2AxisId:=0,
         nQ3AxisId:=0,
         nQ4AxisId:=0,
         nQ5AxisId:=0,
         tTimeOut:=,
         bBusy=> ,
         bErr=> ,
         nErrId=> );

//delete 3D interpotaltion group
dGroup(
         bExecute:=deleteGroup,
         nGroupId:=ItpGetGroupId(sNciToPlc:=stItpToPlc),
         tTimeOut:=,
         bBusy=> ,
         bErr=> ,
         nErrId=> );

//interpreter feedrate
ItpSetOverridePercent(fOverridePercent:=nFeedrate,
sPlcToNci:=stPlcToItp);

//load gcode program (progName)
//default folder= ... \TwinCAT\Mc\Nci\
loadProg(
         bExecute:=loadProgram,
         sNciToPlc:=stItpToPlc,
         sPrg:=progName,
         nLength:=INT_TO_UDINT(LEN(progName)),
         tTimeOut:=,
         bBusy=> ,
         bErr=> ,
         nErrId=> );

//reset interpreter
resetChannel(
         bExecute:=reset,
         tTimeOut:=,
         sNciToPlc:=stItpToPlc,
         bBusy=> ,
         bErr=> ,
         nErrId=> );
```

```
//start or stop iterpreter
startStop(
          bStart:=itpStart,
          bStop:=stop,
          sNciToPlc:=stItpToPlc,
          tTimeOut:=,
          bBusy=> ,
          bErr=> ,
          nErrId=> );

//reset variables to default
createGroup:=FALSE;
deleteGroup:=FALSE;
itpStart:=FALSE;

//load gcode
IF load THEN
          loadProgram:=TRUE;
END_IF

//start printing if interpreter state = READY
IF start THEN
          createGroup:=TRUE;
          nInterpreterState  := ItpGetStateInter-
preter(sNciToPlc:=stItpToPlc);
          IF nInterpreterState = Tc2_NCI.NCI_INTERPRETER_READY THEN
                    itpStart:=TRUE;
          END_IF
END_IF

//reset and delete group
IF reset THEN
          loadProgram:=FALSE;
          deleteGroup:=TRUE;
END_IF
```

Liite 4. TwinCAT 3 FB_MoveAxes-ohjelmakoodi

```
FUNCTION_BLOCK FB_MoveAxes
VAR_INPUT
         fbXPos                    : LREAL;
         fbYPos                    : LREAL;
         fbZPos                    : LREAL;
         fbEPos                    : LREAL;

         fbXAxisMove               : BOOL;
         fbYAxisMove               : BOOL;
         fbZAxisMove               : BOOL;
         fbEAxisMove               : BOOL;

         fbXJogPlus                : BOOL;
         fbXJogMinus               : BOOL;
         fbYJogPlus                : BOOL;
         fbYJogMinus               : BOOL;
         fbZJogPlus                : BOOL;
         fbZJogMinus               : BOOL;
         fbEJogPlus                : BOOL;
         fbEJogMinus               : BOOL;
END_VAR
VAR_OUTPUT
         bXMoveDone                : BOOL;
         bYMoveDone                : BOOL;
         bZMoveDone                : BOOL;
         bEMoveDone                : BOOL;
END_VAR
VAR_IN_OUT
         stX                       : AXIS_REF;
         stY                       : AXIS_REF;
         stZ                       : AXIS_REF;
         stE                       : AXIS_REF;
END_VAR
VAR
         fbXMove                   : MC_MoveAbsolute;
         fbYMove                   : MC_MoveAbsolute;
         fbZMove                   : MC_MoveAbsolute;
         fbEMove                   : MC_MoveAbsolute;

         fbXJog                    : MC_Jog;
         fbYJog                    : MC_Jog;
         fbZJog                    : MC_Jog;
         fbEJog                    : MC_Jog;
END_VAR
```

```
stX.ReadStatus();
stY.ReadStatus();
stZ.ReadStatus();
stE.ReadStatus();

fbXMove(
         Axis:= stX,
         Execute:= fbXAxisMove,
         Position:= fbXPos,
         Velocity:= 100,
         Acceleration:= 1000,
         Deceleration:= 1000,
         Jerk:= 1000,
         BufferMode:= ,
         Options:= ,
         Done=>bXMoveDone ,
         Busy=> ,
         Active=> ,
         CommandAborted=> ,
         Error=> ,
         ErrorID=> );

fbYMove(
         Axis:= stY,
         Execute:= fbYAxisMove,
         Position:= fbYPos,
         Velocity:= 100,
         Acceleration:= 1000,
         Deceleration:= 1000,
         Jerk:= 1000,
         BufferMode:= ,
         Options:= ,
         Done=>bYMoveDone ,
         Busy=> ,
         Active=> ,
         CommandAborted=> ,
         Error=> ,
         ErrorID=> );

fbZMove(
         Axis:= stZ,
         Execute:= fbZAxisMove,
         Position:= fbZPos,
         Velocity:= 100,
         Acceleration:= 1000,
         Deceleration:= 1000,
         Jerk:= 1000,
         BufferMode:= ,
         Options:= ,
         Done=>bZMoveDone ,
         Busy=> ,
```

```
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

fbEMove(
        Axis:= stE,
        Execute:= fbEAxisMove,
        Position:= fbEPos,
        Velocity:= 100,
        Acceleration:= 1000,
        Deceleration:= 1000,
        Jerk:= 1000,
        BufferMode:= ,
        Options:= ,
        Done=>bEMoveDone ,
        Busy=> ,
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

fbXJog(
        Axis:= stX,
        JogForward:= fbXJogPlus,
        JogBackwards:= fbXJogMinus,
        Mode:= ,
        Position:= ,
        Velocity:= 100,
        Acceleration:= 100,
        Deceleration:= 100,
        Jerk:= 100,
        Done=> ,
        Busy=> ,
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

fbYJog(
        Axis:= stY,
        JogForward:= fbYJogPlus,
        JogBackwards:= fbYJogMinus,
        Mode:= ,
        Position:= ,
        Velocity:= 100,
        Acceleration:= 100,
        Deceleration:= 100,
        Jerk:= 100,
        Done=> ,
        Busy=> ,
```

```
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

fbZJog(
        Axis:= stZ,
        JogForward:= fbZJogPlus,
        JogBackwards:= fbZJogMinus,
        Mode:= ,
        Position:= ,
        Velocity:= 100,
        Acceleration:= 100,
        Deceleration:= 100,
        Jerk:= 100,
        Done=> ,
        Busy=> ,
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

fbEJog(
        Axis:= stE,
        JogForward:= fbEJogPlus,
        JogBackwards:= fbEJogMinus,
        Mode:= ,
        Position:= ,
        Velocity:= 100,
        Acceleration:= 100,
        Deceleration:= 100,
        Jerk:= 100,
        Done=> ,
        Busy=> ,
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

Liite 5. TwinCAT 3 FB_ResetAxes-ohjelmakoodi

```
FUNCTION_BLOCK FB_ResetAxes
VAR_INPUT
         bResetAxes                  : BOOL;
END_VAR
VAR_OUTPUT
         bResetAxesDone              : BOOL;
END_VAR
VAR_IN_OUT
         stX                         : AXIS_REF;
         stY                         : AXIS_REF;
         stZ                         : AXIS_REF;
         stE                         : AXIS_REF;
END_VAR
VAR
         fbResetX                    : MC_Reset;
         fbResetY                    : MC_Reset;
         fbResetZ                    : MC_Reset;
         fbResetE                    : MC_Reset;
END_VAR
```

```
fbResetX(
         Axis:=stX,
         Execute:=bResetAxes,
         Done=> ,
         Busy=> ,
         Error=> ,
         ErrorID=> );

fbResetY(
         Axis:=stY,
         Execute:=bResetAxes,
         Done=> ,
         Busy=> ,
         Error=> ,
         ErrorID=> );

fbResetZ(
         Axis:=stZ,
         Execute:=bResetAxes,
         Done=> ,
         Busy=> ,
         Error=> ,
         ErrorID=> );

fbResetE(
         Axis:=stE,
         Execute:=bResetAxes,
         Done=> ,
         Busy=> ,
         Error=> ,
         ErrorID=> );

bResetAxesDone    := fbResetX.Done AND fbResetY.Done AND fbRe-
setZ.Done AND fbResetE.Done;
```