View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Theseus

Opinnäytetyö (AMK)

Kone- ja tuotantotekniikan koulutusohjelma

Tuotantopainotteinen

2011

Jari Lehtonen

MAKRO-OHJELMOINNIN KOULUTUSMATERIAALI FANUC-OHJAUKSELLE

– OPISKELIJAN OPAS

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Kone- ja tuotantotekniikan koulutusohjelma | Tuotantopainotteinen

Syyskuu 2011 | Sivumäärä 60

Ohjaaja: Pekka Törnqvist

Tekijä: Jari Lehtonen

MAKRO-OHJELMOINNIN KOULUTUSMATERIAALI FANUC OHJAUKSELLE

Tämä opinnäytetyö sai alkunsa talvella 2010 20-sivuisesta pikaoppaasta. Valmis teos on yli 50-sivuinen ja sisältä erittäin laajan koulutusmateriaalin makro-ohjelmoinnin kiehtovaan maailmaan. Työ tehtiin yhteistyössä Salon seudun aikuisopiston ja Espoon ammattiopiston kanssa. Yhteistyössä oli lisäksi mukana salolainen alihankintakonepaja Halikko Tools Oy, jossa kaikki ohjelmat testattiin käytännön töissä.

Tätä koulutusmateriaalia voidaan käyttää sekä nuorten että aikuisten koneistajan koulutuksessa. Lisäksi sitä voidaan hyödyntää ammattikorkeakouluissa, esim. koneistukseen liittyvien kurssien lisämateriaalina. Koulutusmateriaalia voidaan hyödyntää koneistuksen ammattilaisten täydennyskoulutuksien pohjana. Materiaali sopii hyvin myös itseopiskeluun. Lähtökohtana on kuitenkin se, että kaikki kursseille osallistujat hallitsevat vähintään FANUC-ohjelmoinnin perusteet. Tätä koulutusmateriaalia voidaan käyttää niin sorvauksessa kuin koneistuksessakin, kunhan käytettävässä koneessa on FANUC 10 tai siitä uudempi ohjaus.

Koulutusmateriaalissa käsitellään kaikki tärkeimmät muuttujat, kuten paikallismuuttujat ja yleismuuttujat sekä niiden käyttö. Materiaali sisältää paljon hyviä esimerkkejä ja makro-ohjelmia lauseittain selitettynä. Siinä on mukana valmiita harjoitustehtäviä ja työkuvia, joihin makroja voidaan hyödyntää.

Opettajan versiossa on kaikkiin harjoituksiin tarvittavat makro-ohjelmat ja niiden käyttöohjeet. Siinä olevia makroja voidaan hyödyntää sellaisenaan suoraan tuotantoon.

ASIASANAT:

Fanuc, CNC-työstökone, CNC-sorvi, makro-ohjelmointi, parametri

BACHELOR´S THESIS | ABSTRACT

UNIVERSITY OF APPLIED SCIENCES

Mechanical and Production Engineering

October 2011| Total number of pages 60

Instructor: Pekka Törnqvist

Author Jari Lehtonen

TRAINING MATERIAL FOR PROGRAMMING BY USEING FANUC CUSTOM MAKRO

I started this thesis in winter 2010. In the beginning it was just a brief quick start guide. But now you have in your hands a very extensive education material for the fascinating world of macro-programming containing more than 60 pages.

This educational material can be used for in vocational schools, occupational adult education, university of applied sciences and self-access learning. The starting point is however that you have the basic information for FANUC programming. This educational material can be used for turning and milling but the machine that you use must be FANUC 10 or more modern control.

Educational material go through all the most significant variables and their use. The material includes a lot of good examples. Exercises are quite easy. In addition, the material includes ready macro-programs which are explained in detail.

If you order the teachers version you get the same exercises with all the necessary macro-programs and instructions for use. You can use them directly for production. All involved in the programs are tested and revised.

KEYWORDS:

Fanuc, CNC-machines, custom makro B, programming, parameter.

ALKUSANAT

Erityskiitokset kuuluvat perheelleni ja ennen kaikkea vaimolleni, joka suonut minulle mahdollisuuden tämän työn tekemiseen. Haluan kiittää myös Juho Kesseliä, joka työskentelee Salon seudun aikuisopistossa, koneistuspuolen kouluttajana ja joka on antanut minulle loistavia neuvoja ja paljon apu työhöni. Kiitos ohjaajalleni Pekka Törnqvistille, jolta olen saanut runsaasti hyviä ohjeita ja mukavasti kannustusta. Suuret kiitokset myös Halikko Toolsin toimitusjohtajalle Jari Juhannusvuorelle, joka oli yhteistyössä mukana ja antoi yrityksen koneita testauskäyttöön.

Olen toiminut koneistuksen parissa n. 10 vuotta, mutta tehdessäni tätä opinnäytetyötä olen saanut valtavasti uusia ulottuvuuksia sorvaamisesta ja koneistamisesta sekä niihin käytettävien koneiden ohjelmoinnista. Samalla tämän projektin aikana minulle on yhä paremmin selvinnyt CNC-koneiden koko toimintaperiaate. Tehtävä on ollut erittäin mielenkiintoinen ja samalla se on ollut myös hyvin haasteellinen.

Haluan vielä kiittää työtäni auttaneita henkilöitä:

Nina Tuupanen, Omnia

Jarmo Palonen, Omnia

Timo Sirkiä, SSKKY

Salossa 27.06.2011

Jari Lehtonen

SISÄLTÖ

KÄYTETYT LYHENTEET JA SANASTO 6
1 JOHDANTO 7
2 FANUC-OHJELMOINTI 8
2.1 Yleistä 8
2.2 Aliohjelmointi 10
3 MAKRO- VAI PARAMETRIOHJELMOINTI 11
3.1 Ohjelmointiesimerkki muuttujaohjelmasta ja makro-ohjelmasta. 13
4 MAKRO-OHJELMOINNIN ETUJA 14
5 MAKRO-OHJELMOINTI VS. CAM-OHJELMOINTI 15
6 MUUTTUJIEN KÄYTTÖ 17
7 MUUTTUJIEN RYHMITTELY 19
7.1 Määrittelemättömät muuttujat 19
7.2 Paikallismuuttujat (#1 - #33) 21
7.3 Yleismuuttujat (#100 - #149 ja #500 - #599) 22
7.4 Järjestelmämuuttujat 24
8 ARITMEETTISET JA LOOGISET OPERAATIOT 34
9 OHJAUSKÄSKYT 35
9.1 Hypyt ja ehdot 36
9.2 Toisto 37
10 RAJOITUKSET 38
11 MAKRO-OHJELMIEN KUTSUT 39
11.1 Makro-ohjelman kutsu G- ja M-koodilla 42
12 MAKRO-OHJELMIEN LAATIMINEN 43
13 MAKROJEN KÄYTTÖ SORVAUSOHJELMIEN APUNA 47
14 LOPPU HARJOITUS 53
15 YHTEENVETO 54
LÄHTEET 55

LIITTEET

Liite 1. Reikäpiirimakron esimerkki
Liite 2. Makroharjoituslevy
Liite 3. Plaanausmakro zigzag-menetelmällä

TAULUKOT

Taulukko 1. Fanuc koneistuskeskuksissa käytettävät osoitteet. 9
Taulukko 2. Kaavio esittää miten ohjelmat etenevät. 11
Taulukko 3. Muuttujan käyttö esimerkki. 17
Taulukko 4. Esimerkkejä muuttujien käytöstä osoitteen yhteydessä. 18
Taulukko 5. Määrittelemätön muuttuja kerto- ja jakolaskussa. 20
Taulukko 6. Määrittelemätön muuttuja ja ehtolauseet. 21
Taulukko 7: Työkalukorjainten vastaavuudet. 25
Taulukko 8. Muuttujat #3001 ja #3002. 27
Taulukko 9. Muuttujan #3003 arvot ja niiden vastaavuudet. 28
Taulukko 10. Muuttujan #3004 arvot ja niiden vastaavuudet. 29
Taulukko 11. Muuttujaa #3007 vastaavat akselit ja alla esim. mukaiset bitit. 31
Taulukko 12. Järjestelmänmuuttujat modaalisille toiminnoille. 32
Taulukko 13. Järjestelmämuuttujat paikoitustietoihin. 33
Taulukko 13. Aritmeettiset ja loogiset operaatiot. 34
Taulukko 14. Makro-ohjelmoinnissa käytettävät ehto käskyt. 36
Taulukko 15. Argumenttien määrittely menetelmällä 1 40
Taulukko 16. Menetelmän 2 argumentit ja niitä vastaavat osoitteet 41
Taulukko 17. G- ja M-koodeja vastaavat parametrit. 42
Taulukko 18. Taulukkolaskennalla tutkittu Z-akselin saavuttamaa mittaa 44

KUVAT

Kuva 1. Modeemin suojakotelo. 10
Kuva 2. Kiilauramakron käyttöohjekuva. 26
Kuva 3. Silmukassa sallitut ketjutukset ja hypyt. 38
Kuva 4. Reikäpiirimakron ohje. 40
Kuva 5. Terän paikoitus ja siirrot plaanauksessa. 45
Kuva 6. Ohjekuva helical-interpolaatio makrolle. 46
Kuva 7. Osatuoteperhe ohjaustapeista ja niiden mitat. 47
Kuva 8. Ohjaustapin ohjelmointiin tarvittavat pisteet. 48
Kuva 9. Ohjauspinnin makro-ohjelman ohjeet. 50
Kuva 10. Tehtävään 4 tarvittavat mitat ja taulukko. 53
Kuva 11. Makroharjoituslevy. 54

KÄYTETYT LYHENTEET JA SANASTO

ATK = Automaattinen tietojenkäsittely. → Information technology (IT).

Back edit = Tausta ohjelman muokkaus. Tila jossa ohjelmaa voidaan muokata tai se voidaan tallentaa koneen käydessä. → Background Editing.

CAD = Tietokoneavusteinen suunnittelu. → Computer Aided Design.

CAM = Tietokoneavusteinen valmistus. → Computer-aided manufacturing.

CNC = Tietokoneistettu numeerinen ohjaus. → Computerized Numerical Control.

FMC = Joustava valmistussolu. → Flexible manufacturing cell.

FMU = Joustava valmistusyksikkö. Flexible manufacturing unit.

FMS = Joustava tuotantojärjestelmä →Flexible Manufacturing System.

MDI =.Manuaalinen sisään syöttö NC ohjauksen muistiin. → Manual data input.

MS-DOS = Microsoftin tekstipohjaisella komentoliittymällä varustettu käyttöjärjestelmä. → Microsoft Disk Operating System.

NC = Numeerisesti ohjattu. → Numerical Control.

Parametri = Tarkoittaa tietotekniikassa ohjelmalle, käynnistyksen yhteydessä välitettäviä tietoja tai ohjelmoinnissa funktiolle välitettäviä tietoja.

PC = Henkilökohtainen pientietokone. → Personal Computer.

1 Johdanto

Lastuamistekniikaltaan ei ole suurtakaan eroa siinä, käytetäänkö manuaalista vai CNC-ohjattua konetta. Molemmissa käytetään samanlaisia työkaluja, eikä työstöarvoissakaan ole huomattavia eroavaisuuksia. Suurin ero muodostuu niiden ohjaustavassa. CNC-konetta ohjaa tietokone ja manuaalista työkonetta vastaavasti ihminen. [2, s. 249]

CNC-koneen käyttäjältä ei vaadita niinkään käden taitoja, vaan enemmänkin koneen ohjelmoinnin osaamista. Oikeiden työstömenetelmien ja -arvojen tuntemus ovat toki ensisijaisen tärkeitä myös automaattisen koneen käytössä. Koneen ohjaus suoritetaan tietokoneohjelman avulla. Ohjelma muodostuu numeroiden ja kirjainten yhdistelmistä eli koodeista. Näillä koodeilla ohjataan koneen sähkömoottoreita, jotka pyörittävät johderuuveja. Jyrsin koneissa johderuuvit liikuttavat mm. koneen työpöytää. Sähkömoottorien avulla hallitaan myös työkalun pyörimisnopeutta. Ohjelman avulla voidaan säädellä erilaisia hydraulisia tai pneumaattisia toimintoja. Tällaisia toimintoja ovat esimerkiksi työkalun vaihdot ja paletin käännöt. CNC-koneen käyttäjän on pystyttävä lukemaan valmiin ohjelman koodia ja tekemään tarpeen vaatiessa siihen tarvittavia muutoksia. Mikäli käyttäjä vielä tuntee koneensa toimintaperiaatteet ja rakenteet, voi hän häiriön yllättäessä tehdä pienempiä huoltotoimenpiteitä. [1, s. 2–3]

CNC-koneissa on käytössä monenlaisia ohjausjärjestelmiä. Joihinkin koneisiin on mahdollista valita haluttu ohjausjärjestelmä useammasta eri vaihtoehdoista. Valinnan perusteena on yleensä ohjelman aikaisempi tuntemus ja tietenkin hinta. Useimpien koneiden ohjelmat muistuttavat lähestulkoon toisiaan eri ohjausjärjestelmistä riippumatta. Käytetään ns. ISO-koodijärjestelmää, jossa monet koodit ovat täysin samoja ohjauksesta riippumatta. [2, s. 262] Tässä tutkielmassa keskitytään pelkästään Fanuc-ohjauksella toimiviin työkoneisiin ja niiden ohjelmointiin muuttuja-aliohjelmia eli makro-ohjelmia hyödyntäen.

2 Fanuc-ohjelmointi

2.1 Yleistä

CNC-ohjelma rakentuu lauseista. Lauseet muodostuvat sanoista. Jokainen sana taas koostuu kirjainosan ja numero-osan yhdistelmästä. Sana on pienin toiminnallinen käsky ja yksi sana voi muodostaa lauseen. Työstökone lukee aina yhden rivin kerrallaan ylhäältä alaspäin. [2 s. 264]

Koneen toiminnan kannalta ei yleensä ole mitään merkitystä sillä, missä järjestyksessä lauseeseen tulevat sanat kirjoitetaan. On kuitenkin hyvä käyttää aina samaa sanajärjestystä, sillä se helpottaa ohjelman lukemista. Yksi yleisesti käytetty tapa on kirjoittaa G-koodit suurimmasta pienimpään. Tämän jälkeen kirjataan koordinaatit, mikäli niitä tulee, aakkosjärjestykseen. Seuraavaksi tulevat apuosoitteet sekä tarvittaessa kierros- ja syöttönopeus. Jos lauseeseen tulee M-koodi, se sijoitetaan yleensä lauseen loppuun. Ohjelmia kirjoitettaessa on syytä ottaa huomioon että useimmat Fanuc-versiot hyväksyvät vain yhden M-koodin per lause. Taulukossa 1 on esitettynä yleisimmät työstökoneissa käytetyistä osoitteista sekä esimerkki niiden käyttötavoista.

Seuraavassa esitetään lyhyt esimerkki ohjelman sanajärjestyksestä:

O2002 (OM100 MALLI);	Ohjelman numero (piirustuksen nro ja kappaleen nimi)
T1 M6 (32.MM OTSAJYRSIN) ;	Työkalun 1 vaihtokäsky (työkalun nimi)
G90 G54 G00 X-70. Y0 S1600 M3;	Paikoitus haluttuun paikkaan ja kara pyörimään
G43 H1 Z5. M8;	Pituuskompensointi ja leikkuu nesteet päälle
G01 Z-6. F200;	Lineaarinen liike haluttuun syvyyteen sopivalla syötöllä
G41 D1 X-50.;	Sädekompensointi vasemmalle, arvo luetaan paikasta 1
Y20. F300;	Työstö
G3 X-20. Y50. R30.;	Työstö
G40 G01 Y30.;	Sädekompensoinnin peruutus
G0 Z50. M5;	Pikaliikkeellä turvaetäisyydelle kpl:n pinnasta
G91 G28 Z0 M9;	Paluu Z-akselin referenssipisteeseen
M30 TAI M2	Ohjelman lopetus

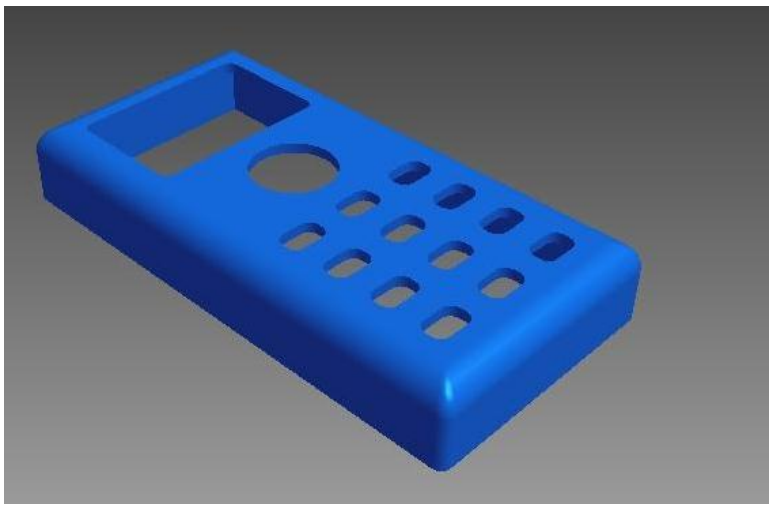
Osoitteet			Esimerkki
O-koodit	ohjelmien numerot		O5123
N-koodit	rivien numerot		N120
M-koodit	aputoiminnot		M3
T-koodit	työkalujen numerot		T12
G-koodit	liikkeet, työkierrot, yms.		G73
S-koodit	kierrosnopeudet		S1250
F-koodit	syöttönopeudet		F425
Apuosoitteet			
R-koodit	ympyränkaaren sädearvot		R10.
D-koodit	työkalun sädearvot		D12
H-koodit	työkalun pituusarvot		H12
L-koodit	toistojen määrät (tai K)		L5
Koordinaatiston osoitteet ja kierrot			
A	B	C	C90.
X	Y	Z	X100. Y50.
I	J	K	I10. J50.
Poraustyökiertojen apuosoitteet			
P	P500	Q	Q15.
viive		lastunkatkaisumatka	

Taulukko 1. Fanuc-koneistuskeskuksissa käytettävät osoitteet.

Fanuc-ohjauksessa on tyypillistä käyttää pistettä mittalukujen yhteydessä. Mittalukuna 10 ilman pistettä, olisi sama kuin 0.010 tai 0.0010, riippuen siitä, onko koneessa tuhannesosan vai kymmenestuhannesosan käyttöjärjestelmä. Koneen parametrien avulla voidaan tosin määritellä, onko desimaalipiste pakollinen vai ei. Fanucin vakio-optiona piste on pakollinen ja kumoavan parametrin kytkentä jälkitoimituksena tulee siinä määrin kalliiksi, että se jätetään yleensä tekemättä.

2.2 Aliohjelmointi

Mikäli työstettävässä kappaleessa toistuu usein sama muoto tai työvaihe, voidaan ohjelmointia helpottaa tekemällä eri vaiheisiin ja muotoihin aliohjelma. Otetaan esimerkiksi kuvan 1 mukainen suojakotelo.



Kuva 1. Modeemin suojakotelo.

Suojakotelossa on 12 samansuuruista ja -muotoista aukkoa. Tehtävänä on koneistaa nuo aukot. Laaditaan pääohjelma, jolla valitaan kaikki tarvittavat työkalut ja niille sopivat työstöarvot. Pääohjelma paikoittaa halutun työkalun ensimmäisen aukon yläpuolelle. Tässä kohtaa pääohjelmassa on komento, jolla hypätään aliohjelmaan. Aliohjelma sisältää näppäinaukon koneistuksen. Kun se on suoritettu loppuun, aliohjelmasta palataan takaisin pääohjelmaan. Pääohjelmassa edetään seuraavan aukon yläpuolelle ja komennolla siirrytään jälleen aliohjelmaan. Pääohjelma jatkaa tätä, kunnes jokainen aukko on koneistettu siinä määrätyllä työkalulla. Mikäli on tarpeen, ohjelmassa valitaan seuraava työkalu, joka on normaalisti edellistä hieman pienempi ja sillä suoritetaan esimerkiksi aukkojen viimeistely. Tässä toimitaan samoin kuin edellä, eli pääohjelmassa paikoitus, aliohjelman kutsu, paluu pääohjelmaan ja uusi paikoitus seuraavan aukon yläpuolelle jne.

Aliohjelmalla pitää olla osoite O, aivan niin kuin pääohjelmallakin. Aliohjelma kutsutaan komennolla M98 P__ __ __ __ L__ __. P-kirjaimen jälkeen annetaan aliohjelman numero ja L-kirjaimella annetaan toistokertojan määrä, mikäli se on tarpeen. Aliohjelmien sisällä voidaan kutsua toista aliohjelmaa ja siitä edelleen kolmatta. Aliohjelmien sisäisiä kutsuja voidaan tehdä kuitenkin enintään viisi. M99-komento lopettaa aliohjelman ja palaa edellisen ohjelman seuraavalle riville. [3 s. 105]

Taulukossa 2 on kuvattu ohjelman etenemistä pääohjelmasta aliohjelmaan ja siitä vielä seuraavaan aliohjelmaan. Ensimmäinen aliohjelma toistetaan kahteen kertaan ja molemmilla kerroilla toinen aliohjelma toistetaan kymmenen kertaa.

Pääohjelma	ensimmäinen aliohjelma	toinen aliohjelma
O1000 N1 M6 T1 (terän nimi) N2 G54 G0 X_ Y_ S_ M_ N3 G43 Z_ H1 M_ N4 M98 P2000 N5 G90 G0 X_ Y_ N6 M98 P2000 N7 G91 G28 Z0 M5 N8 M30	O2000 G1 Z_ F_ G91 Y_ F_ M98 P3000 L10 G1 Y-_ F_ G90 G0 Z_ M99	O3000 G3 J-_ Z-_ F_ M99 EDELLISEN ALIOHJELMAN L-KOODI TOISTAA TÄMÄN OHJELMAN 10 KERTAA JA VASTA SEN JÄLKEEN PALATAAN TAKAISIN ENSIMMÄISEN ALIOHJELMAAN JA L-KOODISTA SEURAAVALLE RIVILLE

Taulukko 2. Kaavio esittää miten ohjelmat etenevät.

3 Makro- vai parametriohjelmointi

Parametrien käytöllä NC-koneiden ohjelmoinnissa tarkoitetaan lähinnä sitä, että mitat ja/tai arvot annetaan muuttujien eli parametrien avulla. Työstöratoja laadittaessa ohjelmoitsija voi käyttää hyväkseen erilaisia muuttujia sekä loogisia että aritmeettisia toimenpiteitä. Ohjelmoinnissa voidaan käyttää hyödyksi myös ehdollisia hyppyjä [4 s. 348]. Kaikkia edellä mainittuja toimenpiteitä voidaan käyttää ohjelmia kirjoitettaessa kahdella eri tavalla. Niitä voidaan kirjoittaa suoraan pää- tai aliohjelmaan, aina kulloinkin tarvittavaan kohtaan. Tämänkaltaista ohjelmointitapaa käytettäessä muuttujien arvonmäärittely tapahtuu yleensä suoraan ohjelman alussa. Tätä ohjelmointitapaa ei pidä sekoittaa varsinaiseen makro-ohjelmointiin, vaan tässä on kyseessä enemminkin parametriohjelmointi.

Aliohjelmat ovat toki hyödyllisiä, mutta ne sisältävät kiinteän ohjelman, joka on aina samanmuotoinen ja suorittaa samat liikkeet. Aliohjelmat ovat usein sidottuja tietynkokoiseen työkaluun ja tietyyn materiaaliin. Varsinaiset makro-ohjelmat, joihin jatkossa syvennytään tarkemmin, ovat eräänlaisia aliohjelmia. Nämä makroaliohjelmat sisältävät kaikki nuo tarvittavat muuttujat ja ehdolliset hypyt sekä kaiken muun

olennaisen tiedon. Jolloin samaa makroa voidaan käyttää, riippumatta siitä, minkäkokoinen terä on käytössä, tai mikä on koneistettavan materiaalin laatu.

Makro-ohjelmat tallennetaan koneelle pysyvästi muistiin. Ne voidaan suojata niin, ettei koneenkäyttäjä pääse edes vahingossakaan poistamaan niitä. Makroja voidaan tarvittaessa kutsua mistä tahansa ohjelmasta. Makroja kutsuttaessa niille voidaan joka kerta määritellä uudet parametriarvot, jotka siten siirtyvät haluttuun ohjelmaan ja muuttavat sen toimintaa tarpeen mukaan. Makrojen kutsumisesta ja muuttuja-arvojen siirtämisestä on kerrottu tarkemmin kappaleessa 11.

Seuraavassa esimerkissä on kaksi erilaista plaanausohjelmointia. Vasemmanpuoleinen ohjelma on muuttujaohjelmoinnilla tehtyä ohjelmaa ja oikeanpuoleinen on vastaavasti tehty makro-ohjelmalla. Molemmat ohjelmat ovat hieman lyhennettyjä versioita, eivätkä siten ole sellaisenaan suoraan käytettävissä koneistukseen.

Vasemmanpuoleisessa on vain yksi ohjelma, joka siis sisältää kaikki tarvittavat muuttujat ja niiden määrittelyn (vertaa kohtia #1 - #9). Oikeanpuoleinen taas sisältää sekä pääohjelman että makro-ohjelman. Pääohjelmasta vain kutsutaan haluttua makroa ja samalla siirretään tarvittavat muuttujat. Todellisuudessa makroja ei joka kerta tarvitse kirjoittaa pääohjelman kanssa, vaan niin kuin aikaisemmin jo todettiin, ne ovat jo koneen muistissa. Tosin, vain siinä tapauksessa, että joku on ne sinne edeltäkäsin taltioinut.

3.1 Ohjelmointiesimerkki muuttujaohjelmasta ja makro-ohjelmasta.

O1007 (MUUTTUJAOHJELMA)	**O1234 (MAKRO-OHJELMAN PÄÄOHJELMA)**
M24	**G90 G80 G49 G40 G17**
0G90G49G40G80	**M6 T1 (63MM PLAANAUSTERA)**
G10L2P1X0Y0Z0	**G90 G54 G0 X0 Y0 S3500 M3**
G10L2P2X0Y0Z0	**G43 Z20. H1 M8**
G10L2P3X0Y0Z0	**G65 P8001 D63. I142. J100. Z0 S70 T5 F1000**
G10L2P4X0Y0Z0	**G90 G0 Z50. M9**
G54	**G91 G28 Z0 M5**
T209M6	**G28 Y0**
G49H209	**M30**
S600M3	
M7	**O8001 (PLAANAUS MAKRO)**
#1=-215(X-ALKU)	**#123=0**
#2=692(Y-ALKU)	**#125=0**
#3=153.5(Z-ALKU)	**#126=0**
#4=215(X-LOPPU)	**#101=#5001 (X-AKSELIN PAIKKA TALTEEN)**
#5=142(Y-LOPPU)	**#102=#5002 (Y-AKSELIN PAIKKA TALTEEN)**
#6=153(Z-LOPPU)	**#103=#5003 (Z-AKSELIN PAIKKA TALTEEN)**
#7=-50(Y-ASKEL)	**#104=#7/2**
#8=-0.5(Z-ASKEL)	**#105=[#19/100]*#7**
#9=0(B-AKSELI)	**#106=[#20+#104]**
WHILE[#3GE#6]DO1	**#107=[#105-#104]**
#11=#3+10(R-TASO)	**#110=[#26+#6]**
#9=0(B-AKSELI)	**IF[#6LE0]GOTO30**
WHILE[#9LT271]DO2	**IF[#17LE0]GOTO30**
G0B#9	**#120=[FIX[#6/#17]]**
#12=#9+90	**IF [#120EQ0] GOTO 30**
#12=#12/90	**#121=[#120*#17]**
#12=#12+53	**#122=[#6-#121]**
G#12	**#123=[#120+#122]**
#10=#2	**N20 #123=#123-1**
WHILE[#10GE#5]DO3	**IF [#123LE0] GOTO 30**
G0X#1Y#10	**GOTO 40**
G0G43Z#11	**N30 #125=#26**
G0Z#3	**GOTO 50**

4 Makro-ohjelmoinnin etuja

Kuten edellä jo mainittiin, makrot toimivat erilaisia parametrejä hyödyntäen. Tästä on huomattavaa apua etenkin silloin, kun yrityksessä valmistetaan ns. osatuoteperheitä, joissa vain osa kappaleen mitoista muuttuu ja perusmuodot pysyvät samoina. Muita yleisiä makro-ohjelmoinnin kohteita ovat taskujen ja urien rouhinnat, reikien poraukset ympyränkehälle sekä suorakaide verkkoon. Reikien avaus helical-toiminnolla eli kierreinterpolaatiolla helpottuu huomattavasti makro-ohjelmia käytettäessä. Myös tasopintojen rouhintaan ja viimeistelyyn kannattaa hyödyntää muuttujien käyttöä. Ilman muuttujaohjelmointia matemaattisten muotojen, kuten pallonpuolikkaan tai ellipsin koneistus olisi erittäin vaikeaa.

Makro-ohjelmia käytettäessä ohjelmointi nopeutuu ja samalla ohjelmista tulee lyhyempiä ja niiden selkeys paranee. Myös ohjelmien muunneltavuus helpottuu. Kun käytetään ennalta hyviksi havaittuja makro-ohjelmia, pääohjelmista tulee varmemmin toimivia. Makro-ohjelmien käyttö motivoi koneenkäyttäjiä, koska toisinaan yksitoikkoiselta tuntuva ohjelmointityö vähenee. Samalla koneiden käyttöaste kasvaa.

Perinteiseen aliohjelmointiin verrattuna makro-ohjelman läpivieminen työstökoneella vie vähemmän aikaa. Tämä korostuu etenkin silloin, kun aliohjelmalla tehdään useita toistoja peräkkäin. Otetaan esimerkiksi tilanne, jossa tarkoituksena on koneistaa suurella nopeudella, mutta pienellä lastunpaksuudella, suorakaiteen muotoinen tasku. Lopullinen syvyys on 15,0 mm ja lastunpaksuus 0,15 mm. Aliohjelmointitekniikalla tämä tarkoittaisi sitä, että taskun ajoon laadittu aliohjelma kutsutaan 100 kertaa. Jokaisen aliohjelman kutsun kohdalla ohjelman luku pysähtyy noin puolen sekunnin ajaksi. Makro-ohjelmoinnissa vastaavaa ongelmaa ei synny, mikäli käytetään silmukka-toimintaa. Esimerkki tapauksessa koneistusaika pitenee siis 50 s pelkästään aliohjelman luvun takia. Luvussa 9.2 on kerrottu tarkemmin silmukkatoiminnon käytöstä.

FMS-ympäristössä on CNC-ohjelmien hallittavuutta voitu parantaa käyttämällä muuttujaohjelmointia. Paletin ohjelma on rakennuttu siten, että pääohjelma vaihtaa työkalua ja kutsuu muuttujia käyttäviä kappaleohjelmia. Kappaleohjelmasta ajetaan kutsuttaessa läpi vain karalla olevan työkalun ohjelman osa. Muuttujilla valitaan koneistettavat rivit. Näin palettia varten tarvitaan huomattavasti vähemmän ohjelmia kuin perinteistä aliohjelmatekniikkaa käyttämällä ja tällöin ohjelmat mahtuvat paremmin koneen muistiin. Muuttujien avulla voidaan lisäksi valita, montako kappaletta kutakin

osaa koneistetaan. Tämä taas auttaa siinä, että ei tarvita eri ohjelmavariaatiota, vaan muuttujilla hoidetaan haluttu osien määrä. [6]

5 Makro-ohjelmointi vs. CAM-ohjelmointi

Mikroprosessorit lähtivät 70-luvun alkupuolella huimaan kehitykseen. Samoihin aikoihin alkoi myös Windowsin kehitys. 1981 ilmestyi markkinoille ensimmäinen PC, jossa oli MS-DOS käyttöjärjestelmä. Se koettiin kuitenkin turhan hankalaksi käyttää ja 1990 julkaistu Windows 3.0 versiosta tuli ensimmäinen todellinen menestys. Lopullisen läpimurron Microsoft teki 1995, kun se julkaisi Windows 95 -käyttöjärjestelmänsä. [8 Microsoft Oy (viitattu 12.5.2011)]

Kotitietokoneiden synnystä on siis vierähtänyt jo 30 vuotta ja tuona aikana niiden hinnat ovat laskeneet roimasti. Samaan aikaan niiden suoritusnopeus ja muistin määrä ovat kehittyneen valtavasti. Samaan aikaan tietokoneiden käytettävyys on helpottunut merkittävästi, niinpä on ymmärrettävää että tietokoneita on nykyisin lähes joka paikassa, myös konepajoissa.

Kotitietokoneiden kehityksen myötä tietokonepelien tekijät ovat osaltaan edistäneet 3D-grafiikan kehitystä. Tämä on taas auttanut CAM-ohjelmien tekijöitä heidän kehitellessä omiin ohjelmiinsa parempia simulointiminaisuuksia. Nykyään markkinoilla olevat CAM-ohjelmat ovat jonkin verran halventuneet ja niillä suoritetut simuloinnit ovat varsin luotettavia. Koska tietokoneiden hinnat ovat nykyään varsin kohtalaiset ja CAM-ohjelmien hinnat ovat myös laskeneet jonkin verran, on CAM-ohjelmia alkanut ilmaantua lähes joka konepajaan.

CAM-ohjelmien ehdoton valtti on siinä, että niiden avulla voidaan tehdä hyvinkin monimutkaisia koneistushjelmia. CAM-ohjelmalla tehty ohjelma tallentuu tietokoneen muistille ja sieltä se saadaan tarvittaessa siirrettyä CNC-koneelle. CAM-ohjelmien yhtenä haittapuolena on kuitenkin se, että ohjelmista tulee helposti valtavan pitkiä ja siksi myös hankalasti tulkittavia. Jos työstökoneelle siirretyssä ohjelmassa ilmenee jokin virhe, on koneenkäyttäjän lähes mahdotonta korjata sitä. Yleensä näissä tapauksissa helpoin vaihtoehto on korjata virhe CAMillä ja lähettää korjattu ohjelma uudelleen koneelle. Tästä taas aiheutuu turhia katkoja tuotantoon ja siitä koituu vain ylimääräisiä kuluja yritykselle.

Makro-ohjelmoinnissa ei yleensä tule samaa ongelmaa, ainakaan jos käytettävä makro on jo aikaisemmin toimivaksi todettu. Makro-ohjelmoinnissa varsinainen ohjelma on suhteellisen lyhyt ja helposti muunneltavissa. Usein konepajoissa valmistettavat tuotteet muistuttavat pääpiirteiltään toisiaan. Niistä plaanataan tasopinnat puhtaiksi ja porataan erikokoisia reikiä joko reikäpiirille tai suorakaidematriisina. Toisinaan koneistetaan ympyrän tai suorakaiteen muotoisia ratoja, sekä ulko- että sisäpuolisina. Edellä mainittujen työstöratojen laadintaan voidaan hyödyntää vanhoja pääohjelmia etenkin silloin, kun niissä on käytetty makro-ohjelmia hyödyksi. Pääohjelmaan korjataan tarvittaessa uudet terät ja niiden työstöarvot sekä muutetaan makron kutsun yhteydessä tarvittavien argumenttien arvoja (katso luku 11). Näin saadaan nopeasti ja pienillä muutoksilla laadittua uuden kappaleen työstöradat. Vanha pääohjelma on syytä tallentaa tietokoneen muistiin ennen kuin sitä ryhdytään muokkaamaan. Näin varmistetaan että se on käytettävissä myös tulevaisuudessa, mikäli kyseistä kappaletta aiotaan koneistaa joskus myöhemminkin.

Yrityksissä on usein koneistettavana useita materiaaleja ja niiden työstämiseen käytettään eri työkaluja. Muuttujia hyödyntäen voidaan laatia makro-ohjelma, joka laskee koneistukseen parhaiten sopivat työstöarvot. Kyseiseen makroon tallennetaan aikaisemman kokemuksen perusteella parhaaksi todetut arvot. Näin saadaan varmistettua se, että koneistuksessa käytetään aina optimaalisesti parhaita arvoja.

Itse makro-ohjelmat ovat hieman vaikeaselkoisia ja useimmiten niihin ei ole syytä kajota. Mikäli sellainen tilanne kuitenkin syntyy, on parasta antaa kyseisen makro-ohjelman laatijan hoitaa se, jos se suinkin on mahdollista.

CAM-ohjelmien lisääntyminen on johtanut siihen, että makro-ohjelmien käyttö on vähentynyt ja samalla osaamisen taito niiden alalla on hiipunut. Omasta kokemuksesta olen huomannut, että yksinkertaisimmatkin ohjelmat pyritään nykyään tekemään CAM-ohjelmia käyttäen. Olen havainnut, että siinä ajassa, missä CAM-ohjelmalla piirretään pelkästään kuva, siis varsinaista ohjelmointia ei ole vielä edes aloitettu, niin työstökoneella sama ohjelma olisi jo tehnyt makro-ohjelmia käyttäen. Kaikkein tehokkainta ohjelmointi olisi tehdä ”back edit”-tilassa, samalla kun koneenkäyttäjä vielä valvoo edellisen työn valmistumista.

Visa Koponen kartoitti omassa opinnäytetyössään ”Oppimisen hallinta konepajoissa”, kuuden varsinaissuomalaisen konepajan osaamis- ja koulutustarvetta. Tämän tutkimuksen perusteella, jotkut koneistajat pitivät CAD/CAM-ohjelmia hankalina. Usein

CAD/CAM-ohjelmien käyttö olikin melko vähäistä. Samasta tutkimuksesta käy ilmi, että joissakin yrityksissä, etenkin koneen käyttäjät pitivät erillisen ohjelmoitsijan palkkaamista huonona vaihtoehtona. Esimiestasolla tuli usein esiin CAD/CAM-osaamisen laajentaminen, kun taas työntekijäpuolella kaivattiin enemmän ohjelmointitaitojen kehittämistä. Kukaan Koposen haastattelemista työntekijöistä ei osannut itse laatia Fanucin makro-ohjelmia, vaikka useimmissa paikoissa niitä hyödynnettiin. [7 s. 27–28, 44, 48,]

6 Muuttujien käyttö

Perusohjelmoinnissa haluttu liikekäsky suoritetaan aina antamalla sitä vastaava G-koodi. Esim. suoraviivainen syöttöliike saadaan aikaan G1-koodilla, jonka lisäksi tarvitaan vielä määränpää, olkoon se vaikka koordinaattipaikka X10., näin muodostuu lause G1 X10. Makro-ohjelmassa numeroarvot voidaan määritellä suoraan tai ne voidaan korvata muuttujilla. Luvussa 2.1 mainittiin, että mittalukujen yhteydessä tulee käyttää pistettä. Makro-ohjelmassa numeroarvot eivät aina vaadi pistettä, mutta mittalukuja annettaessa kannattaa käyttää aina pistettä. HUOM! Tässä on konekohtaisia eroja.

Makro-ohjelmassa voi olla useita muuttujia, jotka erotetaan toisistaan muuttujanumerolla. Muuttujat esitetään aina merkillä # ja sen perään tulee muuttujan numero. Taulukossa 3 on esitetty edellä mainittu liikekäsky makro-ohjelman avulla. [4 s. 349]

#1=10	Muuttujan yksi, arvoksi on annettu 10
G1 X#1	ohjelma siirtyy liikekäskyllä X-koordinaattiin 10.0, joka saatiin edelliseltä riviltä
#1 = 20	G1 X#1 sama kuin G1 X20.
# 1 =30	G1 X#1 sama kuin G1 X30.

Taulukko 3. Muuttujan käyttöesimerkki.

Muuttuja voi sisältää ilmaisun, jolloin se merkitään seuraavasti: #[<ilmaisu>], esim. #[#100]. Osoitetta seuraava arvo voidaan korvata muuttujalla, silloin se merkitään <osoite>#i tai <osoite>-#i. Tämä tarkoittaa sitä, että muuttujan arvo tai sen vastaluku

korvaa osoitteen ilmaiseman käskyarvon. Taulukossa 4 on esiteltynä kolme erilaista esimerkkiä osoitteen ja muuttujan yhteiskäytöstä.

Jos #33=1.5	merkitsee käsky F#33 samaa kuin F1.5
Jos #18=20	merkitsee käsky G0 Z-#18 samaa kuin G0 Z-20.0
Jos #130=3	merkitsee käsky G#130 samaa kuin G3

Taulukko 4. Esimerkkejä muuttujien käytöstä osoitteen yhteydessä.

Poraustyökierto G81 on koneen mukana tuleva makro-ohjelma. G81 Z__ R__ F__, jossa Z = haluttu poraussyvyys, R = lähestymisetäisyys ja F = syöttö mm/min. Seuraavan esimerkin avulla tarkastellaan samaa toimintaa muuttujien avulla.

Tarkoituksena on porata alkureikäporalla 3 mm syvä reikä. Lähtö- ja paluutaso on kahden millimetrin etäisyydellä kappaleen pinnasta. Z-akselin nollataso on aihion yläpinnassa.

Tehdään ensin seuraavanlainen ohjelma;

O1001	Ohjelman numero
N10 M6 T1 (ALKUREIKÄPORA)	Oikean työkalun valinta.
N20 G0 X50. Y50.	Paikoitetaan terä oikeaan kohtaan.
N30 G43 Z50. H1	Pituuskompensointi päälle .
N40 S1000 M3	Kara pyörimään myötäpäivään 1000 $^{r}/_{min}$.
N50 G0 Z2.	Pikaliikkeellä porauksen lähtötasolle, = R taso.
N60 G1 Z-3. F100	Suoritetaan poraus syötöllä 100 $^{mm}/_{min}$.
N70 G0 Z2.	Paluu porauksen lähtötasolle.
N80 G28 Z50 M5	Paluu Z-akselin referenssipisteeseen.
M30	Ohjelman lopetus

Seuraavaksi tehdään sama muuttujia hyödyntäen. Määritellään heti ohjelman alussa tarvittavat muuttujat.

O1002	Ohjelman numero
N1 #18=2	Porauksen lähtötaso, eli R arvo.
N2 #26=-3	Poraussyvyys, eli Z.
N3 #9=100	Syöttö, eli F.
N10 –N40	Samat kuin edellisessä.
N50 G0 Z#18	Muuttamalla alussa olevaa #18 arvoa myös lähtötason arvo muuttuu.
N60 G1 Z#26 F#9	Muuttamalla alussa olevien #26 ja #9 arvoja, muuttuu myös poraussyvyys ja –syöttö.
N70 G0 Z#18.	Paluu porauksen lähtötasolle.
N80 G28 Z50 M5	Paluu Z-akselin referenssipisteeseen.
M30	Ohjelman lopetus

7 Muuttujien ryhmittely

Muuttujatyypit voidaan luokitella kolmeen eri ryhmään, paikallis-, yleis- ja järjestelmämuuttujiin. Näiden lisäksi on vielä ns. nollamuuttujat. Kaikilla muuttujilla on omat ominaispiirteensä ja käyttötarkoituksensa.

7.1 Määrittelemättömät muuttujat

Mikäli muuttujan arvoa ei ole määritelty kone tulkitsee että sen arvo on <vapaa>. Se ei siis ole sama kuin arvo nolla, vaan se on ”tyhjä”. Muuttujat #0 ja #3100 ovat nollamuuttujia, eli ne ovat aina tyhjiä. Niitä ei voi muuttaa, mutta ne voidaan lukea. [4 s. 352]

Kun määrittelemättömään muuttujaan viitataan, jätetään osoitekin huomioimatta. Esimerkki:

Makrossa oleva lause on seuraavanlainen → G0 X#1Y#2

Muuttuja #1 saa arvon 100, eli #1 = 100 ja muuttuja #2 arvo jää määrittelemättä, eli se on vapaa. Tällöin koodattu lause toteutuu muodossa G0 X100.

Muutetaan edellistä sen verran että muuttuja #2 saa arvon 0, eli #2 = 0. Silloin sama lause toteutuu muodossa G0 X100 Y0.

Seuraavassa taulukossa on esitetty miten määrittelemätön muuttuja toimii silloin kun sillä korvataan jokin toinen muuttuja tai sitä käytetään kerto- tai jakolaskussa

Kun #1 = <vapaa>	Kun #1 = 0
#2 = #1 ▼ #2 = <vapaa>	#2 = #1 ▼ #2 = 0
#2 = #1 * 5(= tyhjä x 5, saa arvon 0) ▼ #2 = 0	#2 = #1 * 5(= 0 x 5 = 0) ▼ #2 = 0
#2 = #1 + #1 (tyhjä + tyhjä, saa arvon 0) ▼ #2 = 0	#2 = #1 + #1 (0 + 0 = 0) ▼ #2 = 0

Taulukko 5. Määrittelemätön muuttuja kerto- ja jakolaskussa. [5 s. 373]

Ehdollisessa ilmaisussa <vapaa> eroaa nollasta vain tapauksissa EQ ja NE. Muissa ehdollisissa ilmaisuissa, GE, GT, LE ja LT määrittelemätön muuttuja saa aina arvon 0. Ehdollisista ilmaisuista on tarkempaa tietoa kappaleesta 9.1 Hypyt ja ehdot.

Taulukossa 6 tarkastellaan, miten tilannetta muuttuu ehdollisissa ilmaisuissa, jos #1 on määrittelemätön tai se saa arvon 0.

Kun #1 = <vapaa>	Kun #1 = 0
#1 EQ 0 ▼ Ei toteudu (epätosi)	#1 EQ 0 ▼ Toteutuu (tosi)
#1 NE 0 ▼ Toteutuu (tosi)	#1 NE 0 ▼ Ei toteudu (epätosi)
#1 GE 0 ▼ Toteutuu (tosi)	#1 GE 0 ▼ Toteutuu (tosi)
#1 GT 0 ▼ Ei toteudu (epätosi)	#1 GT 0 ▼ Ei toteudu (epätosi)
#1 LE 0 ▼ Toteutuu (tosi)	#1 LE 0 ▼ Toteutuu (tosi)
#1 LT 0 ▼ Ei toteudu (epätosi)	#1 LT 0 ▼ Ei toteudu (epätosi)

Taulukko 6. Ehtolauseen muutos, jos muuttuja on 0 tai vapaa. [4 s. 352-353]

7.2 Paikallismuuttujat (#1 - #33)

Paikallismuuttujia voidaan käyttää suoraan laskutoimituksissa, mutta yleensä niitä käytetään argumenttien asettamiseen. Argumenttien ja osoitteiden vastaavuuksista kerrotaan myöhemmin makro-ohjelmien kutsumisen yhteydessä luvussa 11. Jos paikallismuuttujalle ei ole asetettu argumentteja, sen alkutila on <vapaa> ja se on vapaasti käytettävissä.

Paikallismuuttujat ovat nimensäkin mukaan sellaisia, joita käytetään vain paikallisesti. Tämä tarkoittaa sitä, että ensimmäisen makron käyttämä paikallismuuttuja on eri kuin seuraavan makron käyttämä sama muuttuja. Jos esimerkiksi makro B kutsutaan makrosta A, niin makrossa B määritelty paikallismuuttuja ei pysty turmelemaan makro A:n käyttämää samaa paikallismuuttujaa. [4 s. 350] (Esimerkki ohjelma sivulla 23.)

7.3 Yleismuuttujat (#100 - #149 ja #500 - #599)

Yleismuuttujat ovat aktiivisia läpi koko pääohjelman, pääohjelman kutsumissa aliohjelmissa, sekä näiden kutsumissa makroissa. Tämä tarkoittaa sitä että makron käyttämä muuttuja #i on sama kuin toisen makron käyttämä vastaava muuttuja. Näin ollen makrossa B voidaan käyttää jo aikaisemmin makrossa A laskettua yleismuuttujan arvoa. [4 s. 350]

Yleismuuttujien käyttöä ei ole määritelty järjestelmässä, vaan käyttäjä voi määritellä sen vapaasti. Pääsääntöisesti yleismuuttujille annetaan jokin lukuarvo joita sitten käytetään erilaisissa laskutoimituksissa. Muuttujat #100 - #149 ovat sellaisia, jotka nollaantuvat silloin kun koneesta katkaistaan virrat. Muuttujien #500 - #599 arvot ovat taas sellaisia, jotka pysyvät koneen muistissa vaikka virrat katkaistaan. [4 s. 350]

Seuraavalla sivulla on esimerkki ohjelma, jossa koneistetaan kaksi ympyränmuotoista taskua helical-toiminnolla. Mallissa on pääohjelma josta kutsutaan modaalisesti makroa A, joka taas kutsuu makroa B.

HOUM! Ohjelma on täysin toimiva ja voidaan käyttää sellaisenaan, mutta se ei suinkaan ole paras mahdollinen kyseiseen työskentelyyn. Viimeistään tämän kurssin loppupuolella osaat jo itsekin tehdä huomattavasti paremman version.

Esimerkki ohjelma kierreinterpolaatiolla tehdystä ympyrätaskusta.

O1000 (PIIR. NRO JA NIMI)	Tehdään kaksi pyöreän muotoista taskua, joiden Ø on 20 mm ja syvyys on pinnasta -10 mm.
M6 T1 (10. MM TAPPIJYRSIN)	Tarvittavan työkalun vaihto
G54 G0 X50. Y50. S1000 M3	Paikoitus koneistettavan reiän keskelle
G43 Z20. H1 M8	Pituuskompensointi päälle ja siirto turvaetäisyydelle
G66 P8101 D10. H20. R1. F100 Q1. T11	Makro A:n kutsu. D = terän Ø, H = haluttu Ø, R = aloitus korkeus, Q = Z-askel ja T = Z-askelten toistojen määrä
G90 G0 X100. Y100.	Paikoitus seuraavan reiän keskelle
G67	Modaalisen kutsun peruutus
G90 G0 Z50. M9	Pikaliikkeellä turvaetäisyydelle kpl:n pinnasta
G91 G28 Z0 M5	Paluu Z-akselin referenssipisteeseen
M30	Ohjelman lopetus
O8101 MAKRO A	Sama nro kuin makron kutsussa on P:n jälkeen
#1=#5003	Muuttuja 1 saa Z-akselin sen hetkisen arvon
G0 Z#18	Pikaliikkeellä R:llä annettuun etäisyyteen
#109=#9*2	Muuttuja 109 saa 2 x F:n arvon, eli on nyt 200
#100=[#11-#7]/2	Muuttuja 100 = $\frac{Haluttu\ halkaisija - terän\ halkaisija}{2}$ eli on nyt 5
#120=#20	Muuttuja 120 saa T:llä annetun arvon, eli on nyt 11
G91 G1 Y#100 F#109	Siirrytään inkrementaalisesti Y5. F200
G65 P8102 Q#17 F#9	Makro B:n kutsu joka vie samalla aiemmin määritellyn Q:n arvon
G3 J-#100	Kun aliohjelma on toistettu haluttu määrä, tasataan vielä taskun pohja (G3 J-5.)
G1 Y-#100 F#109	Siirrytään reiän keskelle (G1 Y-5. F200)
G90 G0 Z#1.	Palataan lähtötasolle (G90 G0 Z20.)
M99	palataan pääohjelmaan
O8102 (MAKRO B)	
WHILE[#120GT0]DO1	Niin kauan kuin muuttujan 120 arvo suurempi kuin 0 toistetaan rivit ennen END1 riviä
G3 J-#100 Z-#17 F#9	Pyöräytetään täysi ympyrä muuttuja 100:n arvolla, samalla mennään Q:n arvon verran alaspäin (G3 J-5. Z-1. F100)
120=#120-1	Laskuri, joka pienenee joka kierroksella yhdellä,
END1	kun muuttuja 120 saa arvon 0 siirrytään tähän
M99	Palataan makro A:han

7.4 Järjestelmämuuttujat

Järjestelmämuuttujien määrä on hyvin runsas ja vaihtelee jonkin verran kone- ja tyyppikohtaisesti. Järjestelmämuuttujia käytetään erilaisten järjestelmätietojen lukemiseen ja kirjoittamiseen. Niillä voidaan esimerkiksi lukea työkalujen kompensaatioarvoja ja hetkellinen koordinaattiasema. Järjestelmämuuttujat ovat aina järjestelmään sidottuja muuttujia ja siksi niitä käytettäessä onkin aina tiedettävä, mitä muuttujaa käytetään ja mihin se vaikuttaa. Väärään muuttujaan syötetty arvo voi pahimmassa tapauksessa sekoittaa jonkin koneen vakiotoiminnon. Työstökoneiden kaikista parametriarvoista onkin hyvä ottaa aika-ajoin varmuuskopiot.

Seuraavaksi tarkastellaan hieman lähemmin muutamia yleisemmin käytettyjä järjestelmämuuttujia. Saman järjestelmämuuttujan numero voi jonkin verran vaihdella konekohtaisesti, joten ainakin uudemmissa koneissa on syytä tarkistaa käytettävän järjestelmämuuttujan arvo koneen käyttöohjeista. Useimmat muuttujista ovat tosin vakioita koneesta riippumatta, mutta mm. työkalujen kompensointiarvojen rekisteröintiin on useampia eri vaihtoehtoja.

7.4.1 Liitäntäsignaali

CNC-kone ja siihen liitetty aputoimilaite, kuten esim. lastunkuljetin, tarvitsevat keskinäiseen tiedonsiirtoon liitäntäsignaaleja. Sisäänmenosignaalit on määritelty järjestelmämuuttujilla #1000 - #1032 ja ulostulosignaalit on välillä #1100 - #1132. Näiden muuttujien arvo on 0 tai 1. Jos muuttujan arvo määrittelemätön, tai se on pienempi kuin 0.00000001, on se silloin <vapaa> ja luetaan arvoksi 0. Muissa tapauksissa muuttuja saa arvon 1. Kun sisäänmenosignaalin muuttujan arvo on 1, niin kontakti on silloin kiinni. Jos taas kontakti halutaan avata, annetaan sille arvoksi 0.

7.4.2 Työkalun kompensointiarvo

Työkalukorjaimien vastaavuuksissa on jonkin verran konekohtaisia eroja, riippuen lähinnä siitä kuinka monta työkalupaikkaa kyseisessä koneessa on. Periaatteeltaan kaikki kuitenkin muistuttavat toisiaan ja alkavat muuttuja-arvosta #2001 ja jatkuvuus on esitetty seuraavassa taulukossa:

Järjestelmämuuttuja	Työkalukorjaimen numero
#2001	**1**
#2002	**2**
#2003	**3**
▼	▼
#2099	**99**

Taulukko 7: Työkalukorjainten vastaavuudet.

Tätä toimintoa voidaan hyödyntää kun halutaan lukea jokin ennalta koneen tiedostoon syötetty korjainarvoja. Offset tiedostoon voidaan lisätä arvoja ja siellä olevia arvoja voidaan muuttaa.

Esimerkki:

Työkalupaikassa 5 on 16 mm tappijyrsin ja sen sädekorjain arvo, joka oletuksena on 8.0, on sijoitettu korjainpaikkaan 25.

#110 = #2025 Muuttuja #110 saa arvon 8.0

Tai vastaavasti:

#2025 =#8 Jos #8 on saanut aikaisemmin arvon 10.0, muuttuu korjainpaikkaan 25 arvoksi 10.0.

7.4.3 Makrohälytys

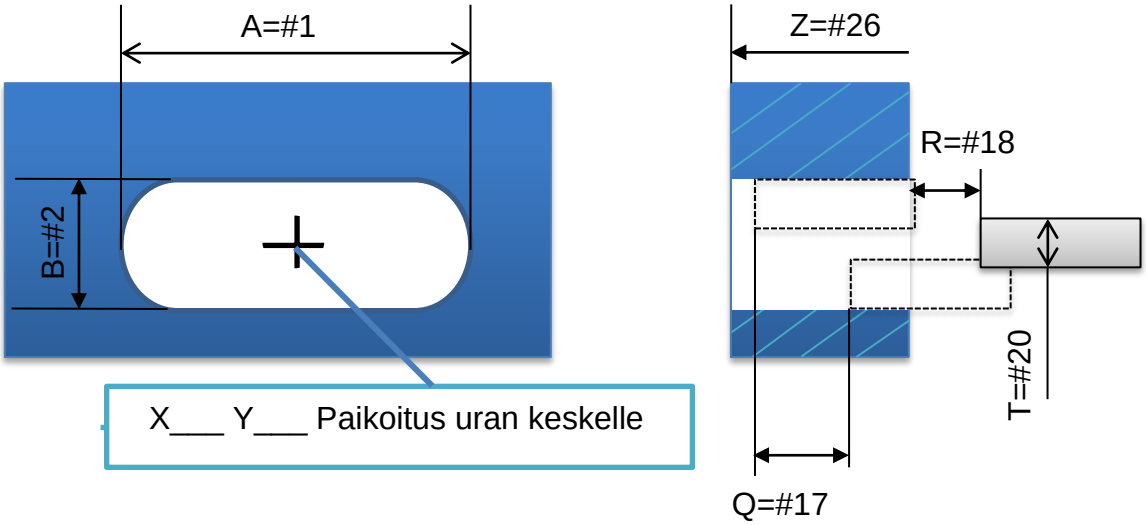
Kun makro on hyvin suunniteltu ja tehty, se tarkistaa jo heti ohjelman alussa mahdolliset virheet pois. Jos on esimerkiksi tarkoitus koneistaa kiilauraa, niin ohjelman tulee antaa virheilmoitus, mikäli siinä yritetään käyttää haluttua uraa suurempaa terää. Tähän toimintaan käytetään järjestelmämuuttujaa #3000.

Hälytysnumero määritellään muuttujassa #3000, välillä 0-200 ja samalla siihen voidaan lisätä hälytysviesti. Hälytysviestin pituus saa olla enintään 26 merkkiä ja se kirjoitetaan kaarisulkujen sisään. Jos makrossa ilmenee virhe, hälytysvalo syttyy ja näytölle tulee ohjelmassa kirjoitettu viesti.

Makron laatijan tulee itse miettiä ja suunnitella kaikki mahdolliset virheet, joita koneenkäyttäjä saattaisi makroa käyttäessään tehdä.

Otetaan esimerkiksi aikaisemmin mainittu kiilaura. Kiilauran leveys on annettu muuttujalla #2 (B) ja sen koneistamiseen käytettävä terä annetaan muuttujalla #20 (T).

MAKRO-OHJELMAN KUTSUMUOTO:
G65 P8200 A___ B___ T___ R___ Z___ Q___



Kuva 2. Kiilauramakron käyttöohjekuva.

O8200 (KIILAURA MAKRO)

IF[#20 GT #2] GOTO100 — Makro suorittaa vertailun ja mikäli käytettävä terä on suurempi kuin kiilauran leveys, niin hypätään riville N100. Silloin ohjelma pysähtyy ja näytölle ilmestyy muuttujalla #3000 annettu hälytysnumero sekä viesti.

(TÄSSÄ VÄLISSÄ VARSINAINEN OHJELMA)

N100 #3000=10 (TERA ON SUUREMPI KUIN URA)
M99

7.4.4 Aikamuuttujat

Järjestelmämuuttujien #3001 ja #3002 avulla voidaan lukea esimerkiksi koneistukseen käytettyä aikaa tai odotusajan luomiseen. Aika voidaan asettaa muuttamalla kyseisten muuttujien arvoa.

Laji	Muuttuja	Yksikkö	Tila virtaa kytkettäessä	Laskee
Kello 1	#3001	1 millisekunti	Asettuu 0:aan	Aina
Kello 2	#3002	1 tunti	Sama kuin virta katkaistaessa	Kun STL-signaali on päällä

Taulukko 8. Muuttujat #3001 ja #3002.

Molempien kellojen tarkkuus on parempi kuin 16 millisekuntia. Kello 1 nollautuu 65536 ms välein. Kello 2 nollautuu itsestään 9544 tunnin välein, mikäli sitä ei ole nollattu aikaisemmin.

Esimerkki: ajastin (vastaa G4 koodia).

G65 P8300 T1000	Makron kutsukäsky ja odotusaika millisekunteina. T = #20.
O8300	Makro-ohjelman numero
#3001 = 0	Alkuarvon nollaus.
WHILE [#3001 LE #20] DO1	Odottaa muuttujalla #20 annetun ajan 1000 ms = 1 s
END1	Kun ehto ei enää toteudu, siirrytään tähän lauseeseen
M99	Paluu edelliseen ohjelmaan

Järjestelmämuuttujalla #3011 voidaan koneeseen asentaa uudelleen vuosi, kuukausi ja päivämäärä. Muuttujan #3012 avulla voidaan vastaavasti kirjata tunnit, minuutit ja sekunnit. Muutosta tehtäessä pitää olla ajastimen näyttösivu käytössä.

Kesäkuun 1. päivänä vuonna 2011 kello 15,50,05 muutetaan seuraavasti

#3011 = 20110601
#3012 = 155005

7.4.5 Ohjelmanajon hallittavuus

Koneenkäyttäjä voi tarvittaessa ajaa ohjelmaa läpi lauseittain, eli silloin ”single block”-toiminto on kytkettynä päälle. Kesken ohjelman voidaan säädellä prosentuaalisesti sekä terän kierroslukua että syöttönopeutta. Joskus koneistuksessa tulee kuitenkin vastaan sellaisia tilanteita, jolloin tästä koituisi vahinkoa.

Otetaan esimerkiksi kierteen koneistus poraustyökiertoa käyttäen. Siinä on ensisijaisen tärkeää että terä kulkee tietyllä nopeudella, eli kierteen nousun verran per kierros. Toinen tärkeä seikka siinä on, ettei koneenkäyttäjä pysty pysäyttämään syöttö kesken porautumisen, silloinhan terä jäisi pyörimään samalle syvyydelle ja näin se pilaisi kierteen. Samasta syystä myös single block toimintoa, eli lauseittain ajoa ei voida käyttää kierrettä valmistettaessa.

Jos Järjestelmämuuttujalle #3003 annetaan jokin seuraavista arvoista, voidaan määritellä, estetäänkö, vai sallitaanko ”single block”-ajo. Samalla määritellään odotetaanko jonkin aputoiminnon, esim. M-koodin toteutumista ennen siirtymistä seuraavaan lauseeseen. Järjestelmämuuttuja #3003 voi saada taulukon 9 mukaiset arvot:

#3003	Single block	Aputoiminnon loppusignaali
0	Käytössä	Odotetaan
1	Ei käytössä	Odotetaan
2	Käytössä	Ei odoteta
3	Ei käytössä	Ei odoteta

Taulukko 9. Muuttujan #3003 arvot ja niiden vastaavuudet. [5 s. 377]

Kun koneeseen kytketään virrat, muuttujan #3003 arvo on aina nolla, jolloin single block ajo on toiminnassa ja aputoimintojen toteutumista odotetaan ennen seuraavaa lausetta.

Esimerkki: Inkrementaalinen poraus sorvauksessa. (Vastaa G81 koodi)

Makron kutsukäsky:

G65 P9081 R (= #18 lähestymissiirto) **Z** (= #26 poraus siirtymä)

Makro-ohjelma on seuraavanlainen:

O9081	Makro-ohjelman numero.
G0 W#18	Pikaliikkeellä lähestytään muuttujalla annettu matka.
#3001 = 3	Lauseittain ajo pois käytöstä ja aputoimintoja ei odoteta. Näin varmistetaan, ettei terä jää pyörimään reiän pohjalle.
G1 W#26	Poraus suoritetaan syötöllä (huomioi että syvyys tulee inkrementaalisesti).
G0 W- [[#18] + [#26]]	Paluu pikaliikkeellä lähtötasolle.
#3003 = 0	Lauseittain ajo taas käytössä.
M99	Paluu pääohjelmaan.

Syötön säätökytkimen, feed hold -napin ja tarkanpysähtymisen toimintaa ohjataan muuttujalla #3004. Toteutuminen tapahtuu taulukon 10 mukaisesti.

#3004	Syötön pysäytys	Syötön säätö	Pysäytystarkkuuden tarkistus
0	Käytössä	Käytössä	Toteutuu
1	Estetty	Käytössä	Toteutuu
2	Käytössä	Estetty	Toteutuu
3	Estetty	Estetty	Toteutuu
4	Käytössä	Käytössä	Ei Toteudu
5	Estetty	Käytössä	Ei Toteudu
6	Käytössä	Estetty	Ei Toteudu
7	Estetty	Estetty	Ei Toteudu

Taulukko 10. Muuttujan #3004 arvot ja niiden vastaavuudet. [5 s. 377]

Kun syötönpysäytys on estetty muuttujalla #3004 ja syötönpysäytys –nappia painetaan, toimii ohjaus seuraavanlaisesti.

1) Syötönpysäytys toteutuu heti seuraavassa lauseessa, mikäli yksittäislauseen ajoa ei ole estetty #3003 ja ”feed hold”-nappia pidetään alas painettuna.

2) Jos feed hold –nappia ei pidetä alas painettuna, syötönpysäytysvalo syttyy ja syötönpysäytys toteutuu vasta kun muuttujan #3004 arvo sen sallii.

Esimerkki: Kierteitystyökierto (vastaa työkiertoa G84)

Makron kutsu:

G65 P9084 R (= #18 lähestymispiste) **Z** (= #26 reiän pohjan lopullinen syvyys)

Makro-ohjelma on seuraavanlainen:

O9084	Makro-ohjelman numero.
G0 Z#18	Pikaliikkeellä lähestymistasolle.
#3001 = 1	Lauseittain ajo pois käytöstä ja aputoimintoja odotetaan.
G1 Z#26	Kierteen poraus.
#3004 = 7	Syötönpysäytyksen, syötönsäädön ja pysäytystarkkuuden tarkistuksen estot päälle.
M5	Karan pysäytys.
M4	Kara pyörii vastapäivään.
G0 Z-#26	Palataan pikaliikkeellä lähestymistasolle.
#3004 = 0	Syötönpysäytys ja syötönsäätö ovat taas käytössä sekä pysäytystarkkuuden tarkistus on voimassa.
M5	Karan pysäytys.
M3	Kara pyörii myötäpäivään.
#3003 = 0	Lauseittain ajo on taas käytössä.
M99	Paluu pääohjelmaan.

7.4.6 Ohjelman pysäytys viestillä

Koneistuksessa saattaa tulla sellaisia vaiheita, joissa ohjelma on hyvä pysäyttää ja suorittaa jokin toiminta ennen kuin koneistamista voidaan jatkaa. Esimerkiksi lastujen poisto poterosta ennen viimeistelyä voisi olla tällainen.

Ohjelma voidaan toki pysäyttää M0-koodilla ja kirjoittaa sen perään kommentti. Itselläni on tästä sellaisia kokemuksia että kun kone pysähtyy, eikä näytössä ole mitään ilmoitusta, painetaan automaattisesti joko ”reset”-näppäintä tai ”cykle start”–näppäintä. Ohjelman läpiviemisen kannalta kumpikaan vaihtoehto ei ole tarkoituksen mukaista.

Parempi vaihtoehto on käyttää muuttujaa #3006, joka keskeyttää ohjelman edelliseen lauseeseen. Kone pysähtyy, ohjelma häviää näytöltä ja vain muuttujan perään kirjoitettu viesti tulee näkyviin. Nyt haluttu viesti on paljon selkeämmin havaittavissa ja sen ohittamisen mahdollisuus vahingossa pienenee huomattavasti. Viestin pituus saa olla enintään 26 merkkiä ja muista että myös välilyönti on merkki.

Esimerkki: #3006 = 1 (POISTA LASTUT ENNEN AVARR)

7.4.7 Peilikuvatoiminto

Toisinaan samasta tuotteesta pitäisi valmistaa vasemman- ja oikeanpuoleiset kappaleet, siis toistensa peilikuvat. Peilikuva toiminto voidaan kytkeä päälle suoraan koneen käyttöpaneelin kautta, tai se voidaan tehdä järjestelmämuuttujien #3007 ja #3008 avulla. Kyseiset koodit kirjoitetaan binäärikoodilla ja jokaisen akseli bitin tila kuvaa onko sen peilaus toiminnassa vai ei. Jokainen akseli voi siis saada vain arvon 0 tai 1. Bitti 0 merkitsee että peilikuva ei ole voimassa. Bitti 1 vastaavasti merkitsee että peilikuva on käytössä.

Jos muuttuja #3007 saa arvokseen luvun 3 on se binäärilukuna sama kuin: 0000 0011. Tässä tapauksessa akselit 1 ja 2 toimisivat peilikuva tilassa. Akseli 1 on koneen X-akseli ja Y-akselia vastaava luku on 2. Seuraavassa taulukossa on muuttujaa #3007 vastaavat akselit. [5 s.378]

8. aks.	7. aks.	6. aks.	5. aks.	4 aks.	3. aks.	2. aks.	1. aks.
0	0	0	0	0	0	1	1

Taulukko 11. Muuttujaa #3007 vastaavat akselit ja alla esim. mukaiset bitit.

7.4.8 Modaalinen informaatio

NC-tekniikassa modaalinen informaatio välittää koneelle tietoa sen hetkisistä toiminnoista. Kaikki modaaliset toiminnot on jaettu omiin ryhmiin ja näitä ryhmiä vastaan on taas omat muuttuja-arvot.

G-koodeja on kahden tyyppisiä. Osa G-koodeista on kertavaikutteisia, käytetään termiä ”ei-modaalinen”. Kyseinen G-koodi on voimassa vain siinä lauseessa, jossa se on määritelty. Modaalinen G-koodi on taas voimassa niin kauan kunnes se kumotaan määrittelemällä toinen samaan ryhmään kuuluva G-koodi. Esimerkiksi G1, joka on lineaarinen liike ja G2, joka on kaariliike myötäpäivään, eivät voi olla vaikutettuna samaan aikaan. Molemmat ovat modaalisia käskyjä ja kuuluvat samaan ryhmään.

Kaikkien modaalisten käskyjen hetkellinen arvo voidaan poimia lukemalla sitä vastaavan järjestelmämuuttujan arvo. Tämä toiminto on makro-ohjelmoinnissa erittäin hyödyllinen, joskus jopa välttämätön. Ajatellaan vaikka tilannetta jossa ohjelmoitsija

laatii pääohjelman absoluuttisesti, mikä on yleisin tapa työstökoneella. Absoluuttista ohjelmointia vastaava koodi on G90 ja se on modaalinen. Pääohjelmasta hypätään makro-ohjelmaan joka puolestaan onkin laadittu inkremantaalisesti. Tätä vastaava koodi on G91 ja myös se on modaalinen. Kun makrosta palataan takaisin pääohjelmaan paikoitukset tapahtuvatkin nyt inkrementaalisesti, eikä absoluuttisesti, niin kuin ohjelmoitsija oli suunnitellut. Pienin vahinko mitä tästä seuraa on terän katkeaminen, mutta siitä voi koitua huomattavankin suuria vahinkoja.

Tästä syystä on makro-ohjelman heti alkuun laatia kohta jossa poimitaan tarpeelliset modaaliset toiminnat talteen. Näitä ovat mm. liikkeen koodi ja ohjelmointi tapa sekä syötön arvo. Vastaavasti ohjelman lopussa palautetaan kyseisten koodien arvot vastaamaan pääohjelmassa voimassa oleviksi arvoiksi.

Seuraavassa taulukossa on tärkeimpiä modaalisia toimintoja.

#4001	Ryhmä 1 G0, G1, G2, G3, G33
#4002	Ryhmä 2 G17, G18, G19
#4003	Ryhmä 3 G90, G91
#4009	Ryhmä 9 G73, G74, G76, G80-G89
#4014	Ryhmä 14 G54 – G59
#4102	B-koodi
#4109	F-koodi
#4111	H-koodi
#4113	M-koodi
#4114	Lauseen numero
#4115	Ohjelman numero
#4119	S-koodi
#4120	T-koodi

Taulukko 12. Järjestelmänmuuttujat modaalisille toiminnoille. [5 s. 379]

7.4.9 Hetkellinen asema

5-tuhatta sarjan muuttujien kirjo on hyvin runsas ja vaihtelee jonkin verran, käytettävästä koneesta riippuen. Kaikille niille on kuitenkin yhteistä se että niiden hetkellinen arvo voidaan vain lukea, mutta ei kirjoittaa. Niitä ei siis voi käyttää suoraan laskutoimituksissa.

Tämän teoksen harjoituksissa ja usein myös käytännön työssäkin pärjää hyvin pitkälle, kun osaa tästä sarjasta muuttujat #5001, #5002 ja #5003. Niiden avulla luetaan sen hetkisten X-, Y- ja Z-akseleiden asemat. Nämä kyseiset tiedot pitää poimia muistiin esim. reikäpiirimakron alussa (katso liite 1). Seuraavaan taulukkoon on kerätty järjestelmämuuttujat hetkellisen aseman lukuun. Muuttujan viimeinen luku vastaa akselin arvoa, #5021 siis vastaa X-akselia jne.

Muuttuja	**Sijainti tieto**	**Koordinaatti järjestelmä**	**Työkalun kompensaatio arvo**	**Toiminnon lukeminen liikkeen aikana**
#5001 - #5008	Luistin loppu piste	Kappaleen koordinaatti järjestelmä	Ei sisälly	Mahdollista
#5021 - #5028	Nykyinen sijainti	Työstökoneen koordinaatti järjestelmä	Sisältyy	Mahdotonta
#5041 - #5048	Nykyinen sijainti	Kappaleen koordinaatti järjestelmä	Sisältyy	Mahdotonta
#5061 - #5068	Ohitussignaalin asema	Kappaleen koordinaatti järjestelmä	Sisältyy	Mahdollista
#5081 - #5088	Työkalun pituuden offset arvot			Mahdotonta
#5101 - #5108	Servoaseman poikkeama			Mahdotonta

Taulukko 13. Järjestelmämuuttujat paikoitustietoihin.[5 s. 380]

8 Aritmeettiset ja loogiset operaatiot

Makro-ohjelmoinnin idea on juuri siinä, että muuttujien kesken voidaan suorittaa erilaisia matemaattisia laskuoperaatioita. Saatuja tuloksia hyödynnetään mm. uusien koordinaattipaikkojen laskentaan.

Taulukossa 13 on esiteltynä valtaosa käytössä olevista aritmeettisista ja loogisista toiminnoista, joita voidaan hyödyntää makro-ohjelmien teossa. Toimintoja löytyy lisää mm. käytettävän koneen manuaalista. [4 s. 411]

# i = # j	Sijoitus	# i = SQRT [# j]	Neliöjuuri
# i = # j + # k	Yhteenlasku	# i = ABS [#j]	Itseisarvo
# i = # j - # k	Vähennyslasku	# i = ROUND [#j]	Pyöristys
# i = # j * # k	Kertolasku	# i = FIX [#j]	pyöristys alaspäin
# i = # j / # k	Jakolasku	# i = FUP [#j]	pyöristys ylöspäin
# i = SIN [# j]	Sini (asteina)	# i = # j OR # k	looginen summa (tai)
# i = COS [# j]	Kosini (asteina)	# i = # j XOR # k	poissulkeva tai
# i = TAN [# j]	Tangentti	# i = # j AND # k	looginen tulo (ja)
# i = ATAN [# j]	Argustangentti	# i = BIN [# j]	BCD-koodin muunnos binäärikoodiksi
		# i = BCD [# j]	binäärikoodin muunnos BCD-koodiksi

Taulukko 13. Aritmeettiset ja loogiset operaatiot.

Pyöristys funktioista muutama huomioon otettava seikka:

1) Jos käytät funktiota ROUND aritmeettisena toimintakäskynä, IF- tai WHILE-komennossa, pyöristyy siinä oleva luku tasaluvuksi.

2) Jos funktiota ROUND käytetään käskyssä NC-osoitteen ilmaisemiseen, pyöristetään se pienimpään mahdolliseen osoitteen tarkkuuteen.

3) Funktiot FUX ja FIX pyöristää aina lähimpään tasalukuun.

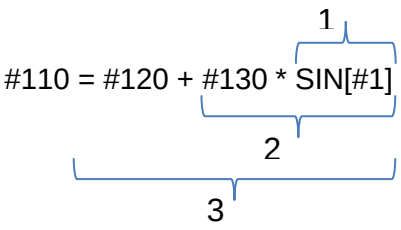
Esimerkkejä:

1. Koneen ohjelmointi tarkkuus on 0.001 mm ja #101 = 1.4567 ja #102= ROUND[#1]. Tällöin muuttuja #102 saa arvon 1.0.
2. Mutta jos X-akselille annetaan seuraava käsky, X[ROUND[#1]], ei X:n arvoksi tulekaan 1.0, vaan 1.457.

Muuttujat #1 = 3.3 ja #2 = -2.2 pyöristyvät seuraavanlaisesti:

1. Jos #103 = FUP[#1], saa muuttuja #103 arvon 4.0.
2. Jos #103 = FIP[#1], saa muuttuja #103 arvon 3.0.
3. Jos #103 = FUP[#2], saa muuttuja #103 arvon -4.0.
4. Jos #103 = FIP[#2], saa muuttuja #103 arvon -3.0.

Aritmeettisia toimintoja ja funktioita voidaan yhdistellä vapaasti. Ensimmäisenä suoritetaan funktio, sitten kertolasku ja viimeisenä yhteenlasku. Kuten matematiikassakin, niin myös tässä voidaan laskujärjestystä muuttaa sulkujen avulla. MUISTA käyttää aina hakasulkuja [ ]. Yhdessä lauseessa saa käyttää enintään viisi kertaa sulkeita.



9 Ohjauskäskyt

Ohjauskäskyt ovat matemaattisten ja loogisten operaatioiden ohella koko makro-ohjelmoinnin ydin. Muuttujien kesken voidaan suorittaa erilaisia vertailuja. Saatuihin vertailutuloksiin voidaan kohdistaa ehtoja tai hyppykäskyjä. Hyppykäskyt voivat olla joko ehdottomia tai ehdollisia.

9.1 Hypyt ja ehdot

Ehdoton hyppykäsky ohjelmoidaan muotoon ”GOTOn”, jossa n:llä on annettu sen lauseen numero, johon halutaan hypätä. Ohjelmoitaessa CNC-koneita, lauseen numerointi ei ole välttämätöntä, paitsi siinä lauseessa johon GOTO-käskyllä halutaan mennä. Ehdoton hyppykäsky aiheuttaa välittömän hypyn n:llä osoitettuun lauseeseen.

GOTO-käskyn käyttö hidastaa hieman ohjelman kulkua, koska ohjaus pysähtyy siksi aikaa, kun järjestelmä lukee ohjelmaa ja löytää halutun rivinumeron. Tämä viiveaika kasvaa sitä mukaa, kun GOTO-käskyn ja hypättävän lauseeseen välimatka pitenee. Siksi ne onkin syytä ohjelmoida kulkusuunnassa niin lähelle toisiaan kuin on mahdollista.

Ehdollisessa hyppykäskyssä suoritaan ensin vertailu ”IF[ehto] GOTOn”, ja jos hakasulkujen sisällä oleva ehto toteutuu, niin silloin suoritetaan hyppy n:llä osoitetulle riville. Mikäli ehto ei toteudu, ohjelman luku jatkuu suoraan seuraavalta riviltä. [4 s. 425]

Käytettävä ehto	Käytettävä lyhenne		
Equal to	EQ	yhtä suuri kuin	=
Not equal to	NE	eri suuri kuin	≠
Greater than	GT	suurempi kuin	>
Less than	LT	pienempi kuin	<
Greater than or equal to	GE	suurempi tai yhtä suuri	≥
Less than or equal to	LE	pienempi tai yhtä suuri	≤

Taulukko 14. Makro-ohjelmoinnissa käytettävät ehto käskyt.

Esimerkki:

IF [#21 GT 100] GOTO 10 (jos muuttujan #21 arvo on suurempi kuin 100, hyppää ohjelma lauseeseen numero 10).

Uudemmissa ohjauksissa voidaan IF-komennon yhteydessä käyttää määritelmää THEN. Silloin siihen voidaan vielä lisätä loogisia toimintoja.

Esimerkki:

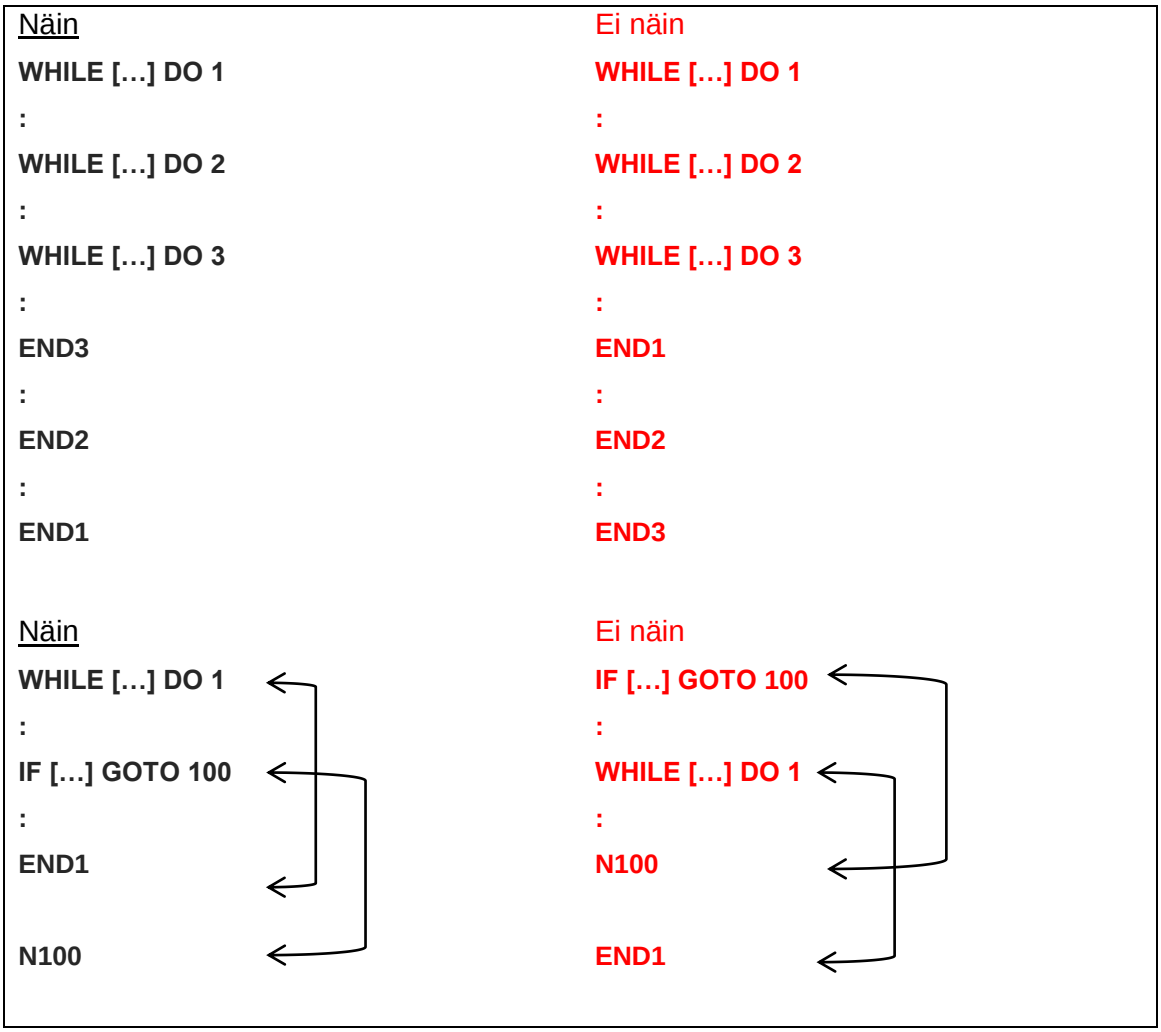
IF [#3 LT #2] THEN #1 = 0
IF [#5 LT #4] OR [#3 LT #2] THEN #1 = 0

9.2 Toisto

Toisto eli silmukka voidaan rakentaa WHILE-koodin avulla. Se ohjelmoidaan muotoon ”WHILE [ehto] DOm” ja silmukan loppuun pitää aina määrittää vielä ”ENDm”. Ohjelman käskyt ”WHILE [ehto] DOm” ja ”ENDm” muodostavat aina parin, eli silmukan. Silmukka tunnistetaan numeron ”m” perusteella. Kun hakasulkeissa annettu ehto toteutuu, toistetaan lauseita välillä DOm ja ENDm. Jos ehto ei toteudu, hypätään ENDm:ää seuraavaan lauseeseen. WHILE-komennon yhteydessä ei myöskään synny GOTO-käskyn kaltaista viivettä lainkaan, joten sen käyttö onkin siinä suhteessa suositeltavampaa. Seuraavassa on esimerkki silmukan käytöstä.[4 s. 427]

#501 = 0	Laskuri, eli muuttujan #501 arvo nollataan.
WHILE [#501 LT 99] DO 1	Niin kauan kuin muuttujan #501 arvo on pienempi kuin 99, tehdään alla olevat toimenpiteet. Kun muuttuja #501 arvo on 100 eli suurempi kuin 99 poistutaan silmukasta, jolloin M30 lopettaa ohjelman.
#501 = #501 + 1	Laskurin eli muuttujan 501 arvoa kasvatetaan yhdellä.
#100 = #100 + 50.	Poraustyökierron X-arvon muuttujaa kasvatetaan 50 mm.
G81 X#100 Y30. Z-20. R5.	Suoritetaan poraus.
END 1	Silmukka 1:n loppukohta
M30	

WHILE-silmukoita voidaan ketjuttaa, mutta enintään kolmella tasolla. Tällöin silmukat erotellaan toisistaan numeroilla 1, 2 ja 3 ja niitä voidaan käyttää niin monta kertaa kuin on tarpeen. Numerot tulee kirjata molempiin kohtiin, sekä WHILE-lauseen DO-komentoon, että kyseisen silmukan lopettavaan END-komentoon. MUISTA, mikäli käytät useampaa silmukkaa, ne eivät saa mennä ristiin. Huomio myös, että silmukasta voidaan poistua GOTO-käskyllä, mutta sillä ei voida hypätä silmukan sisään.



Kuva 3. Silmukassa sallitut ketjutukset ja hypyt [4 s. 428].

10 Rajoitukset

Makro-ohjelmia laadittaessa on hyvä muistaa että NC-ohjaus suorittaa kaikki laskutoimitukset binäärilukuina. Oletetaan että muuttuja #1 saa arvon 0.002 ja suoritetaan yksinkertainen laskutoimitus: #2 = 1000 * #1. Kymmenjärjestelmällä laskettuna vastaukseksi pitäisi tulla 2, mutta binäärilukuina laskettaessa vastaukseksi tuleekin 1.99999997. Mittalukuna tämän suuruinen virhe on mitättömän pieni, eikä sitä pystyttäisi normaalisti edes mittaamaan. Yhteen- ja vähennyslaskuissa sekä loogisissa toiminnoissa näinkin pienestä erosta saattaa tulla ongelmia. [4 s.416]

Binäärijärjestelmästä johtuen myös tallennustarkkuudessa voi ilmetä virheitä. Suurimmassa osassa vielä käytössä olevista koneista, ohjaukset käsittelevät lukuja kahdeksan bittisenä. Siitä saattaa seurata seuraavan esimerkin kaltaisia ongelmia.

Annetaan muuttujille #1 ja #2 seuraavat arvot.

#1 = 9876543277777.777
#2 = 9876543210123.456

Binäärijärjestelmä muuttaa lukemat seuraavanlaisesti:

Muuttuja	Kymmenjärjestelmä	Binäärijärjestelmä
#1	9876543277777.777	9876543300000.000
#2	9876543210123.456	9876543200000.000
Tulos	67654.321	100000.000

Yhtälön #1 - #2 vastaukseksi pitäisi siis tulla 67654.321, mutta binäärijärjestelmällä lopputulos onkin 100000.000, jotenka ero on aika suuri. Käytännössä, harvemmin lasketaan näin suurilla luvuilla, mutta kyseinen virhemahdollisuus on kuitenkin syytä muistaa.[4 s.416]

Edellä mainittuja virheitä pystytään välttämään pienillä muutoksilla. Loogisissa toiminnoissa voidaan ehtoasettelua muuttaa. Laskutoimituksissa voidaan käyttää pyöristyksiä ylös- tai alaspäin. Voidaan myös lisätä niin pieni luku, että ehto toteutuu, mutta toiminto ei vielä muutu.

11 Makro-ohjelmien kutsut

Makro-ohjelmankutsu voi olla kertakutsu, jolloin se on voimassa vain siinä lauseessa, jossa se on ohjelmoitu. Käytettävä koodi on silloin G65. Makro-ohjelma voidaan kutsua myös modaalisesti, jolloin se on voimassa siksi, kunnes se peruutetaan erillisellä käskyllä. Modaalinen kutsu suoritetaan koodilla G66 ja sen peruutus tapahtuu koodilla G67. G66 koodia käytetään etenkin silloin, kun makro-ohjelmalla on tehty omia työkiertoja. Molemmissa tapauksissa makro-ohjelmasta palataan takaisin pääohjelmaan M99-koodilla. [4 s. 431]

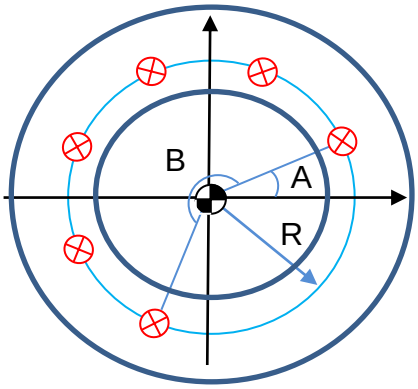
Käytetäänpä makro-ohjelman kutsussa sitten kumpaa tahansa, G65 tai G66 koodia, niin molempien yhteydessä pitää antaa tarvittavat argumentit. Näiden argumenttien avulla saadaan makro-ohjelmaan siirrettyä halutut lähtöarvotiedot. Lähtöarvojen siirtämiseen voidaan käyttää kahta eri tapaa, menetelmä 1 tai menetelmä 2. Taulukossa 15 on esitelty helpomman ja yleisemmin käytetyn menetelmän 1 argumentit. [4 s. 431]

A	#1	I	#4	T	#20
B	#2	J	#5	U	#21
C	#3	K	#6	V	#22
D	#7	M	#13	W	#23
E	#8	Q	#17	X	#24
F	#9	R	#18	Y	#25
H	#11	S	#19	Z	#26

Taulukko 15. Argumenttien määrittely menetelmällä 1.[5 s. 393]

Osoitteita G, L, N, O ja P ei voida käyttää argumentteina. Tarpeettomat osoitteet voidaan jättää pois. Osoitteiden järjestyksellä käään ei ole väliä, kuitenkin poikkeuksena tulee huomioida I, J ja K, jotka pitää ohjelmoida aakkosjärjestyksessä. [4 s. 431]

Seuraavassa esimerkissä on malli reikäpiirimakron kutsumisesta sekä siihen liittyvät ohjeet ja kuva. -



REIKÄPIIRI ALKU- JA LOPPUKULMALLA
MAKRO-OHJELMAN KUTSUMUOTO:
G65 P8100 A___ B___ C___ R___
A (#1) = REIKÄPIIRIN ALKUKULMA
B (#2) = REIKÄPIIRIN LOPPUKULMA
C (#3) = REIKIEN LUKUMÄÄRÄ
R (#18) = REIKÄPIIRIN SÄDE
PAIKOITETAAN PIIRIN KESKELLE.
B VOIDAAN JÄTTÄÄ ANTAMATTA MIKÄLI REIKÄPIIRI KIERTÄÄ TÄYDEN YMPYRÄN.
KÄYTÄ PORAUSKIERTOKUTSUN YHTEYDESSÄ L0-OSOITETTA,
JOS ET HALUA ETTÄ PORAUS SUORITETAAN MYÖS REIKÄPIIRIN KESKELLE

Kuva 4. Reikäpiirimakron ohje.

Menetelmässä 2 käytetään argumenttien siirtoon vain osoitteita A, B ja C sekä I, J ja K. Osoitteen I, J ja K voidaan ohjelmoida enintään kymmenen kertaa.

A__ B__ C__ I__ J__ K__ I__ J__ K__ ….

Taulukkoa 16 on esitetty makron kutsun yhteydessä käytettävät osoitteet ja niitä vastaavat ositteet käytettäessä menetelmää 2. Jos samalla osoitteella ohjelmoidaan enemmän kuin yksi sarja, täytyy niiden määrittely suorittaa täsmällisesti oikeassa järjestyksessä. Osoitteet, joita ei tarvita voidaan jättää pois. Mielestä tämä menetelmä on hieman sekavampi ja siinä tulee tehtyä helpommin virheitä ja siksi en ainakaan itse suosi sen käyttöä.

Jos haluat käyttää argumenttien siirtoon menetelmää 2, on sinun muutettava koneen parametri nro 6008 ja sen bitti 7 arvosta 0 arvoon 1.

A	#1	K_3	#12	J_7	#23
B	#2	I_4	#13	K_7	#24
C	#3	J_4	#14	I_8	#25
I_1	#4	K_4	#15	J_8	#26
J_2	#5	I_5	#16	K_8	#27
K_1	#6	J_5	#17	I_9	#28
I_2	#7	K_5	#18	J_9	#29
J_3	#8	I_6	#19	K_9	#30
K_2	#9	J_6	#20	I_{10}	#31
I_3	#10	K_6	#21	J_{10}	#32
J_4	#11	I_7	#22	K_{10}	#33

Taulukko 16. Menetelmän 2 argumentit ja niitä vastaavat osoitteet. [5 s. 394]

Käytetään sitten argumenttien siirtoon kumpaa tahansa, menetelmää 1 tai 2. On ohjelman tekijän syytä tehdä tarkat ja selkeät ohjeet makro-ohjelman käytöstä sekä siinä vaadittavien osoitteiden toiminnasta. Siis tehdään selkeä kuva, josta näkyy kaikki tarvittavat argumentit ja mihin tarkoitukseen ne tulevat. Kuvasta tulee myös ilmetä mihin kyseinen makro soveltuu. Tarvittaessa kuvaa voidaan vielä täydentää lyhyellä selosteella, josta makron käyttäjälle selviää loput tarvittavat tiedot.

11.1 Makro-ohjelman kutsu G- ja M-koodilla

Työstökoneessa olevat poraustyökierrot tai sorvissa valmiina olevat työkierrot ovat todellisuudessa makro-ohjelmia. Arvot, joita niiden kutsun yhteydessä annetaan, ovat argumentteja. Työkalun vaihtokoodi, M6, on sekin oikeasti makron kutsu.

Koneen ohjelmoija voi laatia vastaava toiminnon itse. Kutsuttaessa makroa G-koodilla tulee sen ohjelmanumeron olla välillä O9010- O9019 ja niiden vastaavuus määritellään parametrien #6050 – #6059 välillä. M-koodia taas käytettäessä, ohjelman tulee olla välillä O9020 – O9029 ja niitä vastaavat parametrit ovat #6080 – #6089. Jos halutaan kutsua ohjelmaa O9010 G-koodilla 100, niin annetaan parametrinumeron #6050 arvoksi 100. Nyt annettaessa komento G100 ja siinä tarvittavat muuttujat, kutsutaan todellisuudessa makro-ohjelmaa joka on tallennettu koneelle numerolla 9010. Taulukko 17 esittää G- ja M-koodeja vastaavat parametrit.

G-koodia vastaavat parametrit		M-koodia vastaavat parametrit	
Ohjelmanumero	Parametrinumero	Ohjelmanumero	Parametrinumero
O9010	#6050	O9020	#6080
O9011	#6051	O9021	#6081
O9012	#6052	O9022	#6082
O9013	#6053	O9023	#6083
O9014	#6054	O9024	#6084
O9015	#6055	O9025	#6085
O9016	#6056	O9026	#6086
O9017	#6067	O9027	#6087
O9018	#6058	O9028	#6088
O9019	#6059	O9029	#6089

Taulukko 17. G- ja M-koodeja vastaavat parametrit.

12 Makro-ohjelmien laatiminen

On hyvä muistaa että ei ole olemassa yhtä ja ainoaa tapaa laatia jokin tiettyä makro. On vain erilaisia tapoja toteuttaa ne, aivan kuten perusohjelmoinnissakin.

Otetaan lähtökohdaksi makron laatiminen viimeistelyplaanaukseen. Ensimmäiseksi kannattaa laatia ohjelma ilman muuttujia. Sitä varten lähdetään pohtimaan mm. seuraavia kysymyksiä.

Mikä on käytettävän terä leveys? Mistä kohtaa terä siirtyy haluttuun syvyyteen? Käytetäänkö alaspäin liikkeessä pienempää syöttöä? Koneistetaanko myötäjyrsinnällä vai käytetäänkö zigzag-menetelmää? Kuinka suuri on seuraava sivuttainen siirtymä?

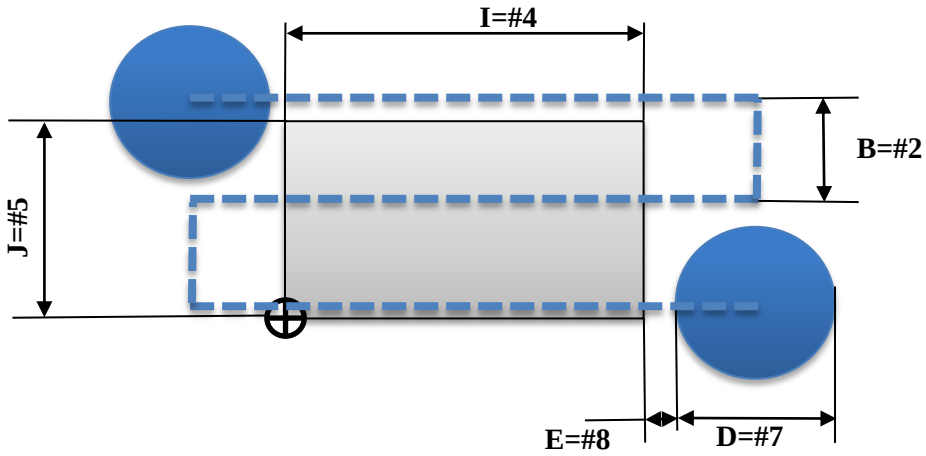
Tämän jälkeen piirustuksen mitat korvataankin jollain argumenttitaulukossa annetulla kirjaimella. Tähän kannattanee käyttää paperia ja kynää, mutta voit toki käyttää jotain tietokoneohjelmaa apuna. Huomioi että editori-ohjelmalla ei voi simuloida makroja, mutta sen sijaan voit tarkastella esim. EXCEL-taulukkolaskennalla, muuttujien tuloksia. Taulukossa 18 on siitä esimerkki.

O8100 (PLAANAUS MAKRO)	Syöttö kenttä	Lähtö
(A #1 = Tason leveys X-suunnassa)	150	X-akseli
(B #2 = Tason leveys Y-suunnassa)	75	0
(C #3 = Terästä käytettävä leveys prosentteina)	70	Y-akseli
(D #7 = Terän halkaisija)	50	0
(E #8 = Sivuturvaetäisyys)	5	
(F #9 = Syöttö)	500	
(H #11 = Poistettava määrä Z-suunnassa)	5	
(Q #17 = Rouhintalastun paksuus)	2,5	
(R #18 = Lähestymis- ja paluunpiste Z-suunnassa)	7	
(U #21 = Ulkoreunan ylitysmatkan minimi)	5	
(V #22 = Viimeistelylastun paksuus)	0,5	
(Z #26 = Z-akselin lopullinen syvyys)	2	

Laskurit		
#111=#7/2 (TERAN SADE)		25
#112=[#3/100]*#7 (SIVUSIIRTYMAN SUURUUS)		35
#113=#8+#111+ABS[#101](TURVAVALI + TERAN SADE + X-AKS LAHDON ITSEISARVO)		30
#114 =#8+#111+#1+#101 (TURVAVALI + TERAN SADE +KPL:N LEVEYS +X:N LAHTOKOHTA)		210
#115=#102-#112+#111] (Y:N LAHDON ARVO - SIVUSIIRTYMAN SUURUUS + TERAN SADE)		-10
#116=-[#2+#21] (KPL:N KORKEUS + ULKOREUNAN YLITYS = Y:N MINIM ARVOI)		-80
IF [#11LT#17]THEN#17 =#11 (JOS H:N ARVO < Q:N ARVO SILLOIN Q SAA H:N ARVON)		
#117=0 (ARVON NOLLAUS)		
IF[#22GT0]THEN#117=1	Uudet arvot	
#118 =#17		
#120=#11	5	
#121=#17	2,5	
#122=#22	0,5	
IF[#11EQ=]THEN#120=1	5	
IF[#17EQ=]THEN#121=1	2,5	
IF[#22EQ=]THEN#122=1	1	
#124 =#26+#11	7	Z-TASO ENNEN ROUHINTAA
#125 =#26+#22	2,5	Z-TASO ENNEN VIIMEISTELYÄ
#130=FUP[[#120-#22]/#121]	2	ROUHINTALASTUJEN LASKURI
#131=#130+#117	3	KOKONAISLASTUKERTOJEN LASKURI
WHILE[#131GT0]DO1		TEHDÄÄN DO1
#132=#132+1	3	ROUHINTALASTUJEN MÄÄRÄ
#141=#124-[#17*#132]	-0,5	
IF[#141LT#125]THEN#141=#125	2,5	
IF[#132EQ#131]THEN#141=#26	2	
VARSINAINEN OHJELMA ALKAA TÄSTÄ		
N10 G90 G0 X30,00 Y-10,00		
G0 Z7,00		
G1 Z4,50 F250 TUTKITTAVA TULOS		

Taulukko 18. Taulukkolaskennalla tutkittu Z-akselin saavuttamaa mittaa.

Seuraavaksi lähdetään muuttamaan tehtyä ohjelmaa niin, että käytetäänkin kaikissa mahdollisissa siirtymissä muuttujia. Kyseisessä esimerkissä lähtöpaikan piste on laskettu niin että terän sädemittaan on lisätty haluttu turvaetäisyys. Tämän jälkeen makroon sijoitetaan sama lasku niin, että korvataan vakioarvo muuttujalla.

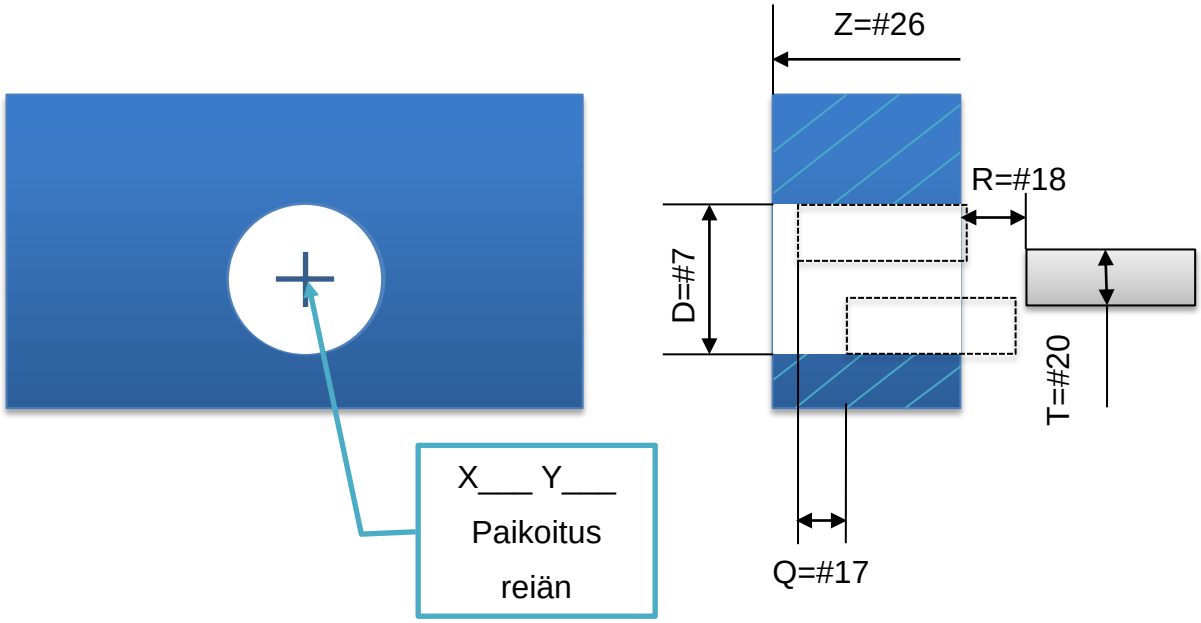


Kuva 5. Terän paikoitus ja siirrot plaanauksessa.

Yksi erittäin tärkeä asia, joka uutta makroa laadittaessa tulee tehdä, on lisätä siihen hyvät ja selkeät ohjeet, jotta muutkin kykenevät sitä käyttämään. Lisäksi makro-ohjelmaan on hyvä laittaa kommentteja tärkeimpiin kohtiin, jotta sitä on helpompi korjata tarvittaessa. Kun uusia makro-ohjelmia on valmis, on ne syytä testata erittäin huolella. Samalla tulee testata, ettei siinä ole mahdollista tehdä virheitä. Katso tarkemmin luvusta 7.4.3 Hälytys.

Harjoitustehtävä 1: Laadi plaanausmakro niin, että siinä on pelkästään viimeistely. Käytä kuvaa 5 apuna.

Harjoitustehtävä 2: Käytä seuraavaa kuvaa apuna ja laadi makro-ohjelma helical-interpolaatiolle. Tee vain viimeistely ja äläkä turhaan käytä sädekompensointia. Se voidaan hoitaa muuttamalla halutun halkaisijan D-arvoa.



- D → #7 = Reiän lopullinen halkaisija
- Q → #17 = Lastun syvyys / kierros
- R → #18 = Lähestymis- ja paluunpiste Z-suunnassa
- T → #20 = Terän Ø
- Z → #26 = Z-akselin lopullinen syvyys

Kuva 6. Ohjekuva helical-interpolaatio makrolle.

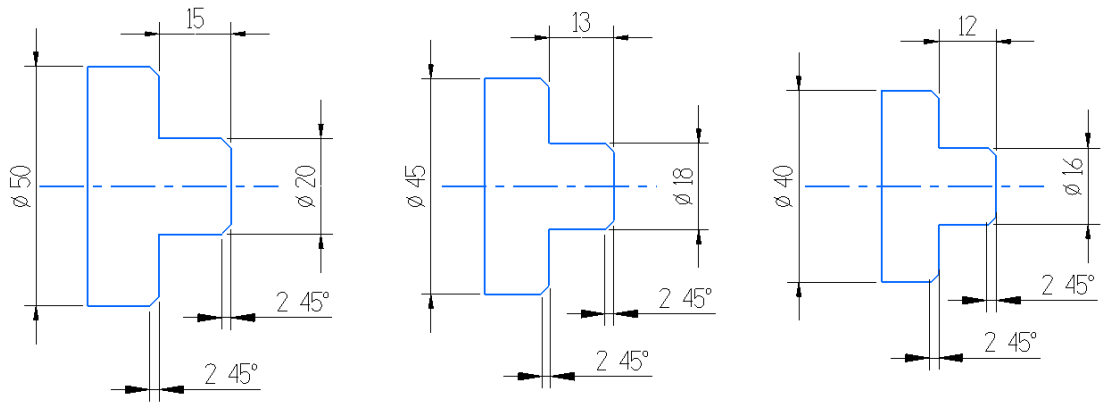
13 Makrojen käyttö sorvausohjelmien apuna

Myös sorvausohjelmoinnin apuna voidaan onnistuneesti hyödyntää makro-ohjelmia. Uuden makron luomisessa toimitaan vastaavasti kuin luvussa 12 jo kerrottiin. Sorvausohjelmissa on kohtalaisen helppoa käyttää sekä absoluuttista että inkrementaalista ohjelmointia ja jopa samassa lauseessa sekaisin. Siksi sorville voidaan laatia monenlaisia makroja. Oivallisia käyttökohteita ovat mm. seuraavat:

- Urien pistot, niin että saadaan aikaan haluttu leveys ja niihin tulee samalla pyöristykset tai viisteet.
- Kiilahihnat, niin että ne on suoraan taulukoitu, jolloin valitaan vain kappaleen halkaisija sekä standardinmukaisen hihnan tyyppi ja uran sijainti.
- Kartiokulmat, niin että ohjelma ajetaan kartiokulman alkuun ja siinä kohtaa kutsutaan makro, joka laskee kahden tunnetun tekijän avulla tarvittavat loppupisteet. Tämän tyyliset makrot on hyvä muuttaa G-koodilla kutsuttaviksi.
- Osatuoteperheet, niin että laaditaan esim. taulukko josta poimitaan kulloiseenkin kappaleeseen tarvittavat muuttujat.

Jos sorvissa on jo olemassa makroja, niin niitä kutsutaan ja käytetään aivan kuten työstökeskuksessakin.

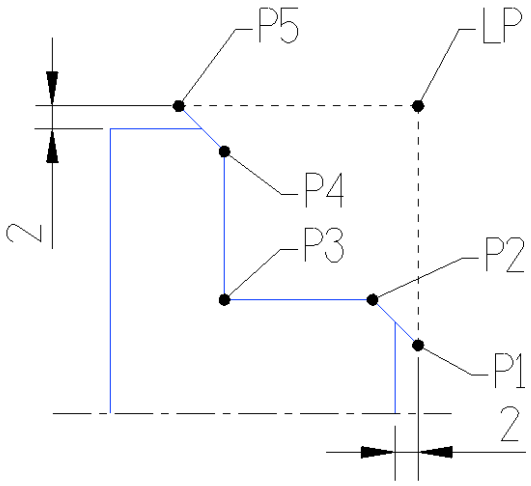
Tarkastellaan seuraavaksi kuvan 7 esittämää osatuoteperhettä. Siinä on kolme ohjaustappia joiden mitat poikkeavat vain hieman toisistaan. Kokonaispituus on kaikissa sama.



Kuva 7. Osatuoteperhe ohjaustapeista ja niiden mitat.

Kyseisissä ohjaustapeissa on vain viisi ohjelmointiin vaadittavaa pistettä ja niiden lisäksi tarvitaan vielä lähestymispiste (LP), joka on tässä tapauksessa kahdenmillin etäisyydellä kappaleen otsa- ja kehäpinnasta. Muodonkuvaukseen vaadittavat pisteet on esitetty kuvassa 8.

Kun tähän kappaleeseen lähdetään laatimaan perusohjelmaa, se tapahtuu näiden viiden pisteen avulla. Ohjelmointiin käytetään rouhintatyökiertoa (G71) ja viimeistelytyökiertoa (G70) apuna. Alkuun laitetaan vielä kierrosrajoitukset, oikeat lastuamisnopeudet ja valitaan haluttu työkalu sekä laitetaan kara pyörimään oikeaan suuntaan. Seuraavassa esimerkissä on laadittu rouhinta- ja viimeistelyohjelma halkaisijan 50 omaavalle kappaleelle, niin että se noudattaa kuvan mukaisia pisteitä.



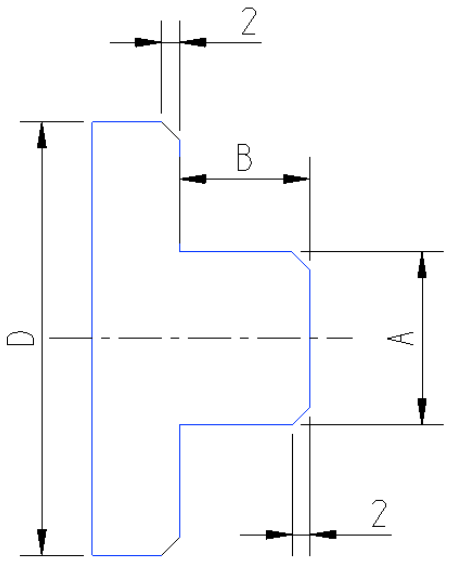
Kuva 8. Ohjaustapin ohjelmointiin tarvittavat pisteet.

Esimerkki ohjauspinnin perusohjelmasta:

O1000	Ohjelman numero.
G0 T0101 (ROUHINTA)	Rouhintatyökalun ja sen korjaimien valinta.
G96 S180 M3	Vakiolastuamisnopeuden asetus.
G0 X54. Z2. M8	Paikoitus lähestymispisteeseen, joka on otsapinnasta ja ulkokehältä säteessä 2 mm. Samalla leikkuuneste menee päälle.
G94 X-1.8 Z0 F0.2	Pääntasaus.
G71 U2. R1.	Rouhintalastun paksuus 2 mm ja irtivetomatka 1 mm.
G71 P10 Q20 U0.5 W0.1 F0.35	Muodonkuvauksen ensimmäisen ja viimeisen lauseen numerot. Viimeistelyvarojen suuruudet ja rouhinta syöttö.
N10 G42 G0 X12.	Paikoitetaan muodonkuvauksen ensimmäiseen pisteeseen P1. Kompensointi oikealle.
G1 X20. Z-2. F0.2	Siirto muodonkuvauksen pisteeseen P2 ja sopiva syöttö viimeistelyyn.
Z-15.	Siirto muodonkuvauksen pisteeseen P3
X46.	Siirto muodonkuvauksen pisteeseen P4
N20 X54. Z-17.	Siirto muodonkuvauksen pisteeseen P5
G40 G0 X100. Z200. M9	Kompensoinnin peruutus ja siirto pikaliikkeellä turvallisen matkan päähän, jossa voidaan suorittaa tarvittaessa uuden terän vaihto. Leikkuunesteen peruutus.
T0100 M5	Työkalun mittojen nollaus ja karan pysäytys.
M1	Valinnainen ohjelman pysäytys.
G0 T0303 (VIIMEISTELY)	Viimeistelytyökalun ja sen korjaimien valinta.
G96 S250 M3	Vakiolastuamisnopeuden asetus viimeistelylle.
G0 X54. Z2. M8	Paikoitus lähestymispisteeseen.
G70 P10 Q20	Viimeistelytyökierron kutsu.
G40 G0 X100. Z200. M9	Pikaliikkeellä turvallisen matkan päähän.
T0300 M5	Työkalun mittojen nollaus ja karan pysäytys.
M30 (M2)	Ohjelman lopetus.

Seuraavaksi piirretään mallinmukainen kuva, johon muuttuvat mitat korvataan argumenttien avulla. Muuttuvista mitoista kannattaa vielä laatia taulukko. Tämän jälkeen suunnitellaan miten ohjelma halutaan toteuttaa, eli tässä vaiheessa suunnitellaan myös kaikki loput muuttujat joita makro sisältää.

Tässä mallissa ohjelma on toteutettu niin, että koneenkäyttäjä laatii vain lyhyen pääohjelman, jossa määrätään ainoastaan rouhintatyökalun valinta ja maksimi kierrokset. Tämän jälkeen tulee makronkutsu ja ohjelman lopetus.



Ohjauspinni	A	B	D
1	16	12	40
2	18	13	45
3	20	15	50

G65 P7001 A__ B__ D__ Q__ R__ S__ T__ U__ W__ F__

Q = ROUHINTALASTUN PAKSUUS
R = ROUHINNAN SYÖTTÖ
S = VIIMEISTELYN VAKIOLASTUAMISNOPEUS
T = VIIMEISTELYTYÖKALU
U = VIIMEISTELYVARA X-SUUNNASSA
W = VIIMEISTELYVARA Z-SUUNNASSA
F = VIIMEISTELYN SYÖTTÖ

Kuva 9. Ohjauspinnin makro-ohjelman ohjeet.

Varsinainen pääohjelma on siis hyvin yksinkertainen ja näyttää tältä:

O1000	Ohjelman numero.
G0 T0101 (ROUHINTA)	Rouhintatyökalun ja sen korjaimien valinta.
G96 S180 M3	Vakiolastuamisnopeuden asetus.
G65 P7001 A20 B15 D50 Q2 R0.35 S250 T0303 U0.5 W0.1 F0.25	Makron kutsu ja tarvittavien muuttujien asetus.
M2 (M30)	Ohjelman lopetus.

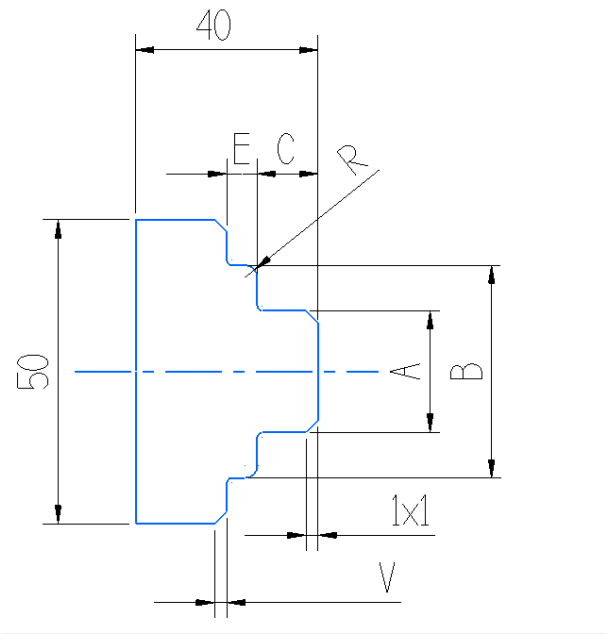
Käytännöllisin tapa toteuttaa tämän tapaisen osatuoteperheen sorvaus, lienee ratkaista makro niin, että sorvarin tulee määrittää vain makrokutsun yhteydessä ohjauspinnin numero ja käytettävät työkalut sekä niiden työstöarvot. Laaditaan kuitenkin ensin edellisen ohjeen mukainen makro ja lisätään siihen vasta myöhemmin tarvittavat vertailut ja muuttujat. Seuraavan sivun esimerkistä näkee, miltä ohjeen mukainen makro-ohjelma näyttää.

Harjoitus 3. Muuta makro-ohjelmaa niin, että koneenkäyttäjän ei tarvitse määritellä muuta kuin pinnin numero, viimeistelytyökalun numero sekä viimeistelyyn tarvittava lastuamisnopeus ja syöttö. Jos yrityksessä on aina vakiotyökalut rouhintaan ja viimeistelyyn sekä materiaali on ohjauspinneissä aina sama, voidaan ne sisällyttää suoraan makroon. Nyt makrokutsuun riittää pelkästään pinnin numero ja tietenkin makron numero.

Makro-ohjelma näyttää tältä:

O7001	Makro-ohjelman numero.
G0 X[#7+2*2] Z2. M8	Muuttuja D:n arvoon lisätään turvaväli, säde 2 mm x 2.
G94 X-1.6 Z0 F0.2 **(G94 X-[#2201*2] Z0 F0.2)**	Pääntasaus. X-akselin arvoksi voitaisiin poimia offset tiedoista nirkonsäteen arvo, tässä tapauksessa #2201 ja kertoa se kahdella. HUOM! EI TOIMI KAIKISSA KONEISSA.
G71 U#17. R1.	Rouhintalastun paksuus on muuttuja Q
G71 P10 Q20 U#21 W#23 F#18	Viimeistelyvarat ja rouhintasyöttö muuttujista.
N10 G42 G0 X[#1-8].	Paikoitetaan terä pisteeseen P1, joka saadaan vähentämällä muuttuja A:n arvosta 2 x 2 viisteen arvo ja saman verran lähestymisvaraa. Muista että sorvissa X-akselin mitat annetaan halkaisija mittoina, eli kerrotaan vielä kahdella jolloin vastaukseksi tulee 8.
G1 X#1. Z-2. F0.2	Siirto pisteeseen P2.
Z-[#B].	P3.
X[#7-4]	P4. Muuttuja D:stä vähennetään 2 x 2 viiste.
N20 [X#7+4]. Z-17.	P5. D:n arvoon lisätään 2 x 2 ylitysmatka.
G40 G0 X100. Z200. M9	Pikaliikkeellä turvallisen matkan päähän ja leikkuunesteet sekä kompensointi pois päältä.
#100=#4120	Työkaluarvon luku
#100=#100*100	T1*100=100
T#100 M5	Työkalun mittojen nollaus ja karan pysäytys.
M1	Valinnainen ohjelman pysäytys.
G0 T#20 (VIIMEISTELY)	Viimeistelytyökalun ja sen korjaimien valinta.
G96 S#19 M3	Vakiolastuamisnopeuden asetus viimeistelylle.
G0 X[#7+2*2] Z2. M8	Paikoitus lähestymispisteeseen.
G70 P10 Q20	Viimeistelytyökierron kutsu.
G40 G0 X100. Z200. M9	Pikaliikkeellä turvallisen matkan päähän.
#100=#4120	Työkaluarvon luku
#100=#100*100	T3*100=300
T#100 M5	Työkalun mittojen nollaus ja karan pysäytys.
M30 (M2)	Ohjelman lopetus.

Harjoitustehtävä 4. Laadi alla olevan kuvan ja taulukon mukaiselle osatuoteperheelle sopiva makro-ohjelma niin, että sorvarin täytyy määritellä ainoastaan, mikä pinni kulloinkin tehdään. Tee makrolle myös selkeät käyttöohjeet.



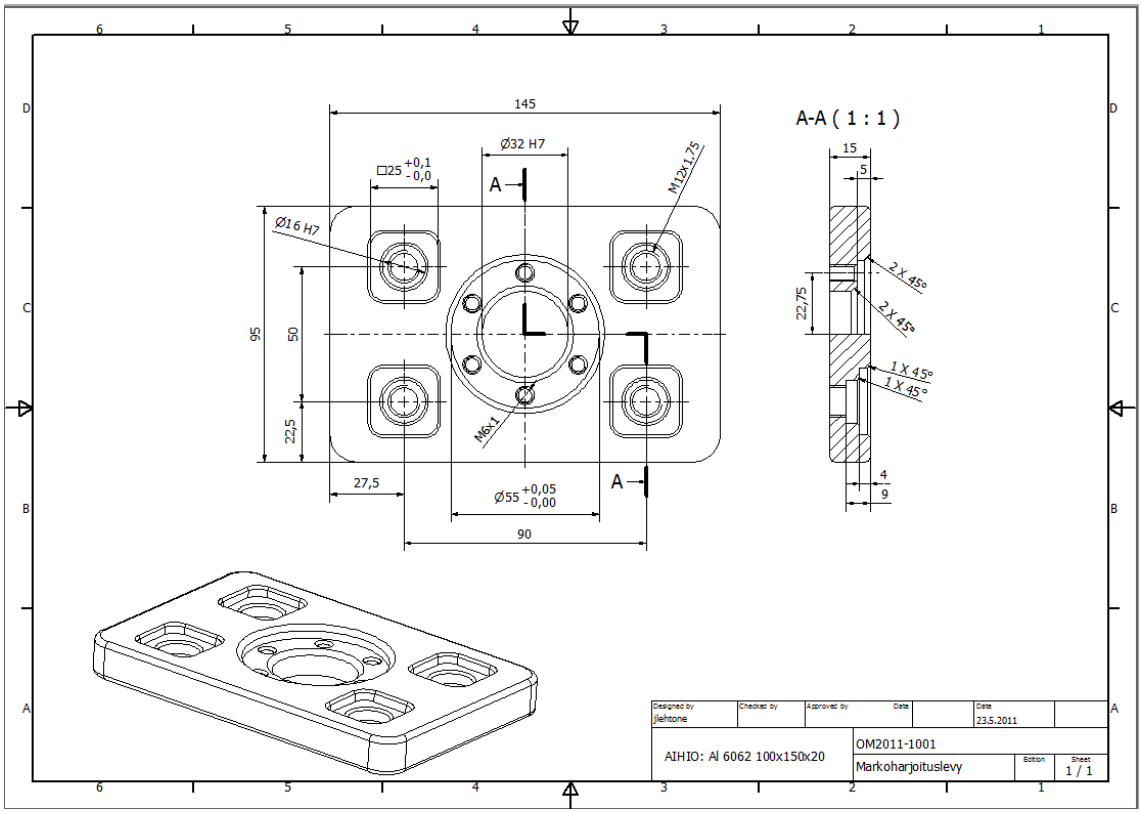
Pinni	A	B	C	E	R	V
1	12	25	10	4	4	1,5x1,5
2	14	28	10	5	4	1,5x1,5
3	16	30	12	6	5	2x2
4	18	32	12	6	5	2x2
5	20	35	13	7	6	2,5x2,5

Kuva 10. Tehtävään 4 tarvittavat mitat ja taulukko.

14 Loppuharjoitus

Tee tarvittavat makro-ohjelmat kuvan 11 levyyn ja laadi sellainen pääohjelma, joka hyödyntää niitä kaikissa työvaiheissa (pyydä tarvittaessa parempi kuva opettajalta). Käytä hyväksi jo aikaisemmin laadittuja makroja. Tarvitset mm. suorakaiteen ympäriajoon sellaisen makron, jossa on nurkkien pyöristys. Tee suorakaidetaskun rouhintaan ja viimeistelyyn makro-ohjelmat. Vaikka porattavia reikiä onkin suorakaidepisteverkossa vain neljä, tee silti myös niiden poraukseen oma makro. MUISTA laatia selkeät ohjeet joka makro-ohjelmaan.

Tämä harjoitus voidaan toteuttaa myös niin, että opettaja laatii kaikki tarvittavat makrot sekä niiden käyttöohjeet ja oppilas ainoastaan laatii pääohjelman, jossa hyödynnetään olemassa olevia makroja.



Kuva 2. Makroharjoituslevy.

15 Yhteenveto

Opinnäytetyö oli mielestäni erittäin opettavainen ja se onnistui hyvin. Sain arvokasta työkokemusta juuri siltä alalta, johon haluan opintojeni jälkeen sijoittua. Projektin aikana pidin kurssin ”ohjelmointikoulutus koneistajille”. Se toteutettiin Salon seudun aikuisopistossa jatkokurssina koneistusalan osaajille. Monipuoliselta koulutettavien joukolta sain itsekin arvokasta tietoa erilaisten pulmien ratkaisuihin, joita pystyn varmasti tulevaisuudessa hyödyntämään.

Projektin toinen osa-alue oli Espoossa, Omniassa, nuorten ammatillisella koulutuksen puolella ja siellä ohjelmointikoulutuksen järjestäminen. Näiden lisäksi projektissa oli vielä mukana salolainen alihankintaan erikoistunut konepaja HalikkoTools.

Näin makro-ohjelmointikoulutusta päästiin kokeilemaan niin nuorille kuin aikuisillekin, sekä alan ammattilaisille. Samalla saatiin hyvää ja arvokasta käytännön kokemusta makrojen käytöstä konepajassa.

LÄHTEET

Kirjat

[1] Ritakallio, Topi & Kivinen, Eero

Tekninen tiedotus 33/84 NC-koulutuspaketti. Helsinki: Metalliteollisuuden kustannus Oy, 1984, s. 1–3

[2] Maaranen, Keijo

Koneistustekniikat. Helsinki: Werner Söderström Osakeyhtiö, 2004, s. 248–251,262, 264–268

[3] Vesamäki, Hannu

Lastuavan työstön NC-ohjelmointi. Helsinki: Teknologia teollisuus ry, 2007 s.105, 189–191

[4] FANUC Series 30i/300i/300is-MODEL A, Yhteinen sorvausjärjestelmien ja koneistuskeskusten käyttäjän käsikirja. B-63944FI 1/02

[5] FANUC Series 16/18/160/180-MC, OPERATOR'S MANUAL B62764EN/01

[6] Laaksonen, Jyrki (Fanuc makro-ohjelmointi kurssi 24.10.2001 AEL)

[7] Koponen, Visa

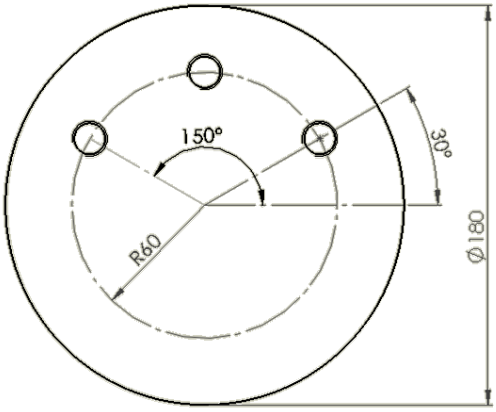
Oppimisen hallinta konepajoissa, Turun ammattikorkeakoulu 2009 s. 27, 28, 44, 48

[8] Microsoft Oy

http://windows.microsoft.com/fi-Fi/windows/history [viitattu 12.05.2011]

Liite 1: Reikäpiirimakron esimerkki

REIKÄPIIRIMAKRON KÄYTTÖESIMERKKI

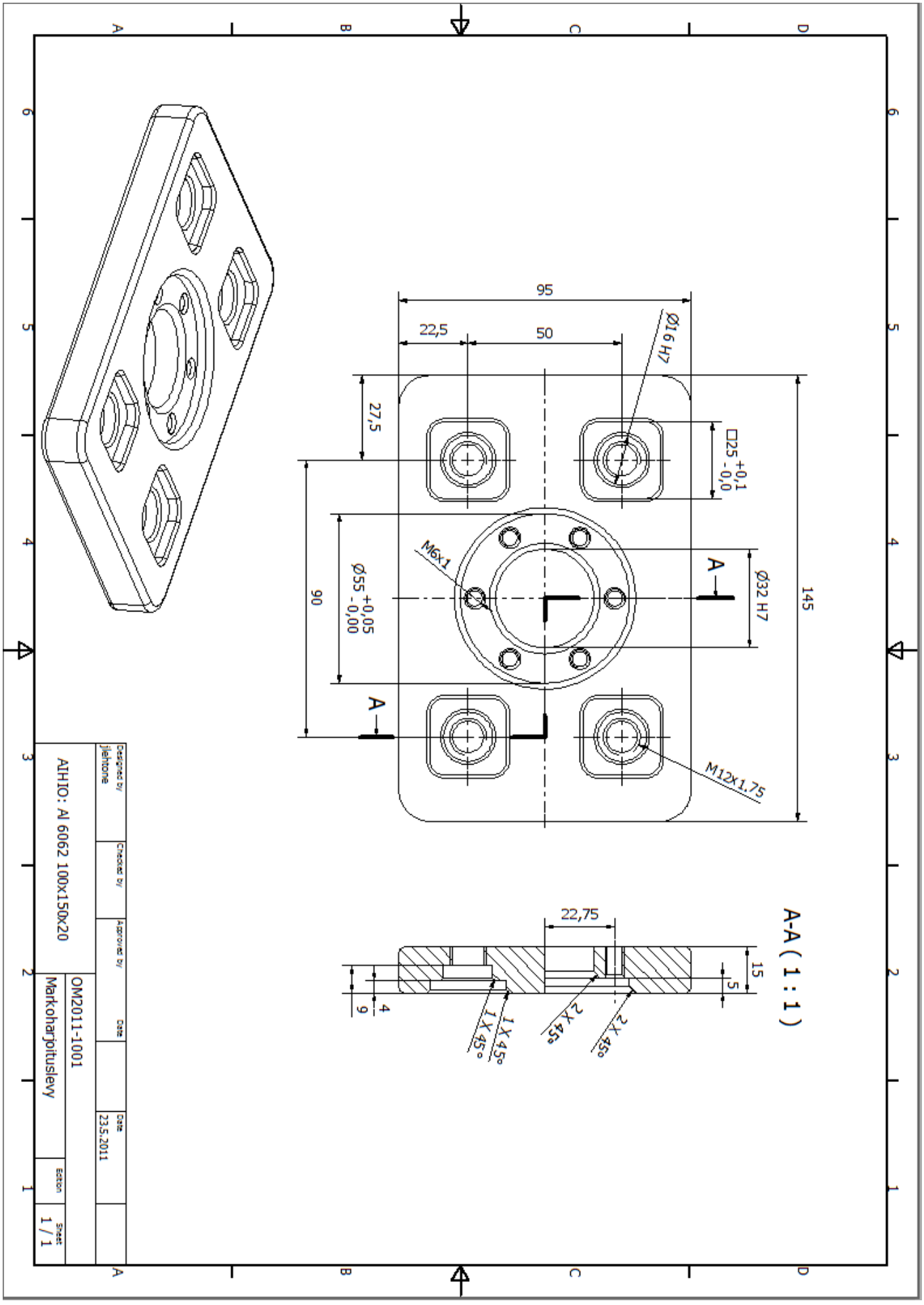



```
O1010 REIKÄPIIRI PORAUS
G90 G80 G49 G40 G17
T1 M6 (KESKIOPORA)
G54 G0 X0 Y0 S1500 M3
G43 Z20. H1 M8
G98 G81 Z-3. R2. L0 F100
(L0 JOS EI HALUTA PORAUSTA
REIKÄPIIRIN KESKELLE.)
G65 P8100 A30 B150 C3 R60.
(MAKRO-OHJELMAN O8100 KUTSU)
G80 G0 Z50. M9
G91 G28 Y0. Z0. M5
M1
T2 M6 (PORA D14)
G54 G90 GO X0. Y0. S550 M3
G43 Z20. H2   M8
G98 G73 Z-35. Q5. R2. F110 L0
G65 P8100 A30 B150 C3 R60.
G80 G0 Z50. M9
G91 G28 Y0. Z0. M5
M1
T3 M6 (SENKKAUS TERA D20.5)
G54 G90 GO X0. Y0. S700 M3
G43 Z5. H3 M8
G98 G82 Z-6.75 R2. F100 P0.5
G65 P8100 A30 B150 C3 R60.
G80 G0 Z50. M9
G91 G28 Y0. Z0. M5
M1
T4 M6 (KIERRETAPPI M16)
G54 G90 GO X0. Y0. S200 M3
G43 Z30. H4 M8
G98 G84 Z-25. R8. F400 L0
G65 P8100 A30 B150 C3 R60.
G80 G0 Z50. M9
G91 G28 Y0. Z0. M5
```

Liite 1: Reikäpiirimakron esimerkki

Makro-ohjelma	Selitykset
O8100 (REIKAPIIRI)	Huom! Kommentit suluissa ja ilman ä ja/tai ö kirjaimia.
#130=#5001	X-koordinaatin asema tallennetaan muuttujaan #130.
#131=#5002	Y-koordinaatin asema tallennetaan muuttujaan #131.
#132=1	Laskuri, asetetaan arvoon 1.
#136=360	Muuttujalle #136 annetaan arvo 360, eli täysi ympyrä.
IF [#2 EQ 0] GOTO10	Jos muuttujan #2 arvo = 0, niin silloin siirrytään riville numero 10. Mikäli muuttujan #2 arvo on ≠ 0, siirrytään suoraan seuraavalle riville. Jos argumentti B jätetään tyhjäksi, sen arvo on nolla.
#136=#2	Jos argumentille B on annettu jokin arvo, niin se tallennetaan muuttujaan #136. Muussa tapauksessa tämä lause jää huomioimatta. Vertaa edelliseen.
N10 #133 = [#136-#2] / [#3-1]	Jakokulmalaskuri, eli loppukulmasta vähennetään aloituskulma ja saatu tulos jaetaan reikien kokonaismäärä #3 vähennettynä yhdellä.
WHILE [#132 LE [#6]] DO1	Silmukka DO1 ja EDN1 välillä olevaa ohjelmaa toteutetaan niin kauan kuin muuttujan #132 arvo on ≤ muuttujan #6. Kun #132 arvo on ≥ kuin #6 arvo, niin silloin siirrytään END1 lausetta seuraavalle riville.
#134 = #1 + [#132 - 1] * #133	Laskuri. Muuttujaan #134 tallentuu aina seuraavaksi porattavan reiän kulma. Kuinka mones reikä miinus yksi, saatu tulos kerrotaan jakokulmalla ja summa lisätään alkukulmaan.
#101 = #130 + #18 * COS [#134]	Tämä laskutoimitus laskee X-akselin suuntaisen siirtymän suuruuden, käyttäen edellä saatua kulma-arvoa. Saatu tulos summataan aikaisemmin muuttujaan #130 tallennettuun X-koordinaatin arvoon.
#102 = #131 + #18 * SIN [#134]	Tämä laskutoimitus laskee Y-akselin suuntaisen siirtymän suuruuden, käyttäen edellä saatua kulma-arvoa. Saatu tulos summataan aikaisemmin muuttujaan #131 tallennettuun Y-koordinaatin arvoon.
X#101 Y#102	Tässä lauseessa työkalu siirtyy uusiin, edellä laskettuihin X- ja Y-koordinaattiarvoihin ja pääohjelmassa annettu porauskäsky toteutuu.
#132 = #132 + 1	Laskuri joka lisää aikaisempaan muuttujan #132 arvoon aina yhden. Kun silmukka palaa takaisin WHILE komentoon, niin tätä arvoa verrataan haluttujen reikien määrään, siis muuttujaan #6.
END1	Silmukan loppukohta. Tästä kohtaa hypätään takaisin WHILE komentoon.
M99	Makro-ohjelmasta palataan takaisin pääohjelmaan.

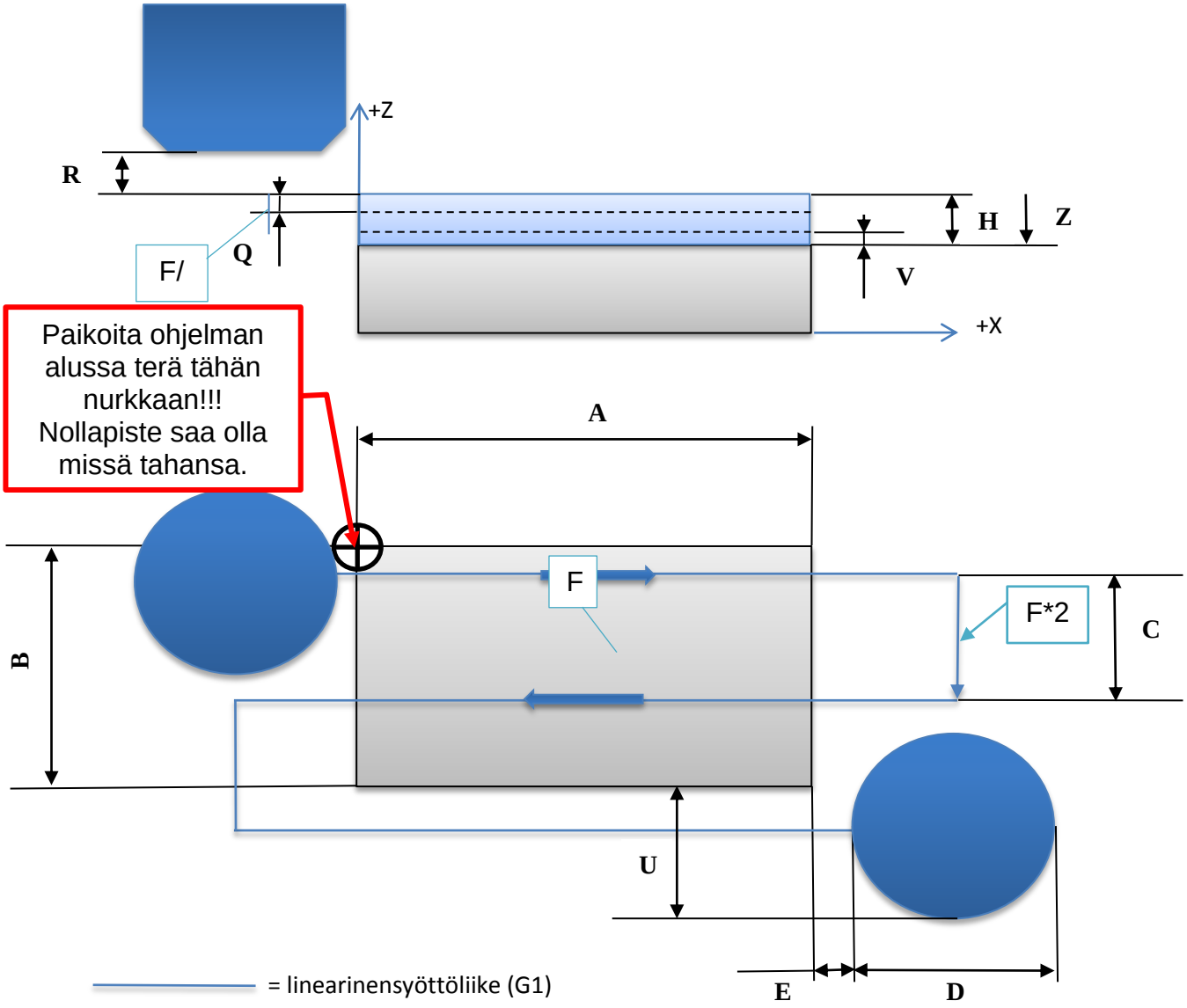
Liite 2. Makroharjoituslevy



Liite 3. PLAANAUS MAKRO ZIGZAG-MENETELMÄLLÄ

O8020

G65 P8020 A__ B__ C__ D__ E__ F__ H__ Q__ R__ V__ Z__



OHJEET SEURAAVALLA SIVULLA

O8020

PLAANAUS MAKRO ZIGZAG-MENETELMÄLLÄ

G65 P8020 A__ B__ C__ D__ E__ F__ H__ Q__ R__ V__ Z__

Paikoita terä vasempaan yläkulmaan, haluamallesi turvaetäisyydelle.

- A → #1 = Tason leveys X-suunnassa
- B → #2 = Tason leveys Y-suunnassa
- C → #3 = Terästä käytettävä leveys prosentteina
- D → #7 = Terän halkaisija
- E → #8 = Sivuturvaetäisyys
- F → #9 = Syöttö
- H → #10 = Poistettava määrä Z-suunnassa
- Q → #17 = Rouhintalastun paksuus
- R → #18 = Lähestymis- ja paluunpiste Z-suunnassa
- U → #21 = Ulkoreunan ylitysmatkan minimi
- V → #22 = Viimeistelylastun paksuus
- Z → #26 = Z-akselin lopullinen syvyys, jos arvo on 0, niin sitä ei tarvitse määritellä

Liite 3. PLAANAUS MAKRO ZIGZAG-MENETELMÄLLÄ

```
%

O1234

G90 G80 G49 G40 G17

M6 T1 (63MM OTSAJYRSIN)

G90 G54 G0 X0 Y0 S900 M3

G43 Z20. H1 M8

G65 P8100 A200. B100. C70 D63. E5. F200 H5. Q2. R7. U5. V0.2 Z0

G90 G0 Z50. M9

G91 G28 Z0 M5

G28 Y0

M30


O8100 (PLAANAUS MAKRO)

IF [#1LE0 ]GOTO1001

IF [#2LE0 ]GOTO1002

IF [#3LE0 ]GOTO1003

IF [#7LE0 ]GOTO1007

IF [#8LE0 ]GOTO1008

IF [#18LT#26+#11 ]GOTO1018

#101 =#5001(X-AKSELIN PAIKKA TALTEEN)

#102 =#5002(Y-AKSELIN PAIKKA TALTEEN)

#103 =#5003(Z-AKSELIN PAIKKA TALTEEN)

#104 =#4001(G-KOODI TALTEEN- RYHMA 1)

#105 =#4003(ABS. / INKR. TOIMINTO TALTEEN)

#111 =#7/2(TERAN SADE)

#112 =[#3/100]*#7(SIVUSIIRTYMAN SUURUUS)

#113 =#8+#111+ABS [#101](TURVAVALI + TERAN SADE + X-AKS LAHDON ITSEISARVO)

#114 =[#8*2]+#7+#1(TURVAVALI*2 + HALKAISIJA +PKL:N LEVEYS = X:N SIIRTYMA)

#115 =[#102-#112+#111](Y-AKS LAHDON ARVO - SIVUSIIRTYMA + TERAN SADE)

#116 =-1*[#2+#21-#111](Y:N ALIN KOORDINAATTI)
```

Liite 3. PLAANAUS MAKRO ZIGZAG-MENETELMÄLLÄ

```
IF [#11LT#17 ]THEN#17 =#11

#117 =FUP [#22]

#118 =#17

#120 =#11

#121 =#17

#122 =#22

IF [#11EQ0 ]THEN#120 =1

IF [#17EQ0 ]THEN#121 =1

IF [#22EQ0 ]THEN#122 =1

#124 =[#26+#11](Z-TASO ENNEN PLAANAUSTA)

#125 =[#123+#22](KPL:EN PINNAN TASO ENNEN VIIMEISTELYA)

#130 =FUP [[#120-#22]/#121](ROUHINTALASTUJEN LASKURI)

#131 =#130+#117(KOKONAISLASTUKERTOJEN LASKURI)

#132 =0 (LASKURIN NOLLAUS)

#132 =#132+1 (ALKUARVOKSI 1)

WHILE [#131GT0 ]DO1

(MIKALI ANNETTU EHTO ON TOSI, NIIN SUORITETAAN OPERAATIOT DO1 JA END 1
VALILLA)

#141 =#124-[#17*#132](LASTU SYVYYDEN ASETUS)

IF [#141LT#125 ]THEN#141 =#125

IF [#132EQ#131 ]THEN#141 =#26

N10 G90 G0 X-#113 Y#115(SIIRTO ABS TYOSTON LAHTO KOHTAAN)

G90 Z#18(SIIRTO ABS LAHESTYMISTASOLLE)

G1 Z#141 F [#9/2]

G91 X#114 F#9

WHILE [#115GT#116 ]DO2

(JOS EHTO TOTEUTUU, NIIN SUORITETAAN OPERAATIOT DO2 JA END 2 VALILLA)

G91 Y-#112 F [#9*2]

G90 X-#113 F#9

#115 =#115-#112
```

Liite 3. PLAANAUS MAKRO ZIGZAG-MENETELMÄLLÄ

```
WHILE [#115GT#116 ]DO3
(JOS EHTO TOTEUTUU, NIIN SUORITETAAN OPERAATIOT DO3 JA END 3 VALILLA)
G91 Y-#112 F [#9*2]
#115 =#115-#112
X#114 F#9
END2
END3
N20 G90 G0 Z#18
#131 =#131-1
#132 =#132+1
END1
G90 G0 Z#113 (Z-AKSELI PALAA TURVATASOLLE)
N1001#3000 =2(KAPPALEEN LEVEYS PUUTTUU)
N1002#3000 =2(KAPPALEEN KORKEUS PUUTTUU)
N1003#3000 =3(SIVUSIIRRON SUURUUS PUUTTUU)
N1007#3000 =7(TERAN HALKAISIJA PUUTTUU)
N1008#3000 =8(TERA TORMAA ALASMENOSSA)
N1018#3000 =18([TURVA TASO ON PINNAN ALAPUOLELLA)
G#104 G#105 F#9
N100 M99
%
```