USBCNC

User Manual

Document Release 3.52

Document History

Version	Date	Author	Comment
1	2006-03-10	Bert Eding	Initial version
1.01	2006-08-13	Bert Eding	Added I/O info and language extensions
1.02	2006-08-19	Bert Eding	Added hardware descriptions
1.03	2006-08-22	Bert Eding	Added backlash compensation description.
1.04	2006-08-31	Bert Eding	Added new (graph) interface description and work tutorial.
1.05	2006-09-02	Bert Eding	Added hint for manual homing
1.06	2006-09-05	Bert Eding	Added setup parameter A-Axis Linear
1.07	2006-09-12	Bert Eding	Added HPGL import and configurable output inversions. All outputs are now set to zero at startup and at shutdown.
1.08	2006-10-07	Bert Eding	Added DXF EasyCAM
1.09	2006-10-19	Bert Eding	Corrected mistake wit external supply at CPU_2
2.10	2006-12-05	Bert Eding	Changed GUI with respect to homing and zeroing.
2.11	2007-01-22	Bert Eding	Introduction of USBCNC_CPU_3
2.12	2007-04-04	Bert Eding	Remove unused reference, add description of G61/G64.
2.12	2007-05-23	Bert Eding	Minor correction about homing.
2.13	2007-05-28	Bert Eding	Changed install description, added Vista32 and Vista64 AMD install.
2.14	2007-11-10	Bert Eding	New release of USBCNC V2.62, new M functions to control all outputs from the interpreter, PWM control for spindle.
2.15	2007-12-28	Bert Eding	Added description of USBCNC_CPU_4
3.00	2008-01-30	Bert Eding	Modified GUI descriptions for V3.XX
3.01	2008-02-26	Bert Eding	Small corrections in hardware connection examples.
3.02	2008-04-12	Bert Eding	Implemented line simplification, merges small G1's to one bigger with specified accuracy. The same accuracy is also used in the blending algorithm. Partially blending instead of full

			blending now possible. The behavior is similar to that of the EMC2 project, however, I have implemented this myself in the EMC1 interpreter and also implemented a new modified trajectory generator. Next step will be Look Ahead Feed, later. See chapter: Set Path Control Mode - G61, and G64, or G64 Px
3.14	2008-06-08	Bert Eding	Manual version matched to SW version. Initial support for Vista drivers added. (This requires a firmware upgrade for everybody who uses the Virtual COMM port driver now). The drive installation instructions are updated. Initial support for **automatic tool** change added via macro.cnc file. See sections 2.4.1 and 2.4.2 and 4.3.
3.16	2008-07-30	Bert Eding	Look Ahead Feed added.
3.20	2008-07-30	Bert Eding	Small updates for V3.20
3.21	2008-07-30	Bert Eding	Small updates for V3.21
3.22	2008-07-30	Bert Eding	Small updates for V3.22, driver install
3.26	2008-10-25	Bert Eding	Made consistent with V3.26
3.27	2008-10-25	Bert Eding	Tool data stored into user variables.
3.28	2008-10-25	Bert Eding	Added interpreter commands. Added Advantronix GPIO card support for extra IO. Fully configurable XYZABC axes. A-axis various calculation options Rotational, linear for foam cutting and 4th axis milling. Graphic display and jog keys for lathe when X/Z is selected as first two axes.
3.30	2009-1-2	Bert Eding	Basic Lathe/Turning functions added - Added X offset to tool table - Added Orientation to tool table Tool radius compensation G41, G41D.. and G42, G42D.. works in G18 plane for Lathe's and tool orientation is taken into account. G41.1D..L.. , G42.1D..L.. where D is the tool diameter and L is the tool orientation (1..9) Tool length compensation: G43, G43H.. takes both tool Z offset and tool X offset into account. G43.1 I.. K.. where I is tool X-Offset and K is tool Z offset. Not yet implemented: thread cutting functions G33 and G76 thread cutting and also spindle functions G96.
3.34	2009-2-1	Bert Eding	Added JOGPAD description and G41.1/G41.2 plus examples.
3.36	2009-2-12	Bert Eding	Added more description to LAF.
3.42	2009-09-20	Bert Eding	Added G33, G76 for turning. Note that turning required special firmware. Contact me if you want to do turning.

3.42.1	2009-09-20	Bert Eding	Modified some textual mistakes.
3.43	2009-09-20	Bert Eding	New USB drivers, compatible with XP, VISTA and Windows 7
3.44	2009-11-14	Bert Eding	Updated thread cutting + driver installation
3.46	2010-01-03	Bert Eding	Updated thread cutting + driver installation
3.47	2010-01-10	Bert Eding	Updated driver installation
3.48	2010-01-11	Bert Eding	Updated driver installation for 64 Bit Windows
3.49	2010-04-09	Bert Eding	Updated for V3.49, Tandem axes, Tangential knife, user buttons and more.
3.50	2010-11-21	Bert Eding	Updated for 3.50 - CPU5 support with USB and Ethernet. - long and super long file modes allowing files up to 4G containing 100M lines of G-Code. - Machine and Tool change Area collision guarding. - Job time estimation and progress monitor. - Improved tan knife - A few other Features and improvements.
3.50.5	2010-11-23	Bert Eding	Added description of User 1 button and User 2 button. Updated for up to software version V3.50.5
3.51	2011-01-26	Bert Eding	Added X down tool length. Added description for activation (CPUOPT) of 4th axis on CPU 5A. Added spindle settings. Added Icon directory setting. Added Macro file name setting.
3.51.1	2011-02-17	Bert Eding	Added description of reset button. Added return value for dlgmsg function.
3.52	2011-03-27	Bert Eding	Added new functions for major release V3.52 New hand wheel settings and behavior.

ACKNOWLEDGEMENTS

The G-Code part of this user manual has been derived from the full report of the RS274/NGC language. Parts that are less relevant to USBCNC users or parts that are not supported are left out. The original report has the following title:

The NIST RS274NGC Interpreter -Version 3

Thomas R. Kramer

Frederick M.Proctor

Elena Messina

Intelligent Systems Division National Institute of Standards and Technology Administration
U.S. Department of Commerce Gaithersburg, Maryland 20899

NISTIR 6556August 17, 2000

On the issue date of this user manual (020225) the report was available on the World Wide Web using the URL:

http://www.linuxcnc.org/handbook/RS274NGC_3/RS274NGC_3TOC.html

Table of contents

1 Introduction

This user manual is intended to be useful to machine operators running machining centers with USBCNC. USBCNC uses the RS274/NGC interpreter of the EMC project before it was GPLed.

The RS274/NGC Interpreter (the Interpreter) is a software system that reads numerical control code in the "NGC" (Next Generation Controller) dialect of the RS274 numerical control language.

I extended the language with constructs that allow programming inside the language, see the language extensions chapter.

1.1 BACKGROUND

1.1.1 Machine context for USBCNC

1.1.1.1 USBCNC HARDWARE

The descriptions of USBCNC CPU's can be found on the website of Eding CNC.
Go to the download page of www.edingcnc.com and download de CPU flyer you need.
General explanation and connection examples are provided in next chapters about the I/O.

1.1.1.2 THE AXES OUTPUTS

A maximum of 6 axes named X, Y, Z, A, B or C can be controlled by USBCNC software. X, Y, Z are the linear axes and A, B, C are rotation axes usually.

The schematic here shows the configuration of the motor control outputs.	
All CPU's have a direction and step output for each axis.	
Some boards also have an amplifier enable output for each axis.	
Additionally, there is the GND and +5V connection.	
The motor outputs may source or sink a maximum current of 50mA together or 17 mA per output.	
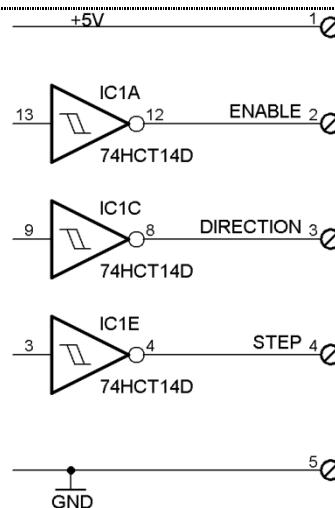
This is the way to connect CPU4 or CPU5B

To a LEADSHINE type of drive or compatible.

CPU5 B standard delivers a negative step-pulse of approx 3uS at 125 KHz.
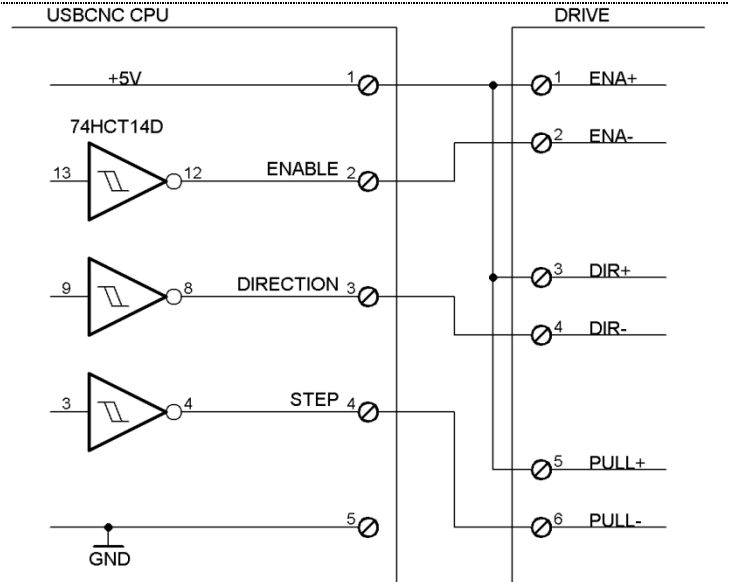
This is the way to connect CPU5A

To a LEADSHINE type of drive or compatible.

CPU5 A can be configured to give a positive step-pulse of approx 3uS at 125 KHz.

CPU5A has no enable output per axis. To enable/disable the drives we can use the WATCHDOG output.

The LEADSHINE drives are enabled when there is no current at ENA+/-, so when leaving the drive ENA+/- unconnected it is always enabled.
This is also a possibility.
For heavy Z axes that fell downwards due to gravity, this might be a safer solution.
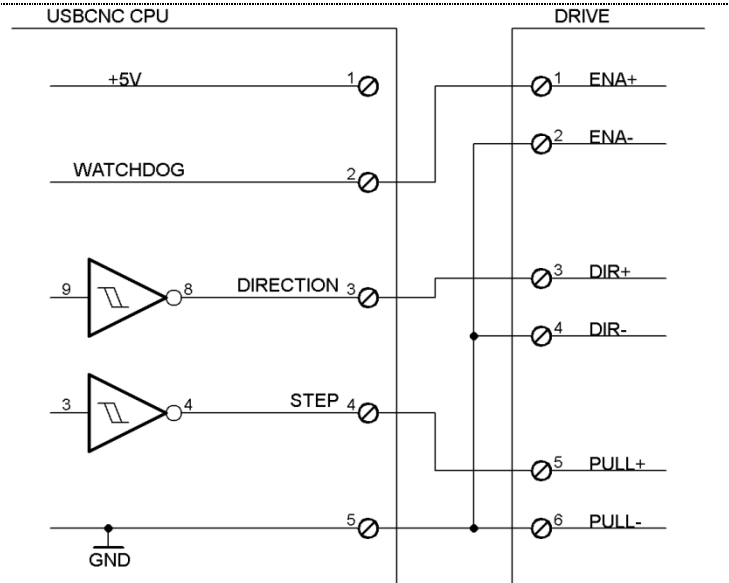
1.1.1.3 STANDARD CNC OUTPUTS

The standard CNC outputs are the ones coupled to the standard M functions:

M3/M4:   Spindle on and spindle direction.
M5:   Spindle off
M7:   Mist coolant on
M8:   Flood coolant on
M9:   Both coolants off
M80/81:   Amplifier on/off (USBCNC specific)

OPEN COLLECTOR EXPLANATION
Connection point 2 here is the actual output.
It is connected to the collector of transistor T1 and further to nothing.
That is why this type of output is called open collector.
All CNC outputs of the USBCNC CPU's have this type of output.

The transistor T1 behaves like a switch between point 2 and 3. The switch is controlled by the microcontroller on the board.
Sometimes I get the question, why do I not measure anything at the output?

I hope this schematic clarifies this question, when you put a voltmeter between GND and the output, you will measure in the air if the switch is open. When the switch is closed, you will measure a voltage near zero volt.

If you put a resistor (1k .. 10k) between the output 2 here and the +5, then you will measure correctly.

OPEN COLLECTOR OUTPUT
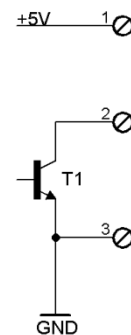
OPEN COLLECTOR TO SOLIDSTATE RELAY

This example shows how to control a solid state relay from an open collector output.

The solid state relay has a LED on the input which is to be switched on and off. A LED needs a current limiting resistor, R1 here. Sometimes the resistor is inside the solid state relay, sometimes you have to add it externally.

Always read the description of the used solid state relay.

OPEN COLLECTOR OUTPUT
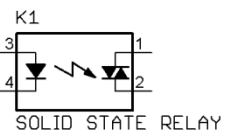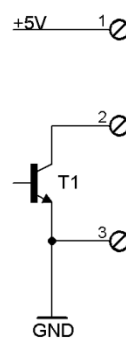
OPEN COLLECTOR TO NORMAL RELAY

This example shows how to control a solid normal relay from an open collector output.

We use a 24 Volt relay here, so we need an external 24V supply.

Note that there is a diode connected anti-parallel to the relay. This is needed because a normal relay can generate huge voltages when switched off. Without the diode T1 on the CPU and also the microcontroller can get damaged. **So never forget the diode!**.

Note that the diode is a fast diode in the order of 100-200Volt, 1 – 2 ampere.
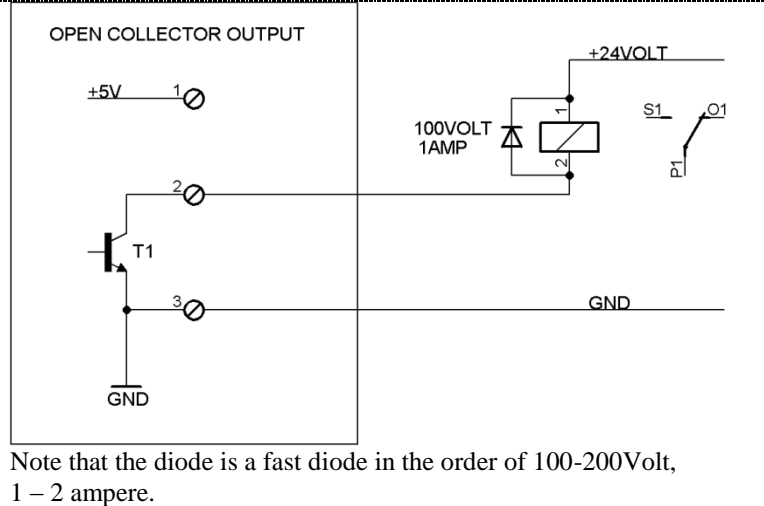
1.1.1.4 STANDARD CNC INPUTS

Home [x..c]|:   the home <axis> command uses the home input of the axis.

G38.2:   probing, uses the probe input

EStop: Uses the EStop input

Inputs of the CPU

Are configured like the schematic here:

Important to know, the input has a pull-up resistor of 10K to +5V. The input is filtered and also connected to an input buffer with hysteresis. This all helps to prevent glitches on the.
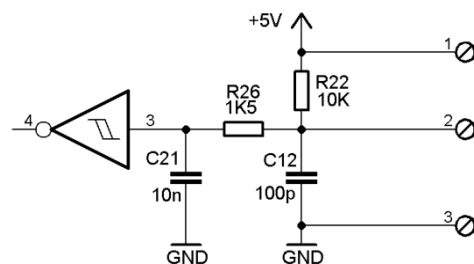
Connecting a switch to an input

A switch can be connected between the input and GND. This apply's to al switches, home, E-Stop etc.

The switch can be normally open (as drawn) or normally closed.

'
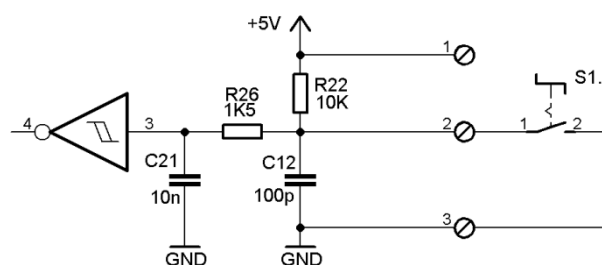
| Connecting a sensor

This example applies to home sensors that use an external supply and have an NPN open collector output.

Read the user manual of the sensor carefully. | |
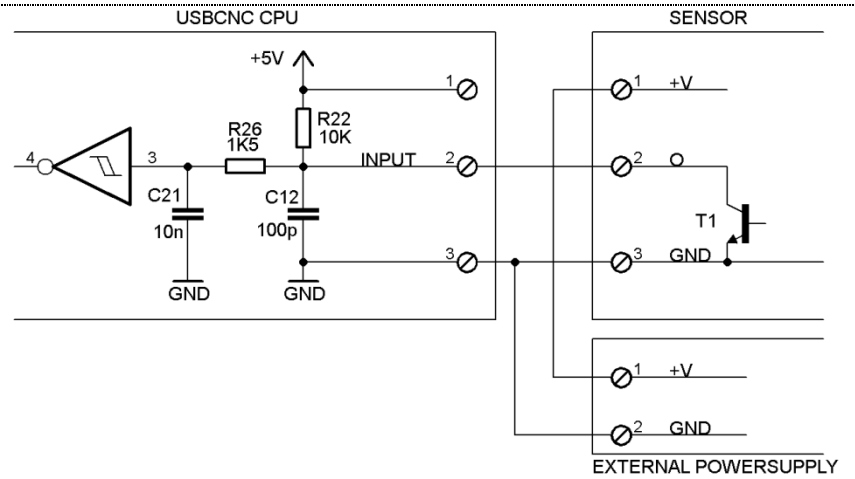
1.1.1.5 EXTRA INPUTS AND OUTPUTS

It depends on the CPU variant, which extra I/O there is available. The extra or AUX IO can be used in wt M functions M54 M55 and M56.

1.1.2 Numerical Control Programming Language RS274

RS274 is a programming language for numerically controlled (NC) machine tools, which has been used for many years. The most recent standard version of RS274 is RS274-D, which was completed in 1979. It is described in the document "EIA Standard EIA-274-D" by the Electronic Industries Association [EIA] (see page 4-1). Most NC machine tools can be run using programs written in RS274. Implementations of the language differ from machine to machine, however, and a program that runs on one machine probably will not run on one from a different maker.

1.1.2 The RS274/NGC Language

The NGC architecture has many independent parts, one of which is a specification for the RS274/NGC language, a numerical control code language for machining and turning centers. The specification was originally given in an August 24, 1992 report "RS274/NGC for the LOW END CONTROLLER -First Draft" [Allen-Bradley] (see page 4-1) prepared by the Allen-Bradley company. A second draft of that document was released in August 1994 by the National Center for Manufacturing Sciences under the name "The Next Generation Controller Part Programming Functional Specification (RS-274/NGC)" [NCMS] (see page 4-1). All references in this user manual are to the second draft. The RS274/NGC language has many capabilities beyond those of RS274-D.

1.2 DEFINITIONS, ACRONYMS AND ABBREVIATIONS

| CNC | Computerized Numerical Control |

CPU	Central Processor Unit, a PCB board with a Processor on it.
DXF	Drawing Exchange Format) is a CAD data file format developed by Autodesk
FIFO	First In First Out Buffer
HPGL	Hewlet Packard Graphical Language
GUI/UI	Graphical User Interface
G-Code	CNC language
INTERPRETER	A software function that is able to read a text file and execute the commands contained therein.
JOBFILE	A *job* is the text file (G code) that will be executed by the interpreter.
GUI	Graphical User Interface.
PWM	Pulse Width Modulation

1.3 MINIMUM PC REQUIREMENTS

- 1.4 GHz Atom.
- Pentium, duo-core recommended for Ethernet.
- 1024 MB RAM for XP, 4G for Windows 7.
- Windows XP or Windows 7, 32 or 64 bit.
- Minimum Screen resolution 1024 x 768.
- USB-2 connection / Ethernet connection for Ethernet CPU's
- Intel 100Mbit Ethernet card for Ethernet CPU's.

Windows XP and Windows 7 is proven to work fine with USBCNC. Windows Vista is not.

USBCNC requires soft-real-time behavior of your PC. Sometimes a bad driver of your video-card, sound etc may be the cause of problems with USBCNC. USBCNC requires a USB communication speed of about 150 times/second to and from the USBCNC CPU.
There are PC's with bad USB chipset on which cannot handle this. One of such PC's is the ACER notebook time-line series. From the better brand notebooks like Dell, HP, Sony, Toshiba I have not yet heard problems. When you encounter this, you may be able to solve it by adding a PCI USB card.

1.4 INSTALLATION OF USBCNC

Download the installation executable from the website download page. Click on it to install the software. Follow the screens. On Windows 7, click with the right mouse button, start as administrator.

After installation reboot the PC, when it is rebooted connect the CPU, after 10 – 60 seconds, you will see that windows has found an USBCNC COM port.
You will have the USBCNC icon on your desktop:

USBCNC must be started as Administrator. This can be done by right clicking the mouse on the USBCNC Icon and then select run as Administrator on Windows 7. On XP this is generally not needed because most of the time you are already Administrator.
To avoid having to do this every time, you can change the properties of the ICON by right clicking on the ICON and set the check "Run this program as Administrator".
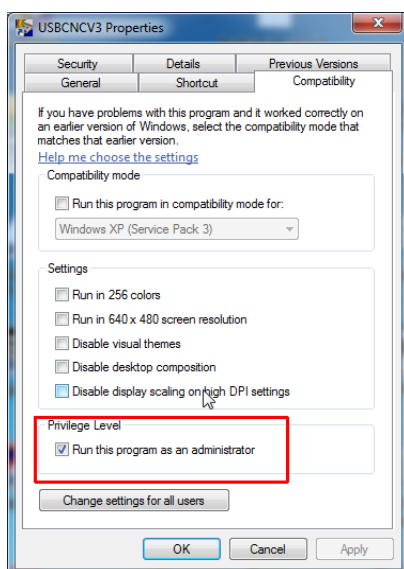
You can start USBCNC now by clicking the shortcut. When everything is OK, you will see a flashing led on the CPU board, meaning that the PC software and the CPU are communicating.

2 The user interface

There are several views:
Operate , Program, Tools, Variables, Setup and Help.
Using control-tab, you can tab through them.

It is important that USBCNC is started started as administrator. On windows 7 this is not
automatically done like in XP.
Click on the right mouse button and select "Run As administrator". You can also set this in the ICON
properties, compatibility.

When you start USBCNC for the first time you will get the Terms/Guarantee page click the language
you read the text. Then click agree if you agree. The operate page is shown. This is the main screen
to do all machine operation's from. But first some settings must be filled in before you can start, so go
to the setup page.

**Running a program and fast jogging is only possible after the machine is correctly homed, so
this must be setup first. The reason is that collision prevention is not active when the machine
isn't homed, so damage to the machine may happen when homing is not performed.**

2.1 SETUP PAGE'S

Before the system is actually used we have to setup the system to accommodate the machine, we do that in the setup page's. There a two main setup pages:

Setup Page 1:
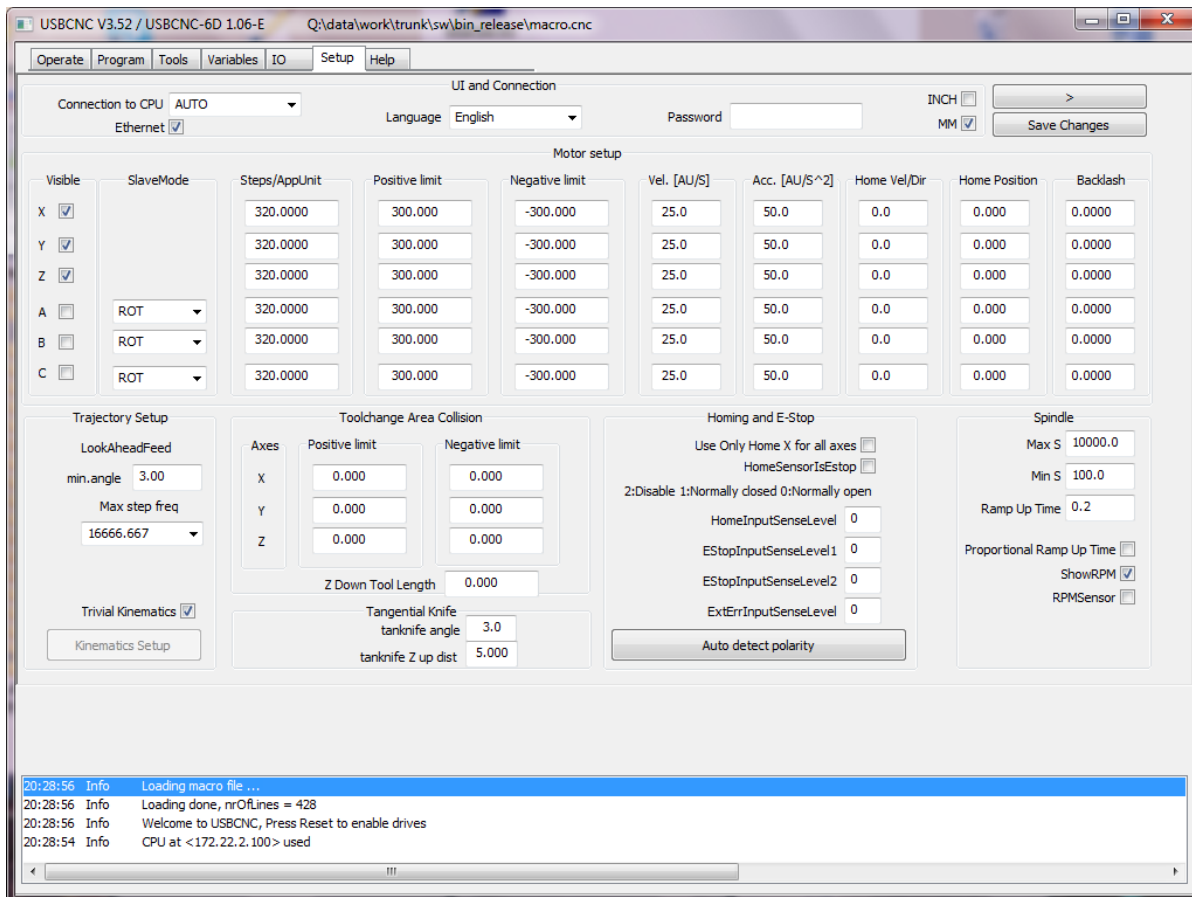
2.1.1 UI and Connection

Connection to CPU: If you have 1 board connected to your PC, leave the setting at AUTO, the software will find the board automatically. Otherwise choose the here the CPU you want to work with. For CPU's with USB, you see the COMx ports here, in case of a CPU5 with Ethernet, you will see the IP-Address here.

If you have a CPU with Ethernet, check the Ethernet checkbox.

Language setup: Speaks for itself. After it is set, save the changes, then close USBCNC and restart so that everything will be in the correct language. The translations are in 2 files, cncgui-lang.txt and cncserver-lang.txt, if you find translation mistakes you can correct this here. Please send the corrected file to Eding CNC, the corrections will then be incorporated into new versions.

Password: You can protect the setup parameters from being modified by unauthorized persons by using a password. Leave empty if no password is desired.

| **INCH:** | Machine setup is in inch mode. |
| **MM:** | Machine setup is in mm mode. |

2.1.2   Motor setup

Visible: Check if the axis should be visible in the GUI.

Mode :   Select mode for rotation axes, slave or special function:

❖ **ROT**, default, axis behaves as a normal rotation axis.
❖ **SLAVE** X, SLAVE Y or SLAVE Z axis is slave of X or Y or Z axes, for Gantry machines with two independent (Tandem) motors on the main axes. See also the Homing chapter for details on Slave axes.
❖ **FOAM CUT** for A-Axis, if used as a Foam cutter with 4 linear axes. X is the left horizontal axis, Y is the left vertical axis, A is the right horizontal axis and Z is the right vertical axis. Feed calculation are based on the X/Y or A/Z combination which ever makes the biggest distance,
❖ **4th MILL**, if used in 4 axes milling. Feed calculations are optimized such that the tooltip gets the correct speed relative to the material.
❖ **Tangential Knife**, this option is available for the C-Axis only. The Knife will rotate in the movement direction of X-Y. See also trajectory setup.

Steps/AppUnit :   Fill in number of steps per millimeter for millimeter mode or number of steps per inch for inch mode. Fill in a **negative number to reverse** the motor direction. Example: Suppose your driver is set to 1600 steps/revolution (1/8 micro step) and you have coupled the motor directly to a spindle with 5mm pitch. The number to be filled in here = 1600 / 5 = 320.
If the movement direction is wrong, change it to -320.

| Positive limit: | Maximum machine position. |
| Negative limit: | Minimum machine position; |

Vel :   Maximum axis velocity, all velocities, whether jogging, G0/G1/G2/G3 are limited to this value.
Acc: Maximum acceleration.

2.1.3   Homing and ESTOP setup

Home Vel/Dir: Homing velocity, **sign** determines homing **direction**.

When the velocity is set to 0, the axis is homed **manually**, see the homing and coordinate systems chapter.

Home Position: Machine position at the moment the home switch activated. This determines the machine coordinates. It is not really relevant where the machine zero point lies, it should only match with the MIN/MAX position.

Homing sensors should be setup such that they remain active until the mechanical end of the machine. The space from home sensor activation to mechanical end is required to ramp down the movement.

Machine Mechanical Range

Home Sensor behavior, good

Home Sensor behavior, WRONG!
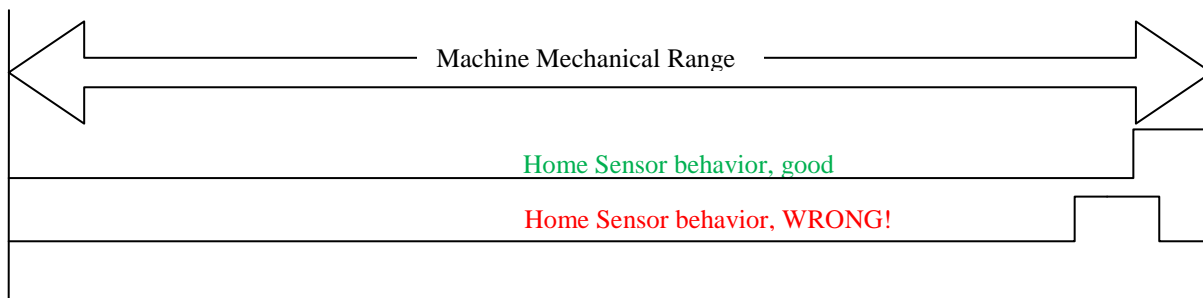
Use only home X for all axes:
Check this option if you have all home sensors wired to one input.

HomeSensorIsEStop:
The home sensors can also be used as limit switch which generate an E-Stop when activated. When this function is required, the sensors should be mounted outside the normal machine area.
Check this option if the home sensors work as EStop when activated. This option will work after homing is complete. The reason is that otherwise homing itself will generate an E-Stop.

EStopInputSenseLevel:
use this if you have an external emergency button connected.

HomeInputSenseLevel:
Defines EStop input behavior,
0 = low active (normally open switch),
1 = high active, (normally closed switch).

set the level of your end of stroke switches, these are used for homing the machine. First check that the home sensors or switches are working, activate them and look at the home-LED's at the lower left side of the main Operate screen. If you see it working, take care that the machine axes are at the working area, so that none of the sensors are activated. Look at GUI "LEDs"

Home- x 🔴
Home- y 🔴
Home- z 🔴
Home- a 🔴
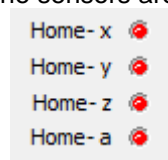
EStopInputSenseLevel1:

Defines EStop input behavior,

0 = low active (normally open switch),

1 = high active, (normally closed switch).

2 = OFF

EStopInputSenseLevel2:

Defines EStop input behavior for second EStop input (CPU5B only),

0 = low active (normally open switch),

1 = high active, (normally closed switch).

2 = OFF

ExtErrInputSenseLevel, CPU5B ONLY:

Defines External Error input behavior (CPU5B only),

0 = low active, e-stop (normally open switch),

1 = high active, e-stop (normally closed switch).

2 = OFF

3 = low active, smoothstop

4 = high active, smoothstop

With smoothstop the axes speed is ramped down, this means that there is no position loss.

The polarity settings for the home inputs, E-stop's and Extern error can be automatically determined by pressing the "Auto detect polarity" button.

2.1.4   Backlash setup

Backlash: Set the amount of backlash for each axis that the software should compensate. Experiment with velocities and acceleration, the backlash compensation demands more from your motors than without backlash compensation. Do not try to compensate more than 0.25 millimeters. If there is more backlash, try to reduce it mechanically first. The backlash compensation superimposes a second movement (the backlash) on top of the normal movement when the direction reverses. You can see the impact on the motion profile in the figures below. Here you can see the extra demands on the motors. Especially look at the extra acceleration that is caused by the backlash compensation. A non micro step drive in combination with a relative good motor may not be able to follow the profile.
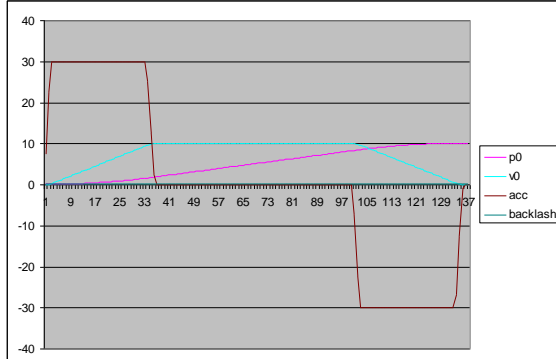
A movement from 0 to 10 mm without direction reversal. It is a beautiful 3rd order profile as you can see. The numbers on the X axis are 10 ms units, total move is about 1.37s.
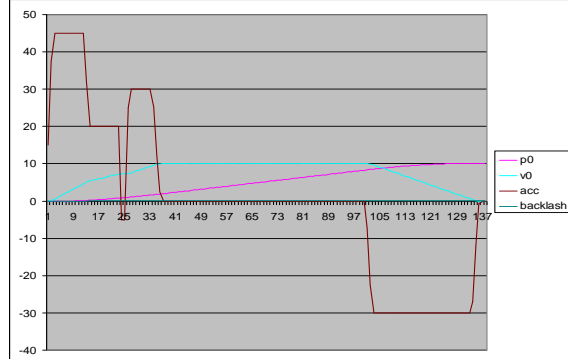
The next figure shows the same movement with direction reversal. The compensation value is 0.25 mm for this measurement, this is a relative large value.

2.1.5   Trajectory setup

MAXFREQ: The maximum step frequency that the CPU will generate. It is sometimes required to lower the maximum frequency, e.g. in case because the drive is unable to handle the high step rate. For instance, when you build the PICSTEP driver and want to use it, do not set the max frequency to 50 KHz, because the PICSTEP driver cannot handle frequencies above 50 KHz.

LAF minimum angle:
Look Ahead Feed calculations: Motion segments that are connected with a smaller angle as specified in **min.angle** will accelerate through which will give higher speeds especially with programs consisting of small motion segments. This is a unique feature which you don't find easily on low cost CNC controllers. Be carefully with the min.angle setting because this cause acceleration spikes, it depends on your machine and the speed up till what extend this is possible. I suggest performing tests with en check whether you get step pulse loss.
A value of 0.1 .. 3 degrees is generally safe. Segments that are really tangential connected will move fast that way.

An example of what I use:
When using CorelDraw, a circle is drawn of 100mm in diameter, and exported as HPGL CorelDraw generates small line segments of approximately 6 degrees. Now I have set the min.angle to 6, this gives the possibility to mill the circle with a speed of F6000 while without LAF the speed would be approx F1300 on my machine.

2.1.6   Kinematic Setup

Trivial kinematics: It is not needed for normal Cartesian machines, leave the Trivial 1:1 kinematics checked. Please contact Eding CNC if you have a special machine or robot with non Cartesian axes.

2.1.7   Tool change Area

XYZ Limits: By setting the limits here to a value different from zero, the TCA (Tool Change Area) guard will be activated. Using the values here you define an area on the machine which is restricted to tool change. A normal work piece program is not allowed to enter this area.

Z DownToolLength: For machine configurations where the tool chuck does not touch the machine bed when the machine is at its lowest Z position. Here you specify the tool length of the tool that fits when Z is at its lowest position.
This information is important for collision guarding.

2.1.8   Tangential knife setup

TanKnife Angle: Tangential Knife is a rotation motor (the C Axis) around Z. Tangential Knife works with normal G1, G2, G3 without tool-radius compensation G41, G42. The knife is rotated automatically in the direction of the X-Y move. This parameter determines the angle which 2 lines/Arcs can make without lifting the Z. If the angle is greater as this value, the Z will move up (G0), rotate the knife (G0), then move down again (G1). If the angle is lower, the rotation will take place without moving Z up.

TanKnife Z up distance:

        Specifies the distance to lift up Z when detected angle is greater than Tan Knife Angle.

2.1.9 Spindle and PWM setup

MaxS: The speed of your PWM controlled spindle when the PWM signal is at 100%.

MinS: The lowest possible speed for you spindle. If a command for a lower S values is
                       used, then this minimum value is applied.

Ramp up Time: The software waits this time between switching on the spindle and starting the
                       further machining.

Proportional Ramp up Time: Ramp up time is proportional with requested speed. Suppose your
                                 Maximum speed is 24000 and ramp up time 10 second. The a
                                 speed command of 12000 will give a ramp up time of 5 second.

ShowRPM: Check this when you want to see the RPM's, this works also if you have no RPM
                       sensor, a calculated value is displayed in this case.

RPMSensor: Check if you have connected a spindle speed sensor to the Sync input of the
                       CPU.

Setup Page 2:
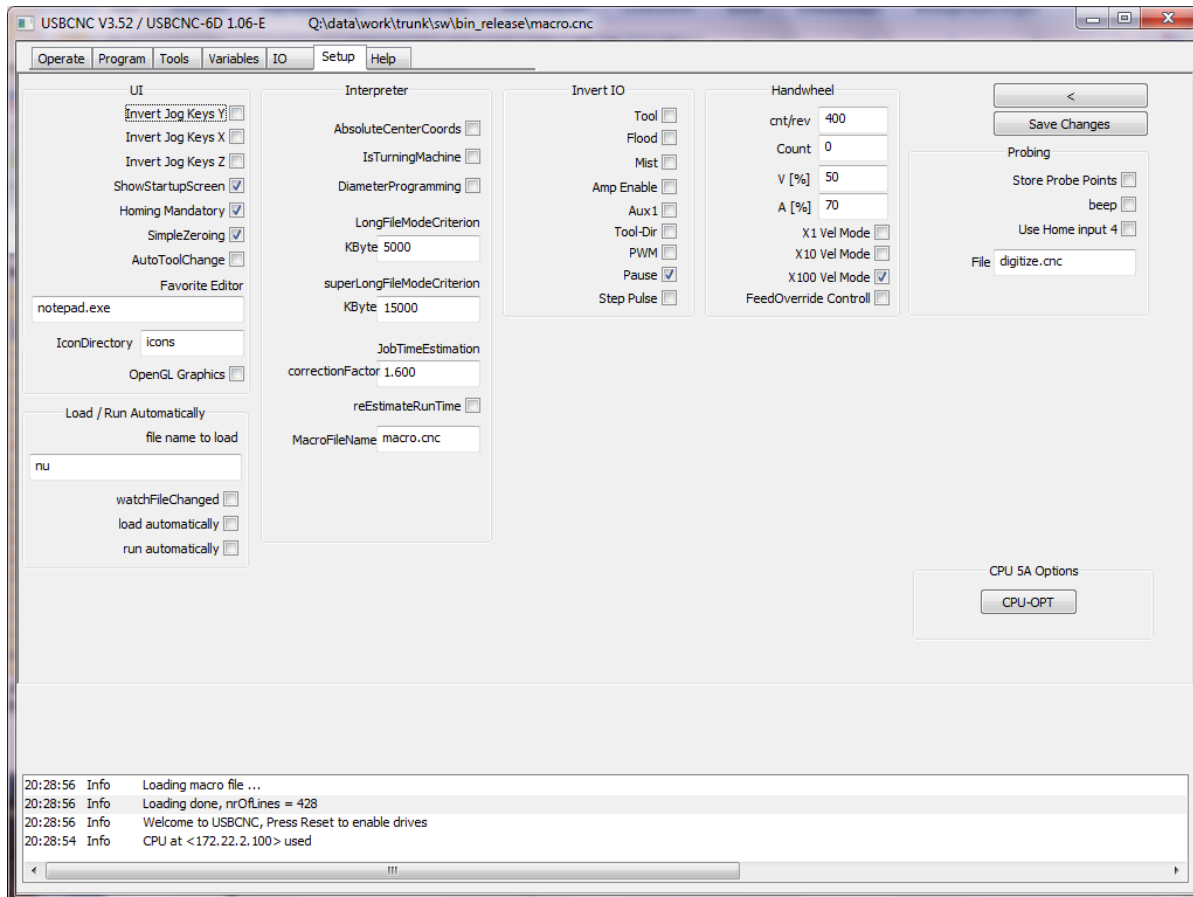
2.1.10 UI setup items

Invert JogKeys: Inverts the movement of the keyboard keys, for moving bed machines, the bed moves in the direction you press the arrow.

IsTurningMachine: Check if your machine is a Lathe, this effects mainly the 3D display which shows the X-Z plane for turning. Also the jog keys operate differently. Futher the working plane is set to G18 (X-Z).

ShowStartupScreen: When checked, the startup screen is shown when USBCNC starts.

HomingMandatory: When checked, running a job and mdi is not allowed before the machine is homed. Also the jog speed is limited to 5% speed. This feature prevents damage to your machine because when the machine isn't homed, the limit guards are not working. So, I advise to leave this item checked always.

SimpleZeroing: If checked, the zero buttons (beside the position display), will simply set the work position to zero. If this item is not checked, a dialog will be shown in which you can set the position. Default it shows a value which is – tool-radius of the current tool. This is handy when zeroing from the lower left corner with the endmill against the material.

AutoToolChange: If checked the running job will not stop when a toolchange is encountered. Use this when you have a ATC or if you simply always have the tool already in.

Favorite Editor: Specify your favorite editor here. I recommend notepad++, it is freely
                          downloadable at internet. E.g. for notepad++, specify
                          c:\program files\notepad++\notepad++.exe.
                          The advantage of notepad++ is that the editor jumps to the actual G-Code
                          line immediately, very handy when programming G-Code.

IconDirectory: The name of the directory where the GUI icons are located.
                          nu means **n**ot **u**sed.
                          If you want to change the Icons on the buttons, you can make first a copy of
                          the entire **icons** and name that directory to **myIcons**. Make your changes an
                          place the directory name in this field.

OpenGL: Check to use OpenGL graphics. This allows smooth panning, zooming and
                          rotation using the mouse.
                          Left mouse key; **Pan**
                          Right mouse key: **Zoom**
                          Control+Left mouse key: **Rotate**.

2.1.11 Load/Run Automatically

watchFileChanged: If checked USBCNC will watch the loaded g-code file for changes on disk if
                          USBCNC is not running. When it is changed, e.g. by an editor or because it
                          is saved by a CAM software, then USBCNC will ask you to reload the file:
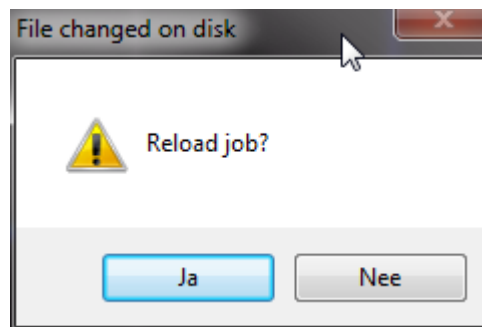
load automatically: If this is checked, the file is automatically loaded when it changes on disk, no
                          dialog will appear.

run automatically: If this is checked and also the load automatically check, then the file will be
                          loaded and immediately start running when changed on disk.

fileName: This is the name of the file that USBCNC watches at startup. So if USBCNC
                          is started and this file time/date changes on disk, it will be loaded.
                          If manually another g-code file is loaded, then USNCNC will watch that one.

2.1.12 IO setup

Invert IO: Check if you want to invert the output, e.g. the PICSTEP 4 card amp is enabled
                          when the signal is low. Goto the main operate window and press reset, the
                          amplifiers should be enabled. Try to jog by pressing the arrow key's. If there is
                          no movement, go back to the setup screen and check the Amp Enable
                          inversion. Press save, go back to Operate, press reset and try again to move. If
                          still not moving, check your hardware.

2.1.13 Interpreter settings

DiameterProgramming:

Check if you want diameter programming for turning, all X-axis values are interpreted as diameter. The effect is that all movements in the X-axis are divided by 2.

AbsoluteCenterCoords:

If Checked, the I,J,K value is interpreted as absolute value. Incremental is used mostly.

LongFileModeCriterion:

Specify a number of Kbytes here. When the loaded job file is larger, the UI switches to long file mode. The program listbox changes and the graphics will show only outlines when a program is loaded. This is al needed to preserve memory and speed for large files.
In this mode the file itself is still executed from memory and allows complex G-Code constructs (While, If then else, sub routines).

SuperLongFileModeCriterion:

Specify a number of KBytes here where super long file mode starts. This number should be equal or bigger as LongFileModeCriterion.
For very long files from 20MByte and UP to 4G this mode is required. It also puts the GUI in the same mode as with LongFileMode, but as extra, the file itself is no longer executed from memory. The means that complex G-Code constructs are no longer possible. These type of files generally contain only G1, sometimes G2, G3, and Tool changes M6Tx. The toolchanges are still executed from the macro.cnc file, so full automatic toolchange is still available.
Files with up to 100.000.000 lines of G-Code have been tested with this.

Macro Filename: Name of the macro file, it can be changed, the default is macro.cnc.

2.1.14 JobTimeEstimation

During the Render phase, after loading the job, the job time is estimated. But this is just a quick estimation because a real calculation of time would take too much time, therefore these parameters:

CorrectionFactor: Correction factor for the time calculations, you can change this if you see that your type of jobs require a correction.

RestimateRunTime: When checked, you will see the remaining estimated time of job based on the average speed measured and the total distance to go.

2.1.15 Hand wheel Setup

Cnt/Rev: The number of counts of the hand wheel for one revolution, usually 400 for most CNC hand wheels.

Count: Shows the actual Hand wheel count value, try to turn the hand wheel and see it change.

V[%]:   Percentage of velocity from selected axis, this is the maximum **velocity** the axis will move when using the hand wheel.

A[%]:   Percentage of acceleration from selected axis, this is the maximum **acceleration** the axis will move when using the hand wheel..

X1..X100 Vel Mode: In velocity mode the most important is that the movement stops immediately when the rotation of the hand wheel stops. The position of the hand wheel will not be maintained if velocity mode is on. The position of the handheld is maintained if velocity mode is off. This also means that the axis may not immediately stop if the hand wheel rotation stops. When turning beyond the limits of the axis, you have to turn back the hand wheel the same amount before the axis starts moving again.
                               My own experience is that it works best to use velocity mode at X100 only. Jus play with it to experience the behavior and make your own choice.

FeedOverride: When checked, hand wheel is used to control the FeedOverride. The feed override can be controlled from 0% (Stop) to 300%. The maximum velocity and acceleration specified for the motors will not be violated.
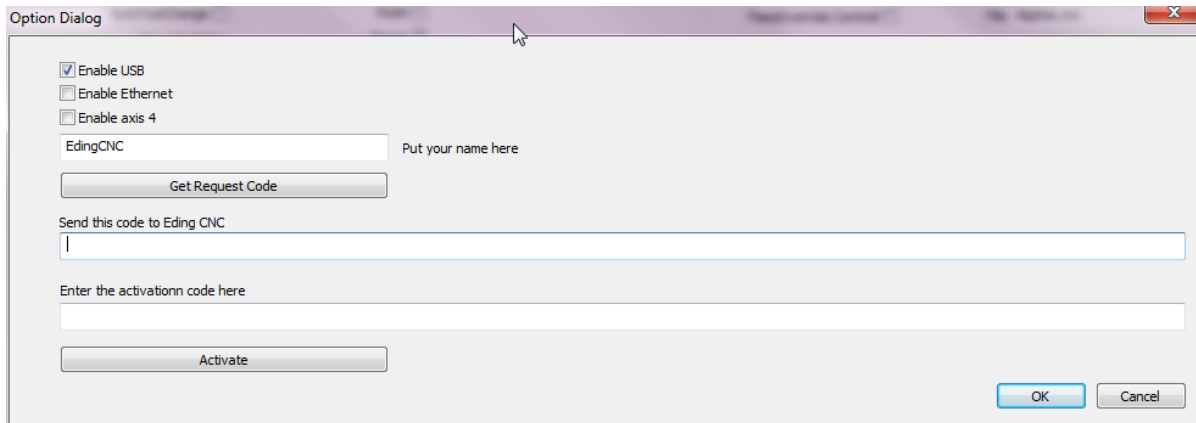
2.1.16 Probing Setup

StoreProbePoints: The touch points are stored in a file when this is checked. This is used for digitizing.

Use Home input 4: If checked home input 4 is used in stead of the standard probe input.

File: The file name for storing the touch points. The file is opened at the first probe touch en closed when a M30 command is encountered, usually at the end of the G-Code program.

2.1.17 CPUOPT

This is special for CPU5A.
This button allows to add the 4th axis function on a CPU5A3. So it upgrades from 5A3 to 5A4.
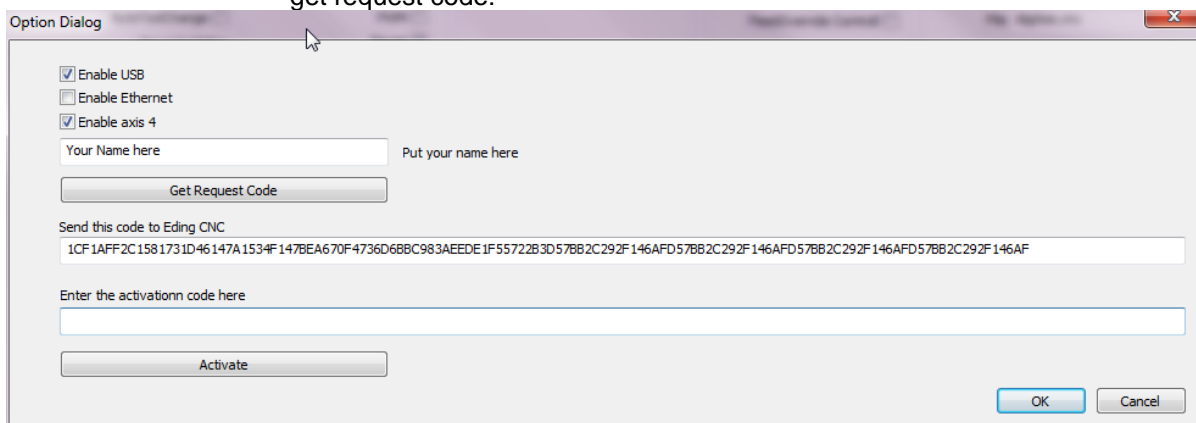
These are the steps to follow:
In the dialog check the "enable axis 4" checkbox, enter tour name and press get request code:

Send the request code to the supplier.
Copy and paste it into an email and send it to your USBCNC supplier.
To do this double click the code, press ctrl-c, in your e-mail press control-v.

Your suplier will send you a activation code.
Copy and paste this into the activation code area. then press activate.
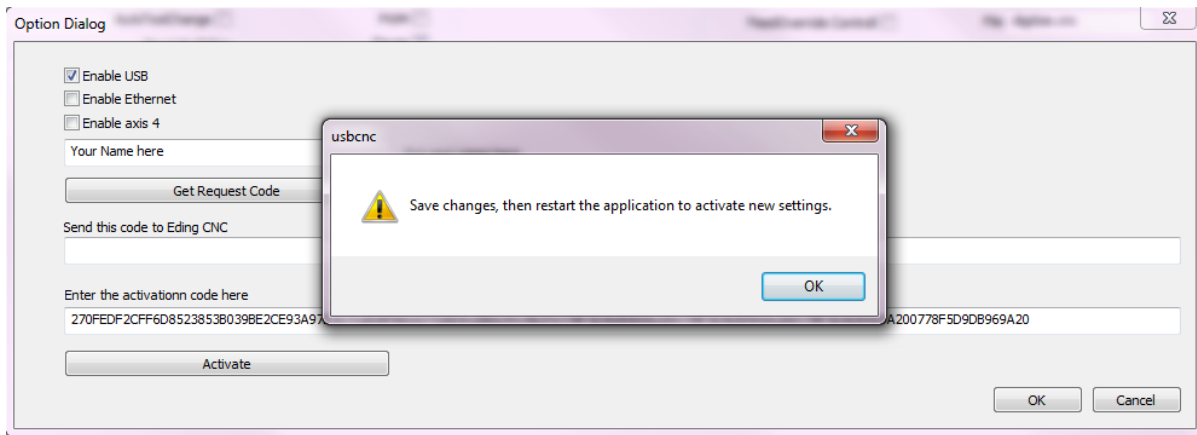
Do what the dialog tells you, press ok twice, press save changes. and restart. When you press the CPU-OPT button again, you will see that the 4th axis is enabled and that it is registered with your name.
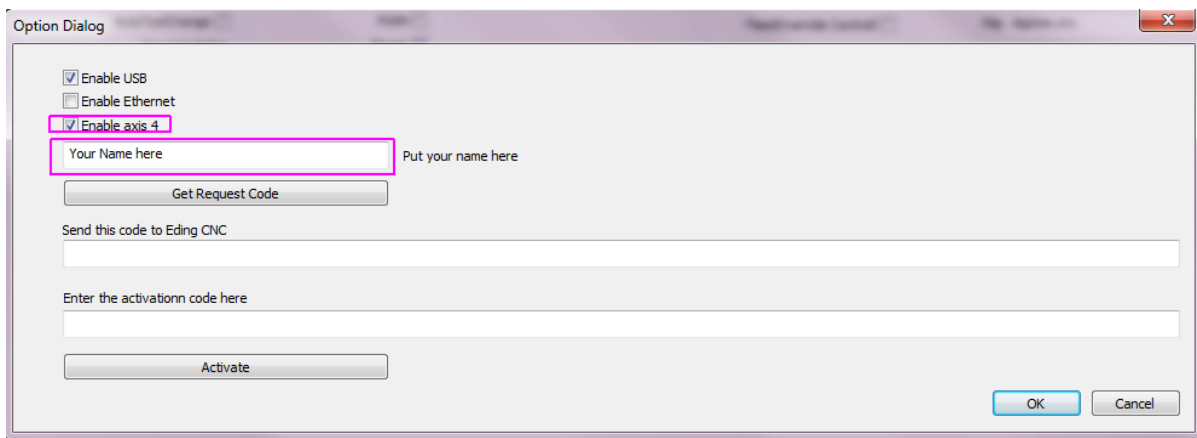
2.2 OPERATE PAGE
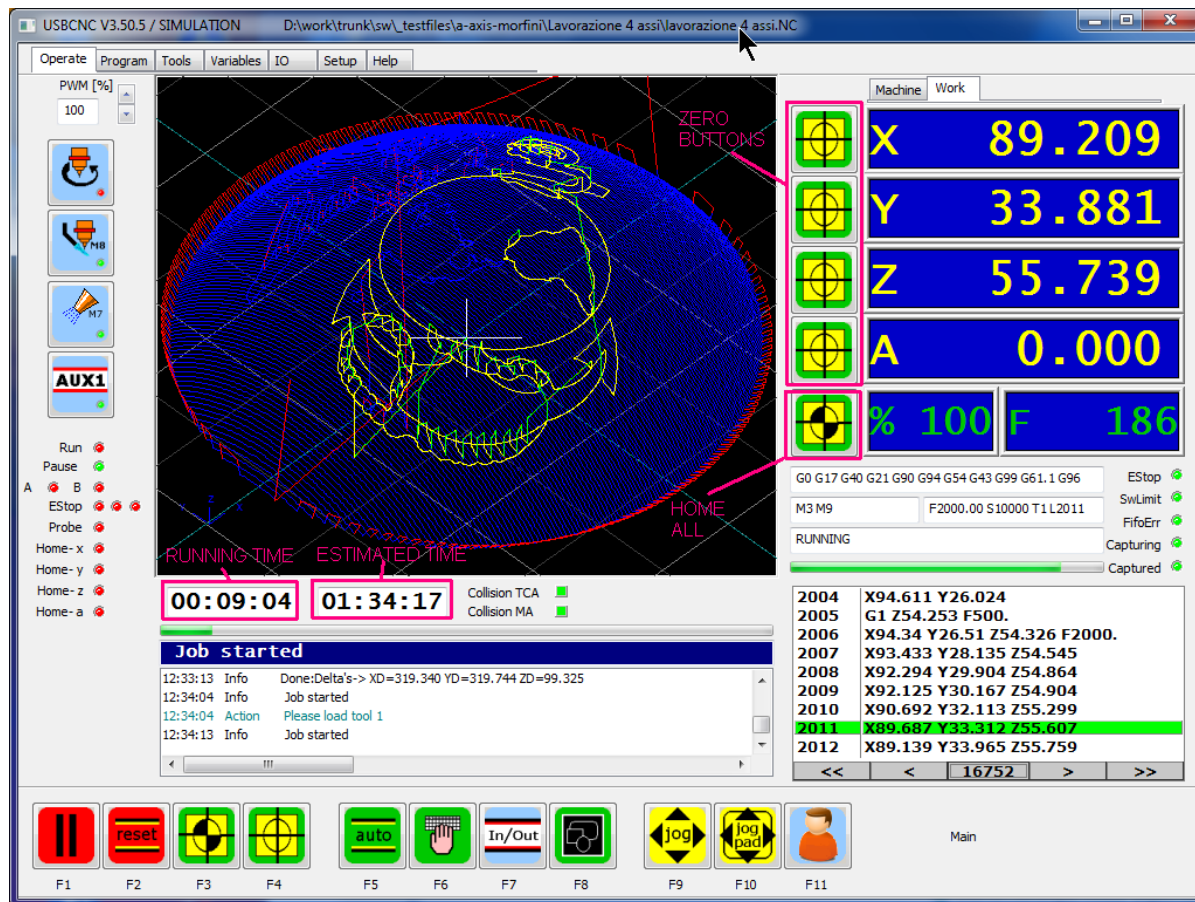
2.2.1 Operate page introduction

From this screen all machine operation like jogging running etc can be executed.
The Operate screen is designed such that it is mouse and touch-screen friendly.

In the middle we see the graphics showing the tool path. Blue/Red when loaded and rendered. Yellow/Green when actually running. So it shows the tool path real-time.

At the left side there are buttons for common used IO:
    Spindle speed for PWM controlled spindels,
    Spindel on-off,
    Coolant on-off, and
    AUX on-off (e.g. for the machine light).

The right part of the screen shows the axes positions, when homing you use the **machine coordinates** and for all other operations the **work coordinates**.
The buttons beside the axes positions are for zeroing the work position, on the background a G92 command is executed to perform this. The zero buttons can also be found in the zero submenu, especially for people who do not like using the mouse at the machine.

Below the axis positions you see the actual feed Override percentage and the actual feed when the machine is running.

Below that you see info about the G-Codes, M-Codes and Feed/Speed/Tool.
Further you see the overall machine status, "Ready" when at rest, "Running" when executing a G-Code file, etc. When the machine is in error, it says ERROR, or ABORTED when ESC was pressed. The

Escape key behaves like emergency stop, when pressed the machine stops immediately and all I/O's are switched off. Since the stop is immediately without ramping down the speed, the position of the motors will be lost and you need to HOME the machine again. So use the escape key only in emergency situations.

The progress indicator and the LED's below the machine status are the status of the USBCNC CPU. The progress indicator shows the fullness of the buffered motion and the LEDS show other status:

E-Stop: Turns red when the external E-Stop is activated, this can be caused by the EStop input(s), External Error input and also the HomeSensor inputs if HomeIsEStop is selected in the setup.

SWLimit: Turns red when during execution of a G-Code file the SW limits where crossed, in this case a smooth stop is executed without losing the position, so when that happens, just reset and continue without homing.

FifoErr: The motion send to the CPU is buffered into a Fifo buffer at the CPU side, while running this buffer may never run empty, when that happens anyway, there is a problem with the speed of the USB communication, the machine will stop immediately. This will normally never occur when your machine is setup correctly and when the USB speed is OK.

Capturing: When homing or probing the internal position latch in the CPU is armed to latch the position when the home switch or probe triggers. Capturing means the CPU is Armed to do this.

Captured: Becomes active when the latch has captured the axes positions, this happens very accurate at every stepper pulse.

All functions of the Operate screen and a list of G-Codes can be found in the Help screen, look at the help TAB when running the program.

All operation can be done without having to use the mouse. Under the function keys are all the functions you need. There is a 2 level menu structure, see the menu structure or help for explanation.

2.2.2 Reset Button

The amplifiers are switched on when pressing the **reset button.** Try this, you can feel at the motor shaft if the amplifier is on. if you can still turn the motor by hand, you probably need to reverse the amplifier enable polarity in the setup.

But the reset button does more:
- Enable the amplifier
- Recover from Error after you get one
- Stop a running program
- Rewind a job
- Render a program if not rendered

2.2.3 Load a G-code file (.iso .tab .nc .cnc .ngc ...)

Example: to load a G-Code file, press F5 to get into the AUTO sub menu, then press F3 to load the file.

When a file is loaded it is rendered first. Rendering is the process of interpreting all lines in the file without motion, check for machine and tool change collisions, estimate the time and create a visual

graphic. The Job can only start if it is collision free. If a collision is encountered this is very clear to see, see figure below.
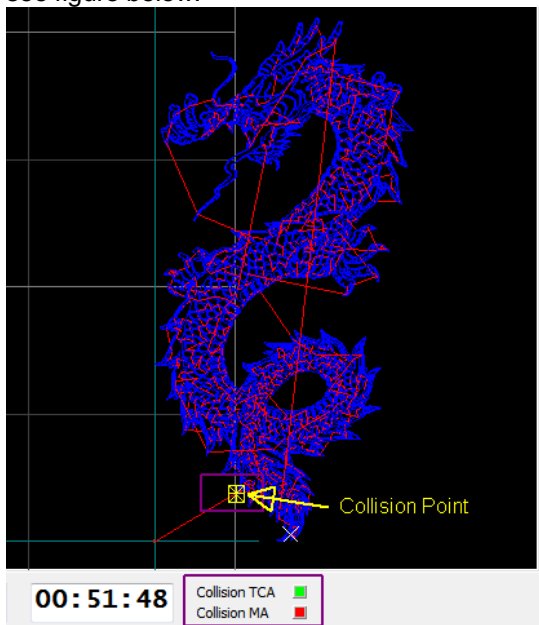
What to do in case of collision?

In this case the figure is small enough to fit the machine. So set the zero point a bit more to the left. You can simply do this by jogging to the left and set the new X zero point, press the button besides the X position display, then press reset to render again.

Now the image is shifted and collision free, ready to run, if the machine is homed. For homing see chapter 2.7 first.
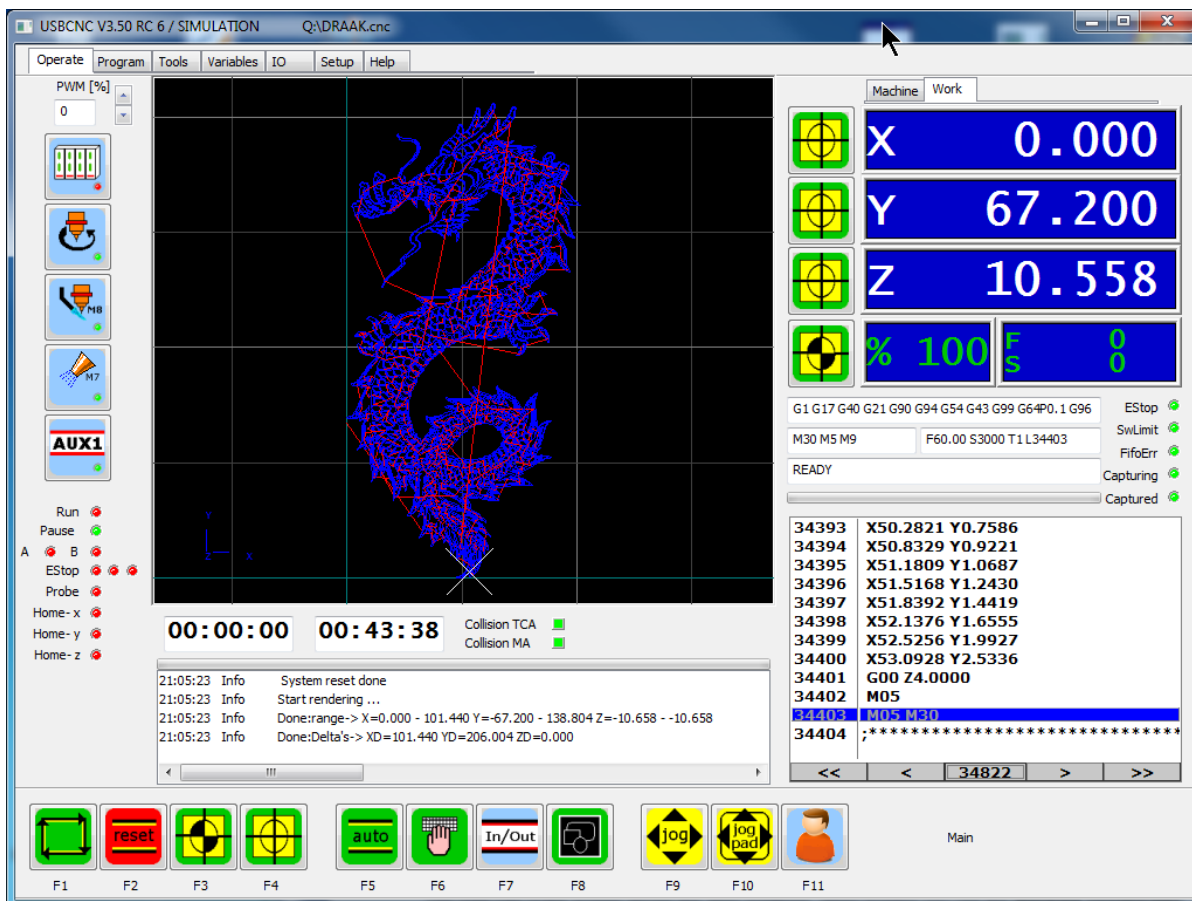
Press F1 to RUN and start milling.

Below the graphics screen there is the logging window, it provides user messages, warnings and errors, always keep an eye at this window, especially when things go wrong.

2.2.4   Menu structure of the operate Functions keys

Several of the function keys are configurable by modifying the macro.cnc file. This is applicable for all Home related keys and also for the user definable keys, see "macro.cnc" file in the install directory of the software.

F-KEY	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
Main Menu	Run/ Pause	Reset	Home	Zero work coord inates	Auto matic	MDI	I/O	Grap hics	Jog	Jog Pad	User defin ed	
Menu	F3- Home											
		Home X	Home Y	Home Z	Home A	Home B	Home C	Park 1 (G28)	Park 2 (G30)	G0 X0Y0 Work		Home All	Back to main menu

Menu	F4-Zero											
	❌	❌	❌	❌	❌	❌					❌	❌
	Zero X	Zero Y	Zero Z	Zero A	Zero B	Zero C					Zero All	Back to main menu
Menu	F5-Auto											
	❌	❌	❌	❌	❌	❌		❌	❌		❌
	Run/ Pause	Reset	Load G-Code File	Edit File	Start from selected line	Back to Pause Point		Feed-	Feed+		Back to main menu
Menu	F6-IO											

In/Out	Run/ Pause	Reset	Enable Amplifier	Aux1	Flood	Mist	Tool	Tool Direction	Tool speed -	Tool Speed +		Back to main menu
Menu	Graph											
	Run/ Pause	Reset	Load G-Code File	Edit current file	2D/3D View	Zoom Fit	Zoom -	Zoom +	Zoom Full Machine	Erase	Re-render graph	Back to main menu
Menu	Jog											

jog	Continu	Step 0.01	Step 0.05	Step 0.1	Step 0.5	Step 1	User defined Step		Hand-wheel x1	Hand-Wheel x10		Back to Main menu
Menu	User											
	Call sub user_1 Macro. cnc	Idem User_ 2	Idem User_ 3	Idem User_ 4	Idem User_ 5	Idem User_6	Idem User_ 7	Idem User_ 8	Idem User_ 9	Idem User_1 0	Idem User_ 11	Back to Main menu
Short-cuts	Ctrl-s	Toggle Single line execution					Ctrl-F6	Toggle MDI window On/off				
	Ctrl-i	Toggle simulation mode On/off										
	Ctrl_b	Toggle Block Delete On/off										
	Ctrl-W	Toggle Machine/Work Coordinates										
	Ctrl+sh ift-s	Show startup screen										

2.2.5 User button 2

Use for automatic tool-length measurement using a fixed toolsetter, the tool length is stored into the tool table. This is a more complex example of a user macro. It is bound to subroutine "user_2" in file "macro.cnc". In this case te user_2 macro calls a new macro named m_tool, see "macro.cnc" file.
Some calibrated variables are used:

#4996 : Tool measurement safe height
#4997 : Tool setter X position
#4998 : Tool setter Y position
#4999 : Z position where the tool chuck would touch the tool setter (tool-length = 0)

This macro first checks the variables above, if they all are 0 a warning I given to calibrate the tool setter values first:
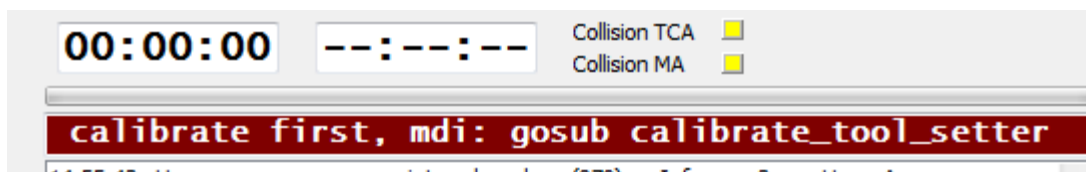
Note that variables #4000 - #4999 are saved when USBCNC is closed and restored when started.
These positions can be calibrated by executing the subroutine calibrate_tool_setter.
So if you get this message, then do what the message says:
Start the subroutine by entering in MDI:
>**gosub calibrate_tool_setter**.

If the calibration is already done the macro performs its job:

Then Z will move to the safe height.
The tool-length measurement asks for the tool data, number, diameter, approx length.
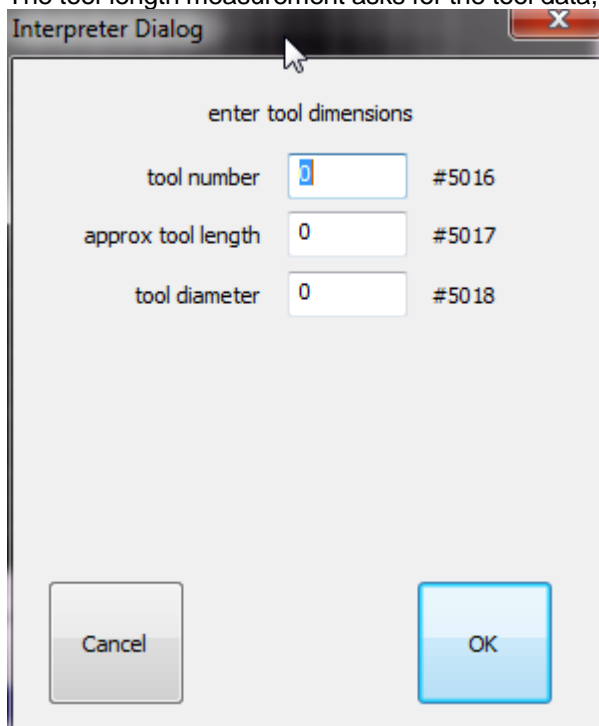
Then the machine X,Y move to the calibrated X,Y positions.

Then the machine Z moves move down to a height which is 10 mm above the tool setter. This is done by taking the tool chuck height + given approx tool length + 10.
Then a probe movement down is made until the tool touches the tool setter.
The tool-length is calculated and stored into the tool-table together with the given tool radius.

Feel free to adapt these macro's to your own needs.

See also chapter 2.10 for explanation about the macro.

2.2.6 The JOGPAD
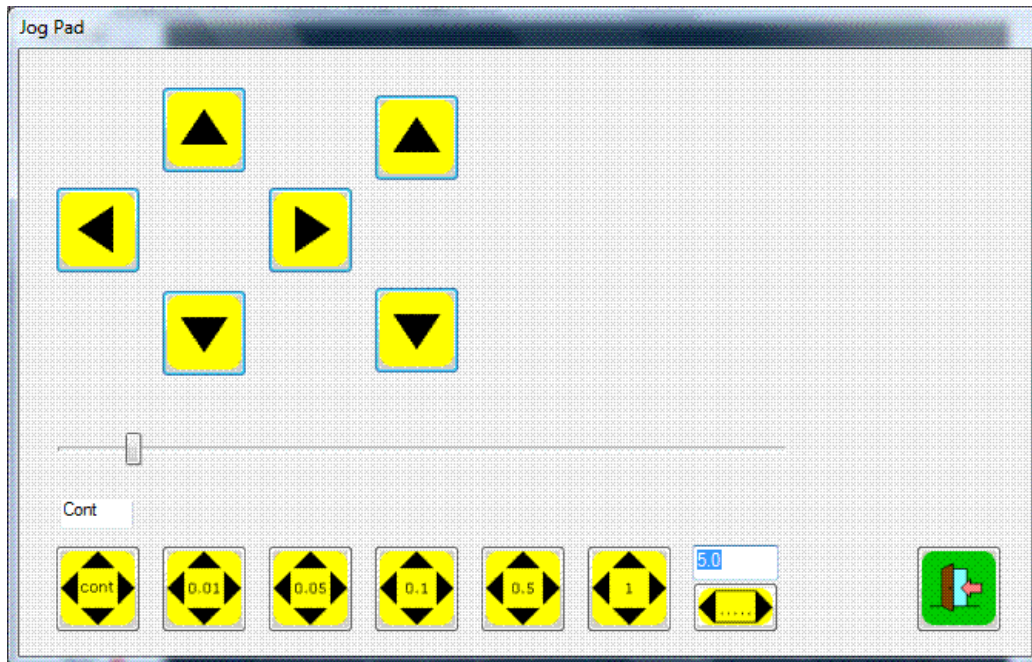
With the jog pad you can jog the axes using the mouse or a touch screen.
- Continuous mode the axis moves as long as the button is pressed and stops when the button is released.
- Step mode 0.01 to 1 allows stepping the axes. Each button press moves the axis one step.
- User step mode is the same as step mode but allows the specify the step-size.

When step mode is chosen, shift coordinates becomes visible:
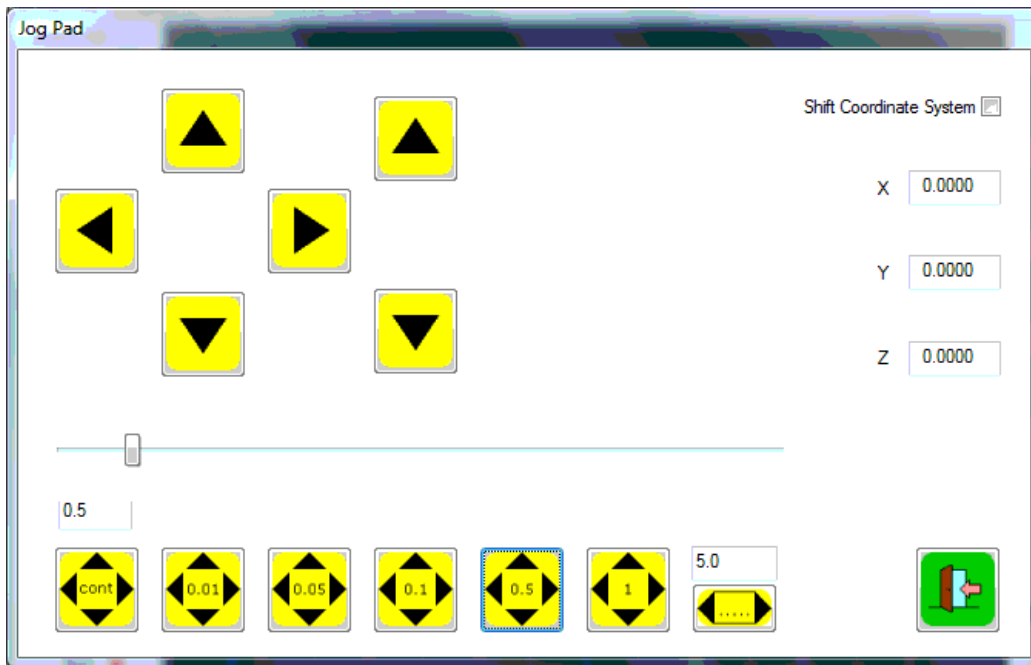
When "Shift Coordinate System" is checked, jog-step functions as normal, the axes move one step at a time. The work position however remains the same. This is accomplished by modifying the active G92 offset. It is useful when e.g. during engraving you want to run the G-Code program again, but a little deeper in Z. E.g. you want to run the program 0.1 mm deeper, select jog step 0.1 and check "shift coordinate system" . Now press de arrow down button to move Z 0.1 mm down. Notice that the axis moves down but that the position remains the same. When you run your engraving program again the engraving will be 0.1 mm deeper into the material.

This option is also very handy during turning. Your program has run and you measure the work piece and see its diameter is still a bit too big. So now use the –X button to compensate the diameter. Run the program again and your work-piece diameter will be correct.

The amount of shift is shown at the right side. To reset the value to 0, which has no influence on the active offset nor machine position, uncheck, and then check "shift coordinate system".

2.2.7   The graph menu and view

The new graph view shows a grid of **50mm** in mm mode or **2 Inch** in inch mode projected on the machine bed (X-Y surface). For a representative view it is important that the axes limits are correctly filled in and that the machine is homed manually or automatic. The current work coordinate system origin is shown as a cyan colored cross in the x-y plane. When you press the preview update button, a preview is shown of the loaded G-Code program. The preview is created by running the entire g-code file through the interpreter. So when interpreter errors occur, it shows in the log window and in the operate view the program list box shows the wrong line in red color. Note that there can be inaccuracy in what the display shows, this is there because of performance and memory usage limitation reasons.
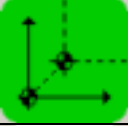
Zooming, rotate, 2D/3D view and other possibilities are found in the graph sub-menu, see the example below.
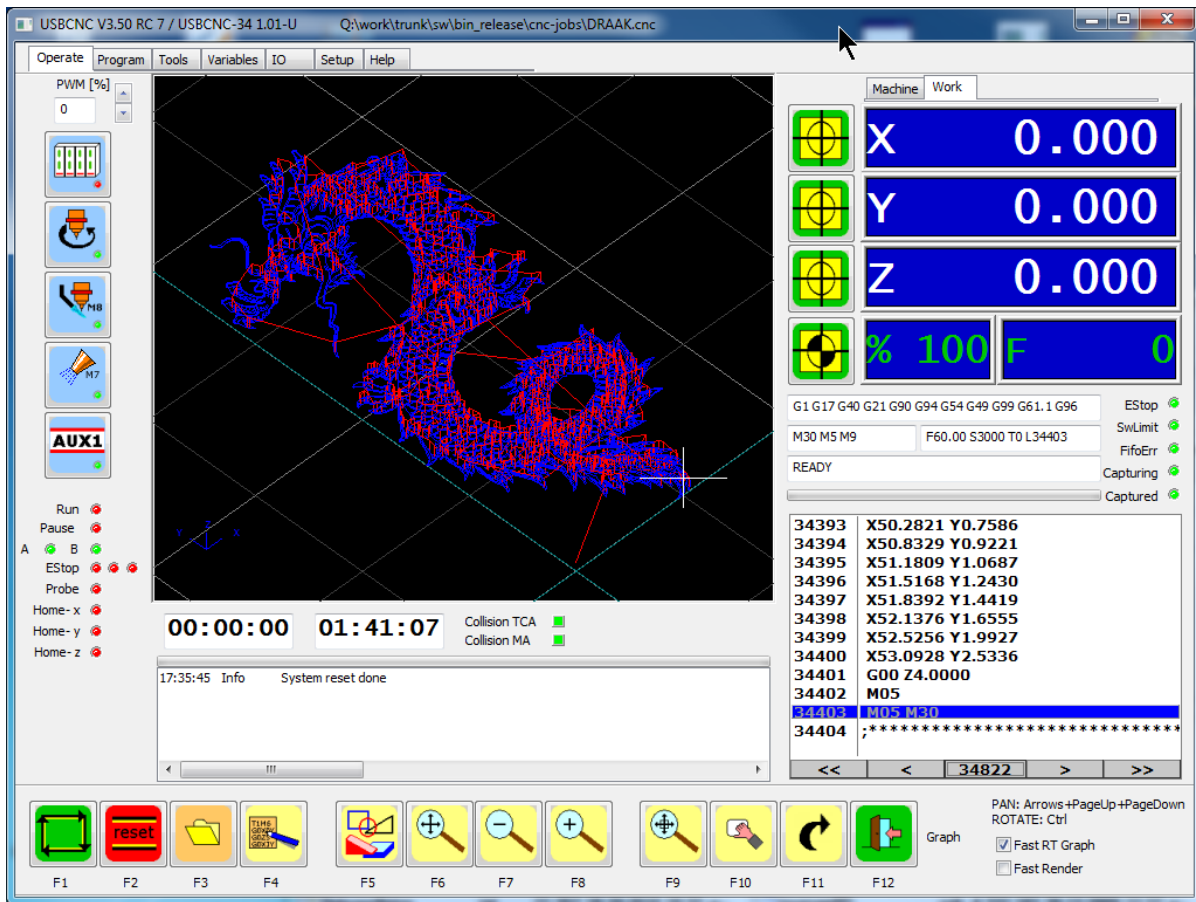
2.3 PROGRAM PAGE, DXF AND HPGL IMPORT
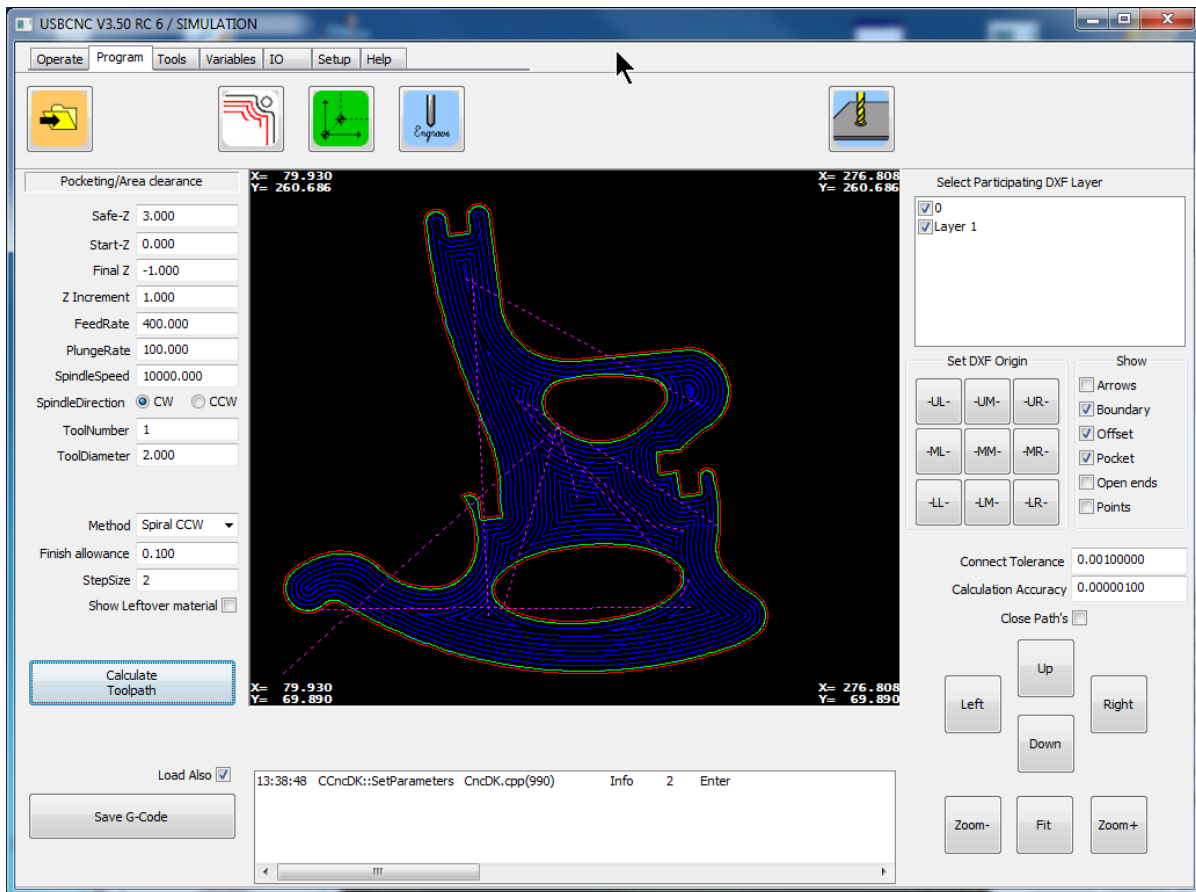
USBCNC uses a build in CAD/CAM library for these advanced import functions. You can load a file and then perform one of these operations:

	Loads a DXF or HPGL file
	Select engraving, this is milling over the lines from the drawing.
	Select profiling, this is for milling out objects and taking the tool diameter into account.
	This is for pocketing, to mill out the complete object.
	Drilling, draw points in the DXF file to use this.

After loading a DXF file, all layers will be visible. You can unselect layers at the right side, such that you see only the part that you want to use.
You also can change the origin of the drawing by pressing the appropriate button under the layer selection list box. The positions of the buttons give the positions of the origin. So e.g. when you press the upper right button, then the most upper right position of the drawing will become x=0, y=0 when milling.

The DXF import supports:
- Lines
- Arcs
- Circles
- Poly lines with arcs
- Points for drilling

The workflow of using these features is:
1) Load drawing
2) Select the correct layers
3) Apply origin offset if wanted
4) Set correct parameters
5) Calculate tool path
6) Save tool path and optionally immediately load it for milling.

Parameters involved:

Save-Z	When moving from one region to another, the machine goes to this height.
Start-Z	Z value where the tool touches the material to be machined.
Final-Z	Z value specifying the milling depth lowest Z value. Final Z must be lower than Start Z.
Z Increment	This specifies the step size when machining in passes.
Feed rate	Milling feed (F) in mm/min
Plunge rate	Feed (F) that the Z moves down into the material also mm/min
Spindle speed	S value for spindle.
CW/CCW	Spindle direction (M3/M4)
Tool number	This is only used for the M6 tool change command
Tool Diameter	Diameter of the tool for the offset and pocketing calculations.
Method	Outside/inside/clockwise/counterclockwise operation
Finish allowance.	Material that is left for the finishing pass when pocketing. This finishing pass is at full depth for getting a clean edge.
Step size	Step oversize for pocketing, this value should be lower than the tool diameter.
Laser mode	For profiling, when switched on, the tool will be switched off when moving from one region to another.
Make bridges	Leave small pieces of material, that prevent you object from falling out (and get damaged) when profiling.

Bridge distance	Approx distance, the exact distance is calculated such that all bridges have equal distance.
BridgeFinalZ	Lowest Z value for bridge, this value should be between startZ and finalZ
BridgeWidth	The width of a bridge.

When the parameters are set, press calculate tool path, it will be visualized on the screen.

Here an example of profiling with bridges:
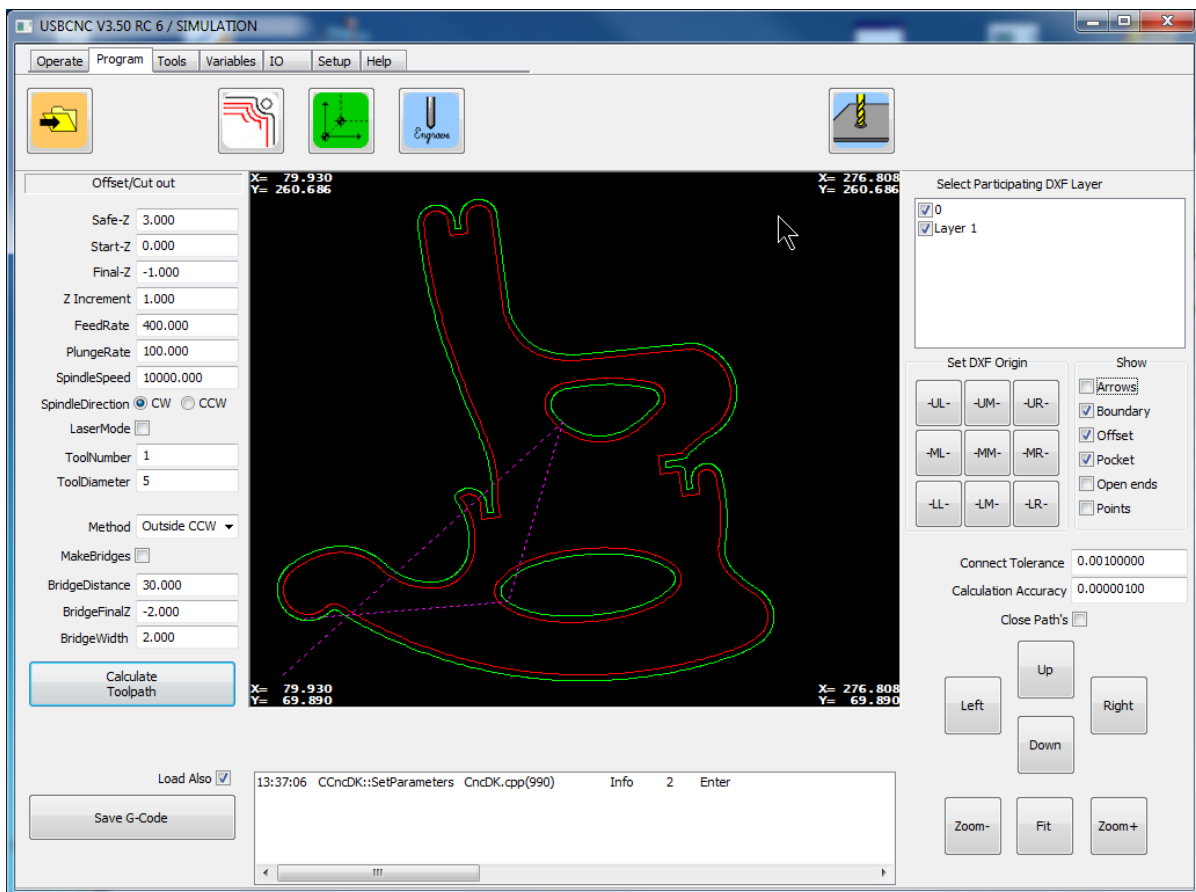
Note: The offset and pocket calculation might not always work, this is usually because of small errors in the drawing like lines over each other or not connecting lines. Experimenting with the Calculation Accuracy might help. Also check/correction of your drawing may help.

The engraving function is robust and will always work.

2.4 TOOLS PAGE

2.4.1 Milling
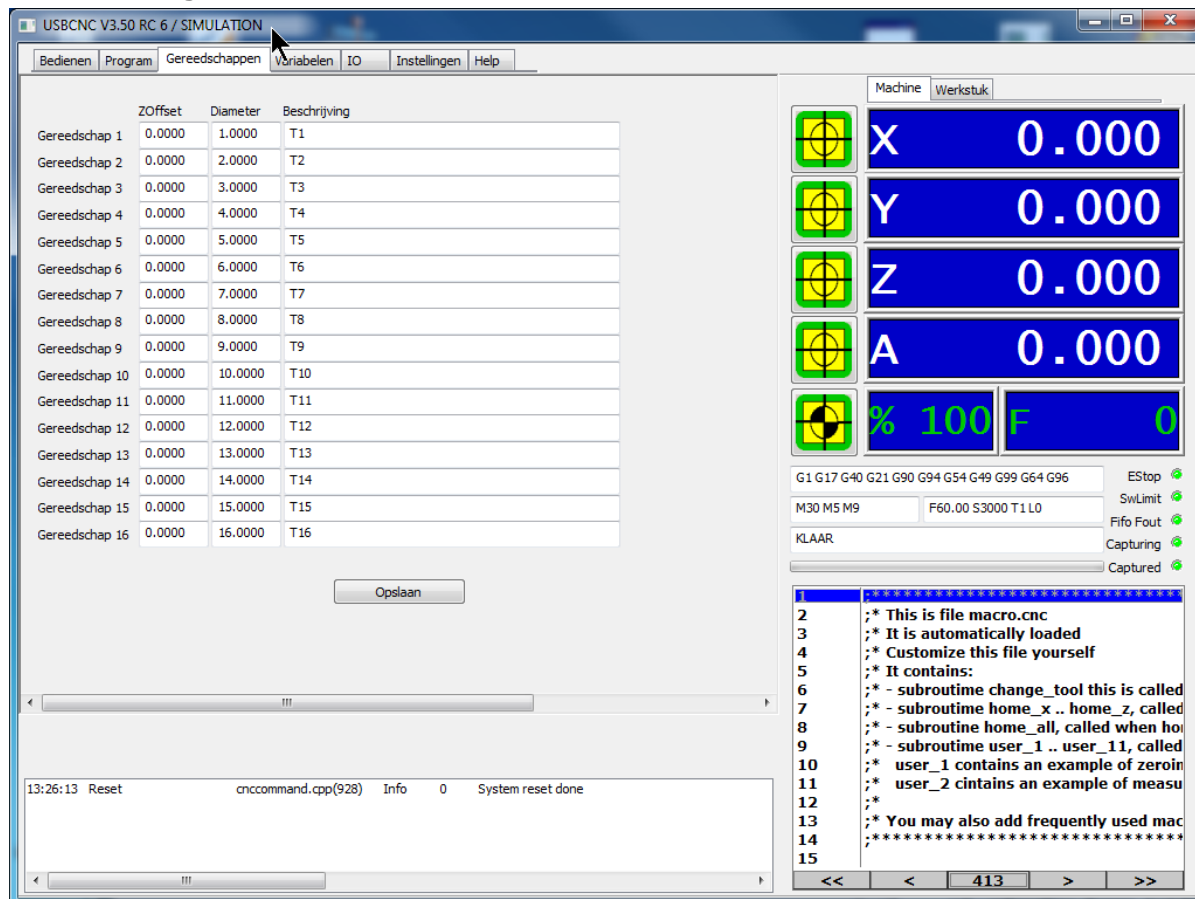
In this view you can define 16 tools with a length diameter and description.
For Lathe operation there is additional and X-Offset and tool Orientation parameter.
The tool information is used when you use the tool radius and or tool length
compensation functions of the G-Code interpreter commands (G40 – G43). See
chapter 3.6. and further.

2.4.2 Tool change

A tool change is performed in G-Code by M6 Tx where Tx is the new tool number.
Tool number 0 means no tool.
Normally the program is stopped on a tool change, with a user message to change
the tool, pressing run again will continue the program. If you don't want the program
to stop, check AutoToolChange in the automatic menu bar. This setting is saved
when you press save INI file in the setup screen.

2.4.3 Automatic user defined Tool change ATC

When you want to define you own tool change cycle, you can edit the file
"macro.cnc" in the USBCNC directory. When an M6 Tx is encountered, this is
translated to a GOSUB of subroutine change_tool in the "macro.cnc" file. This

suproutine then calls further subroutines drop_tool_x and pick_tool_x, if you have a toolchanger, you can add extra movements to the right tool position and control I/O for actually changing the tool.

2.4.4 Turning
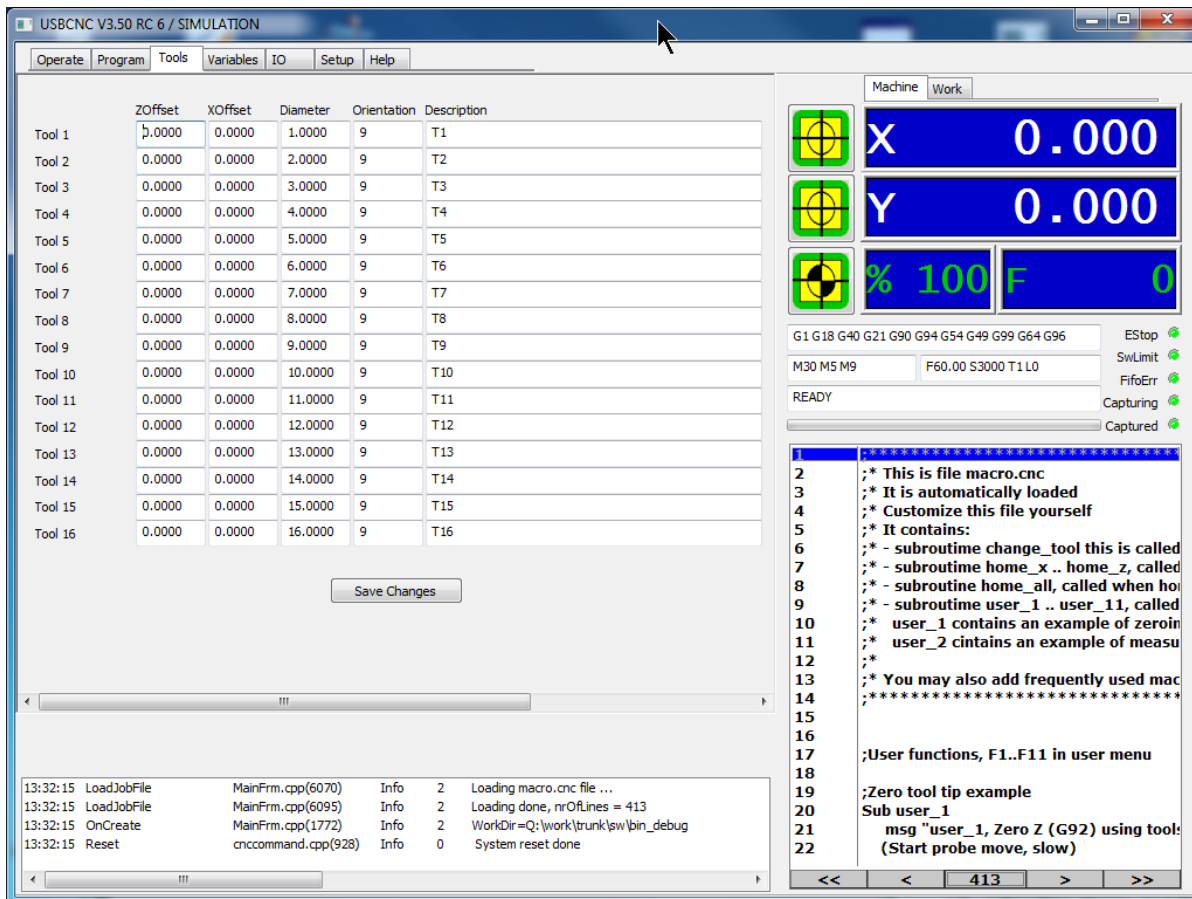
As you can see, there are 2 additional parameters for turning, X-Offset and Orientation.

2.5 THE VARIABLE PAGE

This page shows the standard variables used by the G-Code interpreter.
It also contains 4 watches to show your own variables if you are going to use the
extended programming features. You will understand the meaning of this window
after reading the G-Code interpreter functions and extended programming with
variables.

.
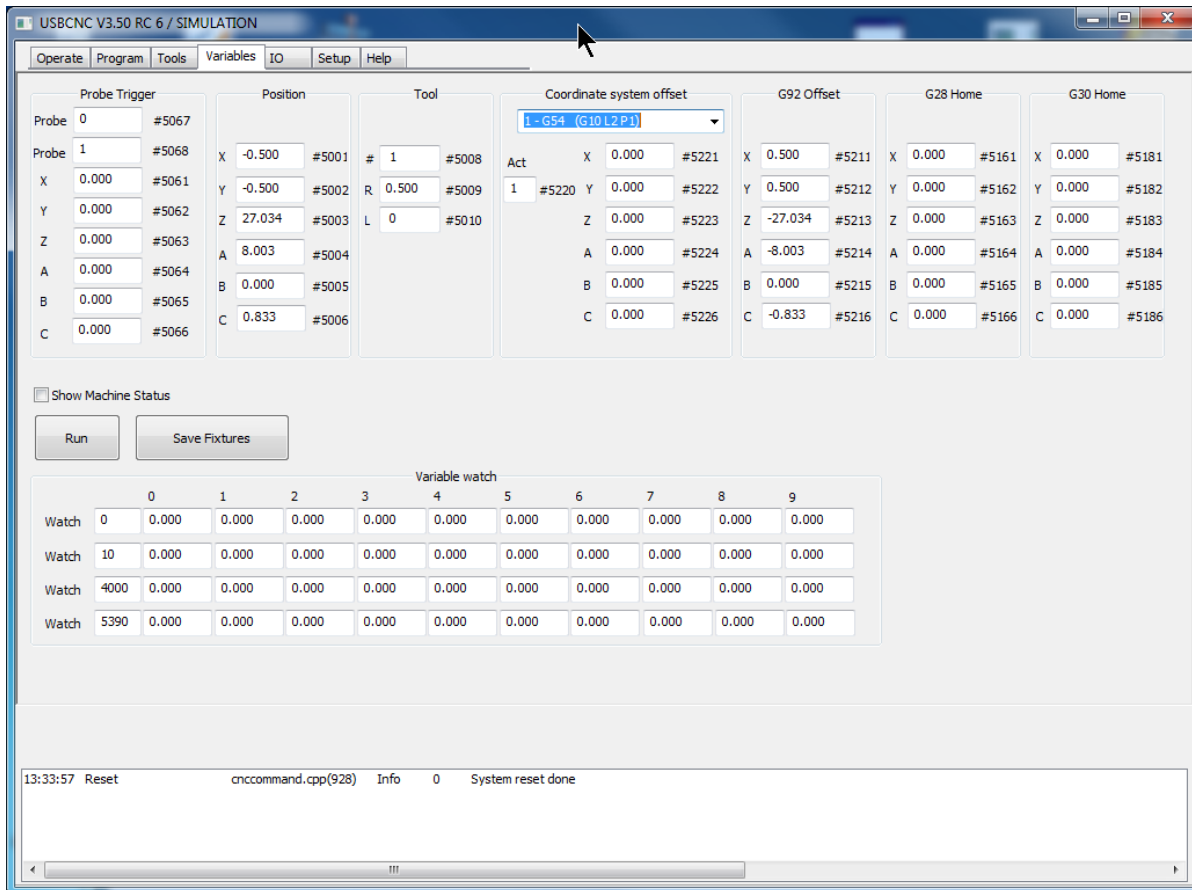
2.6 IO PAGE
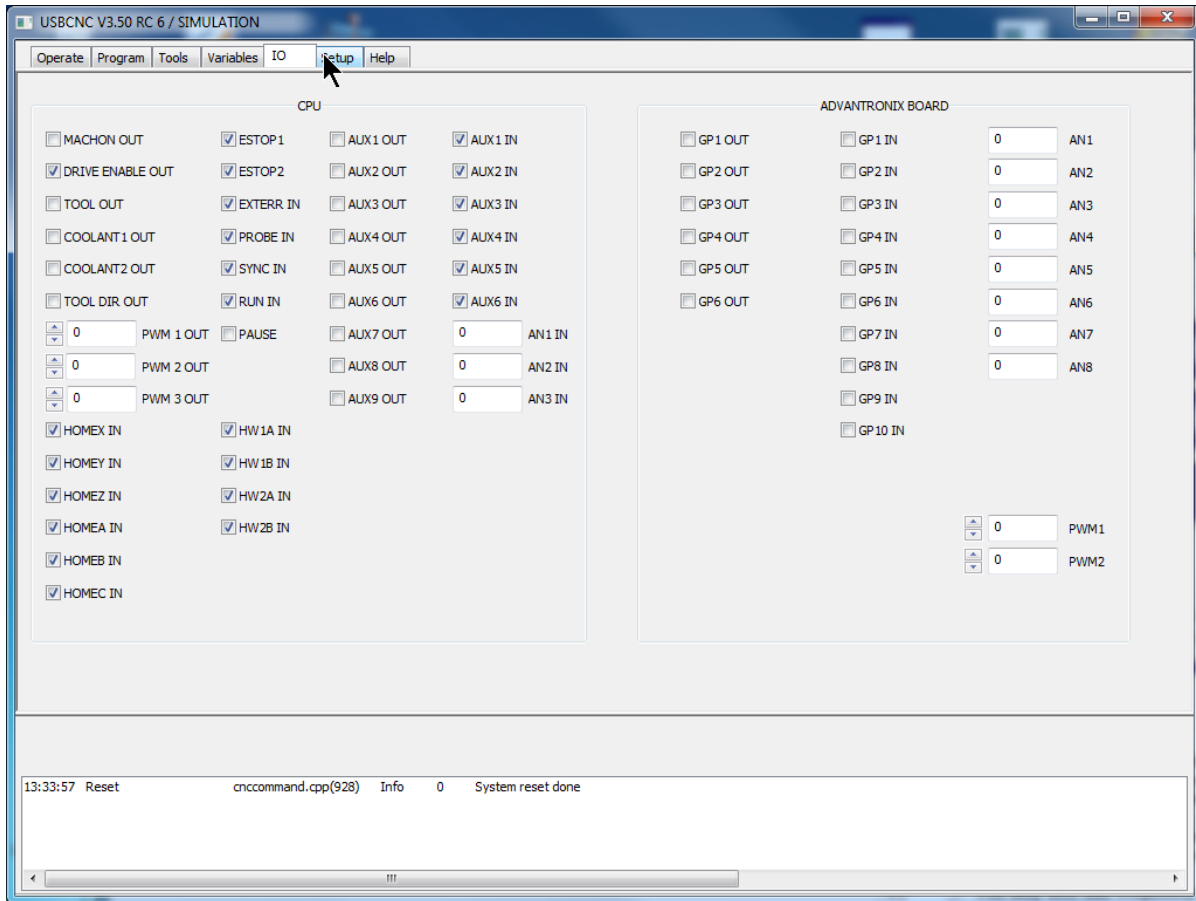
At this page you can monitor and set the I/O signals.
The page shows only the I/O that are applicable for the attached hardware.
The Advantronix part is only available when you have an Advantronix USB I/O card connected.

2.7 HOMING AND COORDINATE SYSTEMS

As I am like most people and don't want to read a comprehensive manual, but start right away. So I have written this little tutorial, it explains how to home the machine and use the coordinate systems in a simple way. This part is very important to read, you will enjoy operating the machine more if you use the coordinate systems the right way!

When your machine is switched on, all axes can be at any position, these positions are unknown by the software. The software however needs to know the position to show a correct graphic in the graph screen and also for preventing damage to your machine by running beyond the machine limits. The process to match the machine position with the software is called homing. Homing can be done either manually or automatic if end of stroke switches are mounted. This tutorial describes homing.

Here are the homing buttons, F3 from the main menu.
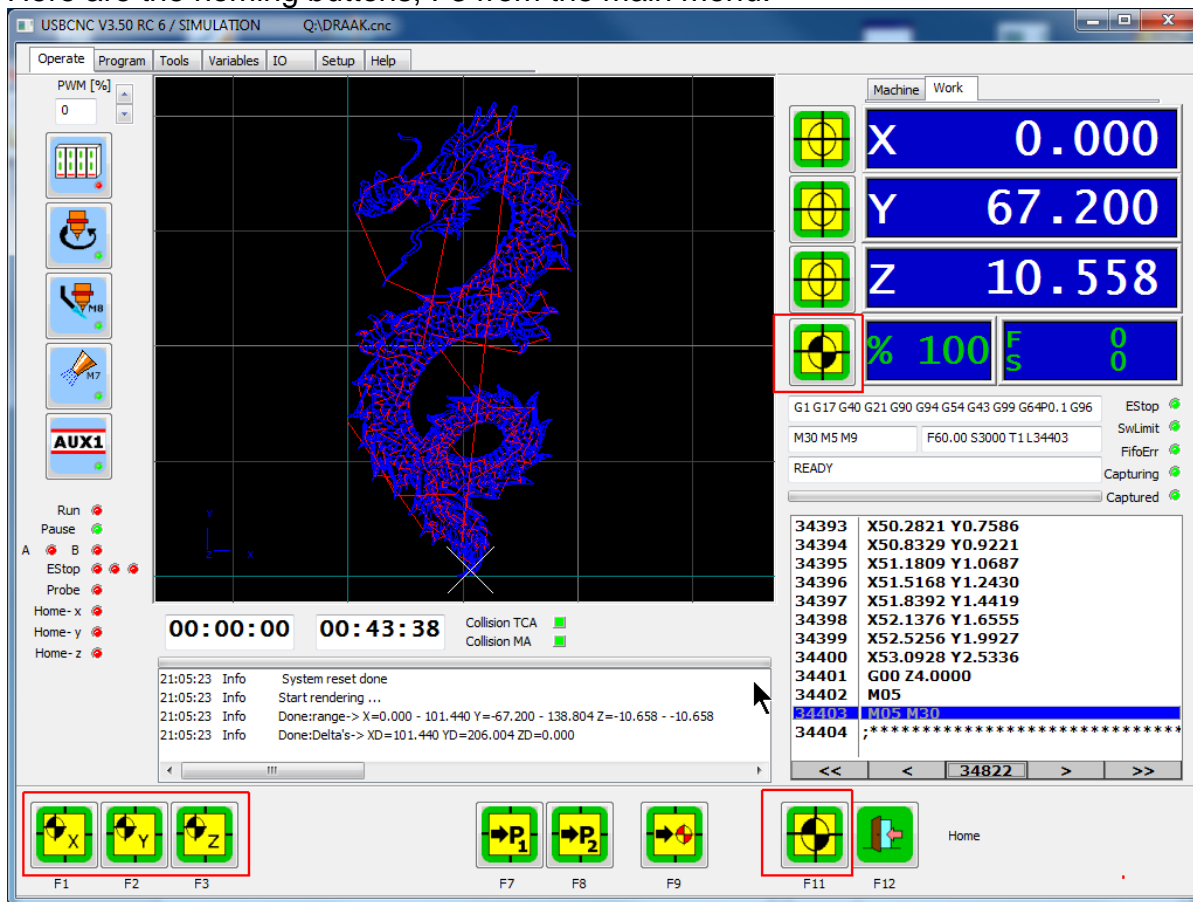
With F1 .. F3 the X..Z axes can be homed individually.
With F11, the home sequence can be started to home all axes in a sequence.
F11 is the same as the button besides the %100 feedOverride display.

What happens is that a few subroutines are called. The subroutines are in the macro.cnc file in your USBCNC installation folder. They look like this:

```
;Homing per axis
Sub home_x
    home x
Endsub
```

```
Sub home_y
    home y
Endsub

Sub home z
    home z
Endsub


;Home all axes, uncomment or comment the axes you want.
sub home all
    gosub home z
    gosub home_x
    gosub home_y
endsub
```

A good reader has seen that the order of homing is defined by the **home_all** subroutine and can be customized to your own needs.

2.7.1   Manual homing the machine

Homing is the first thing you always do after switching on the machine, I recommend making a habit of it. Suppose your machine limits are:

X: +300 mm and -300 mm
Y: +200 mm and -200 mm
Z: +100 mm and 0 (0 is the bottom surface of the bed)

Set the Home velocity to 0 for all axes that have no EOS switch.

Mark a point somewhere on the machine that you want to use as home reference point, let's say X= -200.0mm, which is 100.0mm from the left edge and Y= -150, which is 50.0 mm from the lower edge. For Z we take to bottom of the bed at Z=0 mm. This position x=-250, Y=-150, Z=0 is entered in the Home Position values in the set up screen, this need to be done once.

Using the arrow keys, jog the X,Y axes to the marked position on the bed and move the Z completely up to the surface of the machine. When the machine is at the position press the Home button in the Home submenu, F1-F6 for X-C.

Be sure that you have set the home velocities of the axes to zero, otherwise the axes will start to move. Now click the buttons X, Y, Z, and A if you have an A-axis. That is all, the axes are now homed and the software now knows the machine position.
As a side effect, now also the software limit switches are enabled which protect you from jogging further than the machine can go. Also the Software limit guard is on that will stop a running program when going beyond the limits.

You may also have noticed that the position mode is set to "machine", this is because homing directly affects the machine coordinate system. From this point the machine coordinate system, is not changed any more, it stays as is.

HINT: Move your machine always back to the home position if you are done with the machine. You don't have to move manually to this point next time when you switch back on the machine. You can do a fast move in machine coordinates like this: g53 g0 x0 y0 z0, or first undo the preset (preset dialog, undo preset) and then do a regular G0.

Another possibility to move quickly to the home positions is using g28,
In the variable window set G28 home positions to the same value as the home positions in the set up window. Now you have to type only g28 to go to the home position.

2.7.2   Automatic homing the machine and HomeIsEstop

The machine needs a homing sensor or switch for each axis connected the its home input on the CPU board.

The homing switch is placed at a small distance of the mechanical end of the machine. This distance is needed to ramp down the velocity after the switch is activated.
The sensor should be mounted such that it remains active until the mechanical limit of the machine.

For automatic homing the home velocity needs to be set to another value than zero, use an equal or lower speed than the axis maximum speed. The axis should start to move in the direction where your homing switch is mounted, when it is needed to reverse the direction add a minus sign to the homing velocity. Setup the **HomeInputSenseLevel** correctly. When the led's are green when the input is not activated put a 1 here, when the led's are red when the switch is not activated, put a 0. This depends whether you have used normally open or normally closed switch. I recommend normally closed switches here.
Use the homing sub menu to home your axes.

1st Move: The machine first moves until the switch activates, then ramps down and stops.
2nd Move: Then the direction reverses and ramps down when the switch releases.
At the moment of the release of the switch, the position is captured and used to set your machine position correctly.

2.7.2.1   Tandem axes homing
Tandem axes, one main axis has 2 motors, the correct rotation axis option is set to be slave of X, Y or Z. If the TANDEM axis has individual home sensors for master and slave, the home sequence can be customized such that the TANDEM sets itself straight after homing.

For tandem axes these special interpreter commands exist:

PrepareTandemHome
MoveStaveToMaster

HomeTandem

For the explanation, I assume that the **master axis is X** and the **Slave axis is A**.

1. **PrepareTandemHome X**, Both slave and master are moved towards the home sensor. The axes stop when both axes are on the sensor. When one

axis reaches the home sensor first, this one is stopped and the other moves further. This movement is done when both axes have reached.

2. **Home X**, home the X, the slave will just follow. Because the axes are on the sensor, the move will be off the sensor. The position is latched at the moment the sensor de-activates. Then the movement stops and then the correct position is calculated end set for the X.
3. **Home A**, exactly the same, but now the A is master temporarily and X will follow. At the end the position is calculated and set for the A.
4. At this point both master and slave have a correct known position. It is very important that the home position in the setup matches the actual machine. Now we can straighten the bridge by the command **MoveSlaveToMaster** A. The slave will move to the same position as the master. The bridge is set straight and we are done. (If the bridge is not straight, the home positions in the setup are not correct).

To make this whole sequence more easy, the **HomeTandem X** can be used to do it all at once.

When testing with the individual commands PrepareTandemHome, Home master, Home slave, MoveSlaveToMaster is done, the HomeTandem command can be used.

So if X has a slave Axis, then modify the **macro.cnc** so that subroutine home_x contains:

```
Sub home_x
    homeTandem X
Endsub
```

For a normal, non tandem it would contain:

```
Sub home_x
    home_x
Endsub
```

2.7.3 Work versus Machine coordinate system and zeroing

The machine coordinate system does not change, however we want to be able to do the milling of our part anywhere we want on the machine. We will normally use the "work" coordinate system, we can shift it anywhere we want. This can be done with several G-Codes, which are explained in chapter 3, it can also be done using the "preset" button on the operator screen, we'll see this in a minute.

Suppose our g-code file containing the work piece is created with an origin of X=0, Y=0, Z=0. This is because you have drawn your part in a CAD program beginning from these coordinates and then converted to G-Code.
Now you have put your raw material somewhere on the machine, probably not at coordinates X=0, Y=0, Z=0.
By the way, I prefer to define the upper surface of the material as Z=0, such that a negative Z value goes into the material.

Just move to the zero point of the work piece and there press the zero buttons in the operate screen besides the position display.

For the advanced users: The zeroing can also be done using a measuring probe connected to the probe input. An example is provided in the standard macro.cnc file. Under user_1 you find automatic zeroing. Under user_2 you find interactive tool length measurement.

If you want to do it a more advanced way, look at G55 .. G59.3 and also at the G92 variants.

When homing and zeroing is performed, the milling can start:

When the program is loaded, go to the graphics screen (Alt-g) and press update preview, you will now see exactly where the part is going to be milled at the surface of you machine bed.

Now press the F4 key or the run button to start milling, go to the graphic screen and switch real time graph on to see what the machine is doing.

That's all for this tutorial, happy milling!

2.8 KEYBOARD SHORTCUTS

Besides the already explained keys for jogging etc, there are a few extra, these are special for pendant builders.

```
 Alt-gr + g = run/pause

        + h = run
        + i = pause

        + r = reset
        + s = start/stop spindle
        + m = start/stop mist coolant
        + f = start/stop flood coolant
        + n = start/stop aux 1
        + 1 = +feed
        + 2 = -feed
        + 3 = +Speed
        + 4 = -Speed
        + 5 .. + 0 =  Zero axis 1 .. Zero axis 6

        + u = handwheel x1
        + v = handwheel x10
        + d = handwheel x100
        + w = handwheel off

        + x = handwheel on x
        + y = handwheel on y
        + z = handwheel on z
        + a = handwheel on a
        + b = handwheel on b
        + c = handwheel on c

        + e = abort
        + j = home all
```

2.9 ZERO TOOL MACRO

User button 1 contains:

```
;Zero tool tip example
Sub user_1
     msg "user_1, Zero Z (G92) using toolsetter"
    (Start probe move, slow)
    f30
    g38.2 z-100
    (Move back to touch point)
    g0 z#5063
    (Set position, the measuring device is 43mm in height, adapt for your measuring device)
    G92 z43
    (move 5 mm above measuring device)
    g91 (incremental distance mode)
    g0 z5
    g90 (absolute distance mode)
    m30
Endsub
```

The idea is to use a flexible position tool setter and put it on top of the workpiece.
Start this function and when done, the Z coordinate is set to 0 at the surface of the workpiece.
The feed is set slow F30. A probe move G38.2 is started towards -Z, when the tool setter is touched
the position is stored and the movement is stopped. The machine moves exactly to the touch point.
G92 is used with a Z value that specifies the height of you tool setter 43 mm in this case. Change to
match your tool setter. An incremental movement is started 5 mm upwards, so you can remove the
tool setter. The machine goes back to absolute mode and is done.

WARNING: Take care that you fully understand this macro before using it and that you have adapted
it for your toolsetter.

2.10 TOOL MEASUREMENT MACRO

Under user menu button 2 you'll see this:
;Tool length measurement example
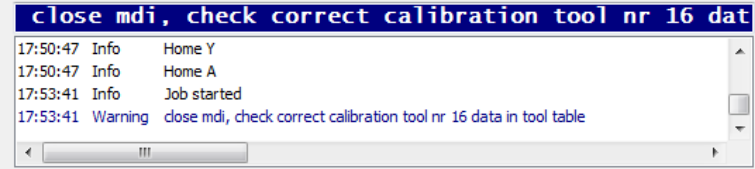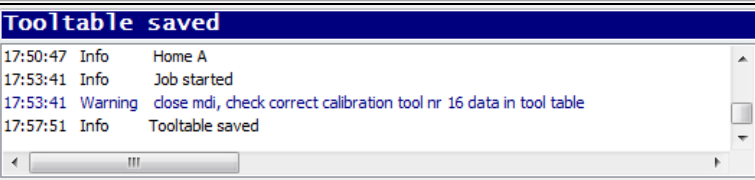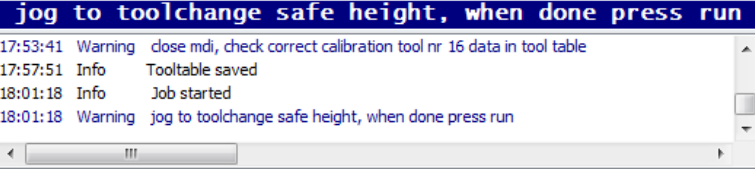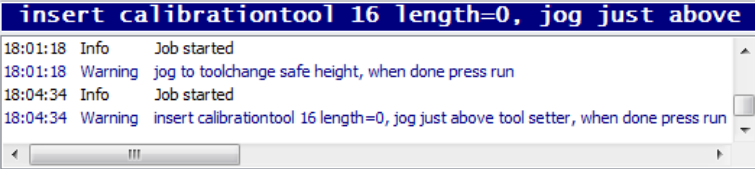
```
Sub user_2
    goSub m_tool ;See sub m_tool
Endsub
```
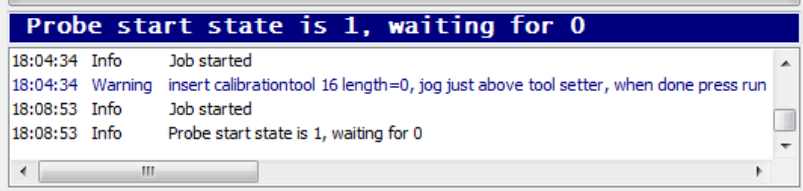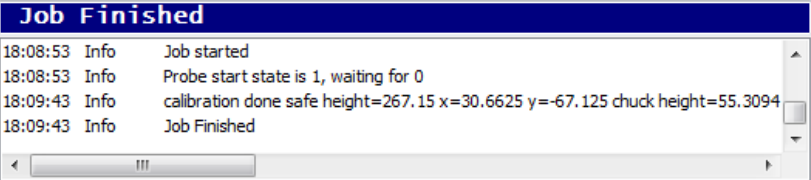
The user 2 button calls subroutine **m_tool**.
This subroutine needs a few values that are stored:

#4996, Z coordinate at tool change safe height
#4997, X coordinate for tool change
#4998, Y coordinate for tool change
#4999, Z coordinate at tool length equals zero or calibration tool height.
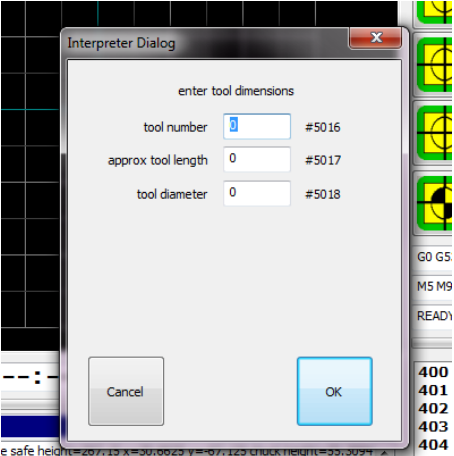
Tool #16 is used as reference tool and should have filled in its tool length before you start. This tool length can be 0 if you use the tool-chuck itself instead of a calibration tool.

The values #4996 .. #4999 are to be determined once. This can be done using the **calibrate_tool_setter** function below. Make sure the machine is homed before you start this.

Your action	Machine message
Open MDI and type : gosub calibrate_tool_setter Press enter to execute.	**close mdi, check correct calibration tool nr 16 dat** 17:50:47 Info Home Y 17:50:47 Info Home A 17:53:41 Info Job started 17:53:41 Warning close mdi, check correct calibration tool nr 16 data in tool table
Close the MDI window using F6, Go to the tools tab and check the tool length of tool 16. For me it is 0 because I use the tool chuck without calibration tool. Press save changes. and go back to the operate tab.	**Tooltable saved** 17:50:47 Info Home A 17:53:41 Info Job started 17:53:41 Warning close mdi, check correct calibration tool nr 16 data in tool table 17:57:51 Info Tooltable saved
The program is still inside subroutine **calibrate_tool_setter.** Press RUN to continue.	**jog to toolchange safe height, when done press run** 17:53:41 Warning close mdi, check correct calibration tool nr 16 data in tool table 17:57:51 Info Tooltable saved 18:01:18 Info Job started 18:01:18 Warning jog to toolchange safe height, when done press run
Do what the message says: Jog Z to safe height. In my case this is completely up. Press RUN to continue	**insert calibrationtool 16 length=0, jog just above** 18:01:18 Info Job started 18:01:18 Warning jog to toolchange safe height, when done press run 18:04:34 Info Job started 18:04:34 Warning insert calibrationtool 16 length=0, jog just above tool setter, when done press run

Do what the messages says: **Insert the calibration tool** if you have one, or just leave the tool chuck empty. **Jog using X, Y, Z just above the tool setter.** When done **jogging, press RUN**.	**Probe start state is 1, waiting for 0** 18:04:34 Info Job started 18:04:34 Warning insert calibrationtool 16 length=0, jog just above tool setter, when done press run 18:08:53 Info Job started 18:08:53 Info Probe start state is 1, waiting for 0 The machine will move down to touch the tool setter, The measured tool-chuck height is stored into #4999. Then the Z is moved up to safe height. **Job Finished** 18:08:53 Info Job started 18:08:53 Info Probe start state is 1, waiting for 0 18:09:43 Info calibration done safe height=267.15 x=30.6625 y=-67.125 chuck height=55.3094 18:09:43 Info Job Finished
Calibration DONE **We need to do this only once.** **You need to do this again if you have changed something that influences the calibrated data.**	

When all calibrated, **the user button F2** can be used to measure the tool length.
Make sure the correct tool is loaded before you start.

Press USER BUTTON 2.	The machine moves to safe height. The dialog is shown: Interpreter Dialog enter tool dimensions tool number 0 #5016 approx tool length 0 #5017 tool diameter 0 #5018 Cancel OK
Type correct values for tool number, tool length and diameter. **Press OK**.	The machine moves to the correct X,Y The machine moves 10 mm above the tool setter. So make sure the approx tool length above is OK. The machine does the move towards the tool setter, Then calculates and stores the values. Then machine moves Z to safe height.
Tool Length measurement Complete.	


```
Sub calibrate tool setter
    warnmsg "close MDI, check correct calibration tool nr 16 data in tool table"
    warnmsg "jog to toolchange safe height, when done press RUN"
    #4996=#5073 ;Store toolchange safe height machine coordinates
    warnmsg "insert calibrationtool 16 length=" #5416 ", jog just above tool setter, when done press RUN"
    ;store x y in non volatile parameters (4000 - 4999)
    #4997=#5071 ;machine pos X
    #4998=#5072 ;machine pos Y
    ;Determine minimum toochuck height and store into #4999
    g38.2 g91 z-20 f30
    #4999=[#5053 - #5416] ;probepos Z - calibration tool length = toolchuck height
    g90
    g0 g53 z#4996
    msg "calibration done safe height="#4996 " X="#4997 " Y="#4998 " Chuck height="#4999
endSub

sub m_tool
    ;Check if toolsetter is calibrated
    if [[#4996 == 0] and [#4997 == 0] and [#4998 == 0] and [#4999 == 0]]
        errmsg "calibrate toolsetter first open mdi, enter gosub calibrate tool setter"
    else
        g0 g53 z#4996 ; move to safe z
        dlgmsg "enter tool dimensions" "tool number" 5016 "approx tool length" 5017 "tool diameter" 5018

        if [[#5016 < 1] OR [#5016 > 15]]
            ErrMsg "Tool must be in range of 0 .. 15"
        endif

        ;move to toolsetter coordinates
        g00 g53 x#4997 y#4998
        ;move to 10mm above chuck height + approx tool length + 10
        g00 g53 z[#4999+10+#5017]
            ;measure tool length and pull 5mm back up
        g38.2 g91 z-20 f30
        g90
        ;back to safe height
        g0 g53 z#4996
        ;Store tool length, diameter in tool table
        #[5400 + #5016] = [#5053-#4999]
        #[5500 + #5016] = #5018
        #[5600 + #5016] = 0 ;Tool X offset is 0
        msg "tool length measured="#[5400 + #5016]" stored at tool "#5016
    endif
endsub
```

3 Input: the RS274/NGC Language

This section describes the input language, RS274/NGC.

3.1 OVERVIEW

The RS274/NGC language is based on lines of code. Each line (also called a "block") may include commands to a machining center to do several different things. Lines of code may be collected in a file to make a program.

A typical line of code consists of an optional line number at the beginning followed by one or more "words." A word consists of a letter followed by a number (or something that evaluates to a number). A word may either give a command or provide an argument to a command. For example, "G1 X3" is a valid line of code with two words. "G1" is a command meaning "move in a straight line at the programmed feed rate," and "X3" provides an argument value (the value of X should be 3 at the end of the move). Most RS274/NGC commands start with either G or M (for miscellaneous). The words for these commands are called "G codes" and "M codes."

The RS274/NGC language has no indicator for the start of a program. The RS274/NGC language has two commands (M2 or M30), either of which ends a program.

3.2 RS274/NGC LANGUAGE VIEW OF A MACHINING CENTER

The RS274/NGC language is based on a particular view of what a machining center to be controlled is like. The view is as described in Section 2.1, with the changes described below. The RS274/NGC language view includes one mechanical component not known to the canonical machining functions: a cycle start button. The use of the button is described in Section 3.6.1.

The RS274/NGC language contains commands that change the way subsequent commands are to be interpreted, but do not tell the machining center to do anything. These are not covered in this section, but are dealt with as they arise in Section 3.5.17, Section 3.5.19, and Section 3.5.20.

3.2.1 Parameters/Variables

In the RS274/NGC language view, a machining center maintains an array of 5400 numerical parameters. Many of them have specific uses. The parameter array should persist over time, even if the machining center is powered down. USBCN stores the parameters that have specific use only. This is performed when the user presses the "Save Fixtures" button in the variable view. The specific parameters are listed in the table below. Other parameters in range of 1..5400 are free to use in your G-Code program.

A simple example of usage:
#1=100 ; assign the value 100 to variable #1
G0[#1] ; use #1 to move to 100

Parameter number	Meaning
1-4999	Free to use, note that 4996 – 4999 are used by the tool length measurement function under user button 2.
4000-4999	Free to use, persistent.
5001-5006	POS X – C, interpreter position = work position
5008	Actual TOOL #
5009	Actual TOOL Radius
5010	Actual TOOL Z offset
5011	New tool during tool change
5012	Actual tool X offset
5013	Actual G43 Z offset
5014	Actual G43 X offset
5015..5050	Used in tool change sub routine
5051 - 5056	Probe position X – C in machine coordinates
5061 - 5066	Probe position X – C in work coordinates
5067	1 if probe is triggered after G38.2, 0 otherwise.
5068	Actual Probe value
5069	Handwheel counter.
5071-5076	POS X – C, interpreter position without offsets = Machine position.
5161 - 5166	G28 home X - C
5181 - 5186	G30 home X - C
5211 - 5216	G92 offset X - C
5220	Coord. System number
5221 - 5226	Coord. System 1 X – C
5241 - 5246	Coord. System 2 X – C
5261 - 5266	Coord. System 3 X – C
5281 - 5286	Coord. System 4 X – C
5301 - 5306	Coord. System 5 X – C
5321 - 5326	Coord. System 6 X – C
5341 - 5346	Coord. System 7 X – C
5361 - 5366	Coord. System 8 X – C
5381 - 5386	Coord. System 9 X – C
5398	Return value for dlgmsg (+1 OK, -1 Cancel)
5399	Return value for M55, M56
5401 – 5416	Tool z offset (Length) Tool 1 – Tool 16
5501 – 5516	Tool diameter Tool 1 – Tool 16
5601 – 5616	Tool x offset (for Turning) Tool 1 – Tool 16
5701 – 5716	Tool orientation (for Turning) Tool 1 – Tool 16
	(Currently supported only Tool 0 .. Tool 16)

3.2.2 Tool data

Tool ID	zOffset (Length)	xOffset (For turning)	Diameter	orientation
1				1-9
2				1-9
..				..
16				
				1-9

3.2.2.1 TOOL ORIENTATION FOR LATHES

When the G18 plane (X-Z) is selected, special LATHE tool radius compensation can be used (G41, G42).
Depending on the tool orientation and tool radius an extra offset is applied.
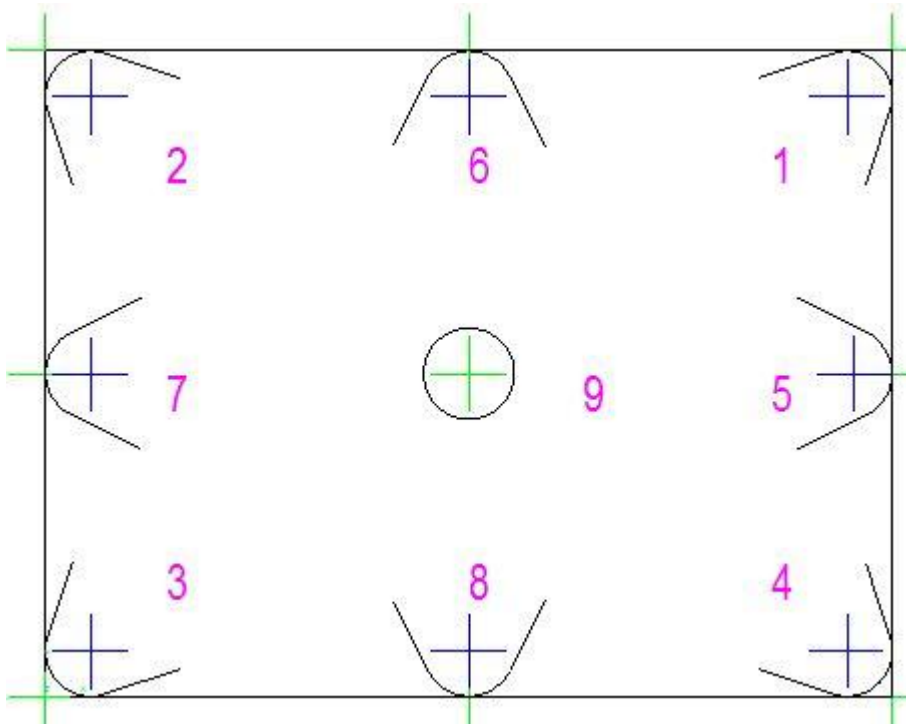
The blue crosses show the radius center of the tool.
The green crosses show the controlled point depending on the tool orientation. For orientation = 9 there is no offset compensation. For orientation = 2, the compensation in X is –tool radius, in Z also –tool radius.

3.2.3 Coordinate Systems

In the RS274/NGC language view, a machining center has an absolute coordinate system and nine program coordinate systems.

You can set the offsets of the nine program coordinate systems using G10 L2 Pn (n is the number of the coordinate system) with values for the axes in terms of the absolute coordinate system.

You can select one of the nine systems by using G54, G55, G56, G57, G58, G59, G59.1, G59.2, or G59.3 (see Section 3.5.13). It is not possible to select the absolute coordinate system directly.

You can offset the current coordinate system using G92 or G92.3. This offset will then apply to all nine program coordinate systems. This offset may be cancelled with G92.1 or G92.2. See Section 3.5.18.
You can make straight moves in the absolute machine coordinate system by using G53 with either G0 or G1.

Data for coordinate systems is stored in parameters, see the previous section. During initialization, the coordinate system is selected that is specified by parameter 5220. A value of 1 means the first coordinate system (the one G54 activates), a value of 2 means the second coordinate system (the one G55 activates), and so on. It is an error for the value of parameter 5220 to be anything but a whole number between one and nine.

3.3 FORMAT OF A LINE

A permissible line of input RS274/NGC code consists of the following, in order, with the restriction that there is a maximum (currently 256) to the number of characters allowed on a line.
- An optional line number.
- Any number of words, parameter settings, and comments.

Any input not explicitly allowed is illegal and will cause the Interpreter to signal an error.
Spaces and tabs are allowed anywhere on a line of code and do not change the meaning of the line, except inside comments. This makes some strange-looking input legal. The line "g0x +0. 12 34y 7" is equivalent to "g0 x+0.1234 y7", for example.

Blank lines are allowed in the input. They are to be ignored.
Input is case insensitive.

3.3.1   Line Number

A line number is the letter N followed by an integer (with no sign) between 0 and 99999 written with no more than five digits (000009 is not OK, for example). Line numbers may be repeated or used out of order, although normal practice is to avoid such usage.
Line numbers may also be skipped, and that is normal practice. A line number is not required to be used, but must be in the proper place if used.

3.3.2   Word

A word is a letter other than N followed by a real value.
Words may begin with any of the letters shown in Table 3-2. The table includes N for completeness, even though, as defined above, line numbers are not words. Several letters (I, J, K, L, P, and R) may have different meanings in different contexts.

Letter	Meaning
A	A-axis of machine
D	Tool radius compensation number
F	Feed rate
G	General function (see Table 3-4)
H	Tool length offset index
I	X-axis offset for arcs X offset in G87 canned cycle
J	Y-axis offset for arcs Y offset in G87 canned cycle
K	Z-axis offset for arcs Z offset in G87 canned cycle
L	number of repetitions in canned cycles key used with G10
M	miscellaneous function (see Table 3-6)
N	line number
P	dwell time in canned cycles dwell time with G4 key used with G10
Q	feed increment in G83 canned cycle
R	arc radius, clear_z distance in canned cycle
S	spindle speed
T	tool selection
X	X-axis of machine
Y	Y-axis of machine
Z	Z-axis of machine
A	A-axis of machine
B	B-axis of machine
C	C-axis of machine

A real value is some collection of characters that can be processed to come up with a number. A real value may be an explicit number (such as 341 or -0.8807), a parameter value, an expression, or a unary operation value. Definitions of these follow immediately. Processing characters to come up with a number is called "evaluating". An explicit number evaluates to itself.

3.3.2.1 NUMBER
The following rules are used for (explicit) numbers. In these rules a digit is a single character between 0 and 9.

- A number consists of (1) an optional plus or minus sign, followed by (2) zero to many digits, followed, possibly, by (3) one decimal point, followed by (4) zero to many digits - provided that there is at least one digit somewhere in the number.
- There are two kinds of numbers: integers and decimals. An integer does not have a decimal point in it; a decimal does.
- Numbers may have any number of digits, subject to the limitation on line length. Only about seventeen significant figures will be retained, however (enough for all known applications).

- A non-zero number with no sign as the first character is assumed to be positive.

Notice that initial (before the decimal point and the first non-zero digit) and trailing (after the decimal point and the last non-zero digit) zeros are allowed but not required.
A number written with initial or trailing zeros will have the same value when it is read as if the extra zeros were not there.

Numbers used for specific purposes in RS274/NGC are often restricted to some finite set of values or some to some range of values. In many uses, decimal numbers must be close to integers; this includes the values of indexes (for parameters and carousel slot numbers, for example), M codes, and G codes multiplied by ten. A decimal number which is supposed be close to an integer is considered close enough if it is within 0.0001 of an integer.

3.3.2.2 PARAMETER VALUE
A parameter value is the pound character # followed by a real value. The real value must evaluate to an integer between 1 and 5399. The integer is a parameter number, and the value of the parameter value is whatever number is stored in the numbered parameter.
The # character takes precedence over other operations, so that, for example, "#1+2" means the number found by adding 2 to the value of parameter 1, not the value found in parameter 3. Of course, #[1+2] does mean the value found in parameter 3. The # character may be repeated; for example ##2 means the value of the parameter whose index is the (integer) value of parameter 2.

3.3.2.3 EXPRESSIONS AND BINARY OPERATIONS
An expression is a set of characters starting with a left bracket [and ending with a balancing right bracket]. In between the brackets are numbers, parameter values, mathematical operations, and other expressions. An expression may be evaluated to produce a number. The expressions on a line are evaluated when the line is read, before
anything on the line is executed. An example of an expression is [1 + acos[0] - [#3 ** [4.0/2]]].

Binary operations appear only inside expressions. Nine binary operations are defined. There are four basic mathematical operations: addition (+), subtraction (-), multiplication (*), and division (/). There are three logical operations: non-exclusive or (OR), exclusive or (XOR), and logical and (AND). The eighth operation is the modulus operation (MOD). The ninth operation is the "power" operation (**) of raising the number on the left of the operation to the power on the right.

The binary operations are divided into three groups. The first group is: power. The second group is: multiplication, division, and modulus. The third group is: addition, subtraction, logical non-exclusive or, logical exclusive or, and logical and. If operations are strung together (for example in the expression [2.0 / 3 * 1.5 - 5.5 / 11.0]), operations in the first group are to be performed before operations in the second group and operations in the second group before operations in the third group. If an expression contains more than one operation from the same group (such as the first / and * in the example), the operation on the left is performed first.

Thus, the example is equivalent to: [((2.0 / 3) * 1.5) - (5.5 / 11.0)] , which simplifies to [1.0 - 0.5] , which is 0.5.

The logical operations and modulus are to be performed on any real numbers, not just on integers. The number zero is equivalent to logical false, and any non-zero number is equivalent to logical true.

3.3.2.4 UNARY OPERATION VALUE
A unary operation value is either "ATAN" followed by one expression divided by another expression (for example "ATAN[2]/[1+3]") or any other unary operation name followed by an expression (for example "SIN[90]"). The unary operations are: ABS (absolute value), ACOS (arc cosine), ASIN (arc sine), ATAN (arc tangent), COS (cosine), EXP (e raised to the given power), FIX (round down), FUP (round up), LN (natural logarithm), ROUND (round to the nearest whole number), SIN (sine), SQRT (square root), and TAN (tangent). Arguments to unary operations which take angle measures (COS, SIN, and TAN) are in degrees. Values returned by unary operations which return angle measures (ACOS, ASIN, and ATAN) are also in degrees.

The FIX operation rounds towards the left (less positive or more negative) on a number line, so that FIX[2.8] =2 and FIX[-2.8] = -3, for example. The FUP operation rounds towards the right (more positive or less negative) on a number line; FUP[2.8] = 3 and FUP[-2.8] = -2, for example.

3.3.3 Parameter Setting
A parameter setting is the following four items one after the other: (1) a pound character # , (2) a real value which evaluates to an integer between 1 and 5399, (3) an equal sign
= , and (4) a real value. For example "#3 = 15" is a parameter setting meaning "set parameter 3 to 15."

A parameter setting does not take effect until after all parameter values on the same line have been found. For example, if parameter 3 has been previously set to 15 and the line "#3=6 G1 x#3" is interpreted, a straight move to a point where x equals 15 will occur and the value of parameter 3 will be 6.

3.3.4 Comments and Messages
Printable characters and white space inside parentheses is a comment. A left parenthesis always starts a comment. The comment ends at the first right parenthesis found thereafter. Once a left parenthesis is placed on a line, a matching right parenthesis must appear before the end of the line. Comments may not be nested; it is an error if a left parenthesis is found after the start of a comment and before the end of the comment.
Here's an example of a line containing a comment: "G80 M5 (stop motion)". Comments do not cause a machining center to do anything.

A comment contains a message if "MSG," appears after the left parenthesis and before any other printing characters. Variants of "MSG," which include white space and lower case characters are allowed. The rest of the characters before the right

parenthesis are considered to be a message. Messages should be displayed on the message display device. Comments not containing messages need not be displayed there.

3.3.5   Item Repeats

A line may have any number of G words, but two G words from the same modal group (see Section 3.4) may not appear on the same line.

A line may have zero to four M words. Two M words from the same modal group may
not appear on the same line.

For all other legal letters, a line may have only one word beginning with that letter.

If a parameter setting of the same parameter is repeated on a line, "#3=15 #3=6", for example, only the last setting will take effect. It is silly, but not illegal, to set the same parameter twice on the same line.

If more than one comment appears on a line, only the last one will be used; each of the other comments will be read and its format will be checked, but it will be ignored thereafter. It is expected that putting more than one comment on a line will be very rare.

3.3.6   Item order

The three types of item whose order may vary on a line (as given at the beginning of this
section) are word, parameter setting, and comment. Imagine that these three types of item are divided into three groups by type.

The first group (the words) may be reordered in any way without changing the meaning of the line.

If the second group (the parameter settings) is reordered, there will be no change in the meaning of the line unless the same parameter is set more than once. In this case, only the last setting of the parameter will take effect. For example, after the line "#3=15 #3=6" has been interpreted, the value of parameter 3 will be 6. If the order is reversed to "#3=6 #3=15" and the line is interpreted, the value of parameter 3 will be 15.

If the third group (the comments) contains more than one comment and is reordered, only the last comment will be used.

If each group is kept in order or reordered without changing the meaning of the line, then the three groups may be interleaved in any way without changing the meaning of the line. For example, the line "g40 g1 #3=15 (foo) #4=-7.0" has five items and means exactly the same thing in any of the 120 possible orders (such as "#4=-7.0 g1 #3=15 g40 (foo)") for the five items.

3.3.7 Commands and Machine Modes

In RS274/NGC, many commands cause a machining center to change from one mode to another, and the mode stays active until some other command changes it implicitly or explicitly. Such commands are called "modal". For example, if coolant is turned on, it stays on until it is explicitly turned off. The G codes for motion are also modal. If a G1 (straight move) command is given on one line, for example, it will be executed again on
the next line if one or more axis words are available on the line, unless an explicit command is given on that next line using the axis words or cancelling motion.

"Non-modal" codes have effect only on the lines on which they occur. For example, G4 (dwell) is non-modal.

3.4 MODAL GROUPS

Modal commands are arranged in sets called "modal groups", and only one member of a modal group may be in force at any given time. In general, a modal group contains commands for which it is logically impossible for two members to be in effect at the same time - like measure in inches vs. measure in millimeters. A machining center may be in many modes at the same time, with one mode from each modal group being in effect. The modal groups are shown in Table 3-3.

Table 3-3 Modal Groups

The modal groups for G codes are:
group 1 = {G0, G1, G2, G3, G38.2, G76, G80, G81,   G82, G83, G84, G85, G86, G87, G88, G89} motion group 2 = {G17, G18, G19} plane selection group 3 = {G90, G91} distance mode group 5 = {G93, G94} feed rate mode group 6 = {G20, G21} units group 7 = {G40, G41, G42} cutter radius compensation group 8 = {G43, G49} tool length offset group 10 = {G98, G99} return mode in canned cycles group 12 = {G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3} coordinate system selection group 13 = {G61, G61.1, G64} path control mode
The modal groups for M codes are: group 4 = {M0, M1, M2, M30, M60} stopping group 5 = {54, M55, M56, M64, M65, M66} AUX and general purpose I/O group 6 = {M6} tool change group 7 = {M3, M4, M5} spindle turning group 8 = {M7, M8, M9} coolant (special case: M7 and M8 may be active at the same time) group 9 = {M48, M49} enable/disable feed and speed override switches
In addition to the above modal groups, there is a group for non-modal G codes: group 0 = {G4, G10, G28, G30, G53, G92, G92.1, G92.2, G92.3}

For several modal groups, when a machining center is ready to accept commands, one member of the group must be in effect. There are default settings for these modal groups. When the machining center is turned on or otherwise re-initialized, the default values are automatically in effect.

Group 1, the first group on the table, is a group of G codes for motion. One of these is always in effect. That one is called the current motion mode.

It is an error to put a G-code from group 1 and a G-code from group 0 on the same line if both of them use axis words. If an axis word-using G-code from group 1 is implicitly in effect on a line (by having been activated on an earlier line), and a group 0 G-code that uses axis words appears on the line, the activity of the group 1 G-code is suspended for that line. The axis word-using G-codes from group 0 are G10, G28, G30, and G92.

3.5 G CODES

G codes of the RS274/NGC language are shown in Table 3-4 and described in this Section.

The descriptions contain command prototypes, set in **bold** type.

In the command prototypes, three dots (…) stand for a real value. As described earlier, a real value may be (1) an explicit number, 4, for example, (2) an expression**, [2+2]**, for example, (3) a parameter value, **#88**, for example, or (4) a unary function value, acos[0], for example.

In most cases, if axis words (any or all of **X…, Y…, Z…, A…, B…, C…**) are given, they specify a destination point. Axis numbers are in the currently active coordinate system, unless explicitly described as being in the absolute coordinate system. Where axis words are optional, any omitted axes will have their current value. Any items in the command prototypes not explicitly described as optional are required. It is an error if a required item is omitted.

In the prototypes, the values following letters are often given as explicit numbers. Unless stated otherwise, the explicit numbers can be real values. For example, **G10 L2** could equally well be written **G[2*5] L[1+1]**. If the value of parameter 100 were 2, **G10 L#100** would also mean the same. Using real values which are not explicit numbers as just shown in the examples is rarely useful.

If **L…** is written in a prototype the "…" will often be referred to as the "L number". Similarly the "…" in **H…** may be called the "H number", and so on for any other letter.

3.5.1   Rapid Linear Motion - G0

For rapid linear motion, program **G0 X… Y… Z… A…**, where all the axis words are optional, except that at least one must be used. The G0 is optional if the current motion mode is G0. This will produce coordinated linear motion to the destination point at the current traverse rate (or slower if the machine will not go that fast). It is expected that cutting will not take place when a G0 command is executing.

It is an error if:
• All axis words are omitted.

If cutter radius compensation is active, the motion will differ from the above; see Appendix A. If G53 is programmed on the same line, the motion will also differ; see Section 3.5.12.

Table 3-4 G Codes

G Code	Meaning
G0	rapid positioning
G1	linear interpolation
G2	circular/helical interpolation (clockwise)
G3	circular/helical interpolation (counterclockwise)
G4	dwell
G10	coordinate system origin setting
G17	XY-plane selection
G18	XZ-plane selection
G19	YZ-plane selection
G20	inch system selection
G21	millimeter system selection
G28	move to park position 1, setup on variable page
G30	move to park position 2, setup on variable page
G33	Lathe, motion synchronized to spindle
G38.2	straight probe
G40	cancel cutter radius compensation
G41	start cutter radius compensation left
G42	start cutter radius compensation right
G43	tool length offset (plus) , tool X offset for lathe
G49	cancel tool length offset
G53	motion in machine coordinate system
G54	use preset work coordinate system 1
G55	use preset work coordinate system 2
G56	use preset work coordinate system 3
G57	use preset work coordinate system 4
G58	use preset work coordinate system 5
G59	use preset work coordinate system 6
G59.1	use preset work coordinate system 7
G59.2	use preset work coordinate system 8
G59.3	use preset work coordinate system 9
G61	set path control mode: exact path
G61.1	set path control mode: exact stop
G64	set path control mode: continuous
G76	Lathe, threading
G80	cancel motion mode (including any canned cycle)
G81	canned cycle: drilling
G82	canned cycle: drilling with dwell
G83	canned cycle: peck drilling
G84	canned cycle: right hand tapping
G85	canned cycle: boring, no dwell, feed out
G86	canned cycle: boring, spindle stop, rapid out
G87	canned cycle: back boring
G88	canned cycle: boring, spindle stop, manual out
G89	canned cycle: boring, dwell, feed out
G90	absolute distance mode
G91	incremental distance mode
G92	offset coordinate systems and set parameters
G92.1	cancel offset coordinate systems and set parameters to zero
G92.2	cancel offset coordinate systems but do not reset parameters
G92.3	apply parameters to offset coordinate systems
G93	inverse time feed rate mode
G94	units per minute feed rate mode
G98	initial level return in canned cycles
G99	R-point level return in canned cycles

3.5.2   Linear Motion at Feed Rate - G1

For linear motion at feed rate (for cutting or not), program **G1 X… Y… Z… A…**,
where all the axis words are optional, except that at least one must be used. The G1
is optional if the current motion mode is G1. This will produce coordinated linear
motion to the destination point at the current feed rate (or slower if the machine will
not go that fast).

It is an error if:
- All axis words are omitted.

If cutter radius compensation is active, the motion will differ from the above; see
Appendix A. If G53 is programmed on the same line, the motion will also differ; see
Section 3.5.12.

3.5.3   Arc at Feed Rate - G2 and G3

A circular or helical arc is specified using either G2 (clockwise arc) or G3
(counterclockwise arc). The axis of the circle or helix must be parallel to the X, Y, or
Z-axis of the machine coordinate system. The axis (or, equivalently, the plane
perpendicular to the axis) is selected with G17 (Z-axis, XY-plane), G18 (Y-axis, XZ-
plane), or G19 (X-axis, YZ-plane). If the arc is circular, it lies in a plane parallel to
the selected plane.

If a line of RS274/NGC code makes an arc and includes rotational axis motion, the
rotational axes turn at a constant rate so that the rotational motion starts and
finishes when the XYZ motion starts and finishes. Lines of this sort are hardly ever
programmed.
If cutter radius compensation is active, the motion will differ from what is described
here. See Appendix A.

Two formats are allowed for specifying an arc. We will call these the center format
and the radius format. In both formats the G2 or G3 is optional if it is the current
motion mode.

3.5.3.1   RADIUS FORMAT ARC

In the radius format, the coordinates of the end point of the arc in the selected plane
are specified along with the radius of the arc. Program **G2 X… Y… Z… A… R…** (or
use G3 instead of G2). R is the radius. The axis words are all optional except that at
least one of the two words for the axes in the selected plane must be used. The R
number is the radius. A positive radius indicates that the arc turns through 180
degrees or less, while a negative radius indicates a turn of 180 degrees to 359.999
degrees. If the arc is helical, the value of the end point of the arc on the coordinate
axis parallel to the axis of the helix is also specified.

It is an error if:
- both of the axis words for the axes of the selected plane are omitted,
- the end point of the arc is the same as the current point.

It is not good practice to program radius format arcs that are nearly full circles or are semicircles (or nearly semicircles) because a small change in the location of the end point will produce a much larger change in the location of the center of the circle (and, hence, the middle of the arc). The magnification effect is large enough that rounding error in a number can produce out-of-tolerance cuts. Nearly full circles are outrageously bad, semicircles (and nearly so) are only very bad. Other size arcs (in the range tiny to 165 degrees or 195 to 345 degrees) are OK.

Here is an example of a radius format command to mill an arc: **G17 G2 x 10 y 15 r 20 z 5**.

That means to make a clockwise (as viewed from the positive Z-axis) circular or helical arc whose axis is parallel to the Z-axis, ending where X=10, Y=15, and Z=5, with a radius of 20. If the starting value of Z is 5, this is an arc of a circle parallel to the XY-plane; otherwise it is a helical arc.

3.5.3.2 Center Format Arc

In the center format, the coordinates of the end point of the arc in the selected plane are
specified along with the offsets of the center of the arc from the current location. In this format, it is OK if the end point of the arc is the same as the current point. It is an error if:

- When the arc is projected on the selected plane, the distance from the current point to the center differs from the distance from the end point to the center by more than 0.0002 inch (if inches are being used) or 0.002 millimeter (if millimeters are being used).

When the XY-plane is selected, program **G2 X… Y… Z… A… I… J…** (or use G3 instead of G2). The axis words are all optional except that at least one of X and Y must be used. I and J are the offsets from the current location (in the X and Y directions, respectively) of the center of the circle. I and J are optional except that at least one of the two must be used. It is an error if:
- X and Y are both omitted,
- I and J are both omitted.

When the XZ-plane is selected, program **G2 X… Y… Z… A… I… K…** (or use G3 instead of G2). The axis words are all optional except that at least one of X and Z must be used. I and K are the offsets from the current location (in the X and Z directions, respectively) of the center of the circle. I and K are optional except that at least one of the two must be used. It is an error if:
- X and Z are both omitted,
- I and K are both omitted.

When the YZ-plane is selected, program **G2 X… Y… Z… A… B… C… J… K…** (or use G3 instead of G2). The axis words are all optional except that at least one of Y and Z must be used. J and K are the offsets from the current location (in the Y and Z directions, respectively) of the center of the circle. J and K are optional except that at least one of the two must be used. It is an error if:
- Y and Z are both omitted,
- J and K are both omitted.

Here is an example of a center format command to mill an arc:
G17 G2 x10 y16 i3 j4 z9.

That means to make a clockwise (as viewed from the positive z-axis) circular or helical arc whose axis is parallel to the Z-axis, ending where X=10, Y=16, and Z=9, with its center offset in the X direction by 3 units from the current X location and offset in the Y direction by 4 units from the current Y location. If the current location has X=7, Y=7 at the outset, the center will be at X=10, Y=11. If the starting value of Z is 9, this is a circular arc; otherwise it is a helical arc. The radius of this arc would be 5.

In the center format, the radius of the arc is not specified, but it may be found easily as the distance from the center of the circle to either the current point or the end point of the arc.

3.5.4   Dwell - G4

For a dwell, program G4 P… . This will keep the axes unmoving for the period of time in seconds specified by the P number. It is an error if:
*   the P number is negative.

3.5.5   Set Coordinate System Data -G10

To set the coordinate values for the origin of a coordinate system, program
G10 L2 P … X… Y… Z… A…, where the P number must evaluate to an integer in the range 1 to 9 (corresponding to G54 to G59.3) and all axis words are optional. The coordinates of the origin of the coordinate system specified by the P number are reset to the coordinate values given (in terms of the absolute coordinate system). Only those coordinates for which an axis word is included on the line will be reset.

It is an error if:
*   the P number does not evaluate to an integer in the range 1 to 9.

If origin offsets (made by G92 or G92.3) were in effect before G10 is used, they will continue to be in effect afterwards.

The coordinate system whose origin is set by a G10 command may be active or inactive at the time the G10 is executed.
Example:**G10 L2 P1 x 3.5 y 17.2** sets the origin of the first coordinate system (the one selected by G54) to a point where X is 3.5 and Y is 17.2 (in absolute coordinates). The Z coordinate of the origin (and the coordinates for any rotational axes) are whatever those coordinates of the origin were before the line was executed.

3.5.6   Plane Selection - G17, G18, and G19

Program **G17** to select the XY-plane, **G18** to select the XZ-plane, or **G19** to select the YZ-plane. The effects of having a plane selected are discussed in Section 3.5.3 and Section 3.5.16.

3.5.7   Length Units - G20/G21 and G70/G71

Program G20 to use inches for length units. Program G21 to use millimeters.
It is usually a good idea to program either G20 or G21 near the beginning of a program before any motion occurs, and not to use either one anywhere else in the program. It is the responsibility of the user to be sure all numbers are appropriate for use with the current length units. G70/G71 is added for CAM software compatibility.

3.5.8   Return to Home - G28 and G30

Two home positions are defined (by parameters 5161-5166 for G28 and parameters

5181-5186 for G30). The parameter values are in terms of the absolute coordinate system, but are in unspecified length units.
To return to home position by way of the programmed position, program
G28 X… Y… Z… A… (or use G30). All axis words are optional. The path is made by a traverse move from the current position to the programmed position, followed by a traverse move to the home position. If no axis words are programmed, the intermediate point is the current point, so only one move is made.

3.5.9 G33, G33.1 Spindle-Synchronized Motion

For spindle-synchronized motion in one direction, program **G33 X… Y… Z… K…**
where K gives the distance moved in XYZ for each revolution of the spindle. For instance, if starting at Z=0, G33 Z-1
K.0625 produces a 1 inch motion in Z over 16 revolutions of the spindle. This command might be
part of a program to produce a 16TPI thread.
A move to the specified coordinate, synchronized with the spindle at the given ratio and starting with a spindle index pulse

All the axis words are optional, except that at least one must be used.
It is an error if:
- all axis words are omitted.
- the spindle is not turning when this command is executed.
- the requested linear motion exceeds machine velocity limits due to the spindle speed.

3.5.10 Straight Probe - G38.2

3.5.10.1 THE STRAIGHT PROBE COMMAND

Program **G38.2 X… Y… Z… A…** to perform a straight probe operation. The rotational axis words are allowed, but it is better to omit them. If rotational axis words are used, the numbers must be the same as the current position numbers so that the rotational axes do not move. The linear axis words are optional, except that at least one of them must be used. The tool in the spindle must be a probe.
It is an error if:

*   the current point is less than 0.254 millimeter or 0.01 inch from the programmed point.
*   G38.2 is used in inverse time feed rate mode,
*   any rotational axis is commanded to move,
*   no X, Y, or Z-axis word is used.

In response to this command, the machine moves the controlled point (which should be at the end of the probe tip) in a straight line at the current feed rate toward the programmed point. If the probe trips, the probe is retracted slightly from the trip point at the end of command execution. If the probe does not trip even after overshooting the programmed point slightly, an error is signaled.

After successful probing, parameters 5061 to 5066 will be set to the program coordinates of the location of the controlled point at the time the probe tripped. The variables 5051 to 5056 will contain the machine coordinates. Useful for measuring tools in absolute machine positions. G53 G38.2 will move in machine coordinates.

3.5.10.2 USING THE STRAIGHT PROBE COMMAND

Using the straight probe command, if the probe shank is kept nominally parallel to the Z-axis (i.e., any rotational axes are at zero) and the tool length offset for the probe is used, so that the controlled point is at the end of the tip of the probe:

*   without additional knowledge about the probe, the parallelism of a face of a part to the XY-plane may, for example, be found.
*   if the probe tip radius is known approximately, the parallelism of a face of a part to the YZ or XZ-plane may, for example, be found.
*   if the shank of the probe is known to be well-aligned with the Z-axis and the probe tip radius is known approximately, the center of a circular hole, may, for example, be found.
*   if the shank of the probe is known to be well-aligned with the Z-axis and the probe tip radius is known precisely, more uses may be made of the straight probe command, such as finding the diameter of a circular hole.

If the straightness of the probe shank cannot be adjusted to high accuracy, it is desirable to know the effective radii of the probe tip in at least the +X, -X, +Y, and -Y directions. These quantities can be stored in parameters either by being included in the parameter file or by being set in an RS274/NGC program.
Using the probe with rotational axes not set to zero is also feasible. Doing so is more complex than when rotational axes are at zero, and we do not deal with it here.

3.5.10.3 EXAMPLE CODE

As a usable example, the code for finding the center and diameter of a circular hole is shown in Table 3-5. For this code to yield accurate results, the probe shank must be well-aligned with the Z-axis, the cross section of the probe tip at its widest point must be very circular, and the probe tip radius (i.e., the radius of the circular cross section) must be known precisely. If the probe tip radius is known only approximately (but the other conditions hold), the location of the hole center will still be accurate, but the hole diameter will not.

In Table 3-5, an entry of the form <description of number> is meant to be replaced by an actual number that matches the description of number. After this section of code has executed, the X-value of the center will be in parameter 1041, the Y-value of the center in parameter 1022, and the diameter in parameter 1034. In addition, the diameter parallel to the X-axis will be in parameter 1024, the diameter parallel to the Y-axis in parameter 1014, and the difference (an indicator of circularity) in parameter 1035. The probe tip will be in the hole at the XY center of the hole.

The example does not include a tool change to put a probe in the spindle. Add the tool change code at the beginning, if needed.

Table 3-5 Code to Probe Hole

```
N010 (probe to find center and diameter of circular hole)
N020 (This program will not run as given here. You have to)
N030 (insert numbers in place of <description of number>.)
N040 (Delete lines N020, N030, and N040 when you do that.)
N050 G0 Z <Z-value of retracted position> F <feed rate>
N060 #1001=<nominal X-value of hole center>
N070 #1002=<nominal Y-value of hole center>
N080 #1003=<some Z-value inside the hole>
N090 #1004=<probe tip radius>
N100 #1005=[<nominal hole diameter>/2.0 - #1004]
N110 G0 X#1001 Y#1002 (move above nominal hole center)
N120 G0 Z#1003 (move into hole - to be cautious, substitute G1 for G0 here)
N130 G38.2 X[#1001 + #1005] (probe +X side of hole)
N140 #1011=#5061 (save results)
N150 G0 X#1001 Y#1002 (back to center of hole)
N160 G38.2 X[#1001 - #1005] (probe -X side of hole)
N170 #1021=[[#1011 + #5061] / 2.0] (find pretty good X-value of hole center)
N180 G0 X#1021 Y#1002 (back to center of hole)
N190 G38.2 Y[#1002 + #1005] (probe +Y side of hole)
N200 #1012=#5062 (save results) N210 G0 X#1021 Y#1002 (back to center of hole)
N220 G38.2 Y[#1002 - #1005] (probe -Y side of hole)
N230 #1022=[[#1012 + #5062] / 2.0] (find very good Y-value of hole center)
N240 #1014=[#1012 - #5062 + [2 * #1004]] (find hole diameter in Y-direction)
N250 G0 X#1021 Y#1022 (back to center of hole)
N260 G38.2 X[#1021 + #1005] (probe +X side of hole)
N270 #1031=#5061 (save results)
N280 G0 X#1021 Y#1022 (back to center of hole)
N290 G38.2 X[#1021 - #1005] (probe -X side of hole)
N300 #1041=[[#1031 + #5061] / 2.0] (find very good X-value of hole center)
N310 #1024=[#1031 - #5061 + [2 * #1004]] (find hole diameter in X-direction)
N320 #1034=[[#1014 + #1024] / 2.0] (find average hole diameter)
N330 #1035=[#1024 - #1014] (find difference in hole diameters)
N340 G0 X#1041 Y#1022 (back to center of hole)
N350 M2 (that's all, folks)
```

3.5.11 Cutter Radius Compensation - G40, G41, G41.1, G42, G42.1

To turn cutter radius compensation off, program G40. It is OK to turn compensation off when it is already off. Cutter radius compensation may be performed only if the XY-plane is active.

To turn cutter radius compensation on **left** (i.e., the cutter stays to the left of the programmed path when the tool radius is positive), program **G41 D…** . To turn cutter radius compensation on **right** (i.e., the cutter stays to the right of the programmed path when the tool radius is positive), program **G42 D…** . The D word is optional; if there is no D word, the radius of the tool currently in the spindle will be used. If used, the D number should normally be the slot number of the tool in the spindle, although this is not required. It is OK for the D number to be zero; a radius value of zero will be used.

It is an error if:

- the D number is not an integer, is negative or is larger than the number of carousel slots,
- the XY-plane is not active or for turning the ZX plane is not active,
- cutter radius compensation is commanded to turn on when it is already on.
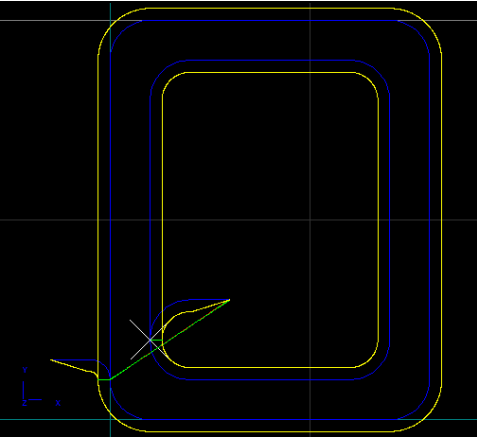
The behavior of the machining center when cutter radius compensation is on is described in Appendix A.

With G41.1 D… is the same as G41 D… except now the D number is not a tool number but a tool diameter.
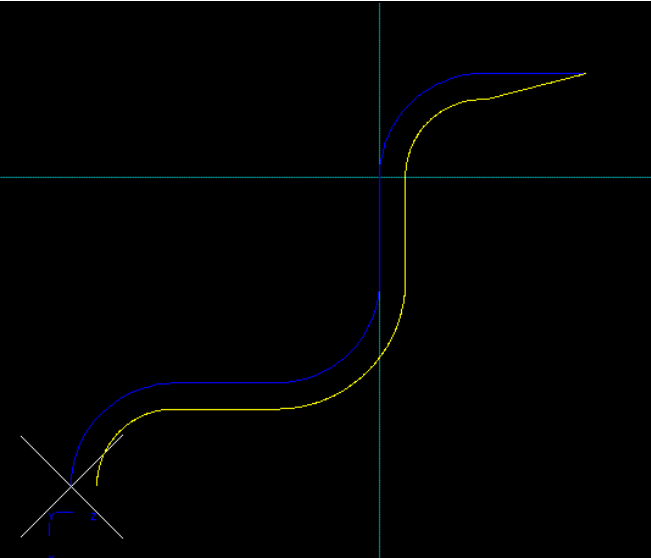
With G42.1 D… is the same as G42 D… except now the D number is not a tool number but a tool diameter.

3.5.11.1 EXAMPLE CODE FOR MILLING

This example mills out a rectangular object from the outside and inside.
On the outside we use G42, tool radius compensation right and for the inside G41, tool radius compensation left is used.
For both contours a tool-radius-compensation entry move is programmed consisting of a line which must be longer than the tool-radius used and a circle, of which also the radius is bigger than the tool.

By the way, all arc radii should be bigger than the tool radius. If you have inside corners, there should be always an arc, so that the tool fits.

```
g0 z3
g0 x-15 y15
f500
/g42.1 D6
g1 x-5      (cutter comp entry move 1)
g2 x0 y10 r5 (cutter comp entry move 2)
g1 z-3      (plunge down)
g3 x10 y0 r10
g1 x70
g3 x80 y10 r10
g1 y90
g3 x70 y100 r10
g1 x10
g3 x0 y90 r10
g1 x0 y10
/g40
g0 z3
g0 x30 y30
/g41.1 d6
g1 x20
g3 x10 y20 r10
g1 z-3
g3 x20 y10 r10
g1 x60
g3 x70 y20 r10
g1 y80
g3 x60 y90 r10
g1 x20
g3 x10 y80 r10
g1 y20
/g40
g0 z3
m30
```

The G42, G41 and G40 codes are programmed with a / (block delete sign) in front. This makes it easy to debug tool comp programs. The program is loaded with block delete on, this is the blue curve.
Then the program is run with block delete off resulting in the yellow curve.

It is clear to see what the entry move does.

3.5.11.2 EXAMPLE CODE FOR TURNING

	The movement starts at the right upper corner. The blue line is the programmed contour. The yellow is the contour with tool-radius compensation G41. The first G1 line is the tool comp entry move. You can get this figure by putting a / character in front of the G41/G40 codes. The load the program with block delete on and execute it with block delete off. With block delete on the tool comp is skipped.
(Diameter programming) (Use R word for Arcs) g0 x-20 z20 /g41.1 d5 g1 x-20 z10 g3 x0 z0 r10 g1 x20 g2 x40 z-10 r10 g1 z-20 g3 x60 z-30 r10 /g40 m30	(Radius programming) (Use R word for arc's) g0 x-10 z20 /g41.1 d5 g1 x-10 z10 g3 x0 z0 r10 g1 x10 g2 x20 z-10 r10 g1 z-20 g3 x30 z-30 r10 /g40 m30
(Diameter programming) (Use I,K programming for arc's) g0 x-20 z20 /g41.1 d5 g1 x-20 z10 g3 x0 z0 i10 k0 g1 x20 g2 x40 z-10 i0 k-10 g1 z-20 g3 x60 z-30 i10 k0 /g40 m30	(Radius programming) (Use I,K programming for arc's) g0 x-10 z20 /g41.1 d5 g1 x-10 z10 g3 x0 z0 i10 k0 g1 x10 g2 x20 z-10 i0 k-10 g1 z-20 g3 x30 z-30 i10 k0 /g40 m30

3.5.12 Tool Length Offsets - G43, G43.1, and G49

To use a tool length offset from the tool table, program **G43 H…**, where the H number is the desired index in the tool table. It is expected that all entries in this table will be positive. The H number should be, but does not have to be, the same as the slot number of the tool currently in the spindle. It is OK for the H number to be zero; an offset value of zero will be used.

If the H number is omitted, the actual tool in the spindle is used.

It is an error if:
- the H number is not an integer, is negative, or is larger than the number of carousel slots.

To use dynamic tool compensation (not from the tool-table), use G43.1 I.. K.. where I.. gives the tool X offset (turning) and K.. gives the tool Z offset (for turning and milling)

To use no tool length offset, program **G49**. It is OK to program using the same offset already in use. It is also OK to program using no tool length offset if none is currently being used.

#5401 - #5416 is the tool-Z (length) offset.
#5501 - #5516 is the tool-diameter (length).
#5601 - #5616 is the tool-X (width for turning) offset.
The variables can be modified runtime (in the G-Code file) if needed to compensate for tool-wear.

3.5.13 Move in Absolute Coordinates - G53

For linear motion to a point expressed in absolute coordinates, program **G1 G53 X… Y… Z… A…** (or use G0 instead of G1), where all the axis words are optional, except that at least one must be used. The G0 or G1 is optional if it is the current motion mode. G53 is not modal and must be programmed on each line on which it is intended to be active. This will produce coordinated linear motion to the programmed point. If G1 is active, the speed of motion is the current feed rate (or slower if the machine will not go that fast). If G0 is active, the speed of motion is the current traverse rate (or slower if the machine will not go that fast).

It is an error if:
- G53 is used without G0 or G1 being active,
- G53 is used while cutter radius compensation is on.

3.5.14 Select Coordinate System - G54 to G59.3

To select coordinate system 1, program G54, and similarly for other coordinate systems. The system-number-G-code pairs are: (1-G54), (2-G55), (3-G56), (4-G57), (5-G58), (6-G59), (7-G59.1), (8-G59.2), and (9-G59.3).

It is an error if:

- one of these G-codes is used while cutter radius compensation is on.

3.5.15 Set Path Control Mode - G61, and G64, or G64 Px

Some work pieces require absolute accuracy and some other require nonstop milling for best surface quality. One must understand that it is physically impossible to move around sharp corners without a standstill at the corner. This isn't possible with a car and also not with a CNC machine, since that would require infinite acceleration. The user can make a choice here, absolute accuracy with standstill at every corner (G61) or no standstill and corner round off with specified accuracy (G64Px.x).

G61 puts the machining center into exact path mode, In G61, the motion velocity between motion segments goes to zero, the end position in corners is exactly reached, use this if you require maximum accuracy. When a work piece consists of many small lines this gives a quite vibrating machine because of the continuous acceleration-deceleration-stop behavior.

G64 Px.x for continuous mode. In G64, subsequent moves are blended, when previous move starts to decelerate and reaches a velocity such that the specified accuracy isn't violated, the next move starts to accelerate, the two motions are added. The result is smooth motion with highest possible speed to achieve required accuracy. The corners however are rounded. specifies the distance reached to the corner while blending. The next move is blended with current such that the tool path remains no more than P from the corner. The figure below is a rectangle of 10x10 milled with F 2000. This is done with P values from 0.1 to 1, you can see the impact. This gives the best compromise between accuracy and smooth motion.
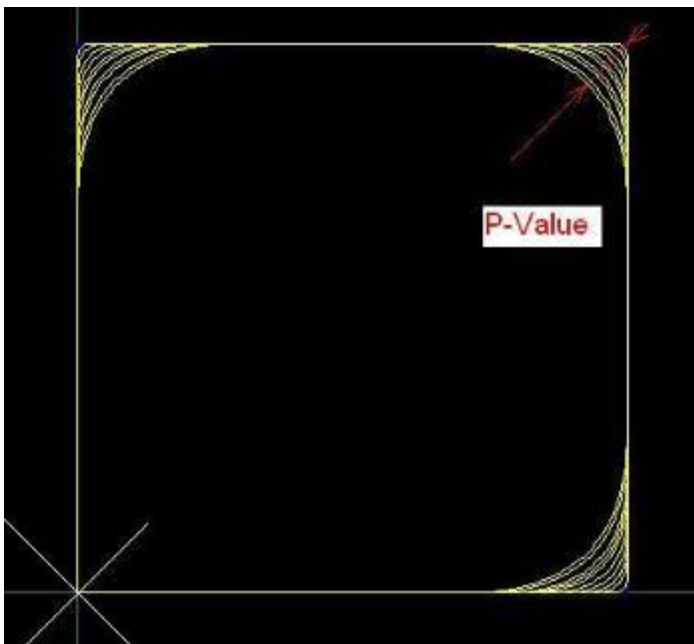
To make a move from stand still we need to accelerate, then have a certain cruising speed and after decelerate. Short moves typically never reach the requested velocity the accelerate and then at half the distance the decelerate.

This table shows the distance traveled so that the given speed is reached. When the line segments generated by the CAD/CAM program are shorter, the actual speed on the machine will be lower than

requested value. Example, you want a milling speed of 900 mm/minute, then the segments generated by the CAD/CAM program must be smaller than 1.88 mm. If the lines are only 0.21 mm, the feed will go down to 300

Velocity	Feed	Accel	Distance
48	2880	120	19.20
30	1800	120	7.50
25	1500	120	5.21
20	1200	120	3.33
17	1020	120	2.41
15	900	120	1.88
12	720	120	1.20
10	600	120	0.83
9	540	120	0.68
8	480	120	0.53
7	420	120	0.41
6	360	120	0.30
5	300	120	0.21
4	240	120	0.13
3	180	120	0.08

If you machine has higher accelerations which requires bigger motors, also higher milling velocities are possible. The values given here are for a moderate hobby machine. This illustrates that when the G-Code file exists of small segments, e.g. 0.08 mm, that with an acceleration of 120 a feed can be reached of 180 mm/minute at most.

3.5.16 Look Ahead feed

To explain this, I will compare a running CNC machine with driving a race car.
The road maximum velocity signs have to be obeyed and you have to drive your car exactly over the white line in the middle of the road. You will try to reach the maximum allowed velocity where possible. When you see a curve coming up ahead, you will brake so that you will not drift off the road. You will try to look ahead as far as you can see and you take care that you can stop in time if the road suddenly stops.

When you would maintain your speed in sharp curves, you will drift off the road resulting possibly into a car accident. When the road has many short curves, then you will not be able to reach the desired speed. The more PS you have in the car, the higher speed you will reach because you can accelerate faster.

I think this is a good comparison with a CNC machine, the same issues apply. A machine cannot suddenly change velocity, to reach a velocity the motors must accelerate first for a certain time to reach the velocity.

LAF behaves like the ideal racecar driver, it will reach the highest possible velocity without violating the maximum motor accelerations.

There is one additional problem while running CNC programs, some programs consists of short line pieces. When the line pieces connect tangentially (are in line), then LAF will accelerate through over the lines, reaching the maximum allowed speed.
The angle to which LAF considers the segments in line is a setup parameter. The theoretical ideal value would be very small, so that no acceleration value occurs.
More practical values are in the range of 1 to 4 degrees, the experience learns that most machines can handle acceleration spikes up to a certain limit.

The value can be set up to 180 degrees in this case you must know what you are doing, it can be useful during e.g. foam cutting wing profiles. Be aware however that if the curve contains real sharp angles that step pulse loss may be the result when using large minimum LAF angles.

In practice we have seen that milling times of complex 3D work pieces can be done in 50% of the time compared to competitors who do not have LAF.

3.5.17 Threading (Lathe) – G76

G76 P- Z- I- J- R- K- Q- H- E- L-

P   Pitch
Z   driveline endpoint
I   Outside thread diameter, always positive.
J   First cut is J beyond I, always positive.
R   Depth regression, use 1.0 for constant cutting depths or leave parameter away.
K   Full thread depth beyond thread peak, always positive.
Q   Compound slide angle, typical 30.
H   Additional spring passes at full depth, use 0 for none.
E   Taper distance along drive line.
L   Taper place, none, enter, exit, both.

;Create a thread from z=20 to z=10, outside diameter=15, inside diameter=14, 10 passes.
G0 X20 Z20
G76 P1.0 Z10 I15 J0.1 K1.0

It is an error if:

- The active plane is not the ZX plane
- Other axis words, such as X- or Y-, are specified
- The R- degression value is less than 1.0.
- All the required words are not specified
- P-, J-, K- or H- is negative
- E- is greater than half the drive line length

The "drive line" is a safe line outside the thread material. The "drive line" goes from the initial location to the Z- value specified with G76. The Z extent of the thread is the same as the drive line.

The "thread pitch", or distance per revolution, is given by the P- value.

The "thread peak" is given by the I- value, which is an offset from the drive line. Negative I values indicate external threads, and positive I values indicate internal threads. Generally the material has been turned to this size before the G76 cycle.

The "initial cut depth" is given by the J- value. The first threading cut will be J beyond the "thread peak" position. J- is positive, even when I- is negative.

The "full thread depth" is given by the K- value. The final threading cut will be K beyond the "thread peak" position. K- is positive, even when I- is negative.

The "depth degression" is given by the R- value. R1.0 selects constant depth on successive threading passes. R2.0 selects constant area. Values between 1.0 and 2.0 select decreasing depth and increasing area. Values above 2.0 select decreasing area. Beware that unnecessarily high degression values will cause a large number of passes to be used.

The "compound slide angle" Q- is the angle (in degrees) describing to what extent successive passes should be offset along the drive line. This is used to cause one side of the tool to remove more material than the other. A positive Q value causes the leading edge of the tool to cut more heavily. Typical values are 29, 29.5 or 30.

The number of "spring passes" is given by the H- value. Spring passes are additional passes at full thread depth. If no additional passes are desired, program H0.

Tapered entry and exit moves can be programmed using E- and L-. E- gives a distance along the drive line used for the taper. E0.2 will give a taper for the first/last 0.2 length units along the thread. L- is used to specify which ends of the thread get the taper. Program L0 for no taper (the default), L1 for entry taper, L2 for exit taper, or L3 for both entry and exit tapers.

The tool will pause briefly for synchronization before each threading pass, so a relief groove will be required at the entry unless the beginning of the thread is past the end of the material or an entry taper is used.

Unless using an exit taper, the exit move (traverse to original X) is not synchronized to the spindle speed. With a slow spindle, the exit move might take only a small fraction of a revolution. If the spindle speed is increased after several passes are complete, subsequent exit moves will require a larger portion of a revolution, resulting in a very heavy cut during the exit move. This can be avoided by providing a relief groove at the exit, or by not changing the spindle speed while threading.

The sample program g76.ngc shows the use of the G76 canned cycle, and can be previewed and executed on any machine using the sim/lathe.ini configuration.
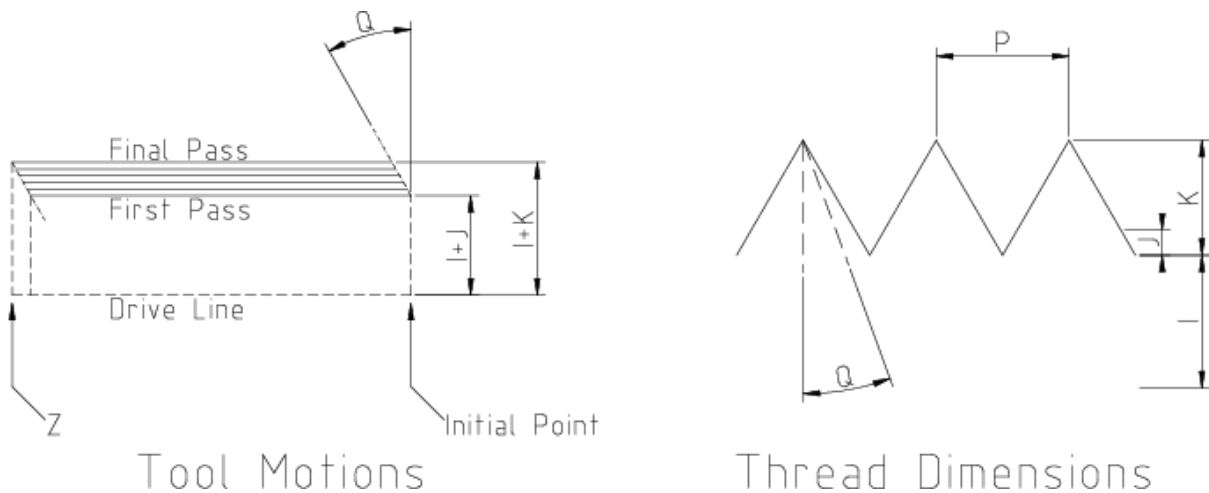
Figure: G76 canned cycle

This is how it works:
1) Before the start, the spindle rate is measured.

2) The feed for de z-axis is calculated: F = pitch * spindleRate

3) The CPU programmed such that a movement is started on the spindle pulse.

4) The movement is calculated and send to the CPU.

5) The movement is started when the spindle pulse passes.

6) Before the treading starts, the spindle-rate is measured, averaged and the feed is calculated from this.

Not that the inside and outside thread diameter are determined by the start position, the position before G76 and the I, K parameters.

3.5.18 Cancel Modal Motion - G80

Program G80 to ensure no axis motion will occur.

It is an error if:
- Axis words are programmed when G80 is active, unless a modal group 0 G code is programmed which uses axis words.

3.5.19 Canned Cycles - G81 to G89

The canned cycles G81 through G89 have been implemented as described in this section. Two examples are given with the description of G81 below.

All canned cycles are performed with respect to the currently selected plane. Any of the three planes (XY, YZ, and ZX) may be selected. Throughout this section, most of the descriptions assume the XY-plane has been selected. The behavior is always analogous if the YZ or XZ-plane is selected.

Rotational axis words are allowed in canned cycles, but it is better to omit them. If rotational axis words are used, the numbers must be the same as the current position numbers so that the rotational axes do not move.

All canned cycles use X, Y, R, and Z numbers in the NC code. These numbers are used to determine X, Y, R, and Z positions. The R (usually meaning retract) position is along the axis perpendicular to the currently selected plane (Z-axis for XY-plane, X-axis for YZ-plane, Y-axis for XZ-plane). Some canned cycles use additional arguments.

For canned cycles, we will call a number "sticky" if, when the same cycle is used on several lines of code in a row, the number must be used the first time, but is optional on the rest of the lines. Sticky numbers keep their value on the rest of the lines if they are not explicitly programmed to be different. The R number is always sticky.

In incremental distance mode: when the XY-plane is selected, X, Y, and R numbers are treated as increments to the current position and Z as an increment from the Z-axis position before the move involving Z takes place; when the YZ or XZ-plane is selected, treatment of the axis words is analogous. In absolute distance mode, the X, Y, R, and Z numbers are absolute positions in the current coordinate system.

The L number is optional and represents the number of repeats. L=0 is not allowed. If the repeat feature is used, it is normally used in incremental distance mode, so that the same sequence of motions is repeated in several equally spaced places along a straight line. In absolute distance mode, L > 1 means "do the same cycle in the same place several times," Omitting the L word is equivalent to specifying L=1. The L number is not sticky.

When L>1 in incremental mode with the XY-plane selected, the X and Y positions are determined by adding the given X and Y numbers either to the current X and Y positions (on the first go-around) or to the X and Y positions at the end of the previous go-around (on the repetitions). The R and Z positions do not change during the repeats.

The height of the retract move at the end of each repeat (called "clear Z" in the descriptions below) is determined by the setting of the retract mode: either to the original Z position (if that is above the R position and the retract mode is G98, OLD_Z), or otherwise to the R position. See Section 3.5.20

It is an error if:
- X, Y, and Z words are all missing during a canned cycle,
- a P number is required and a negative P number is used,
- an L number is used that does not evaluate to a positive integer,
- rotational axis motion is used during a canned cycle,
- inverse time feed rate is active during a canned cycle,
- cutter radius compensation is active during a canned cycle.

When the XY plane is active, the Z number is sticky, and it is an error if:
- the Z number is missing and the same canned cycle was not already active,
- the R number is less than the Z number.

When the XZ plane is active, the Y number is sticky, and it is an error if:
- the Y number is missing and the same canned cycle was not already active,
- the R number is less than the Y number.

When the YZ plane is active, the X number is sticky, and it is an error if:
- the X number is missing and the same canned cycle was not already active,
- the R number is less than the X number.

3.5.19.1 PRELIMINARY AND IN-BETWEEN MOTION
At the very beginning of the execution of any of the canned cycles, with the XY-plane selected, if the current Z position is below the R position, the Z-axis is traversed to the R position. This happens only once, regardless of the value of L.

In addition, at the beginning of the first cycle and each repeat, the following one or two moves are made:
1. a straight traverse parallel to the XY-plane to the given XY-position,
2. a straight traverse of the Z-axis only to the R position, if it is not already at the R position.

If the XZ or YZ plane is active, the preliminary and in-between motions are analogous.

3.5.19.2 G81 CYCLE
The G81 cycle is intended for drilling. Program **G81 X… Y… Z… A… R… L…**
1. Preliminary motion, as described above.
2. Move the Z-axis only at the current feed rate to the Z position.
3. Retract the Z-axis at traverse rate to clear Z.

Example: Suppose the current position is (1, 2, and 3) and the XY-plane has been selected, and the following line of NC code is interpreted.

G90 G81 G98 X4 Y5 Z1.5 R2.8

This calls for absolute distance mode (G90) and OLD_Z retract mode (G98) and calls for the G81 drilling cycle to be performed once. The X number and X position are 4. The Y number and Y position are 5. The Z number and Z position are 1.5. The R number and clear Z are 2.8. Old Z is 3. The following moves take place.
1. a traverse parallel to the XY-plane to (4,5,3)
2. a traverse parallel to the Z-axis to (4,5,2.8)
3. a feed parallel to the Z-axis to (4,5,1.5)
4. a traverse parallel to the Z-axis to (4,5,3)

Example: Suppose the current position is (1, 2, and 3) and the XY-plane has been selected, and the following line of NC code is interpreted.

G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3

This calls for incremental distance mode (G91) and OLD_Z retract mode (G98) and calls for the G81 drilling cycle to be repeated three times. The X number is 4, the Y number is 5, the Z number is -0.6 and the R number is 1.8. The initial X position is 5 (=1+4), the initial Y position is 7 (=2+5), the clear Z position is 4.8 (=1.8+3), and the Z position is 4.2 (=4.8-0.6). Old Z is 3.
The first move is a traverse along the Z-axis to (1,2,4.8), since old Z < clear Z.
The first repeat consists of 3 moves.
1. a traverse parallel to the XY-plane to (5,7,4.8)
2. a feed parallel to the Z-axis to (5,7, 4.2)
3. a traverse parallel to the Z-axis to (5,7,4.8)

The second repeat consists of 3 moves. The X position is reset to 9 (=5+4) and the Y position to 12 (=7+5).
1. a traverse parallel to the XY-plane to (9,12,4.8)
2. a feed parallel to the Z-axis to (9,12, 4.2)
3. a traverse parallel to the Z-axis to (9,12,4.8)

The third repeat consists of 3 moves. The X position is reset to 13 (=9+4) and the Y position to 17 (=12+5).
1. a traverse parallel to the XY-plane to (13,17,4.8)
2. a feed parallel to the Z-axis to (13,17, 4.2)
3. a traverse parallel to the Z-axis to (13,17,4.8)

3.5.19.3 G82 CYCLE
The G82 cycle is intended for drilling. Program **G82 X… Y… Z… A… R… L… P…**
1. Preliminary motion, as described above.
2. Move the Z-axis only at the current feed rate to the Z position.
3. Dwell for the P number of seconds.
4. Retract the Z-axis at traverse rate to clear Z.

3.5.19.4 G83 CYCLE
The G83 cycle (often called peck drilling) is intended for deep drilling or milling with chip breaking. The retracts in this cycle clear the hole of chips and cut off any long stringers (which are common when drilling in aluminum). This cycle takes a Q number which represents a "delta" increment along the Z-axis. **Program G83 X… Y… Z… A… R… L… Q…**

1. Preliminary motion, as described above.
2. Move the Z-axis only at the current feed rate downward by delta or to the Z position, whichever is less deep.
3. Rapid back out to the clear_z.
4. Rapid back down to the current hole bottom, backed off a bit.
5. Repeat steps 1, 2, and 3 until the Z position is reached at step 1.
6. Retract the Z-axis at traverse rate to clear Z.

It is an error if:
• the Q number is negative or zero.

3.5.19.5 G85 CYCLE
The G85 cycle is intended for boring or reaming, but could be used for drilling or milling. Program
G85 X… Y… Z… A… B… C… R… L…
Preliminary motion, as described above.
Move the Z-axis only at the current feed rate to the Z position.
Retract the Z-axis at the current feed rate to clear Z.

3.5.19.6 G86 CYCLE
The G86 cycle is intended for boring. This cycle uses a P number for the number of seconds to dwell.
Program G86 X… Y… Z… A… B… C… R… L… P…
Preliminary motion, as described above.
Move the Z-axis only at the current feed rate to the Z position.
Dwell for the P number of seconds.
Stop the spindle turning.
Retract the Z-axis at traverse rate to clear Z.
Restart the spindle in the direction it was going.

The spindle must be turning before this cycle is used. It is an error if:
the spindle is not turning before this cycle is executed.

3.5.19.7 G87 CYCLE
The G87 cycle is intended for back boring.
Program **G87 X… Y… Z… A… R… L… I… J… K…**

The situation, as shown in Figure 3-1, is that you have a through hole and you want to counter bore the bottom of hole. To do this you put an L-shaped tool in the spindle with a cutting surface on the UPPER side of its base. You stick it carefully through the hole when it is not spinning and is oriented so it fits through the hole, then you move it so the stem of the L is on the axis of the hole, start the spindle, and feed the tool upward to make the counter bore. Then you stop the tool, get it out of the hole, and restart it.

This cycle uses I and J numbers to indicate the position for inserting and removing the tool. I and J will always be increments from the X position and the Y position, regardless of the distance mode setting. This cycle also uses a K number to specify the position along the Z-axis of the controlled point top of the counter bore. The K number is a Z-value in the current coordinate system in absolute distance mode, and an increment (from the Z position) in incremental distance mode.

1. Preliminary motion, as described above.
2. Move at traverse rate parallel to the XY-plane to the point indicated by I and J.
3. Stop the spindle in a specific orientation.
4. Move the Z-axis only at traverse rate downward to the Z position.
5. Move at traverse rate parallel to the XY-plane to the X,Y location.
6. Start the spindle in the direction it was going before.
7. Move the Z-axis only at the given feed rate upward to the position indicated by K.
8. Move the Z-axis only at the given feed rate back down to the Z position.
9. Stop the spindle in the same orientation as before.
10. Move at traverse rate parallel to the XY-plane to the point indicated by I and J.
11. Move the Z-axis only at traverse rate to the clear Z.
12. Move at traverse rate parallel to the XY-plane to the specified X,Y location.

13. Restart the spindle in the direction it was going before.

When programming this cycle, the I and J numbers must be chosen so that when the tool is stopped in an oriented position, it will fit through the hole. Because different cutters are made differently, it may take some analysis and/or experimentation to determine appropriate values for I and J.

Figure 3-1 **G87 Cycle**

The eight subfigures are labelled with the steps from the description above
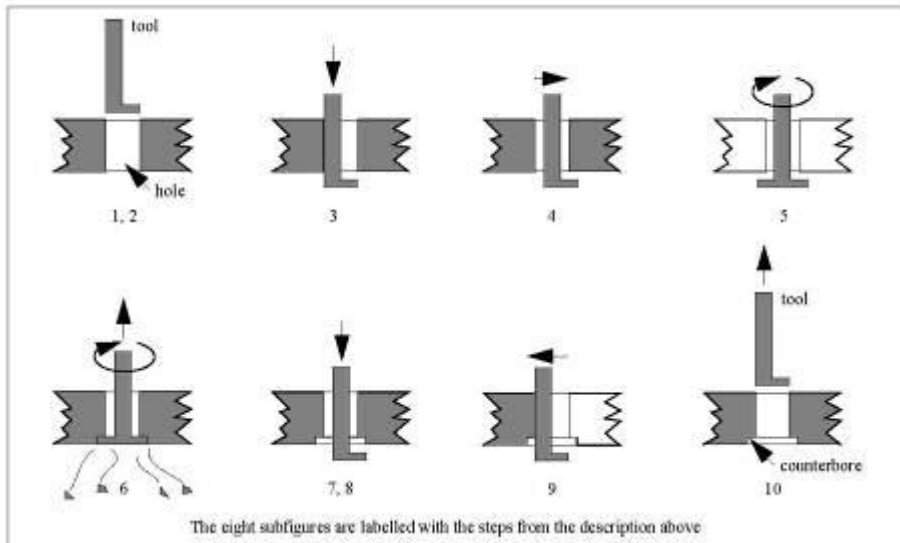
3.5.19.8 G88 CYCLE
The G88 cycle is intended for boring. This cycle uses a P word, where P specifies the number of seconds to dwell. Program **G88 X… Y… Z… A… R… L… P…**
Preliminary motion, as described above.
1. Move the Z-axis only at the current feed rate to the Z position.
2. Dwell for the P number of seconds.
3. Stop the spindle turning.
4. Stop the program so the operator can retract the spindle manually.
5. Restart the spindle in the direction it was going.

3.5.19.9 G89 CYCLE
The G89 cycle is intended for boring. This cycle uses a P number, where P specifies the number of seconds to dwell. program **G89 X… Y… Z… A… R… L… P…**
1. Preliminary motion, as described above.
2. Move the Z-axis only at the current feed rate to the Z position.
3. Dwell for the P number of seconds.
4. Retract the Z-axis at the current feed rate to clear Z.

3.5.20 Set Distance Mode - G90 and G91

To make the current point have the coordinates you want (without motion), program G92 X…
Interpretation of RS274/NGC code can be in one of two distance modes: absolute or incremental.

To go into absolute distance mode, program G90. In absolute distance mode, axis numbers (X, Y, Z, A, B, C) usually represent positions in terms of the currently active coordinate system. Any exceptions to that rule are described explicitly in this Section 3.5.

To go into incremental distance mode, program G91. In incremental distance mode, axis numbers (X, Y, Z, A, B, C) usually represent increments from the current values of the numbers.

I and J numbers always represent increments, regardless of the distance mode setting. K numbers represent increments in all but one usage (see Section 3.5.16.8), where the meaning changes with distance mode.

3.5.21 Coordinate System Offsets - G92, G92.1, G92.2, G92.3

To make the current point have the coordinates you want (without motion), program
G92 X… Y… Z… A… , where the axis words contain the axis numbers you want. All axis words are optional, except that at least one must be used. If an axis word is not used for a given axis, the coordinate on that axis of the current point is not changed.

It is an error if:
- all axis words are omitted.

When G92 is executed, the origin of the currently active coordinate system moves. To do this, origin offsets are calculated so that the coordinates of the current point with respect to the moved origin are as specified on the line containing the G92. In addition, parameters 5211 to 5216 are set to the X, Y, Z, A, B, and C-axis offsets. The offset for an axis is the amount the origin must be moved so that the coordinate of the controlled point on the axis has the specified value.

Here is an example. Suppose the current point is at X=4 in the currently specified coordinate system and the current X-axis offset is zero, then G92 x7 sets the X-axis offset to -3, sets parameter 5211 to -3, and causes the X-coordinate of the current point to be 7.

The axis offsets are always used when motion is specified in absolute distance mode using any of the nine coordinate systems (those designated by G54 - G59.3). Thus all nine coordinate systems are affected by G92.

Being in incremental distance mode has no effect on the action of G92.

Non-zero offsets may already be in effect when the G92 is called. If this is the case, the new value of each offset is A+B, where A is what the offset would be if the old offset were zero, and B is the old offset. For example, after the previous example, the X-value of the current point is 7. If G92 x9 is then programmed, the new X-axis offset is -5, which is calculated by [[7-9] + -3].

To reset axis offsets to zero, program **G92.1** or **G92.2**. G92.1 sets parameters 5211 to 5216 to zero, whereas G92.2 leaves their current values alone.
To set the axis offset values to the values given in parameters 5211 to 5216, program **G92.3**.

You can set axis offsets in one program and use the same offsets in another program. Program G92 in the first program. This will set parameters 5211 to 5216. Do not use G92.1 in the remainder of the first program. The parameter values will be saved when the first program exits and restored when the second one starts up. Use G92.3 near the beginning of the second program. That will restore the offsets saved in the first program. If other programs are to run between the program that sets the offsets and the one that restores them, make a copy of the parameter file written by the first program and use it as the parameter file for the second program.

3.5.22 Set Feed Rate Mode - G93 and G94

Two feed rate modes are recognized: units per minute and inverse time. Program G94 to start the units per minute mode. Program G93 to start the inverse time mode.

In units per minute feed rate mode, an F word (no, not that F word; we mean feed-rate) is interpreted to mean the controlled point should move at a certain number of inches per minute, millimeters per minute, or degrees per minute, depending upon what length units are being used and which axis or axes are moving.

In inverse time feed rate mode, an F word means the move should be completed in [one divided by the F number] minutes. For example, if the F number is 2.0, the move should be completed in half a minute.

When the inverse time feed rate mode is active, an F word must appear on every line which has a G1, G2, or G3 motion, and an F word on a line that does not have G1, G2, or G3 is ignored. Being in inverse time feed rate mode does not affect G0 (rapid traverse) motions.

It is an error if:
- inverse time feed rate mode is active and a line with G1, G2, or G3 (explicitly or implicitly) does not have an F word.

3.5.23 Set Canned Cycle Return Level - G98 and G99

When the spindle retracts during canned cycles, there is a choice of how far it retracts:
(1) retract perpendicular to the selected plane to the position indicated by the R word, or
(2) retract perpendicular to the selected plane to the position that axis was in just before the canned cycle started (unless that position is lower than the position indicated by the R word, in which case use the R word position).

To use option (1), program G99. To use option (2), program G98. Remember that the R word has different meanings in absolute distance mode and incremental distance mode.

3.6 INPUT M CODES

M codes of the RS274/NGC language are shown in Table 3-6

Table 3-6 **M Codes**

M Code	Meaning
M0	program stop
M1 M2	optional program stop
M3 M4	program end
M5 M6	turn spindle clockwise
M7 M8	turn spindle counterclockwise
M9 M30	stop spindle turning
M48	tool change
M49	mist coolant on
M60	flood coolant on
	mist and flood coolant off
	program end, pallet shuttle, and reset
	enable speed and feed overrides
	disable speed and feed overrides
	pallet shuttle and program stop
M64	set general purpose output for Advantronix USB IO card (support is depricated)
M65	clear general purpose output for Advantronix USB IO card (support is depricated)
M66	read general purpose input for Advantronix USB IO card (support is depricated)
M54	set general purpose output for CPU5B
M55	clear general purpose output for CPU5B
M56	read general purpose input for CPU5B

3.6.1 Program Stopping and Ending - M0, M1, M2, M30, M60

To stop a running program temporarily (regardless of the setting of the optional stop switch), program **M0 or M1**. If a program is stopped by an M0, M1, or M60, pressing the cycle start button will restart the program at the following line, so the program will continue.
To end a program, program **M2**.
program **M30** for next effects:
- Selected plane is set to CANON_PLANE_XY (like G17).
- Distance mode is set to MODE_ABSOLUTE (like G90).
- Feed rate mode is set to UNITS_PER_MINUTE (like G94).
- Feed and speed overrides are set to ON (like M48).

- Cutter compensation is turned off (like G40).
- The spindle is stopped (like M5).
- The current motion mode is set to G_1 (like G1).
- Coolant is turned off (like M9).
- Note that the coordinate system are no longer reset, I modified this behavior because I have broken a lot of bits due to this so I modified it.

No more lines of code in an RS274/NGC file will be executed after the M2 or M30 command is executed. Pressing cycle start will start the program back at the beginning of the file.

3.6.2   Spindle Control - M3, M4, M5

To start the spindle turning clockwise at the currently programmed speed, program **M3**.
To start the spindle turning counterclockwise at the currently programmed speed, program **M4**.
To stop the spindle from turning, program **M5**.
It is OK to use M3 or M4 if the spindle speed is set to zero. If this is done (or if the speed override switch is enabled and set to zero), the spindle will not start turning. If, later, the spindle speed is set above zero (or the override switch is turned up), the spindle will start turning. It is OK to use M3 or M4 when the spindle is already turning or to use M5 when the spindle is already stopped.

3.6.3   Tool Change - M6

To change a tool in the spindle from the tool currently in the spindle to the tool most
recently selected (using a T word - see Section 3.7.3), program **M6**. When the tool change is complete:
- The spindle will be stopped.
- The tool that was selected (by a T word on the same line or on any line after the previous tool change) will be in the spindle. The T number is an integer giving the changer slot of the tool (not its id).
- If the selected tool was not in the spindle before the tool change, the tool that was in the spindle (if there was one) will be in its changer slot.
- The coordinate axes will be stopped in the same absolute position they were in before the tool change (but the spindle may be re-oriented).
- No other changes will be made. For example, coolant will continue to flow during the tool change unless it has been turned off by an M9.

The tool change may include axis motion while it is in progress. It is OK (but not useful) to program a change to the tool already in the spindle. It is OK if there is no tool in the selected slot; in that case, the spindle will be empty after the tool change. If slot zero was last selected, there will definitely be no tool in the spindle after a tool change.

The tool change command will call the **change_tool** subroutine inside macro.cnc.
You can adapt the behavior for your own needs in this function e.g:
- Perform automatic tool-length measurement
- Perform tool change with an automatic tool changer.

For a (non functional) example of how to implement automatic tool change for a 16-tool changer.
see the contents of the default_macro.cnc file at the end of this document. It checks whether current tool is already in the spindle. It check that the tool number is in range of 1-4. Then it first drops current tool and picks the new tool:

3.6.4   Coolant Control - M7, M8, M9

To turn mist coolant on, program M7. To turn flood coolant on, program M8. To turn all coolant off, program M9. It is always OK to use any of these commands, regardless of what coolant is on or off.

3.6.5   Override Control - M48 and M49

To enable the speed and feed override switches, program M48. To disable both
switches, program M49. See Section 2.2.1 for more details. It is OK to enable or disable the switches
when they are already enabled or disabled.

3.6.6   IO M Functions

3.6.6.1   Standard CNC IO - M3..M9, M80..M87

To control the outputs, these functions have been added besides the standard M-Functions.

Standard, according to [NIST]
M3 PWM according S value, TOOLDIR = on
M4 PWM according S value, TOOLDIR = off
M5 PWM off, TOOLDIR off.
M7 Mist on
M8 Flood on
M9 Mist/Flood off

Additional, to support the features of the USBCNC CPU's
M80 drive enable on
M81 drive enable off
M82/M61 Aux1 on
M83/M62 Aux1 off
M84 TOOLDIR on
M85 TOOLDIR off
M86 PWM according S value (s/s-max from setup * 100%)
M87 PWM off

3.6.6.2   General purpose IO of CPU5B - M54, M55 and M56

M54 Px
Set output x.

M54 Ex Qy
Set PWM output x to % value y (0 <= y <= 100)

M55 Px
Clear output x.

M56 Px
Read input x.

M56 Px Ly Qx.xx
Read digital input and specify wait mode
Px: x is input number
L0: do not wait
L1, L3: Wait mode High
L2, L4: Wait mode Low
Qx: x is timeout

M56 Ex Ly Qx.xx
=========
Read analogue input and specify wait mode
Ex: x is input number
L0: do not wait
L1, L3: Wait mode High

L2, L4: Wait mode Low
Qx: x is timeout

For all M56 variants, the input value is stored into #5399

For the general purpose I/O of CPU5, use M54, M55, M56 in stead of M64, M65, M66.

3.7 OTHER INPUT CODES

3.7.1 Set Feed Rate - F
To set the feed rate, program F… . The application of the feed rate is as described in
Section 2.1.2.5, unless inverse time feed rate mode is in effect, in which case the feed rate is as
described in Section 3.5.19.

3.7.2 Set Spindle Speed - S
To set the speed in revolutions per minute (rpm) of the spindle, program S… . The spindle will turn at
that speed when it has been programmed to start turning. It is OK to program an S word whether the
spindle is turning or not. If the speed override switch is
enabled and not set at 100%, the speed will be different from what is programmed. It is OK to
program S0; the spindle will not turn if that is done.
The CPU's that support PWM output will have its PWM value set conform the requested spindle
speed if the spindle is turned on.

It is an error if:
  • the S number is negative.

As described in Section 3.5.16.5, if a G84 (tapping) canned cycle is active and the feed and speed
override switches are enabled, the one set at the lower setting will take effect. The speed and feed
rates will still be synchronized. In this case, the speed may differ from what is programmed, even if
the speed override switch is set at 100%.

3.7.3 Select Tool - T
To select a tool, program T…, where the T number is the carousel slot for the tool. The tool is not
changed until an M6 is programmed (see Section 3.6.3). The T word may appear on the same line as
the M6 or on a previous line. It is OK, but not normally useful, if T words appear on two or more lines
with no tool change. The carousel may move a lot, but only the most recent T word will take effect at
the next tool change. It is OK to
program T0; no tool will be selected. This is useful if you want the spindle to be empty after a tool
change.

It is an error if:
  • a negative T number is used,
  • a T number larger than the number of slots in the carousel is used.

On some machines, the carousel will move when a T word is programmed, at the same time
machining is occurring. On such machines, programming the T word several lines before a tool
change will save time. A common programming practice for such machines is to put the T word for
the next tool to be used on the line after a tool change. This maximizes the time available for the
carousel to move.

3.8 ORDER OF EXECUTION
The order of execution of items on a line is critical to safe and effective machine operation. Items are
executed in the order shown in Table 3-7 if they occur on the same line.

Table 3-7 Order of execution

1.	comment (includes message).
2.	set feed rate mode (G93, G94 -inverse time or per minute).
3.	set feed rate (F).
4.	set spindle speed (S).
5.	select tool (T).
6.	change tool (M6).
7.	spindle on or off (M3, M4, M5).
8.	coolant on or off (M7, M8, M9).
9.	enable or disable overrides (M48, M49).
10.	dwell (G4).
11.	set active plane (G17, G18, G19).
12.	set length units (G20, G21).
13.	cutter radius compensation on or off (G40, G41, G42)
14.	cutter length compensation on or off (G43, G49)
15.	coordinate system selection (G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3).
16.	set path control mode (G61, G61.1, G64)
17.	set distance mode (G90, G91).
18.	set retract mode (G98, G99).
19.	home (G28, G30) or change coordinate system data (G10) or set axis offsets (G92, G92.1, G92.2, G94).
20.	perform motion (G0 to G3, G80 to G89), as modified (possibly) by G53.
21.	stop(M0,M1, M2,M30,M60).

4 Language extensions

To provide additional flexibility, I created some extensions in the language that allow for programming.

4.1 FLOW CONTROL

You can use the following flow control commands in a job:
IF[x]-ELSE-ENDIF constructs to define x dependent execution
WHILE[x]-ENDWHILE constructs to define x dependent repeated execution
SUB <name>-ENDSUB constructs to define a subroutine
GOSUB <name> construct to call a subroutine

4.2 SUPPORTED OPERATIONS ON EXPRESSIONS

4.2.1 unary operations

abs   absolute value
acos   arc cosine
asin   arc sine
atan   arc tangent
cos   cosine
exp   e raised to
fix   round down
fup   round
int   integer part
ln   natural log of

round   round
sin   sine
sqrt   square root
tan   tangent
not   logical not

4.2.2   binary operations:

/   divided by
mod   modulo
**   power
*   times
and   logic and
xor   logic exclusive or
-         minus
or   logic non exclusive or
+   plus
>   greater then
>=   greater than or equal
<   less then
<=   less than or equal
==   is equal
<>   not equal
band   bitwise and
bxor   bitwise exclusive or
bor   bitwise non exclusive or
<<   shift left
>>   shift right

See also B.2 for examples on expressions.

4.2.3   An example:

```
sub do_circle_holes

#1=0
g0 z1 x0 y0
while [#1 <> 360]
      #2 = [10 * sin[#1]]
      #3 = [10 * cos[#1]]
      g0 x[#3] y[#2]
      g1 z-1
      g1 z1
      #1 = [#1 + 30]


      if [#1 == 360]
            msg "Done"
      else
            msg "processing at angle "#1
      endif

endwhile

endsub

gosub do_circle_holes
m30
```

This example drills holes at a circle with a radius of 10, each 30 degrees. The code that performs this is put in a subroutine, which can be called as many times as needed in the main program.

4.2.4   Special interpreter commands, non G-Code

Messages:
```
Msg "Hello there, the value of #1 = "#1" and the value of #2 = "#2
```

ErrMsg
Same as Msg, but this one generates an error

Store position
```
SP <filename> [0 or 1]
```
This command stores the actual position in given file name.
The extra parameter 0 means create the file.
1 means add to existing file.
If only file name is given, the position is added to existing file.

DlgMsg
Gives a dialog message for an interactive g-code program.
```
DlgMsg <dialog message> <par1Name> <par1ParNumber> ... <par7Name> <par7ParNumber>
```
Example:
DlgMsg "Give parameters" par1 100 par2 101
The dialog woll have an OK and a Cancel button.
When the user selects OK, variable #5398 is set to 1 and the program automatically continues.
When the user selects CANCEL, variable #5398 is set to -1.
Just try and you will see what this is about.

LogFile, LogMsg
Log anything to a file..

```
LogFile <fileName> <1=append, 0=open new>
LogMsg  your message
```

Example:
LogFile "text.txt" 1
LogMsg "Hi, the current position of X is "#5001

Now check the contents of file text.txt.

TCAGuard [on | off]
Switches on or off the tool change area guard.
This is used during the rendering process, where the job file is checked for collisions with the machine area and tool change area.

HomeIsEstop [on | off]
This allows to control the homeIsEstop feature.
When on, a EStop is generated when one of the home sensors activate.

4.2.5 Special interpreter MDI commands.

M6 TX which is a tool change will call the subroutine change_tool. This subroutine can be customized to match your machine.

Gosub subname moves the interpreter to the first line off the subroutine, allowing you to execute the subroutine without calling it from the main program. This is good for testing subroutines. In combination with DlgMsg you can give your own input parameters.

4.3 MACRO FILE AND AUTOMATIC TOOL CHANGE

Whenever a G-Code file is loaded, also the file "macro.cnc" is loaded. In this file you may put your frequently used subroutines, these can be invoked by the G-Code file through GOSUB subroutineName.
The file contains default one special subroutine called **"change_tool"**, this function is called automatically when a **M6 Tx** command (Tool change) is encountered in the G-Code file. With this it is possible to define your own tool change, especially useful when you have an automatic tool changer. You can put moves and I/O actions there as well as automatic tool length measurement using the probe with G38.2.

The tool change area can be guarded for collision, if it is defined the rendering process will detect eventual collisions and report it. So a normal workpiece program is not allowed to go through the Tool change Area. The tool change itself is allowed to go to this area. Therefor the **change_tool** subroutine contains the statement **TCAGuard off** at the beginning and **TCAGuard on** at the end.

References

[Albus]

Albus, James S; et al; NIST Support to the Next Generation Controller Program: 1991 Final Technical Report; NISTIR 4888; National Institute of Standards and Technology, Gaithersburg, MD; July 1992

[Allen-Bradley]

Allen-Bradley; RS274/NGC for the Low End Controller; First Draft; Allen-Bradley; August 1992

[EIA]

Electronic Industries Association; EIA Standard EIA-274-D Interchangeable Variable Block Data Format for Positioning, Contouring, and Contouring/Positioning Numerically Controlled Machines; Electronic Industries Association; Washington, DC; February 1979

[Fanuc]

Fanuc Ltd.; Fanuc System 9-Model A Operators Manual; Pub B-52364E/03; Fanuc Ltd; 1981

[Kramer1]

Kramer, Thomas R.; Proctor, Frederick M.; Michaloski, John L.; The NIST RS274/NGC Interpreter, Version 1; NISTIR 5416; National Institute of Standards and Technology, Gaithersburg, MD; April 1994

[Kramer2]

Kramer, Thomas R.; Proctor, Frederick M.; The NIST RS274KT Interpreter; NISTIR 5738; National Institute of Standards and Technology, Gaithersburg, MD; October 1995

[Kramer3]

Kramer, Thomas R.; Proctor, Frederick M.; The NIST RS274/NGC Interpreter -Version 2; NISTIR 5739; National Institute of Standards and Technology, Gaithersburg, MD; October 1995

[Kramer4]

Kramer, Thomas R.; Proctor, Frederick M.; The NIST RS274/VGER Interpreter; NISTIR 5754; National Institute of Standards and Technology, Gaithersburg, MD; November 1995

[K&T]

Kearney and Trecker Co.; Part Programming and Operating Manual, KT/CNC Control, Type C; Pub 687D; Kearney and Trecker Corp.; 1980

[NCMS]

National Center for Manufacturing Sciences; The Next Generation Controller Part Programming Functional Specification (RS-274/NGC); Draft; NCMS; August 1994

[Proctor]

Proctor, Frederick M.; Kramer, Thomas R.; Michaloski, John L.; Canonical Machining Commands; NISTIR 5970; National Institute of Standards and Technology, Gaithersburg, MD; January 1997

A Cutter Radius Compensation

This appendix discusses cutter radius compensation. It is intended for NC programmers and machine operators.

See Section 3.5.10 for additional information on cutter radius compensation.

A.1 INTRODUCTION

The cutter radius compensation capabilities of the Interpreter enable the programmer to specify that a cutter should travel to the right or left of an open or closed contour in the XY-plane composed of arcs of circles and straight line segments.

Cutter radius compensation is performed only with the XY-plane active. All the figures in this appendix, therefore, show projections on the XY-plane.

Where the adjacent sides of remaining material meet at a corner, there are two common ways to handle the tool path. The tool may pass in an arc around the corner, or the tool path may continue straight in the direction it was going along the first side until it reaches a point where it changes direction to go straight along the second side. Figure A-1 shows these two types of path. On Figure A-1:

- Uncut material is shaded in the figures. Note that the inner triangles have the same shape with both tool paths.
- The white areas are the areas cleared by the tool.
- The lines in the center of the white areas represent the path of the tip of a cutting tool.
- The tool is the cross-hatched circles.

Both paths will clear away material near the shaded triangle and leave the shaded triangle uncut. When the Interpreter performs cutter radius compensation, the tool path is rounded at the corners, as shown on the left in Figure A-1. In the method on the right (the one not used), the tool does not stay in contact with the shaded triangle at sharp corners, and more material than necessary is removed.

There are also two alternatives for the path that is programmed in NC code during cutter radius compensation. The programmed path may be either (1) the edge of the material to remain uncut (for example, the edge of the inner triangle on the left of Figure A-1), or (2) the nominal tool path (for example, the tool path on the left side of Figure A-1). The nominal tool path is the path that would be used if the tool were exactly the intended size. The Interpreter will handle both cases without being told which one it is. The two cases are very similar, but different enough that they are described in separate sections of this manual. To use the material edge method, read Section A.3. To use the nominal path method, read Section A.4.

Figure A-1
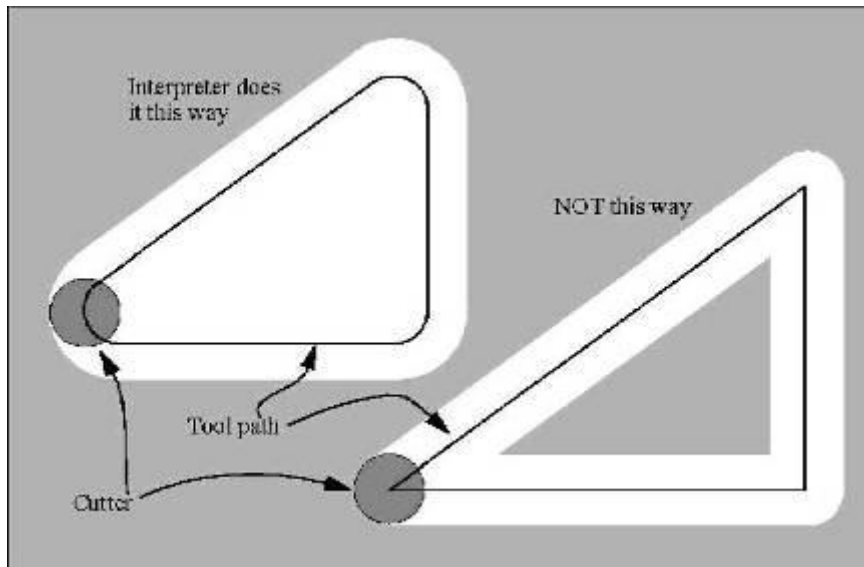
Z-axis motion may take place while the contour is being followed in the XY-plane. Portions of the contour may be skipped by retracting the Z-axis above the part, following the contour to the next point at which machining should be done, and re-extending the Z-axis. These skip motions may be performed at feed rate (G1) or at traverse rate (G0). The Z motion will not interfere with the XY path following. The sample NC code in this appendix does not include moving the Z-axis. In actual programs, include Z-axis motion wherever you want it.

Rotational axis motions (A, B, and C axes) are allowed with cutter radius compensation, but using them would be very unusual.

Inverse time feed rate (G93) or units per minute feed rate (G94) may be used with cutter radius compensation. Under G94, the feed rate will apply to the actual path of the cutter tip, not to the programmed contour.

A.1.1   Data for Cutter Radius Compensation
The Interpreter world model keeps three data items for cutter radius compensation: the setting itself (right, left, or off), program_x, and program_y. The last two represent the X and Y positions which are given in the NC code while compensation is on. When compensation is off, these both are set to a very small number (10-20) whose symbolic value is "unknown". The Interpreter world model uses the data items current_x and
current_y to represent the position of the center of the tool tip (in the currently active coordinate system) at all times.

A.2 PROGRAMMING INSTRUCTIONS

A.2.1   Turning Cutter Radius Compensation On
To start cutter radius compensation keeping the tool to the left of the contour, program **G41 D…**. The D word is optional (see "Use of D Number", just below).

To start cutter radius compensation keeping the tool to the right of the contour, program **G42 D…** .

In Figure A-1, for example, if G41 were programmed, the tool would move clockwise around the triangle, so that the tool is always to the left of the triangle when facing in the direction of travel. If G42 were programmed, the tool would stay right of the triangle and move counter clockwise around the triangle.

A.2.2 Turning Cutter Radius Compensation Off

To stop cutter radius compensation, program **G40**. It is OK to turn compensation off when it is already off.

A.2.3 Sequencing

If G40, G41, or G42 is programmed on the same line as tool motion, cutter compensation will be turned on or off before the motion is made. To make the motion come first, the motion must be programmed on a separate, previous line of code.

A.2.4 Use of D Number

Programming a D word with G41 or G42, is optional.
If a D number is programmed, it must be a non-negative integer. It represents the slot number of the tool whose radius (half the diameter given in the tool table) will be used, or it may be zero (which is not a slot number). If it is zero, the value of the radius will also be zero. Any slot in the tool table may be selected. The D number does not have to be the same as the slot number of the tool in the spindle, although it is rarely useful for it not to be.
If a D number is not programmed, the slot number of the tool in the spindle will be used as the D number.

A.3 Material Edge Contour

When the contour is the edge of the material, the outline of the edge is described in the NC program.
For a material edge contour, the value for the diameter in the tool table is the actual value of the diameter of the tool. The value in the table must be positive. The NC code for a material edge contour is the same regardless of the (actual or intended) diameter of the tool.

A.3.1 Programming Entry Moves

In general, two pre-entry moves and one entry move are needed to begin compensation correctly. However, if there is a convex corner on the contour, a simpler method is available using zero or one pre-entry move and one entry move. The general method, which will work in all situations, is described first. We assume here that the programmer knows what the contour is already and has the job of adding entry moves.

A.3.1.1 GENERAL METHOD

The general method includes programming two pre-entry moves and one entry move. See Figure A-2. The shaded area is the remaining material. It has no corners, so the simple method cannot be used. The dotted line is the programmed path. The solid line is the actual path of the tool tip. Both paths go clockwise around the remaining material. A cutter one unit in diameter is shown part way around the path. The black dots mark points at the beginning or end of programmed or actual moves. The figure shows the second pre-entry move but not the first, since the beginning point of the first pre-entry move could be anywhere.

Figure A-2, Cutting radius compensation entry moves (for material edge contour)
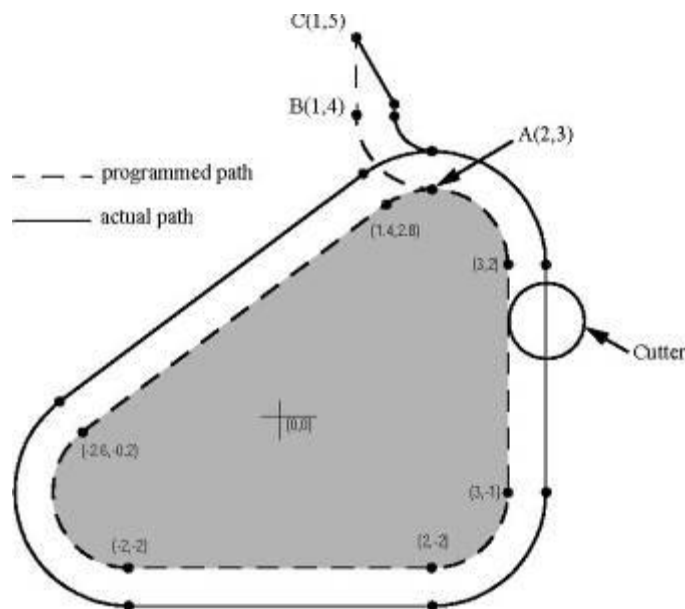
First, pick a point A on the contour where it is convenient to attach an entry arc. Specify an arc outside the contour which begins at a point B and ends at A tangent to the contour (and going in the same direction as it is planned to go around the contour). The radius of the arc should be larger than half the diameter given in the tool table. Then extend a line tangent to the arc from B to some point C, located so that the line BC is more than one tool radius long. After the construction is finished, the code is written in the reverse order from the construction. The NC code is shown in Table A-1; the first three lines are the entry moves just described.

Table A-1 NC program for figure A-2

N0010 G1 X1 Y5 (make first pre-entry move to C)
N0020 G41 G1 Y4 (turn compensation on and make second pre-entry move to point B)
N0030 G3 X2 Y3 I1 (make entry move to point A)
N0040 G2 X3 Y2 J-1 (cut along arc at top)
N0050 G1 Y-1 (cut along right side)
N0060 G2 X2 Y-2 I-1 (cut along arc at bottom right)
N0070 G1 X-2 (cut along bottom side)
N0080 G2 X-2.6 Y-0.2 J1 (cut along arc at bottom left)
N0090 G1 X1.4 Y2.8 (cut along third side)
N0100 G2 X2 Y3 I0.6 J-0.8 (cut along arc at top of tool path)
N0110 G40 (turn compensation off)

Cutter radius compensation is turned on after the first pre-entry move and before the second pre-entry move (including G41 on the same line as the second pre-entry move turns compensation on before the move is made). In the code above, line N0010 is the first pre-entry move, line N0020 turns compensation on and makes the second pre-entry move, and line N0030 makes the entry move.

A.3.1.2 SIMPLE METHOD
If there is a convex (sticking out, not in) corner somewhere on the contour, a simpler method of making an entry is available. See Figure A-3.

First, pick a convex corner. There is only one corner in Figure A-3. It is at A, and it is convex. Decide which way you want to go along the contour from A. In our example we are keeping the tool to the left of the remaining material and going clockwise. Extend the side to be cut

(DA in the figure) to divide the area outside the material near A into two regions; DA extended is the dotted line AC on the figure. Make a pre-entry move to anywhere in the region on the same side of DC as the remaining material (point B on the figure) and not so close to the remaining material that the tool is cutting into it. Anywhere in the diagonally shaded area of the figure (or above or to the left of that area) is OK. If the tool is already in region, no pre-entry move is needed. Write a line of NC code to move to B, if necessary. Then write a line of NC code for a straight entry move that turns compensation on and goes to point A.
If B is at (1.5, 4), the two lines of code for the pre-entry and entry moves would be:

N0010 G1 X1.5 Y4 (move to B)
N0020 G41 G1 X3 Y3 (turn compensation on and make entry move to A)

These two lines would be followed by four lines identical to lines N0050 to N0080 from Table A-1, but the end of the program would be different since the shape of remaining material is different.

It would be OK for B to be on line AC. In fact, B could be placed on the extension outside the part of any straight side of the part. B could be placed on EF extended to the right (but not to the left, for going clockwise), for example.

If DA were an arc, not a straight line, the two lines of code above would still be suitable. In this case, the dotted line extending DA should be tangent to DA at A.
Figure A-3 Simpler cutter radius compensation entry move (for material edge contour)

Figure A-3 Simpler compensation entry move
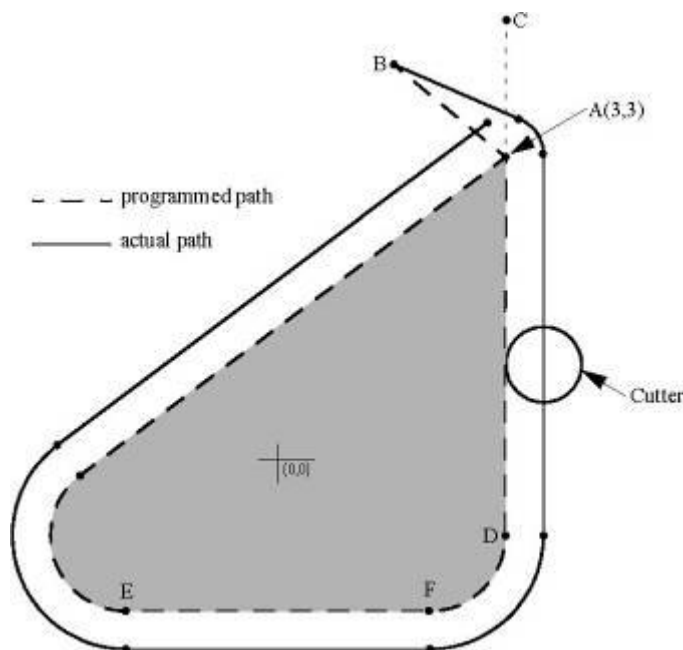
A.4 NOMINAL PATH CONTOUR

When the contour is a nominal path contour (the path a tool with exactly the intended diameter would take), the tool path is described in the NC program. It is expected that (except for during the entry moves) the path is intended to create some part geometry. The path may be generated manually or by a post-processor, considering the part geometry which is intended to be made. For the Interpreter to work, the tool path must be such that the tool stays in contact with the edge of the part geometry, as shown on the left side of Figure A-1. If a path of the sort shown on the right of Figure A-1 is used, in which the tool does not stay in contact with the part geometry all the time, the Interpreter will not be able to compensate properly when undersized tools are used. A nominal path contour has no corners, so the simple method just described will not work.

For a nominal path contour, the value for the cutter diameter in the tool table will be a small positive number if the selected tool is slightly oversized and will be a small negative number if the tool is slightly undersized. If a cutter diameter value is negative, the Interpreter compensates on the other side of the contour from the one programmed and uses the absolute value of the given diameter. If the actual tool is the correct size, the value in the table should be zero. Suppose, for example, the diameter of the cutter currently in the spindle is 0.97, and the diameter assumed in generating the tool path was 1.0. Then the value in the tool table for the diameter for this tool should be -0.03.

The nominal tool path needs to be programmed so that it will work with the largest and smallest tools expected to be actually used. We will call the difference between the radius of the largest expected tool and the intended radius of the tool the "maximum radius difference." This is usually a small number.

The method includes programming two pre-entry moves and one entry moves. See Figure A-4. The shaded area is the remaining material. The dashed line is the programmed tool path. The solid line is the actual path of the tool tip. Both paths go clockwise around the remaining material. The actual path is to the right of the programmed path even though G41 was programmed, because the diameter value is negative. On the figure, the distance between the two paths is larger than would normally be expected. The 1-inch diameter tool is shown part way around the path. The black dots mark points at the beginning or end of programmed moves. The corresponding points on the actual path have not been marked. The actual path will have a very small additional arc near point B unless the tool diameter is exactly the size intended. The figure shows the second pre-entry move but not the first, since the beginning point of the first pre-entry move could be anywhere.

First, pick a point A on the contour where it is convenient to attach an entry arc. Specify an arc outside the contour which begins at a point B and ends at A tangent to the contour (and going in the same direction as it is planned to go around the contour). The radius of the arc should be larger than the maximum radius difference. Then extend a line tangent to the arc from B to some point C, located so that the length of line BC is more than the maximum radius difference. After the construction is finished, the code is written in the reverse order from the construction. The NC code is shown in Table A-2; the first three lines are the entry moves just described.

Table A-2 NC program for Figure A-4

N0010 G1 X1.5 Y5 (make first pre-entry move to C)
N0020 G41 G1 Y4 (turn compensation on and make second pre-entry move to point B)
N0030 G3 X2 Y3.5 I0.5 (make entry move to point A)
N0040 G2 X3.5 Y2 J-1.5 (cut along arc at top)
N0050 G1 Y-1 (cut along right side)
N0060 G2 X2 Y-2.5 I-1.5 (cut along arc at bottom right)
N0070 G1 X-2 (cut along bottom side)
N0080 G2 X-2.9 Y0.2 J1.5 (cut along arc at bottom left)
N0090 G1 X1.1 Y3.2 (cut along third side)
N0100 G2 X2 Y3.5 I0.9 J-1.2 (cut along arc at top of tool path)
N0110 G40 (turn compensation off)

Figure A-4 Cutter radius compensation entry moves
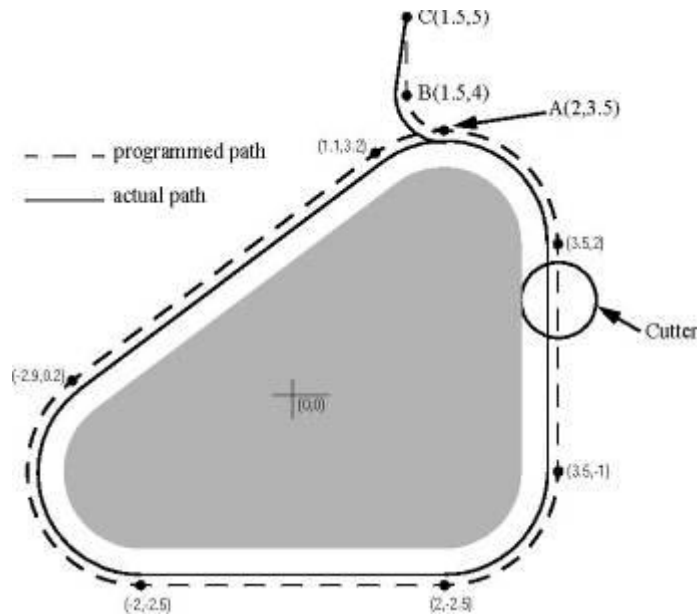
Cutter radius compensation is turned on after the first pre-entry move and before the second pre-entry move (including G41 on the same line as the second pre-entry move turns compensation on before the move is made). In the code above, line N0010 is the first pre-entry move, line N0020 turns compensation on and makes the second pre-entry move, and line N0030 makes the entry move.

A.5 PROGRAMMING ERRORS AND LIMITATIONS

The Interpreter will issue the following error messages involving cutter radius compensation. In addition to these, there are several bug messages related to cutter compensation, but they should never occur.

1. Cannot change axis offsets with cutter radius comp
2. Cannot change units with cutter radius comp
3. Cannot probe with cutter radius comp on
4. Cannot turn cutter radius comp on out of xy-plane
5. Cannot turn cutter radius comp on when on
6. Cannot use g28 or g30 with cutter radius comp
7. Cannot use g53 with cutter radius comp
8. Cannot use xz-plane with cutter radius comp
9. Cannot use yz-plane with cutter radius comp
10. Concave corner with cutter radius comp
11. Cutter gouging with cutter radius comp
12. D word with no g41 or g42
13. Multiple d words on one line
14. Negative d word tool radius index used
15. Tool radius index too big
16. Tool radius not less than arc radius with comp
17. Two g codes used from same modal group.

Most of these are self-explanatory. For those that require explanation, an explanation is given below.

Changing a tool while cutter radius compensation is on is not treated as an error, although it is unlikely this would be done intentionally. The radius used when cutter radius compensation

was first turned on will continue to be used until compensation is turned off, even though a new tool is actually being used.

A.5.1   Concave Corner and Tool Radius Too Big (10 and 16)

When cutter radius compensation is on, it must be physically possible for a circle whose radius is the half the diameter given in the tool table to be tangent to the contour at all points of the contour. In particular, the Interpreter treats concave corners and concave arcs into which the circle will not fit as errors, since the circle cannot be kept tangent to the contour in these situations. See Figure A-5. This error detection does not limit the shapes which can be cut, but it does require that the programmer specify the actual shape to be cut (or path to be followed), not an approximation. In this respect, the NIST
RS274/NGC Interpreter differs from interpreters used with many other controllers, which often allow these errors silently and either gouge the part or round the corner.
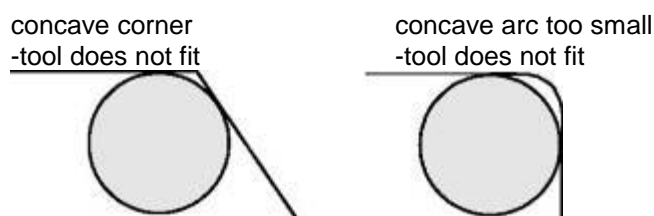
concave corner   concave arc too small
-tool does not fit   -tool does not fit

Figure A-5, Two cutter radius compensation errors

In both examples, the line represents a contour, and the circle represents the cross section of a tool following the contour using cutter radius compensation (tangent to one side of the path.)

A.5.2   Cannot Turn Cutter Radius Comp on When On (5)

If cutter radius compensation has already been turned on, it cannot be turned on again. It must be turned off first; then it can be turned on again. It is not necessary to move the cutter between turning compensation off and back on, but the move after turning it back on will be treated as a first move, as described below.
It is not possible to change from one cutter radius index to another while compensation is on because of the combined effect of rules 5 and 12. It is also not possible to switch compensation from one side to another while compensation is on.

A.5.3   Cutter Gouging (11)

If the tool is already covering up the next XY destination point when cutter radius compensation is turned on, the gouging message is given when the line of NC code which gives the point is reached. In this situation, the tool is already cutting into material it should not cut. More details are given in Section A.6.

A.5.4   Tool Radius Index Too Big (15)

If a D word is programmed that is larger than the number of tool carousel slots, this error message is given. In the SAI, the number of slots is 68.

A.5.5   Two G Codes Used from Same Modal Group (17)

This is a generic message used for many sets of G codes. As applied to cutter radius compensation, it means that more than one of G40, G41, and G42 appears on a line of NC code. This is not allowed.

A.6   First Move into Cutter Compensation

The algorithm used for the first move after cutter radius compensation is turned on, when the first move is a straight line, is to draw a straight line from the programmed destination point which is tangent to a circle whose center is at the current point and whose radius is the radius of the tool. The destination point of the tool tip is then found as the center of a circle of the same radius tangent to the tangent line at the destination point. If the programmed point is inside the initial cross section of the tool (the circle on the left), an error is signaled as described in Section A.5.3. The concept of the algorithm is shown in Figure A-6.

The function that locates the destination point actually takes a computational shortcut based on the fact that the line (not drawn on the figure) from the current point to the programmed point is the hypotenuse of a right triangle having the destination point at the corner with the right angle.

Figure A-6, First cutter radius compensation move - Straight
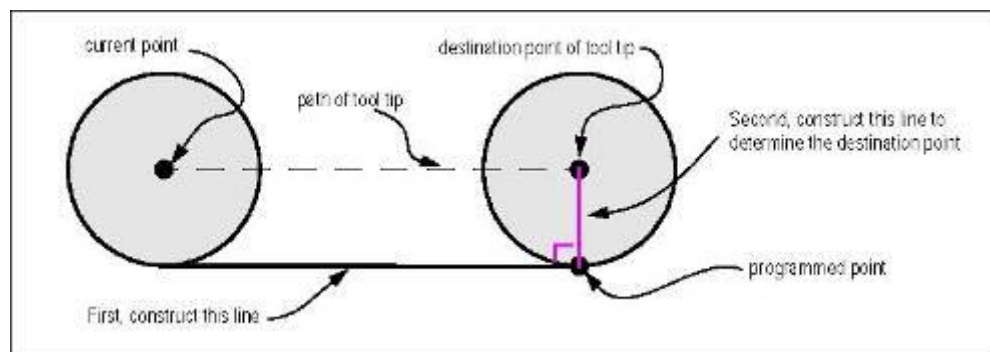
If the first move after cutter radius compensation has been turned on is an arc, the arc which is generated is derived from an auxiliary arc which has its center at the programmed center point, passes through the programmed end point, and is tangent to the cutter at its current location. If the auxiliary arc cannot be constructed, an error is signaled. The generated arc moves the tool so that it stays tangent to the auxiliary arc throughout the move. This is shown in Figure A-7.

Figure A-7, First cutter radius compensation move - Arc
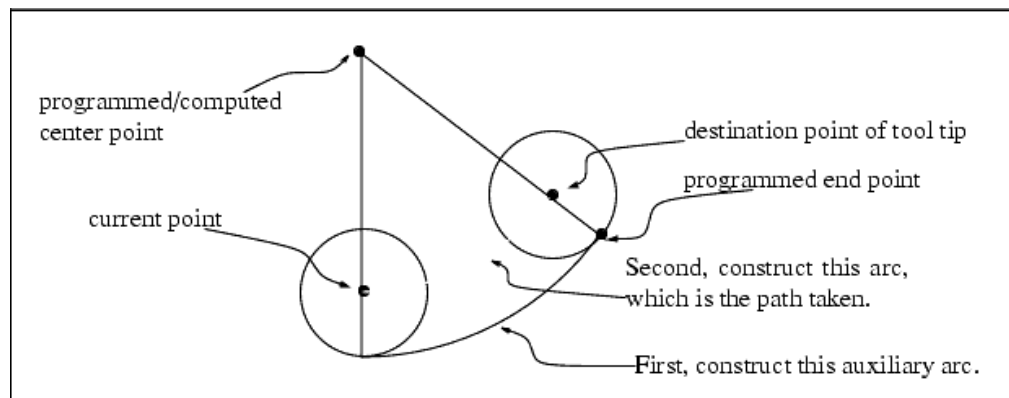
Figure A-7 shows the conceptual approach for finding the arc. The actual computations differ between the center format arc and the radius format arc (see Section 3.5.3).
After the entry moves of cutter radius compensation, the Interpreter keeps the tool tangent to the programmed path on the appropriate side. If a convex corner is on the path, an arc is inserted to go around the corner. The radius of the arc is half the diameter given in the tool table.
When cutter radius compensation is turned off, no special exit move takes place. The next move is what it would have been if cutter radius compensation had never been turned on and the previous move had placed the tool at its current position.

B Sample Programs

This appendix includes three sample programs. This section is intended to be useful to NC programmers and machine operators. Others will probably not find it useful. Additional sample programs are given in Table 3-5 (hole probing), Table A-1 and Table A-2 (cutter radius compensation). Table A-1 and Table A-2 are only sections of programs and need a line at the beginning (to select tool, change tool, start spindle, and set feed and speed rates) and a line at the end (an M2 to end the program).

B.1 SAMPLE SIMPLE PROGRAM

The classical simple computer program prints "Hello world". Here it is in RS274/NGC. The program may be safely run on a machining center.

```
(this program mills "Hello world" between X=0 and X=81 millimeters)
n0010 g21 g0 x0 y0 z50 (top of part should be on XY plane)
n0020 t1 m6 m3 f20 s4000 (use an engraver or small ball-nose end mill)
n0030 g0 x0 y0 z2
n0040 g1 z-0.5 (start H)
n0050 y10
n0060 g0 z2
n0070 y5
n0080 g1 z-0.5
n0090 x 7
n0100 g0 z2
n0110 y0
n0120 g1 z-0.5
n0130 y10
n0140 g0 z2
n0150 x11 y2.5
n0160 g1 z-0.5 (start e)
n0170 x16
n0190 g3 x13.5 y0 i-2.5
n0200 g1 x16
n0210 g0 z2
n0220 x20 y0
n0230 g1 z-0.5 (start l)
n0240 y9
n0250 g0 z2
n0260 x26
n0270 g1 z-0.5 (start l)
n0280 y0
n0290 g0 z2
n0300 x32.5
n0310 g1 z-0.5 (start o)
n0320 g2 x32.5 j2.5
n0330 g0 z2
n0340 x45 y5
n0350 g1 z-0.5 (start w)
n0360 x47 y0n0370 x48.5 y3
n0380 x50 y0n0390 x52 y5
n0400 g0 z2
n0410 x57.5 y0
n0420 g1 z-0.5 (start o)
n0430 g2 x57.5 j2.5
```

```
n0440 g0 z2
n0450 x64
n0460 g1 z-0.5 (start r)
n0470 y5
n0480 y4
n0490 g2 x69 r4
n0500 g0 z2
n0510 x73 y0
n0520 g1 z-0.5 (start l)
n0530 y9 n0540 g0 z2
n0550 x81
n0560 g1 z-0.5 (start d)
n0570 y0
n0580 x79.5
n0590 g2 j2.5 y5
n0600 g1 x81
n0610 g0 z50
n0620 m2
```

B.2 SAMPLE PROGRAM TO TEST EXPRESSIONS

This file is to test the interpretation of expressions. It tests all unary and binary functions implemented. It also tests parameter setting and referencing. The last few lines test more complicated expressions. This program is not intended to be run on a machining center. The tool path is just random back and forth on the X-axis.

```
n0010 g21 g1 x3 f20 (expression test)
n0020 x [1 + 2] (x should be 3)
n0030 x [1 - 2] (x should be -1)
n0040 x [1 --3] (x should be 4)
n0050 x [2/5] (x should be 0.40)
n0060 x [3.0 * 5] (x should be 15)
n0070 x [0 OR 0] (x should be 0)
n0080 x [0 OR 1] (x should be 1)
n0090 x [2 OR 2] (x should be 1)
n0100 x [0 AND 0] (x should be 0)
n0110 x [0 AND 1] (x should be 0)
n0120 x [2 AND 2] (x should be 1)
n0130 x [0 XOR 0] (x should be 0)
n0140 x [0 XOR 1] (x should be 1)
n0150 x [2 XOR 2] (x should be 0)
n0160 x [15 MOD 4.0] (x should be 3)
n0170 x [1 + 2 * 3 - 4 / 5] (x should be 6.2)
n0180 x sin[30] (x should be 0.5)
n0190 x cos[0.0] (x should be 1.0)
n0200 x tan[60.0] (x should be 1.7321)
n0210 x sqrt[3] (x should be 1.7321)
n0220 x atan[1.7321]/[1.0] (x should be 60.0)
n0230 x asin[1.0] (x should be 90.0)
n0240 x acos[0.707107] (x should be 45.0000)
n0250 x abs[20.0] (x should be 20)
n0260 x abs[-1.23] (x should be 1.23)
n0270 x round[-0.499] (x should be 0)
n0280 x round[-0.5001] (x should be -1.0)
n0290 x round[2.444] (x should be 2)
n0300 x round[9.975] (x should be 10)
```

n0310 x fix[-0.499] (x should be -1.0)
n0320 x fix[-0.5001] (x should be -1.0)
n0330 x fix[2.444] (x should be 2)
n0340 x fix[9.975] (x should be 9)
n0350 x fup[-0.499] (x should be 0.0)
n0360 x fup[-0.5001] (x should be 0.0)
n0370 x fup[2.444] (x should be 3)
n0380 x fup[9.975] (x should be 10)
n0390 x exp[2.3026] (x should be 10)
n0400 x ln[10.0] (x should be 2.3026)
n0410 x [2 ** 3.0] #1=2.0 (x should be 8.0)
n0420 ##1 = 0.375 (#1 is 2, so parameter 2 is set to 0.375)
n0430 x #2 (x should be 0.375) #3=7.0
n0440 #3=5.0 x #3 (parameters set in parallel, so x should be 7, not 5)
n0450 x #3 #3=1.1 (parameters set in parallel, so x should be 5, not 1.1)
n0460 x [2 + asin[1/2.1+-0.345] / [atan[fix[4.4] * 2.1 * sqrt[16.8]] /[-18]]**2]n0470 x sqrt[3**2 + 4**2] (x should be 5.0)
n0480 m2

B.3 SAMPLE PROGRAM TO TEST CANNED CYCLES

This file tests canned cycles. It is not intended to be run on a machining center.

n0010 g20 (cycle test)
n0020 g17 g43 h1 m3 s1234 f16 (start in XY-plane)
n0030 g81 x3 y4 r0.2 z-1.1 n0040 x1 y0 r0 g91 l3 (three more g81's an inch apart)
n0050 y-2 r0.1 (one more g81)
n0060 g82 g90 x4 y5 r0.2 z-1.1 p0.6
n0070 x2 z-3.0 (one more G82)
n0080 g91 x-2 y2 r0 l4 (four more g82's)
n0090 g83 g90 x5 y6 r0.2 z-1.1 q0.21
n0100 g84 x6 y7 r0.2 z-1.1
n0110 g85 x7 y8 r0.2 z-1.1
n0120 g86 x8 y9 r0.2 z-1.1 p902.61
n0130 g87 x9 y10 r0.2 z-1.1 i0.231 j-0 k-3
n0135 g91 x1 r0.2 z-1.1 i0.231 j-0 k-3
n0140 g88 x10 y11 r0.2 z-1.1 p0.3333
n0150 g89 x11 y12 r0.2 z-1.1 p1.272
n0160 m4 (run spindle counterclockwise)
n0170 g86 x8 y9 r0.2 z-1.1 p902.61
n0180 g87 x9 y10 r0.2 z-1.1 i0.231 j-0 k-3
n0190 g88 x10 y11 r0.2 z-1.1 p0.3333

n0220 g18 m3 (now run all cycles in the XZ-plane)
n0230 g81 z3 x4 r0.2 y-1.1 n0240 g91 z1 x0 r0 l3
n0260 g82 g90 z4 x5 r0.2 y-1.1 p0.6
n0280 g91 z-2 x2 r0 l4
n0290 g83 g90 z5 x6 r0.2 y-1.1 q0.21
n0300 g84 z6 x7 r0.2 y-1.1
n0310 g85 z7 x8 r0.2 y-1.1
n0320 g86 z8 x9 r0.2 y-1.1 p902.61
n0330 g87 z9 x10 r0.2 y-1.1 k0.231 i-0 j-3
n0335 g91 z1 r0.2 y-1.1 k0.231 i-0 j-3
n0340 g88 z10 x11 r0.2 y-1.1 p0.3333
n0350 g89 z11 x12 r0.2 y-1.1 p1.272
n0420 g19 (now run all cycles in the YZ-plane)

```
n0430 g81 y3 z4 r0.2 x-1.1
n0440 g91 y1 z0 r0 l3
n0460 g82 g90 y4 z5 r0.2 x-1.1 p0.6
n0480 g91 y-2 z2 r0 l4
n0490 g83 g90 y5 z6 r0.2 x-1.1 q0.21
n0500 g84 y6 z7 r0.2 x-1.1
n0510 g85 y7 z8 r0.2 x-1.1
n0520 g86 y8 z9 r0.2 x-1.1 p902.61
n0530 g87 y9 z10 r0.2 x-1.1 j0.231 k-0 i-3
n0535 g91 y1 r0.2 x-1.1 j0.231 k-0 i-3
n0540 g88 y10 z11 r0.2 x-1.1 p0.3333
n0550 g89 y11 z12 r0.2 x-1.1 p1.272
n1000 m2 (the end)
```

B.4 HARDWARE INSTALLATION TIPS

USB is sensitive to EMC noise. Communication problems can occur that result in a "buffer-underun" or "communication failed" and thus a stop of the machine.

To prevent this, the USBCNC CPU should be build in correctly according these general EMC rules:

- Mount all electronics in a metal cabinet or on a metal plate in a plastic cabinet.
- Use a mains filter.
- Create a central GND point near the filter and connect the PE (Protective Earth) as well as the GND from all power supplies to this point.
- Connect the GND point on the green screw connector of CPU4 also to this point.
- Use shielded cables for the motor connections, both inside the cabinet and outside the cabinet. Connect the shield at one side to the central ground point, leave the other side un-connected.
- Use a professional USB2 cable, double shielded with ferrites like this:

- Keep all GND cables especially short and use thick flexible cable
- If not possible to keep it short, then connect it to the metal GND plate.

Here a picture of my own system, it contains various EMC problem makers, like 2 Switched mode power supplies and a frequency inverter for a HF spindle.

Also there a 4 stepper-motor drives working at 75 Volt, motor currents 4,2 Amp.

If all these tips have been applied and there is still an EMC problem, then there is only one solution left, and that is to build the CPU board itself in a small aluminum housing.

B.5 DEFAULT MACRO.CNC FILE

```
;****************************************
;* This is file macro.cnc
;* It is automatically loaded
;* Customize this file yourself
;* It contains:
;* - subroutime change_tool this is called on M6T..
;* - subroutime home_x .. home_z, called when home functions in GUI are activated
;* - subroutine home_all, called when home all button in GUI are activated
;* - subroutime user_1 .. user_11, called when user functions are activated
;*   user_1 contains an example of zeroing Z using a moveable ool setter
;*   user_2 cintains an example of measuring the tool length using a fixed tool setter
;*
;* You may also add frequently used macro's in this file.
;****************************************


;User functions, F1..F11 in user menu

;Zero tool tip example
Sub user 1
    msg "user_1, Zero Z (G92) using toolsetter"
    (Start probe move, slow)
    f30
    g38.2 z-100
    (Move back to touch point)
    g0 z#5063
    (Set position, the measuring device is 43mm in height, adapt for your measuring device)
    G92 z43
    (move 5 mm above measuring device)
    g91 (incremental distance mode)
    g0 z5
    g90 (absolute distance mode)
    m30
Endsub

;Tool length measurement example
Sub user_2
    goSub m_tool ;See sub m_tool
Endsub

Sub user 3
    msg "sub user 3"
Endsub

Sub user_4
    msg "sub user 4"
Endsub

Sub user_5
    msg "sub user_5"
Endsub

Sub user_6
    msg "sub user 6"
Endsub

Sub user_7
    msg "sub user_7"
Endsub

Sub user 8
    msg "sub user_8"
Endsub

Sub user_9
    msg "sub user_9"
Endsub

Sub user_10
    msg "sub user_10"
Endsub

Sub user 11
    msg "sub user 11"
Endsub

;Homing per axis
Sub home_x
    home x
    ;;if A is slave of X uncomment next lines
    ;homeTandem X
Endsub
```

```
Sub home_y
    home y
Endsub

Sub home_z
    home z
Endsub

Sub home_a
    home a
Endsub

Sub home_b
    home b
Endsub

Sub home_c
    home c
Endsub

;Home all axes, uncomment or comment the axes you want.
sub home_all
    gosub home z
    gosub home x
    gosub home_y
    gosub home_a
    ;gosub home_b
    ;gosub home_c
    ;g53g0x50y50 ; move x=50 and y=50 to machine coordinates
        ;; Uncomment next line if youy whish to use the homeIsEstop feature
    ;homeIsEstop on
        m30
endsub


;This example shows how to make your own tool_changer work.
;It is made for 16 tools
;First current tool is dropped, then the new tool is picked
;There is a check whether selected tool is already in the spindle
;Also a check that the tool is within 1-16
;There is a picktool subroutine for each tool and a droptool subroutine for each tool.
;These routines need to be modified to fit your machine and tool changer

sub change_tool
    ;Switch off guard for tool change area collision
    TCAGuard off

    ;Use #5015 to indicate succesfull toolchange
    #5015 = 0 ; Tool change not performed

    if [ [#5011] == [#5008] ]
       msg "Tool already in spindle"
    endif

    ; check tool in spindle and exit sub
    If [ [#5011] <> [#5008] ]
        if [[#5011] > 16 ]
            errmsg "Please select a tool from 1 to 16."
        else
            ;Drop current tool
            If [[#5008] == 1]
                GoSub DropTool1
            endif
            If [[#5008] == 2]
                GoSub DropTool2
            endif
            If [[#5008] == 3]
                GoSub DropTool3
            endif
            If [[#5008] == 4]
                GoSub DropTool4
            endif
            If [[#5008] == 5]
                GoSub DropTool5
            endif
            If [[#5008] == 6]
                GoSub DropTool6
            endif
            If [[#5008] == 7]
                GoSub DropTool7
            endif
            If [[#5008] == 8]
                GoSub DropTool8
            endif
            If [[#5008] == 9]
                GoSub DropTool9
            endif
```

```
            If [[#5008] == 10]
                GoSub DropTool10
            endif
            If [[#5008] == 11]
                GoSub DropTool11
            endif
            If [[#5008] == 12]
                GoSub DropTool12
            endif
            If [[#5008] == 13]
                GoSub DropTool13
            endif
            If [[#5008] == 14]
                GoSub DropTool14
            endif
            If [[#5008] == 15]
                GoSub DropTool15
            endif
            If [[#5008] == 16]
                GoSub DropTool16
            endif

            ;Pick new tool
            if [[#5011] == 1]
                GoSub PickTool1
            endif
            if [[#5011] == 2]
                GoSub PickTool2
            endif
            if [[#5011] == 3]
                GoSub PickTool3
            endif
            if [[#5011] == 4]
                GoSub PickTool4
            endif
            if [[#5011] == 5]
                GoSub PickTool5
            endif
            if [[#5011] == 6]
                GoSub PickTool6
            endif
            if [[#5011] == 7]
                GoSub PickTool7
            endif
            if [[#5011] == 8]
                GoSub PickTool8
            endif
            if [[#5011] == 9]
                GoSub PickTool9
            endif
            if [[#5011] == 10]
                GoSub PickTool10
            endif
            if [[#5011] == 11]
                GoSub PickTool11
            endif
            if [[#5011] == 12]
                GoSub PickTool12
            endif
            if [[#5011] == 13]
                GoSub PickTool13
            endif
            if [[#5011] == 14]
                GoSub PickTool14
            endif
            if [[#5011] == 15]
                GoSub PickTool15
            endif
            if [[#5011] == 16]
                GoSub PickTool16
            endif
        endif
    endif

    If [[#5015] == 1]
        msg "Tool "#5008" Replaced by tool "#5011" G43 switched on"
        m6t[#5011]
        G43
         else
             ;new tool had number 0, which means no tool
        if [[#5011] == 0]
            msg "Tool removed"
            m6t[#5011]
                        G49
        endif
    endif

    ;Set default motion type to G1
```

```
    g1

    ;Switch on guard for tool change area collision
    TCAGuard on

EndSub


;Drop tool subroutines
Sub DropTool1
    msg "Dropping tool 1"
endsub
Sub DropTool2
    msg "Dropping tool 2"
endsub
Sub DropTool3
    msg "Dropping tool 3"
endsub
Sub DropTool4
    msg "Dropping tool 4"
endsub
Sub DropTool5
    msg "Dropping tool 5"
endsub
Sub DropTool6
    msg "Dropping tool 6"
endsub
Sub DropTool7
    msg "Dropping tool 7"
endsub
Sub DropTool8
    msg "Dropping tool 8"
endsub
Sub DropTool9
    msg "Dropping tool 9"
endsub
Sub DropTool10
    msg "Dropping tool 10"
endsub
Sub DropTool11
    msg "Dropping tool 11"
endsub
Sub DropTool12
    msg "Dropping tool 12"
endsub
Sub DropTool13
    msg "Dropping tool 13"
endsub
Sub DropTool14
    msg "Dropping tool 14"
endsub
Sub DropTool15
    msg "Dropping tool 15"
endsub
Sub DropTool16
    msg "Dropping tool 16"
endsub


;Pick tool subroutines
Sub PickTool1
    msg "Picking tool 1"
    #5015 = 1 ; toolchange succes
endsub
Sub PickTool2
    msg "Picking tool 2"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool3
    msg "Picking tool 3"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool4
    msg "Picking tool 4"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool5
    msg "Picking tool 5"
    #5015 = 1 ; toolchange succes
endsub
Sub PickTool6
    msg "Picking tool 6"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool7
    msg "Picking tool 7"
    #5015 = 1 ; Tool change succes
endsub
```

```
Sub PickTool8
    msg "Picking tool 8"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool9
    msg "Picking tool 9"
    #5015 = 1 ; toolchange succes
endsub
Sub PickTool10
    msg "Picking tool 10"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool11
    msg "Picking tool 11"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool12
    msg "Picking tool 12"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool13
    msg "Picking tool 13"
    #5015 = 1 ; toolchange succes
endsub
Sub PickTool14
    msg "Picking tool 14"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool15
    msg "Picking tool 15"
    #5015 = 1 ; Tool change succes
endsub
Sub PickTool16
    msg "Picking tool 16"
    #5015 = 1 ; Tool change succes
endsub

Sub calibrate tool setter
    warnmsg "close MDI, check correct calibration tool nr 16 data in tool table"
    warnmsg "jog to toolchange safe height, when done press RUN"
    #4996=#5073 ;Store toolchange safe height machine coordinates
    warnmsg "insert calibrationtool 16 length=" #5416 ", jog just above tool setter, when done press RUN"
    ;store x y in non volatile parameters (4000 - 4999)
    #4997=#5071 ;machine pos X
    #4998=#5072 ;machine pos Y
    ;Determine minimum toochuck height and store into #4999
    g38.2 g91 z-20 f30
    #4999=[#5053 - #5416] ;probepos Z - calibration tool length = toolchuck height
    g90
    g0 g53 z#4996
    msg "calibration done safe height="#4996 " X="#4997 " Y="#4998 " Chuck height="#4999
endSub

sub m_tool
    ;Check if toolsetter is calibrated
    if [[#4996 == 0] and [#4997 == 0] and [#4998 == 0] and [#4999 == 0]]
             errmsg "calibrate toolsetter first open mdi, enter gosub calibrate tool setter"
         else
        g0 g53 z#4996 ; move to safe z
        dlgmsg "enter tool dimensions" "tool number" 5016 "approx tool length" 5017 "tool diameter" 5018

        if [[#5016 < 1] OR [#5016 > 15]]
            ErrMsg "Tool must be in range of 0 .. 15"
        endif

        ;move to toolsetter coordinates
        g00 g53 x#4997 y#4998
        ;move to 10mm above chuck height + approx tool length + 10
        g00 g53 z[#4999+10+#5017]
             ;measure tool length and pull 5mm back up
        g38.2 g91 z-20 f30
        g90
        ;back to safe height
        g0 g53 z#4996
        ;Store tool length, diameter in tool table
        #[5400 + #5016] = [#5053-#4999]
        #[5500 + #5016] = #5018
        #[5600 + #5016] = 0 ;Tool X offset is 0
        msg "tool length measured="#[5400 + #5016]" stored at tool "#5016
    endif
endsub
```