SpareTimeLabs

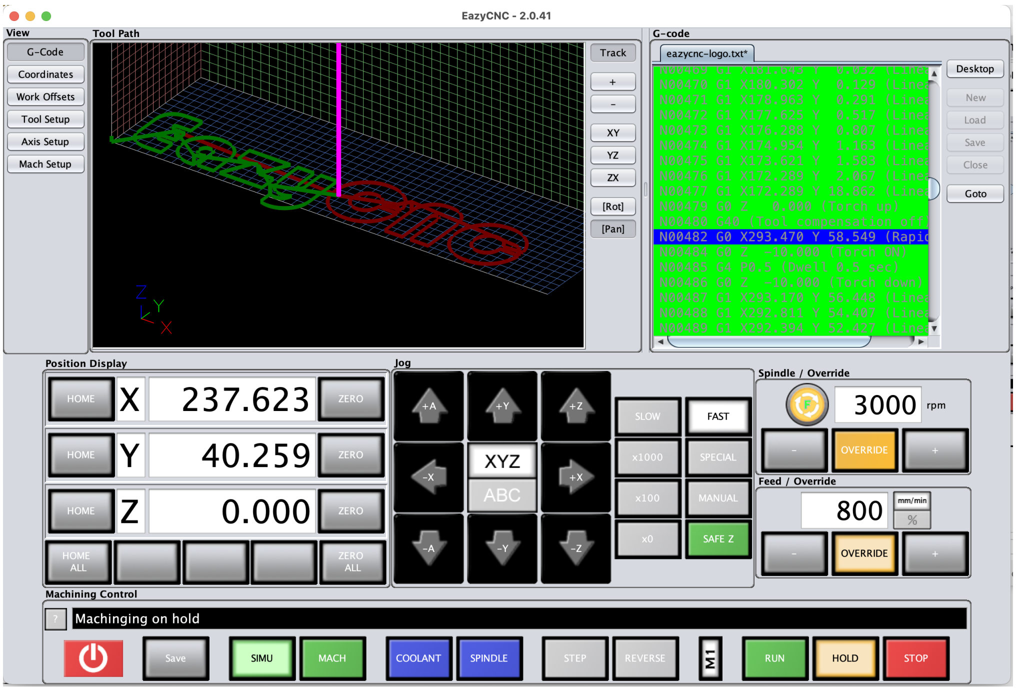
eazycnc@eazycnc.com

# EazyCNC – Manual

Revision 2

for

EazyCNC version 2.0.41




April 7, 2024

# Contents

# List of Tables

# List of Figures

# Preface

# Disclaimer

EazyCNC is program to control the operation of a CNC-Machine Tool.

TOAD4 is a microprocessor based controller board for controlling stepper motors.

EazyCNC and TOAD4/TOAD5 are intended for the hobbyist, they are not intended for for professional/commercial use.

Any machine tool is potentially dangerous.

All electrical system have the potential to cause an electric shock or a fire hazard.

All motorized systems can cause serious personal injury or damage to property.

All computer programs have design or implementation flaws (bugs) some of which can cause serious malfunction of the system.

Most countries and states have regulations and standards that govern the design, construction, use, deployment and placing on the market of electrical and mechanical equipment, including the software used to control them.

*No safety of design or construction or programming nor warranty is implied, instead it is the responsibility of whoever uses or deploys EazyCNC and/or TOAD4 to ensure that he/she understands the implications of using EazyCNC and/or TOAD4 and to comply with any legislation and codes of practice applicable to his/hers country or state. Further it is his/hers responsibility to ensure the safety of the system at all times*.

If you are in any doubt, you must seek guidance from a professionally qualified expert rather than risk injury or liability to yourself or to others.

SpareTimeLabs or Kustaa Nyholm cannot accept any responsibility resulting from the design, construction or use of EazyCNC and/or TOAD4.

All names of products and trademarks used in this manual are for example purposes only, no endorsement of any of them by SpareTimeLabs nor endorsement of EazyCNC or TOAD4 by their respective owners is implied.

# Chapter 1

# Safety First!

Machine tools are dangerous!

Always keep that in mind, both when designing and setting up your system and when operating it on a daily bases.

CNC machine tools are heavy and strong machinery, moving sharp and hot cutting tools or extremely powerful plasma torches under computer control. Computers are complex systems and it is *impossible* to ensure 100% error free and safe operation in every situation. It is perfectly possible that a software design flaw, called bug, cause the system to operate unexpectedly or even run away wild.

Therefore it is very important to take appropriate precautions for such an eventuality.

Every system needs to have an Emergency switch fitted.

The emergency switch needs to be so wired that it will prevent any machine movement and stops spindle or shuts down the torch arc when activated.

The emergency switch needs to be mounted to a place that is easily accessible when operating the machine.

The emergency switch has to be of the latching kind in other words: once activated it must stay activated until manually de-activated.

Depending on the physical layout and power of your machine movements you need to consider if you should activate the emergency switch whenever you have your hands or limbs inside the working area of your machine.

With some machine configurations it may be preferable not to activate the emergency switch if you need to pause the system in the middle of machining, for example to change the tool bit because the axes might lose their positions and it may be acceptable to just ensure that the spindle will not start on its own.

For that purpose a kill switch to the spindle motor controller maybe fitted that will prevent the spindle from running no matter what the control systems does.

Above does not by any means endorse any particular way of ensuring safety and no responsibility or liability is accepted by me. You need to do your own risk and safety assessment and act accordingly.

# Chapter 2

# Introduction to CNC-Machining

## 2.1 Overview of a CNC Machining System

This is probably familiar territory for you otherwise you would not be here in the first place but this section is short introduction to tell you where exactly EazyCNC and TOAD4 fits in the big picture.

Figure 2.1: a CNC Machining System Overview

The CNC machining process starts with a design of the part to be machined which is turned into a sequence of instructions to the computer that controls the motors, typically stepper motors, that move the cutting tool (or work piece) via series of gears, belts, pulleys and/or screws. These tool movements are typically called 'axes', for example X-axis, Y-axis etc.

The 'sequence of instructions' is called G-code and it is basically a text file with coordinate points that define the path the cutting tool will make.

G-code can be hand written but is typically generated automatically from a CAD (Computer Aided Design model) of the part using CAM (Computer Aided Manufacturing) software, either directly by the CAD/CAM program or by a program called post-processor.

The G-code file is read by a program that turns the coordinate information and other commands in the G-code file into motor control pulses in real-time observing programmed feed rates and machine parameters such as number of pulses required to move a unit distance.

This is where EazyCNC/TOAD4 comes into picture because EazyCNC is the program that does the G-code interpretation and TOAD4 is a micro controller that does the real-time motor control.

## 2.2 Understanding Real-Time

In common language real-time means roughly 'as it happens' but in computer jargon real-time has a specific and important meaning.

Real-time here means that the pulses that control the distance and speed of movements need to be generated precisely at appropriate rate because the motors and mechanisms that move the axis have physical limits beyond which they fail to move as required.

EazyCNC runs on a personal computer such as an IBM PC compatible or a Macintosh computer. These computers use an operating system, such as Windows, Linux or Mac OS X, that are not ideally suited to real-time control. You can easily get a feel for this when you plug in a USB-device as often the mouse cursor stops for a second or two – imaging if the system paused like that when it should be turning a corner.

There are different ways out of this difficulty. A Windows program called Mach3 uses a special 'driver' software for the real-time stuff and another program called EMC2 uses a specially 'patched' version of the Linux operating system just to mention two.

EazyCNC takes a different approach.

EacyCNC delegates the most demanding real-time tasks to the TOAD4 micro controller which is better equipped and positioned to do the precise real-time generation of motor control pulses and such because it does not have to deal with the endless variety of the PC hardware and software and because it has been designed from ground up for the very task of performing real-time control.

EazyCNC reads and interprets the G-code and breaks it into bite size chunks that the simple micro controller in the TOAD4 can process in real-time.

These bite size chunks are transferred from the PC to TOAD4 over the USB bus and put into a command queue in the TOAD4 micro controller.

EazyCNC attempts to keep the queue full at all times so that if EazyCNC or the operating system it runs on needs to 'pause' for a second or so there is still data for TOAD4 to process and machining can continue without missing a beat.

This is important because when stepper motors are run at high speed a delayed pulse is equivalent to a sudden deceleration which may cause the system to exceed the stepper motor's max torque in which case the stepper motor will not be able to 'hold' its position and accuracy is compromised.

## 2.3 Understanding Stepper Motors

While EazyCNC and TOAD4 can be used with Servo Motors they really are meant to be used with Stepper Motors.

Stepper Motor are motors with two or more stationary coils and a rotating permanent magnet rotor. By energizing the coils in sequence the rotor can be made to move.

Stepper Motors have some interesting and important properties.

First of all they do not 'run' if you just energize them, at most they make a single small movement called step. This makes them rather safe as a short circuited transistor in the drive system cannot make the motor run wildly.

Secondly when you apply a controlled energizing sequence into the coils the motor makes precise fractional rotational movements called steps, a typical stepper motor step is 1.8°or 200 steps/revolution.

It is this second property that makes steppers very attractive for controlling precise movements.

If you 'step' a stepper motor ten times you can be pretty sure that it actually moved ten steps. So there is no need to measure position of the motor in any other way than counting the pulses we send to it, no need for expensive encoders and or position scales.

This makes stepper motors very cost-effective way to implement motion control.

But the lack of position feedback is also the downside of stepper motors.

The positional control of a stepper motor based system relies on an initial position and keeping track of the steps and their direction.

The initial position can be either given manually or found automatically by using a reference switch.

Manually means that you move the motor/axis to a known position, such as to the end of the movement range, and tell the system that this is it.

If a reference switch is available then the system can move the axis/motor through its range of movements and make a note of the step number on which the reference switch is reached and in this way calibrate its position.

Keeping track of the pulse and their direction is done automatically and precisely by the software but under certain circumstances the motor cannot 'carry out the step' and the system loses track of the real physical position, this is called missing steps.

Missing steps can happen if the stepper is stepped too fast or the load exceeds what the motor can deliver. To prevent that the correct maximum speed and acceleration parameters need to be programmed into the system.

Also note that manually forcing the axes to move, if you manage to overpower the motors, will cause the system to lose its position, so all manual movements must be done via the 'jog' controls of the system.

It must also be mentioned that since we are controlling the motor position, not the cutter position, any backlash or play in the mechanism is *not* automatically compensated for.

Typically not of practical concern but good to know is the fact that stepper motors are not very 'stiff'. Even though the motor has specific torque it actually has very little torque when the magnetic poles of the coil and rotor are aligned i.e. at every full step.

You can think about this as if the rotor and stator poles were connected with springs; when the poles are aligned the spring will not pull the rotor one way or the other, they only exert force and torque when the rotor is moved into misalignment and thus there is almost always a small but measurable error between the physical and ideal step position.

There is of course a lot more to know and understand about Stepper Motors but above is the most import thing to keep in mind when working with systems based on them.

# Chapter 3

# Hardware and Software Requirements

This chapter gives you important information about the software and hardware requirements for a system based on EazyCNC and TOAD4.

## 3.1 Dedicated Computer

The computer used for CNC machining should be dedicated for the CNC system and not shared for other use.

A CNC machine should not be directly connected to Internet because of the possibility of viruses and malware causing havoc in an environment where they can cause actual physical damage and injury.

If the dedicated CNC PC is connected to a local network it is important that the network is secure and runs behind a firewall and the all network connectivity on the CNC PC is kept to minimum.

No other software besides the operating system and EazyCNC should be running on the CNC PC during machining. You should especially watch out for programs that start automatically behind the scenes when the computer boots up. Utility programs to kill non required applications and processes exist.

All screen savers, auto log-off and power save features and modes should be turned off.

Once the system has been set up, configured properly and tested it should be 'freezed' and any upgrades and change

## 3.2 Hardware

It is impossible to give hard limits as to which kind of PC computer should be used with EazyCNC.

In a less demanding application at moderate feed rates you can get by with a less powerful computer.

It is tempting to utilize that old PC that is just lying there gathering dust, but that is not recommended.

Consider that you are building an automated machining tool and for that you want reliability and responsiveness. A suitable PC can be purchased for a few hundred euros/dollars and it is well worth it, considering what you are building and what you are going to do with it.

A fairly recent and decent PC with at least 2 GHz processor, good graphics card, a minimum of 4 GB RAM is recommended as the minimum. To install and run EazyCNC you need at least 400 MB of free disk space.

During machining all the G-code data as well as the tool path graphics is kept in memory so there is no such thing as too much memory, too fast graphics card or too fast CPU!

## 3.3 Operating System

The following operating systems/version have been tested:

- Mac OS X 14.1
- Windows 10
- Linux Ubuntu 18.04 LTS
- Raspberry Pi OS 4.3 (bullseye)

EazyCNC is built on Java which is fairly operating system independent so it is likely it will run with a wide range of versions of above mentioned operating systems but of course it is not possible to guarantee that.

Also worth noting is that operating system version tend to become obsolete and unsupported in a matter of some years and while I try to maintain compatibility with older OS versions it may become impossible at any time.

## 3.4 Anti-Virus Software

An anti-virus software is not recommended on the PC running EazyCNC as such software is very intrusive and can cause real-time violations and machining failures. Further, it is not very necessary because the CNC PC should not be connected to Internet.

It is of course important that anti-virus software or some other means are used to ensure clean operation of the computer producing or transferring the G-code file as a virus might infect the media, such as a USB memory stick, used to transfer the G-code to the CNC system.

# Chapter 4

# Installation

## 4.1 Getting the Application

Note that do not need to have actual hardware or any drivers installed to run and play with the software.

EazyCNC is distributed via Internet so just download the file appropriate for your operating system from the EazyCNC website at:

http://www.eazycnc.com/downloads/downloads.php

Depending on your operating system EazyCNC is distributed as a compressed file which you may have to un-compress with the tools in you operating system.

## 4.2 Installing the Application

For macOS EazyCNC comes as a single executable file which does not require an installer. Just copy the file to where ever you want in your computer's hard disk.

For Windows EazyCNC comes as a single .msi installer file that you need to run by double clicking it. Once installation is complete you can find EazyCNC from the Start Menu along all the other applications.

For Linux EazyCNC comes as single file with .deb extension. This is a Debian package that many browsers will automatically install when you download it and try to open it.

It can also be installed from the command line with:

sudo dpkg -i <filename>

After installation EazyCNC should appear as any other application, the exact details vary between distros.

# Chapter 5

# Overview of EazyCNC

The main purpose of EazyCNC is to read machining instructions in the form of a G-code file and control accordingly the motors that move the machine tool. In addition to that it provides the necessary controls to manually move the machine axes.

## 5.1 Moving Around in the Program

Figure 5.1 shows the 'main screen' of the application. Note that if your screen resolution is small then the titles of the boxes aka panels are not displayed to conserve some screen real estate.

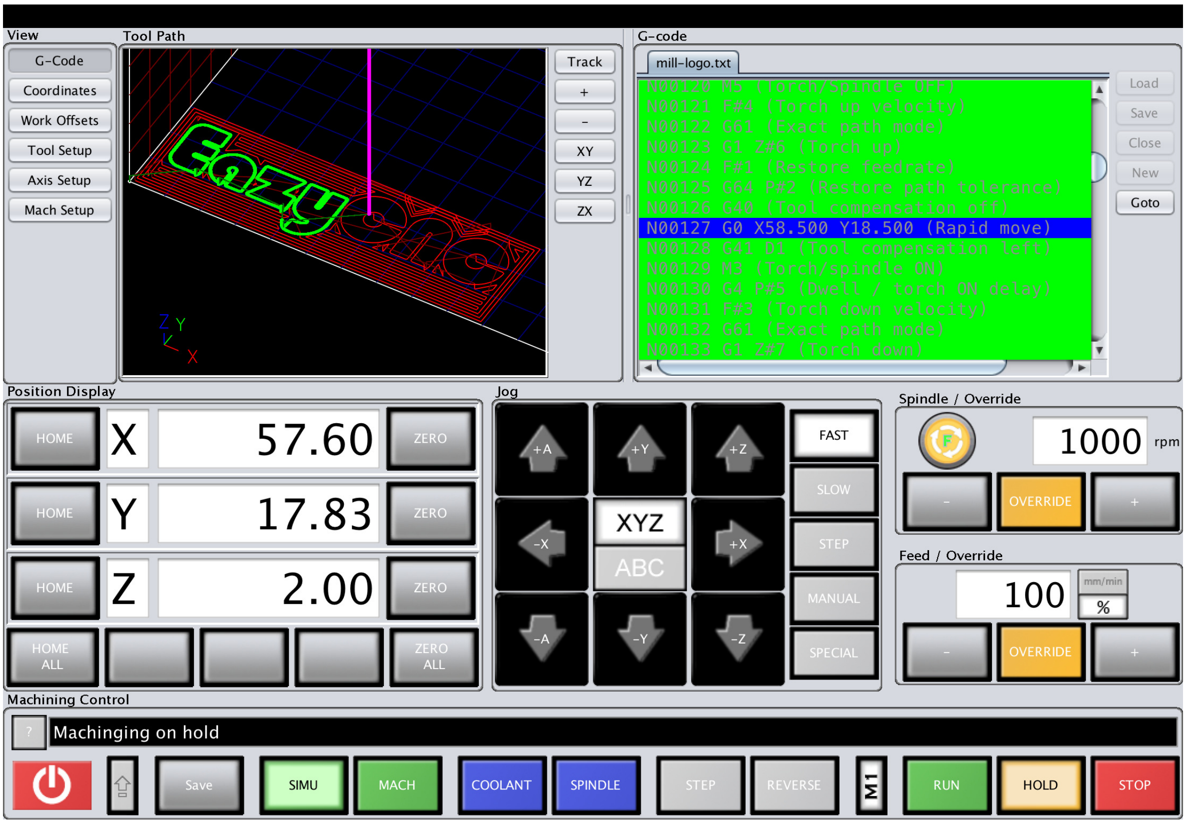



Figure 5.1: EazyCNC Main Screen – the G-code view

The screen is logically dived into two parts, upper and lower half. The lower half is always the same, but the upper half changes depending on which screen or view you are.

On the upper left corner, Figure 5.2 are buttons that control what is show on the upper half. When this manual says 'go to screen' or 'in screen' it refers to these view control buttons and the different screens they bring up.

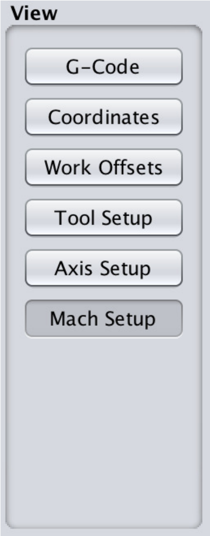



Figure 5.2: The view selection buttons

The lower half of the screen always shows the controls that are used to cause the machine to actually move.

For a typical usage the main screen, 'G-code', provides all the controls that are necessary to open a G-code file and machine the part it represents.

The other screens are for setting up coordinate systems, tool parameters and to configure various aspects of the machining system.

On the main screen the upper right quarter of the screen the current G-code file is shown with the line G-code line being executed highlighted in blue.

On the upper left quarter a 3D view of the tool path is shown with the path already executed shown in green and the path that remains to be machined shown in red.

Below those, from left to right, there are the coordinate read outs (DROs) displaying the tool position, jog controls to manually move the tool, user programmable function keys for repeated tasks and manual spindle controls and feed override controls.

At the bottom row there are buttons to control the actual machining and running of the G-code program, either in simulation mode or actually cutting some metal. With these controls it is also possible to temporarily pause the execution and run the G-code step by step and even backwards.

Usage and cutting metal with EazyCNC is described in detail in Chapter 7.

Setting up of EazyCNC is described in the following chapter.

# Chapter 6

# Setting Up and Configuring

EazyCNC should work out of the box without any setup or configuration, so you can play around and test it right away.

However before you can actually use it with TOAD4 or TOAD5 and machine something there are a few things you need to set up and configure.

This chapter proceeds in the preferred order of setting things up, however I suggest you read it all through once before setting up the system.

The two most important thing to set up is to configure each axis to match the motor characteristics and axis gearing.

## 6.1 Saving Your Setup

EazyCNC stores all setup and configuration information as well as the current machine state in file named 'EazyCNC-Mach-Config.ecnc' in a directory named 'EazyCNC' in your 'home' directory.

Your 'home' directory location depends on your operating system as follows:

| | |
|---|---|
| Mac OS X | `/Users/username` |
| Windows 10 | `C:\Users\username` |
| Linux | `/Users/username` |

where *username* is the name you use when you log into your computer.

Note, EazyCNC does not ever automatically save the settings, this is to protect you from accidentally altering your carefully crafted machine setup and configuration. To save your settings or any changes you've made you need to click the 'SAVE' button.

If the file or folder does not exist EazyCNC will create them with reasonable default settings when you click the 'SAVE' button.

You do not have to 'worry' about the configuration file but it is good to know about it as you may want to make backup copies of it or maintain several different ones for different system configurations.

The file is in a plain text format so it is ok to view and even edit it manually, though that is not recommend unless you know what you are doing.

Many programs like Mac OS X Textedit may mess things up by adding formatting and changing the file name extension with .txt or .rtf, so learn to avoid those if you venture there.

### 6.1.1 Setting up permission in Linux

If you are not running EazyCNC under Linux you can skip this section.

Starting with EazyCNC version 2.0.41 if you install it from the supplied .deb file you do not have to worry about udev rules and can also skip this section.

Most Linux distros take a very serious view of security. This means that by default you are not even allowed to use your own devices! To complicate matters there is no easy way to give yourself the necessary permissions, instead you have to do the following.

To allow EazyCNC to talk to the TOAD4 or TOAD5 you need to create a file named:

```
99-TOADX.rules
```

and put it in the directory

```
 /etc/udev/rules.d
```

In the file '99-TOAD4.rules' you need to put the following line (all on one line):

```
SUBSYSTEMS=="usb", ATTRS{idProduct}=="000a", ATTRS{idVendor}=="0408", MODE="0666", GROUP="plu
```

Basically you can do that with any text editor. Unfortunately you do not have the permission to write it to the directory '/etc/udev/rules.d'. So I recommend creating the file in your home directory and then use the Terminal and type in the following command to copy the file over:

```
sudo cp ~/99-TOAD4.rules /etc/udev/rules.d
```

This command will ask for your password to allow you temporarily write to that directory.

Alternatively following one-liner should accomplish setting up the rules for you:

```
sudo echo 'SUBSYSTEMS=="usb", ATTRS{idProduct}=="000a", ATTRS{idVendor}=="0408", MODE="0666",
```

Note that above absolutely must be on one line, so if you copy paste it from here, make a practice paste to a text file to see that you got it all on one line.

## 6.2 Updating Motor Controller Firmware

You can check which firmware version your TOAD4 or TOAD5 board currently has by activating the MACH button. The version will be displayed on the status display.

At the time of updating this manual for EazyCNC 2.0.41 the latest firmware was 2.0.10-4 for TOAD4 and 2.0.10-5 for TOAD5. The last number of the version number indicates which hardware the firmware is for.

Never update the firmware using a ICSP programmer or PICKit!

You can update the firmware in The Motor Controller screen, see . Figure 6.1


Figure 6.1: Mach Setup when Controller button is selected

To update the firmware do as follows.

Connect your TOAD hardware to a USB port and power up the controller.

For safety it is better if none of the machinery that the controller is controlling are powered up at the same time!

Ensure that EazyCNC is NOT in MACH mode.

Select which firmware version you want to upload from the Firmware -popup.

Click the UPDATE -button.

At this point a dialog box will appear that shows the progress of the firmware update.

*Do not touch anything in the computer or hardware during the update.*

If the firmware update is interrupted you may have to force the TOAD board into the firmware update mode, check the corresponding TOAD hardware manual.

### 6.2.1 Firmware -popup menu

EazyCNC comes with all the firmware versions that have been released at the time of the EazyCNC release and those versions are listed in this popup menu.

If the firmware you want to upload is not listed you can either update your EazyCNC, which should never be taken lightly and without good judgement, or you can obtain the firmware file (.hex) separately.

If you want to use a separate firmware file select the 'From file...' option from the popup menu.

#### UPDATE -button

When you click this button the firmware will be updated.

If you selected 'From file...' in the 'Firmware' -popup then a file dialog will be present at this point and you need to select the correct firmware file.

During the update a dialog box detailing the update progress is shown. This dialog box will disappear after the update is complete. A success/fail message will be show briefly in the Status Display, if you miss that you can always press and hold down the little button marked '?' next to it.

If update was successful the firmware version should show up in the Status Display when you click the MACH button.

### 6.2.2 Testing the Motor Controller Connection

To test that the motor controller is connected power it up and click the MACH button in the EazyCNC user interface.

If everything is as working properly you should see a message next to the button marked with '?' that says something like:

```
Controller: Connected -- firmware version x.y.z-q
```

## 6.3 Enabling Debug Logs

EazyCNC can collect diagnostic information into log files stored inside your EazyCNC directory to help in trouble shooting.

By default the collection of that information is not enabled as it can slow things down which in itself can cause problems.

Your EazyCNC directory location depends on your operating system as follows:

| | |
|---|---|
| Mac OS X | `/Users/username` |
| Windows 10 | `C:\Users\username` |
| Linux | `/Users/username` |

where *username* is the name you use when you log into your computer.

Log files are periodically deleted from the file system if they are older than seven days.

### 6.3.1 Enable Java Console to file logging -checkbox

EazyCNC outputs diagnostic information to a window name Java Console. You can bring up (or hide) the Java Console by pressing the F12 key.

If this checkbox is ticked then anything that is output to the Java Console is also written to a text file in the directory named 'java-console' inside your EazyCNC directory.

These are all text files and you can open them with a text editor should you wish to have a peek.

### 6.3.2 Enable TOAD4 communication logging -checkbox

If this checkbox is ticked then all communication is between EazyCNC and TOAD4/TOAD5 is written to a binary file in the directory named 'debuglogs' inside your EazyCNC directory.

These are binary files and you need some special knowledge to be able to view them.

## 6.4 Setting up the User Interface

Now that the communication works and before we set up the motors and axis we want to select the length units you are comfortable with.

To do that go the 'User Interface' setup screen Figure 6.2

EazyCNC supports working in millimeters or inches. All displays and entry fields will always show values in the selected units and accept values in these units. You can change the units at any time and it will not confuse EazyCNC, so you can use millimeters to set up things and then switch to inches; however even if EazyCNC will not get confused, you may, so it is best to select one system of units and stick with it.

Note that regardless of the units selected here the G-code file can contain coordinates that are expressed in millimeters (G21 mode) or inches (G20 mode), and this is perfectly fine, as long as the correct G20/G21 mode is specified in the G-code file.

### 6.4.1 Units -popup menu

There are two options.

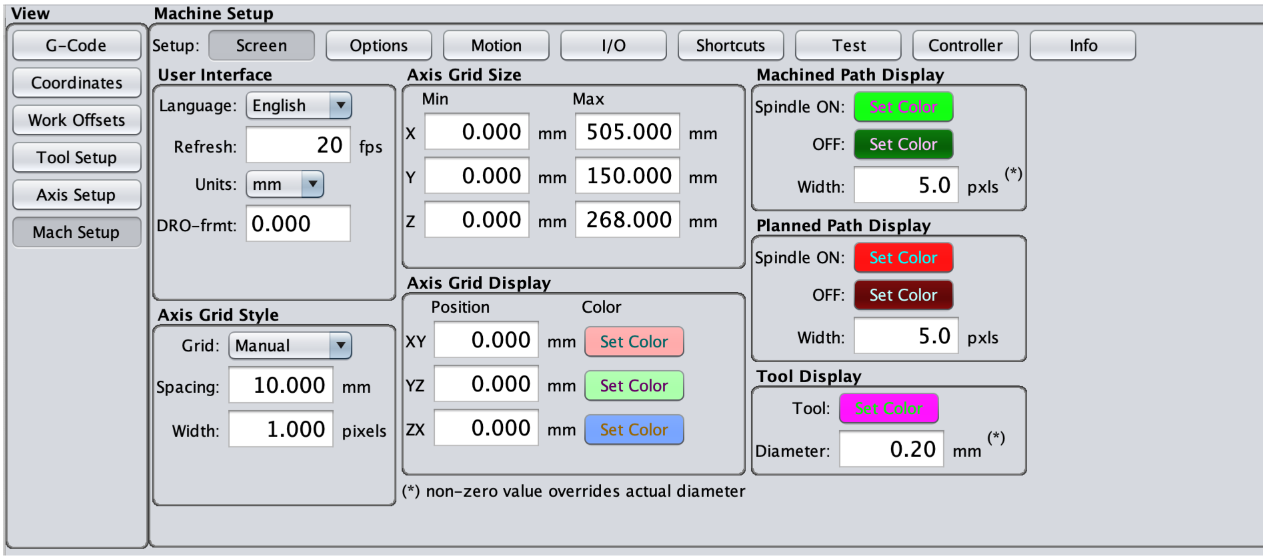



Figure 6.2: The User Interface setup screen

'mm' – with this setting all the entry fields and DROs displays are in millimeters.

'inch' – with this setting all the entry fields and DROs are in inches.

### 6.4.2 DRO-format -entry field

With this entry field you can control how numbers in the entry fields and DROs are displayed.

The main usage is to control the number of decimals you want displayed, to do that just enter '0.' followed by as many '0' characters as you want decimals.

For example : with inches it is probably preferable to use three decimals to see the 'thous' so enter '0.000' in this field, for working with millimeters '0.00' is probably best.

### 6.4.3 Update rate -entry field

This entry field controls how many times per second (frames per second, fps) the toolpath display is updated. Smaller values than 10 may produce jerky updates and higher values than 30 are unnecessary and may bog down the computer.

### 6.4.4 Language -popup menu

This popup allows you to select the language used in the user interface.

### 6.4.5 Screen Size -popup menu

This popup allows you to select between 'Standard' and 'Compact' screen layouts.

Depending on the operating system and the various docks and toolbars in them the 'Compact' layout should fit a screen as small as 1024 x 600 at a pinch.

Note that you need to both Save the configuration and re-start the program. If you are running in a small screen the Save-button may not be visible, in that case use the Alt-Cmd-S for Mac OS X or Alt-Ctrl-S for Windows/Linux to save the the configuration before re-starting EazyCNC.

#### 6.4.6 Machined Path -settings

The settings in this panel control how the tool path for the already cut path is displayed in the Toolpath display panel in the main screen.

A different color can be specified depending on weather the spindle was on or off when the path was cut.

The width of the path in the Toolpath display panel, in *pixels*, can be specified in the 'Width:' -entry field. If this is set to zero then the actual tool width from the tool table for the tool selected by the G-code program is used in displaying the path.

#### 6.4.7 Planned Path -settings

The settings in this panel control how the tool path for the planned i.e. yet to be cut path is displayed in the toolpath display panel in the main screen.

A different color can be specified depending on weather the spindle will be on or off when the path will be cut.

The width of the path in the Toolpath display panel, in *pixels*, can be specified in the 'Width:' -entry field.

#### 6.4.8 Tool Display -settings

The color of the tool in the Toolpath display can be specified as well as the size of it by entering the desired tool width, in *mm or inch*, into the 'Width:' -entry field.

If the tool width is set to zero then the actual tool width from the tool table for the tool selected by the G-code program is used in displaying the path.

#### 6.4.9 Axis Display -settings

The size and colors used to display the axis grids in the Toolpath display can be specified here.

The main purpose of the grids is to allow you check your toolpath against the current working coordinates to ensure that the toolpath and coordinates are set up as you want.

### 6.4.10 Spacing -entry field

This entry field determines the grid line spacing.

### 6.4.11 Width -entry field

This entry field is used to set the line width of the grid lines.

### 6.4.12 Grid -popup menu

The size of the grid can be set manually or automatically. In the automatic mode the grid size is determined by the toolpath and is automatically calculated to just encompass the complete toolpath.

### 6.4.13 Min/Max -entry fields

These entry fields control from which coordinate (min) to which coordinate (max) the grid extends for each axis.

### 6.4.14 Posision -entry fields

These entry fields control at which position each plane crosses third coordinate axis.

### 6.4.15 Color -button

These buttons are used to set color for each grid.

## 6.5 Setting up Inputs and Outputs

For simplicity most of the I/O on the TOAD board has a fixed purpose.

The polarity of some of the inputs can be configured.

To configure the polarity of the home REFIN inputs go to the Axis Setup screen.

### 6.5.1 Probe Input -panel

The polarity of the PROBE input can be configured in here.

Polarity -popup

Select here weather the PROBE signal is zero or one when the probe is activated (i.e. when the probe deflected when it touches the work piece).

Probe -LED

This 'LED' displays in real-time if the probe is activated or not.

To test that your probe is functional and correctly configured, first ensure that you are in the MACH mode. Then just connect the probe to the PROBE INPUT and trip the probe. The PROBE 'LED' on the screen should lit green when the probe is tripped, if it doesn't the just change the polarity. Naturally if the PROBE 'LED' does not change state when you trip the probe then you either have a hardware problem or you are not in the MACH mode.

### 6.5.2 Spindle Speed Calibration -panel

The spindle in a milling machine is driven by Variable Frequency Drive (VFD) that synthesizes three phase AC voltage frequency of which is controlled by a voltage.

The SPEED output of the TOAD board is meant to be connected to that control voltage input in the VFD.

Whenever the spindle has been turned ON with M3 or M4 G-code TOAD will output a voltage from the SPEED output proportional to the S-word value and the voltage that is feeding the SPEED output circuitry.

Because the Digital to Analog Conversion (DAC) of a TOAD board is not linear starting from zero voltage, it needs to be calibrated.

To calibrate the conversion you need to measure the spindle rotation rpm speed at two different DAC setting.

I recommend using 10% and 90% DAC settings.

To obtain the RPM you of course need a tachometer to measure the actual spindle speed and you need to set the DAC to those two values using the Test screen, see section 6.11.3.

So go ahead turn on the spindle and set the DAC to the two values and note down what the tachometer indicates.

Then all you have to do is to enter those rpm and DAC values into the corresponding fields in this panel.

That is it!

If you are curious this is how the math works but you do not need to care.

The output voltage is relative to the voltage fed into the +10V IN input to the TOAD board. The intention is that you get that voltage from the VFD from which typically provides this voltage just for this purpose.

The output voltage is calculated as

$$Uout = (\frac{Sword - LoRPM}{HiRPM - LoRPM} * (HiDAC - LoDAC) + LoDAC)/100 * Uref \qquad (6.1)$$

where

Sword = The S-word value from the G-code program

Uout = 'SPEED' output voltage in TOAD

Uref = '+10V IN' input voltage in TOAD4

LoRPM = Value entered into the 'Low ' entry field

HiRPM = Value entered into the 'High ' entry field

LoDAC = Value entered into the 'Low @DAC' entry field

HiDAC = Value entered into the 'High @DAC' entry field

If above should result in a value outside the range 0..100% of Uref then it is clamped to that range.

### 6.5.3 Spindle Speed Limits -panel

In this panel you should enter the minimum and maximum RPM values want to allow for the spindle.

An S-word in G-code that is outside of that range will cause an error message to be displayed and to protect the spindle EazyCNC will not execute a G-code file which violates these limits.

## 6.6 Configuring Motors and Axes

To configure the motors and axes go to the 'Axis Setup' screen, see Figure 6.3

On the top of the 'Axis Setup' panel you see six or four [TOAD4] buttons. By clicking at those buttons you control which axis parameters are shown on the panel.

There are three group of parameters for each motor and axis plus a panel for testing your motor and mechanics.

Also here you can configure the axis DRO to automatically zero when you start a machining run.

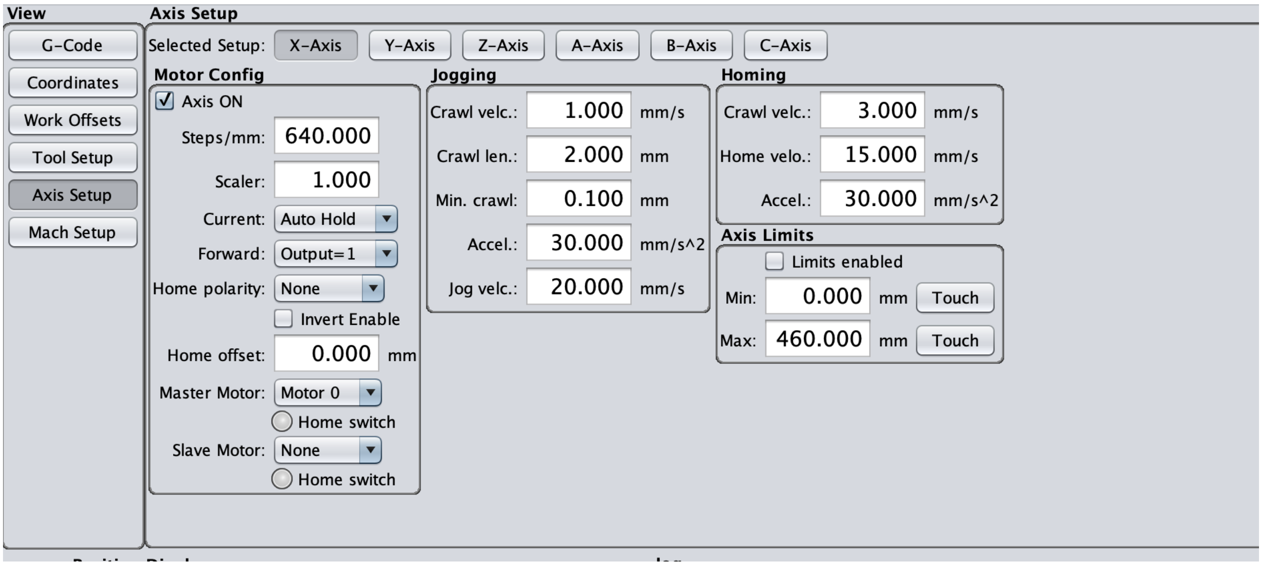



Figure 6.3: Axis Setup screen

### 6.6.1 Motor Config -panel

There are five parameters for each motor.

#### Axis On -checkbox

Axis On -parameter controls weather the G-code commands control that axis/motor or not. If the motor is not controlled by the G-code (Axis On checkbox is not 'ticked') then that motor/axis is available for manual jogging during machining or it can be controlled with EazyCNC plugin extensions.

Typically you want to use G-code to control the motors so make sure the Axis On checkbox is ticked, but if an axis is not used (say you only have a three axis set up) then un-tick the box so that you do not need to set up the motor properly.

#### Current -popup menu

TOAD4 supports two different drive currents for each motor, named High and Low, in addition to which the current can be totally off. The actual motor current depends on the current measurement resistors mounted to the TOAD4 board and the jumper settings on the TOAD4 board, see TOAD4 Hardware Manual for details.

TOAD5 does not support current control, the motor drivers are either enabled via the ENABLE signal implying 100% current, as determined by the driver module used, or not enabled implying zero current.

The Current -popup menu controls how the three different currents are used when driving the motors.

There are four different options.

Low – with this setting the motor current is always set to Low. You might want to use this if the High current is too much for your motor.

High – with this setting the motor current is always set to High. This provides the most 'stiff' setup but means that motors will have full current applied and 'run' hot.

Auto Off – with this setting when the motor/axis is moving or the G-code program is being executed the current will be set to High, but once the movement or machining stops the motor current will be turned off completely within two seconds.

Depending on the mechanics and usage this may not be ideal as the motors may move under external forces if there is no current and thus the axis may lose its accurate position.

Auto Hold – with this setting when the motor/axis is moving or a G-code program is being executed the current will be set to High, but once the movement or machining stops the motor current will be set to Low within two seconds.

This is often the most desirable motor current setting as full current and force is used during machining but the current and heat is reduced when the motors are not being used.

#### Forward -popup menu

The Forward popup menu controls weather the direction output on the TOAD4 board is 1 or 0 when the motor is driven forward.

Forward means that the coordinates of the axis are increasing.

There are two options.

Output = 0 – with this setting the (internal to TOAD4 board) DIR signal is set to logic zero to when the axis/motor is driven forward.

Output = 1 – with this setting the (internal to TOAD4 board) DIR signal is set to one zero to when the axis/motor is driven forward.

You need not to care about zeros or ones, just make sure this setting is right! When you press the axis jog buttons (+X,+Y,+Z or +4) the motor should be running in the direction that you have designated as the increasing coordinate for that axis.

If the motor runs in the wrong direction just change the setting in this popup.

#### Home -popup menu

TOAD4 supports one home/reference position switch input for each axis.

The Home popup menu controls weather the REF input on the TOAD board is 1 or 0 when home switch / reference switch is active.

There are three different options.

None – with this option the REF input is ignored and when you press the HOME button no movement happens, only the DRO for that axis is reset.

Input = 0 – with this setting EazyCNC expects that the REF input is a logical zero (closed) when the reference switch is active.

Input = 1 – with this setting EazyCNC expects that the REF input is a logical one (open) when the reference switch is active.

Again you should not care if the signal is active or non-active, zero or one, just make sure it works for you. If, when you press the 'HOME' button, the axis does not begin to move towards the reference switch the setting of this input is wrong. Note that you should first ensure that DIR signal is correctly configures, see previous section.

When you press the 'HOME' button for an axis EazyCNC will drive that axis until it finds the home/reference position at which point that axis DRO is automatically reset.

The way this works when you press the 'HOME' button is that if the REF input is active the axis is driven to the positive axis direction until the signal becomes non-active. If the signal is non-active to begin with then the axis is driven in the negative direction until the REF signal becomes active and then to the positive direction until it becomes non-active again.

This ensures that even though there is some backlash in the mechanism or hysteresis in the switch the mechanism position will always be correct.

You do not need to use a reference switch but by having one for each axis allows the system to know its absolute physical position which in turn makes it possible for EazyCNC to guard the movements against the physical limits of your system preventing crashes.

Using reference switches it is also possible to continue machining after a sudden loss of power because the absolute axis positions can be re-covered by homing the axes.

*Note that the home or reference switch is no substitute for limit switches that should be installed at each end of the movements and wired to act on the emergency stop system*.

The optimal placement for a reference switch is around the middle of the axis travel, but this requires that the switch is so configured that the REF signal is always on or off depending on which side of the switch the 'axis' is; it should not be possible to drive the axis 'beyond' the switch.

Home polarity -popup menu

TOAD supports one home/reference position switch input for each axis.

The Home popup menu controls weather the REF input on the TOAD board is 1 or 0 when home-/reference is switch is active.

Note that if you are using two motors to control a single axis, ie have the 'Slave Motor' defined for an axis then this polarity setup applies to both.

Home offset -entry field

This entry field controls to which absolute machine coordinates the axis will be re-set when you press the HOME button.

The absolute machine coordinates are more or less irrelevant except for specifying the axis movement limits.

Limits of course only make sense and should only be used if/when an axis is equipped with a Home/Ref switch.

Typically you set the limits so that the Min limit is zero and the Max limit is the total allowable movement for that axis.

For example : say your total X-axis movement is 500 mm. So you set Min=0, Max=500 in the Axis Limits panel. Further suppose your limit switch is set to activate in the exact middle of that range so you set the Home Offset=250.

Steps/unit -entry

This entry fields tells EazyCNC how many steps it takes to move the axis a unit (mm or inch or degrees) amount. We call this value the step ratio.

Note that this is not an integral value and you should enter it with as many significant digits as required to achieve the desired accuracy. As rule of thumb use at least six significant digits in calculations and entry to achieve 0.01 mm accuracy over 1000 mm axis movement range. Note that you can enter more digits that what the entry field will display.

To calculate this value you need to know following:

- $mode$, a factor dependent on step mode
- $steps$, the number of steps per revolution for the motor
- $pitch$, the axis movement per motor revolution (including possible gearing)

Then you calculate the step ratio as follows:

$$step_{ratio} = \frac{mode * steps}{pitch} \tag{6.2}$$

The step mode depends on the jumpers for each motor on the TOAD4 board. With TOAD5 you need to look this up from the documentation of the driver modules you use.

The step mode factor tells how many STEP pulses it takes to make the stepper motor to take one full step.

So for full step mode the mode factor is 1, for half step it is 2 and so on.

Typically you want to run the motors with as small step size as possible for smoother ride, but sometimes speed requirements dictate that a coarser step size is needed because the max STEP output frequency is fixed.

The actual maximum motor rpm (rotation per minute) goes down in direct relation to the step mode factor.

The maximum STEP output frequency is 100 kHz, to keep the jitter of that signal acceptable I recommend to keep the step rate below 10% of that frequency.

Supposing a 200 full steps / revolution motor, this means that the maximum rotation speed of the motor is 100.000 kHz * 10% / 200 steps/rev = 50 rev/ sec = 3000 rpm. With quad steps this is reduced to 750 rpm or 12.5 rev/sec, which is still a high speed for a stepper motor, most likely not achievable in practice.

This may feel a bit complicated so an example maybe useful.

Most stepper motors have 200 steps or step angle of 1.8°so we have:

$$steps = 200(steps/rev)$$

For this example we assume that we want to run X-axis motor at 'Half Step' so we set jumpers on the hardware to that.

A half step this implies that two STEP pulses are needed for a full step, so we have:

$$mode = 2$$

To make this more interesting and life-like let's suppose we use a lead screw to move the X-axis and the screw has a pitch of 3 mm/revolution and that we use a toothed belt to drive it with a 16 tooth pulley on the motor axis and 45 tooth pulley on the lead screw, so we have:

$$pitch = 3 * \frac{16}{45} = 1.066666(mm/rev)$$

Putting it all together we have

$$step_{ratio} = \frac{2 * 200}{1.0666666} = 375.000(steps/mm)$$

Now would be good time to check how fast we can move the X-axis.

As stated the maximum theoretical step rate is about 100 kHz, for jitter and other reasons the maximum recommended pulse rate is about $\frac{1}{10}$ of that, say 10000 pulses/sec. You need to divide this by the $mode$ factor we looked up above so in our example the maximum step rate is 10000 steps/sec and so our max speed is

$$\frac{10000}{375} \approx 26mm/sec$$

Here is a Top Tip!

Even though EazyCNC guides you if you try to enter too big (or small) value to an entry field and even tells you what the limiting parameter is and further tells you what is the maximum value you can use, sometimes this can turn into a bit of a chore.

So before you enter the step ratio, set the maximum jog acceleration and velocity for all axes (Mach Setup / Axis Setup) and movement (Mach Setup / Movement) to a very small values, say 1 mm/sec or 0.01 inch/sec. This will allow you to enter almost any step ratio, which is necessary as the step ratio is a function of your gearing and thus won't budge.

Once you have entered the correct step ratios try to set the accelerations and velocities to a very large values, say 1000 mm/sec or 10 inch/sec and EazyCNC will tell the maximum possible values, so enter and use those.

Scaler -entry field

This entry field is used to specify a scaling factor for each axis which is used when the tool path planner code in the software plans the tool moves.

Normally you set this to 1.0 in which case an F-word in your G-code specifies the feedrate in mm (or inch) per minute for XYZ axes and degrees per minute for the rotating axes ABC.

However, sometimes you may want to change that behaviour.

I'll give you two examples.

Example 1

You have 'knee-mill' or a plasma cutter in which the Z-axis cannot be run at the same speed as X and Y axis. Instead of using the F-word to slow down movement whenever you move the Z-axis you can use thew scaler.

If the Z-axis can only be moved at half the speed of XY axis you set the scaler for Z axis to 1 / 0.5 => 2. This will in effect cause the Z-feed rate to be scaled down by a factor of 2.

Example 2

Suppose you are controlling a mill where the A axis is rotating a cylindrical workpiece of 100 mm in diameter and you are working on the perimeter. To maintain the correct cut rate even when you are using the rotation you need to use the scaler.

In this example case a scale factor of 1 for A axis and F word of 1 would give you a feed rate of 1 degrees per minute which is in effect 100 mm * pi * 1 / 360 = .872 mm/minute which is obviously wrong.

To correct that you would need to set the scale factor for A axis to 100 * pi / 360 = 0.872 .

#### Master Motor -popup menu

EazyCNC allows you to select for each axis which physical motor is used to drive the axis.

You select that with this popup menu.

On the TOAD4 PCB the motors are number like this: X=0, Y=1, Z=2 and A=3. On the TOAD5 PCB the motors are number like this: X=0, Y=1, Z=2, A=3 and B=4.

Typically you don't want to change this to avoid confusion.

Below this pop up there is 'LED' that indicates whether the home/ref switch for selected master motor is active or not.

If you use a home/ref switch in your system you can use this 'LED' to diagnose and ensure that the switch works and is active at the lower (left, near, bottom) end of its movement. To do that move for example the X-axis to the extreme left and ensure that the 'LED' turns ON, then as you move the X-axis to the right, the 'LED' should turn OFF when the switch is no longer active.

If the polarity is wrong, change it with the 'Home polarity' -popup.

#### Slave Motor -popup menu

On some mechanical setups paralleling two motors are handy and allow doubling of the motive power without using larger motors.

EazyCNC allows you to select for each axis a second physical motor that is driven when that axis moves.

You select that with this popup.

Below this pop up there is 'LED' that indicates whether the home/ref switch for selected slave motor is active or not.

When motors are paralleled they perform the exact same movements, except when you press the HOME button. Homing is performed independently for each motor so that the system will align itself correctly based on the home/ref switches.

Note that when motors are paralleled then all the settings for both motors are taken the same motor setup i.e. axis so the gearing, motors and switch setups need to be identical.

### 6.6.2 Axis Limits -panel

EazyCNC can guard movements against set limits to prevent crashing the mechanism.

This is especially useful in Jogging where it is too easy to run too fast to an end of an axis.

However the limit checking is not fool proof and it depends on the operators (that's you!) diligence to work properly. If you for example put a large cutter into the spindle chuck but don't tell EazyCNC about it or move the axis manually by turning the axes handles or forgot to 'home' the axes there is nothing EazyCNC can do about it.

A 'bad' G-code move may still crash the machine and you need to visually satisfy yourself using the simulation mode that this will not happen.

Also worth remembering is that the limits checking does nothing to prevent crashing against the workpiece, fixtures or other obstacles.

The actual movement limits are based a fixed coordinate system independent of all the different G-code coordinate systems. The limits are expressed in the current unit system unscaled and unaffected by any G-code coordinate system transformations.

The origin of the limits coordinate system is at the home/ref switch position, so if the home/ref switch is not used you should not enable the limits because the position is physically undefined.

If you want to use the limits you must remember to 'home' all axes by pressing the 'HOME' buttons if the TOAD4 has been power cycled (it is TOAD4 who maintains the coordinates so it will lose track of the position if it is turned off).

Also worth remembering is that if you use the limits you should have the motors energized at all times otherwise the motors will 'lose' their positions, so don't use the 'Auto Off' current mode.

#### Limits Enabled -checkbox

Limits Enabled -checkbox controls weather EazyCNC enforces the limits for the axes or not.

#### Min/Max - entry fields / Touch -buttons

These entry field controls the minimum and maximum coordinates allowed for the axis, expressed in the current unit system.

The easiest way to set the limit is to disable the limit checking and carefully 'jog' the mechanisms to each end of the movement and press the corresponding 'Touch' button which will then set corresponding limit based on the current axis position.

Remember to 'home' the axis before setting the limits and don't forget to enable the limits once you have set them.

And don't forget to verify your limits!

### 6.6.3 Safe Z -panel

This panel only appears on the Z-Axis setup sub panel.

The 'Safe Z' feature allows you to define a Z coordinate that to which you can move the Z-axis by pressing the Safe Z button in the jog controls.

The idea is that you set this so high that it will always be safe to move in XY at that Z setting without hitting anything.

This can be handy for example for changing tools/cutters.

This feature only works correctly if a HOME reference switch is installed and the Z-axis is HOMEd before you use the feature.

#### Safe Z - entry field

In this entry field you can enter the safe Z value.

This value is in absolute machine coordinates, just like the axis limits. Remember that the absolute machine coordinates are defined by where you HOME-switch is located and the 'Home offset' for that axis.

As a safety measure the initial value of this field is 'NaN' which signals to both you and the software that the Safe Z coordinate has not been set.

### 6.6.4 Jogging -panel

Jogging refers to manually moving the axes with either the 'jog buttons' or with the joystick.

The settings in this panel control the details of jogging.

When you 'jog' an axis, the axis first moves slowly i.e. crawls. This goes on for some short time after which the movement accelerates until it reaches the jog speed. Jogging then continues at that speed as long as you keep jogging or you hit the end of movement, after which the speed decelerates to a halt.

In many CNC mechanisms the axes are not created equal, some motors are by necessity stronger than others and thus the desired jogging characteristics are different and you want to set them individually for each axis.

#### Crawl Velocity -entry field

This entry field controls the initial ie crawl speed of jogging, you typically want to have this pretty slow.

Crawl Length -entry field

This entry field controls how long a distance the crawl will go until the acceleration starts if you keep on jogging.

Min Crawl -entry field

This entry field sets a minimum crawl distance, ie the axis will always move at least this much even if you jog very briefly.

Acceleration -entry field

This entry field control the acceleration/deceleration rate of the movement, you probably want to have this as high as possible but not so high that there is a risk of stalling the motor or the motor skipping steps.

Only experimentation can find a correct value for this, start with a small value and first find the maximum jog speed before you try to maximize the acceleration.

EazyCNC has a motor test function, see section 6.11, to help you to determine the limits of your motors and mechanisms.

Jog Velocity -entry field

This entry field controls the jog or maximum speed the axis will run when you jog it. You probably want to have this set as high as possible but not so high that there is a risk of stalling the motor or the motor skipping steps.

Only experimentation can find a correct value for this, start with a small value and find the maximum at which you can jog back and forth to a given DRO value watching that the tool tip hits exact same position every time.

A manual way to check that the motor has not lost any steps is to 'home' the axis and see that as the axis approaches the home position the DRO will not suddenly jump to the reset value when the reference switch is reached, this is not totally accurate if the motor only loses a small number of steps but typically it is a all or nothing with stepper motors.

## 6.7 Setting the Motion limits

The parameters in the Figure 6.4 screen tell EazyCNC how fast it is safe to accelerate and run the motors, how often the motor position and speed should be updated and how accurately you want EazyCNC to follow the tool path described by the G-code file.

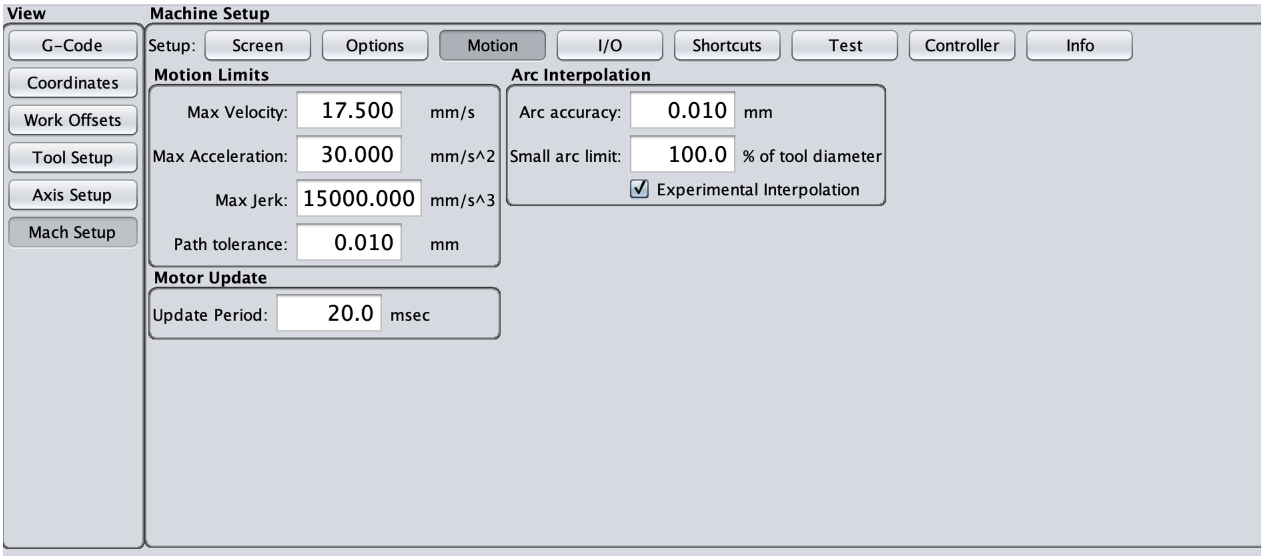



Figure 6.4: The Motion Limits screen.

Next to the individual axis/motor parameters these are the most import parameters to carefully set as if you set the speed and acceleration to too high values the steppers will lose steps and the accuracy is ruined or the motors will completely stall.

On the other hand you will want to have the values as high as reasonable so as not to waste time in machining and with some cutters like plasma torches even the dimensions and quality of cut are dependent on high enough speeds.

Note that these settings here set the maximum values, G-code programs specify the actual value which can be lower or higher that what you specify here. If the G-code specifies a lower value then that applies but if the G-code specifies a higher value then what you have set up here applies.

The only way to find out the maximum acceptable value you can use is to try progressively higher values and verify the accuracy of the motions for each trial.

Note that the values here are common to all axes. See section 6.11) on how to do this for each axis. Once you have found out the maximum velocity and acceleration for each axis, pick the minimum of those values and use that here as the maximum velocity and acceleration for machining.

### 6.7.1 Velocity -entry field

Enter the minimum of the maximum velocities your motors can handle.

Note that it is acceptable to use/have too high feed rate in G-code (the F-word) as EazyCNC will automatically limit the feed rate to the maximum velocity you have set here.

### 6.7.2 Acceleration -entry field

Enter the minimum of the maximum accelerations your motors can handle.

### 6.7.3 Path tolerance -entry field

This field tells EazyCNC how accurately during machining it should try to follow the tool path described by the G-code program.

If you enter a value of zero here then the tool path is accurately reproduced but this necessitates that the tool comes to a complete stop between cutting movements (G-codes G1,G2 and G3).

This is because to have the tool follow the path absolutely without stopping at corners would require infinite acceleration and that strong motors are hard to find.

If you enter a non-zero value then EazyCNC will try to follow the prescribed tool path to within the specified limit but using the leeway given by the tolerance to allow continuous movement and not stopping between cuts.

Rapid tool positioning (G0) always stops at the end of the movement so this parameter does not apply.

If you use EazyCNC with a plasma torch it is important to try to minimize the speed variations as the cut width depends on the travel speed of the torch so use as large path tolerance you can accept.

### 6.7.4 Z-scaler -entry field

Sometimes the Z-axis motor is different from the X and Y axis motors, for example in a plasma cutting machine the X/Y movements need high acceleration and velocities but the Z-movement is rather small and thus a smaller motor that is not capable of such feats can be used. To prevent the system from exceeding the Z-motor capabilities a Z-scaler value (smaller than one) should be entered. This will effectively scale down the accelerations and velocities for G-code movements that involve the Z-axis.

### 6.7.5 Update Period -entry field

Setting a suitable value for the Update Period is also critical.

This value depends on the speed your computer. A faster computer allows for a faster update period, however there are limits on how fast it is acceptable to update the speed and position into TOAD4.

USB limits the update period to a minimum value (maximum update speed) of 1 msec, but typically you should aim to a value of 10 to 20 msec on a modern PC hardware. Note that this has nothing to do with step rate because we are transferring position values to the TOAD4 and the actual steps are generated on the TOAD4 board.

To understand how the update period affects things here is brief description.

TOAD4 maintains a queue of movement commands so that small pauses, interruptions or hiccups in the computer won't affect cutting movements.

If TOAD4 runs out of movements commands ie the queues run empty which happens if the computer does not send new commands fast enough on average then the cutting movements will stop until more commands arrive.

At best this is not desirable as an interrupted cut can leave a mark in the workpiece and at worst the accuracy may be lost if the movement was at such a high feed rate that that the motor cannot be accurately stopped from such speed.

The queue capacity is 16 commands, so an update period of 20 msec means that there are commands for 16 x 20 msec or 320 msec and the system can tolerate a pause, such as Java garbage collection, for that length of time. So longer period allows for longer pauses and hiccups in the host computer system.

On the other hand TOAD4 can only change speed at the interval of the update period so when the speed is changing, like when the machine is cutting a circular path, the speed is always partially 'wrong' for the duration of the update period.

So longer Update Period will result in an increased positional error, which fortunately is not accumulative.

The upper theoretical limit for such an error is 'feed rate * update period', for example if you are cutting at 1200 mm/min which is 20 mm/sec and your update period is 20 msec then the maximum error caused by the update period is 20 mm/sec x 0.02 sec which is 0.4 mm. This would probably be un-acceptable for milling but such high feed rates when milling are rare and for plasma cutting where such high feed rates are common this is in the same ballpark as cutting accuracy anyway. Besides that is a worst case error unlikely to happen in practice.

## 6.8 Configuring Options

While the design goal of EazyCNC is to reduce complexity by limiting options there still are a number options that you may change.

You can change them in the Options screen, see Figure 6.5

### 6.8.1 G-code options -panel

G-code dates back to 1960s with the final revision RS274D approved in 1980. Over the years manufacturers of CNC system have added extension and variations to the standard.

EazyCNC tries to accommodate a common subset of the most popular systems out there by allowing you to fine tune some details of the G-code interpretation to match your G-code program.

For a complete description of G-codes supported by EazyCNC see Chapter 9.

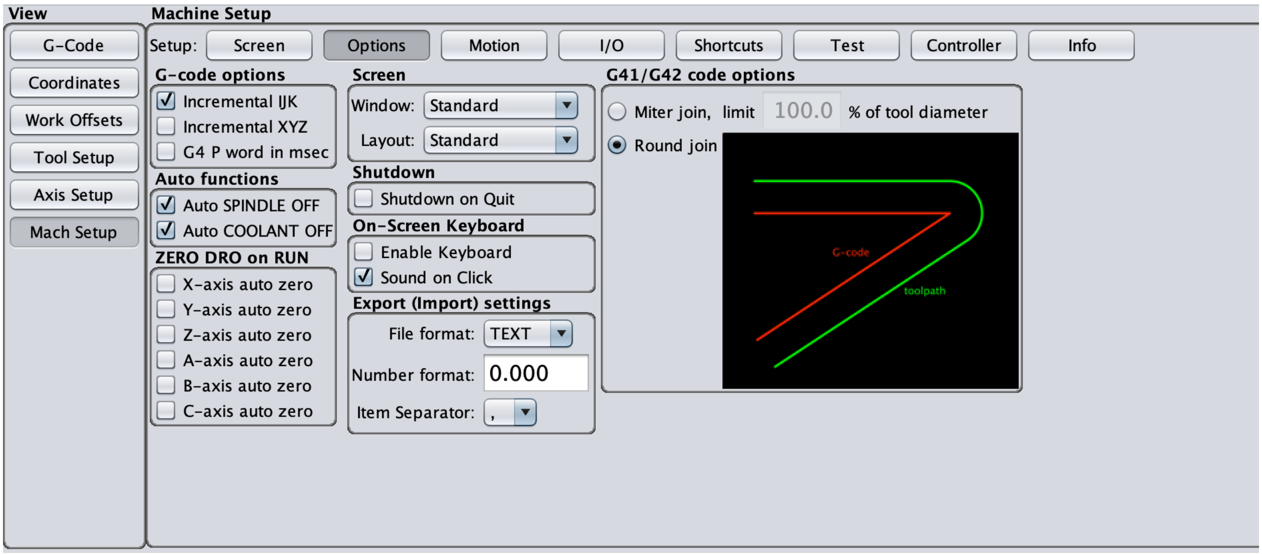



Figure 6.5: The Options screen.

Incremental IJK -checkbox

If this check box is ticked then the I,J and K words in the arc cutting G2 and G3 commands are interpreted relative to the start point of the arc.

If you see a lot of large erroneous arcs in the tool path graphics panel when your are previewing your G-codes then you can be pretty confident that this tick box is in the wrong state.

Incremental XYZ -checkbox

If this check box is ticked then the X,Y and Z words in the movement commands G0,G1,G2 and G3 are interpreted relative to the previous X,Y or Z words/positions.

G4 P in msec -checkbox

If this check box is ticked then the P-word value in the G4 dwell command is interpreted in milliseconds instead of seconds.

If the execution of G-codes seems to stop at G4 commands you can be pretty sure that this check box is not ticked but should be.

### 6.8.2 G41/G42 code options -panel

When tool compensation is turned on with either G41 or G42 G-codes the question arises how the tool should move in external corners.

Traditionally the tool moves around the corner in an arc.

The other option is for the tool to move in a straigth line until it has 'cleared' the corner and the move in a straight line along the next segment. If the (external) corner is very tight then this would cause the tool to move very long away beyond the corner point, therefore a maximum length can be specified.

This later option maybe useful in plasma cutting as it takes the cutting flame further away from the potentially sharp and narrow corner which tends to burn if the torch lingers too long around the corner.

#### Use round join -radio-button

If this is selected then the traditional way of handling external corners is used, i.e. the tool moves in an arc around the corner.

#### Use miter join -radio-button

If this is selected then tool will move in a straigth line until it has 'cleared' the corner and the move in a straight line along the next segment. If, as a result of the corner geometry, the tool would move further from the corner than what is specified in the 'bevel limit' -entry field, then the move is pruned as indicated.

### 6.8.3 Auto functions -panel

LIP

#### Auto SPINDLE OFF -checkbox

If this checkbox is ticked then the spindle will automatically turn off when the system enters HOLD or STOP states. This very useful for plasma cutting, not so much for milling because you will forget to turn the spindle back on when go from HOLD to RUN and break a cutter.

#### Auto COOLANT OFF -checkbox

If this checkbox is ticked the coolant is turned off when the system enters HOLD or STOP states. This can useful for milling when you want to pause the milling for observing work or changing the cutter.

### 6.8.4 ZERO DRO on RUN -panel

#### N-axis auto zero -checkboxes

If 'N-axis auto zero' checkbox is ticked then that axis DRO will be automatically zeroed when you hit the RUN button if the machine was in STOPed state.

The DRO will NOT be zeroed when you hit the RUN button when the system is in HOLD state.

This is handy for plasma cutting where you typically just jog the torch to the origin position, zero the DROs and hit RUN. By ticking this box this becomes automatic and you cannot forget to zero the DROs.

### 6.8.5 Export (Import)settings -panel

#### File format -popup menu

The Tool Setup screen allows you to save and load the tool setup to/from a text file, see section 7.27

With this popup you select the format in which tool setup is stored in the text file. Currently only CSV (Comma Separated Values) format is supported.

#### Number format -entry field

Into this entry field you enter a 'sample' of how the numbers should be formatted. For example : if you want to have three decimals use '0.000' .

#### List item Separator -popup menu

Unfortunately CSV format does not specify whether the decimal separator is '.' (period) or ',' (comma). Worse than that, you probably want to edit the text files in Excel which defaults to different separator depending on which language version of the OS you have.

As mentioned the ',' (comma) can be both decimal separator and list item separator, but it should never be used for both purposes at the same time.

So in this popup you can select which (comma ',' or semicolon ';') character is used as the list item separator, this changes the list decimal separator. If list separator is ',' then decimal separator is ',' and if list separator is ';' then decimal separator is ','.

### 6.8.6 Shutdown -panel

#### Shutdown on Quit -checkbox

A clean shutdown is important, especially on Raspberry Pi running from SD-card, but without a mouse or keyboard this is not possible.

The purpose of this option is to allow operation without a mouse or keyboard by enabling automatic system shutdown.

If this checkbox is ticked then, when press the Quit button EazyCNC will attempt to execute a shutdown script if one exists. If there is no shutdown script then EazyCNC will try to execute the system shutdown command.

A shutdown script is a script named shutdown.BAT on Windows and shutdown.sh on macOS/Linux and which is placed in the EazyCNC directory in your home directory. If one exists, EazyCNC will try to execute it using the operating system shell.

The purpose of the shutdown script is to allow you to customize the shutdown process e.g. by shutting down additional hardware in you machining system.

### 6.8.7 On-Screen Keyboard -panel

#### Enable Keyboard -checkbox

If this checkbox is ticked then an on-screen virtual keyboard will appear whenever you click an entry field. The on-screen virtual keyboard allows operation with touchscreen only without a real physical keyboard. This built-in virtual keyboards is preferable to the operating system provided because it is optimised For CNC operations and it plays nicely with the rest of the user interface.

#### Sound on Click -checkbox

If this checkbox is ticked then clicking any of the buttons in the user interface will emit a click sound.

In addition, on Raspberry Pi a short pulse is generated on the GPIO21 at the same time the click is emitted. This is so that you can connect a tactile feedback device to the GPIO 21 output to give a physical feedback on the touchscreen. A simple tactile feedback device can be fashioned from a solenoid.

## 6.9 Shortcuts setup

EazyCNC is really designed to be used on a PC or tablet computer with a touch screen. However you can use most of the functions with convenient single key keyboard shortcuts on a conventional keyboard or with a gamepad function keys and joystick.

The key assignments are fully user definable so you can configure these as you best please.

To examine or change the key assignments go to the Shortcuts setup screen, see Figure 6.6

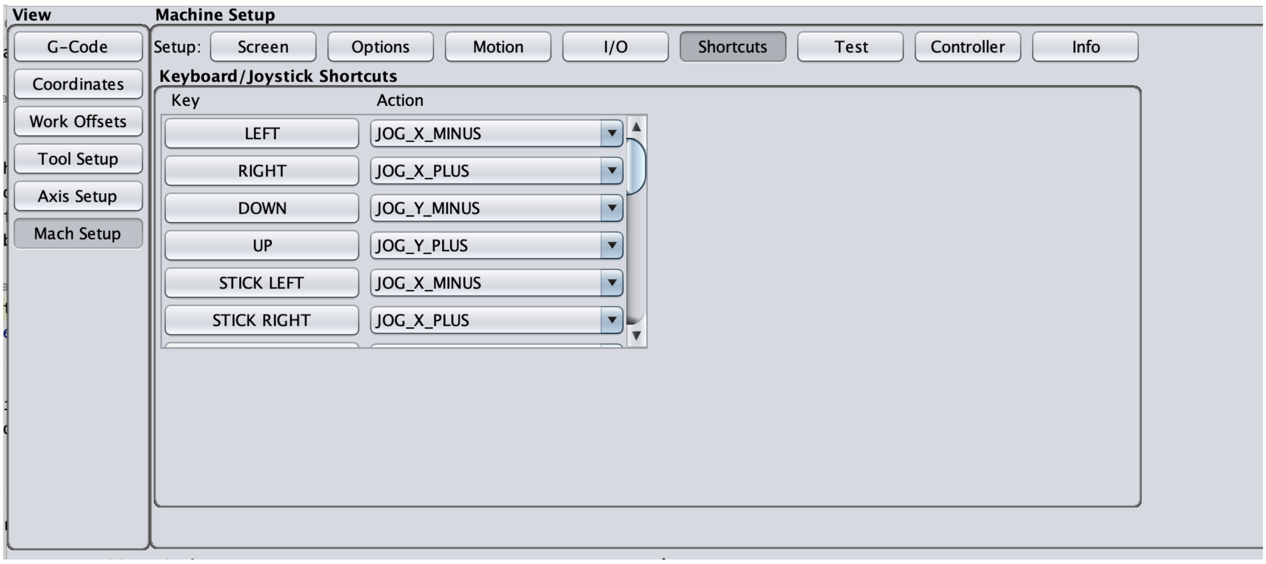



Figure 6.6: The Shortcuts setup screen.

On the left side column you see the keys and on the right side column the corresponding assigned functionality.

To change the key, click on the left side column at the key you want to change; this will popup a dialog prompting you to press the new key you want to assign for that functionality.

To change the functionality click on the right side column of the key assignment you want to change and from the popup pick the functionality you want to assign.

To create a totally new keyboard shortcut scroll to the bottom of the list and click at the 'New Shortcut' button at the bottom of the left column.

To delete a shortcut click on the left column on the shortcut you want to delete and from the dialog that pops up select 'Delete', see Figure 6.7


Figure 6.7: The Redefine Shortcut screen.

## 6.10 Info screen

This screen displays bunch of system related information, see Figure 6.8

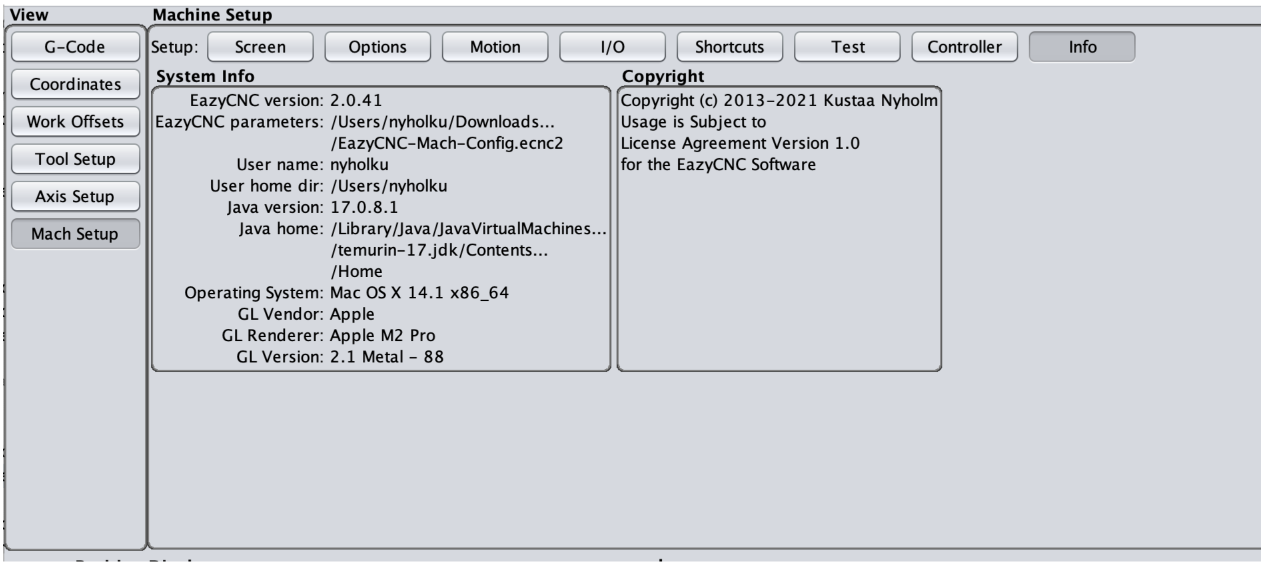



Figure 6.8: The System Info screen

Most of the information displayed is just to report back to eazycnc@eazycnc.com if you are reporting a bug.

The two pieces of information that are useful for you are the 'EazyCNC version' to identify which version you are running (it is also shown in the window title if that is visible) and 'EazyCNC parameter' which tells you the location of the configuration file in case you have doubts about which configuration file is being used.

## 6.11 Test screen

The Test screen allows you to test all the inputs and output of your TOAD boars as well as test or find out the limits of your motors, Figure 6.9

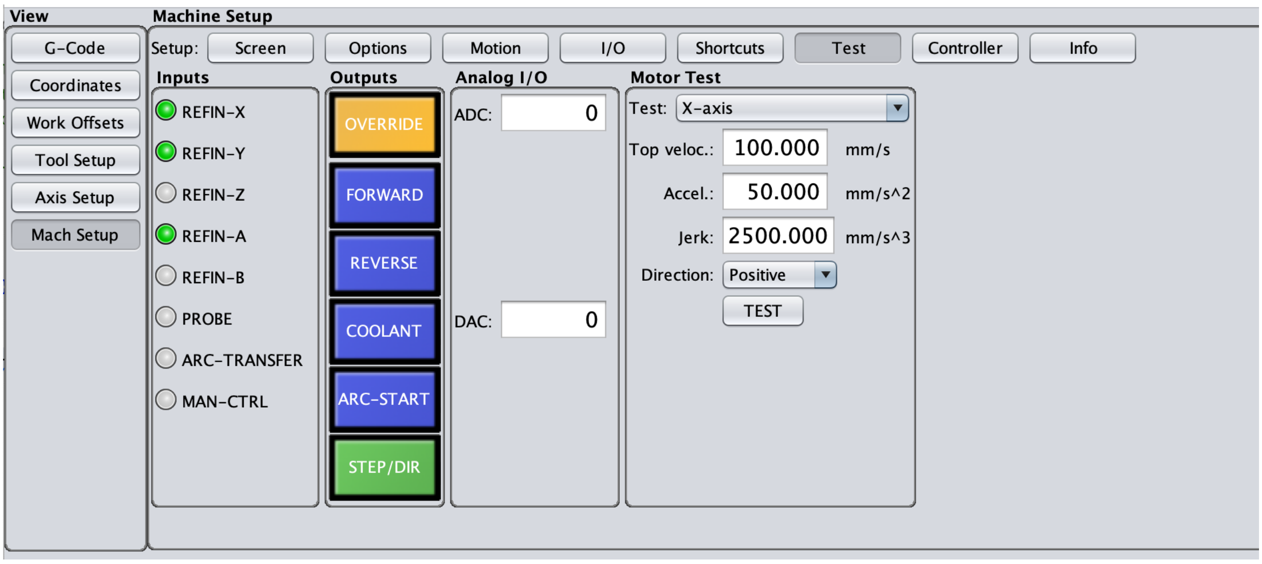



Figure 6.9: The Test screen.

To use this panel the system needs to be in MACH state so that it is communicating with the TOAD board (this actually works in the SIMU mode as well, but then everything is, well, simulated).

### 6.11.1 Inputs -panel

This panel shows in real-time the state of all the digital inputs from the TOAD board.

The different inputs are self-explanatory except perhaps the MAN-CTRL which indicates that the manual control panel is attached AND in manual control mode.

### 6.11.2 Outputs -panel

This panel allows you to manually control all the digital outputs of the TOAD board.

To exercise the outputs you need to first activate the orange OVERRIDE button.

> Warning!
>
> Once the OVERRIDE is enabled all safety features associated with any of the outputs are disabled and the outputs are solely controlled with the buttons in this panel (and the DAC field in the analog I/O panel).
>
> This means that it is perfectly possible and very easy to turn the spindle or plasma torch on or make all the axes run wild if you do not know what you are doing.

For some testing/debugging purposes it is of course necessary to do exactly that, but for some it is best to disconnect or power down the machine being controlled.

The blue buttons (FORWARD, REVERSE, COOLANT, ARC-START) directly toggle the named outputs. When the button is activated the corresponding output is turned on.

Activating the green STEP/DIR button causes the ENABLE output for all axes to turn on and the STEP and DIR outputs emit pulses that effectively drive the connected axis motor back and forth as long as the button is activated.

Proceed with caution.

### 6.11.3 Analog I/O -panel

ADC -field

The ADC (short for Analog to Digital Conversion) field shows the digitized value of the analog ARC VOLTAGE input of TOAD5 in percentages.

0% indicates 0 Volts and 100% indicates 5 Volts at the analog input.

#### DAC -field

The DAC (short for Digital to Analog Conversion) field can be used to control the SPEED output of TOAD5, when the orange OVERRIDE button in OUTPUTS panel is activated.

The DAC conversion is ratio metric which means that the output voltage is always relative to the voltage used to feed the circuitry (see TOAD5 manual).

Suppose the feed voltage is 10 Volts then setting the DAC field to 100 (%) causes the SPEED output to rise to aproximately 10 Volts.

The DAC output is not linear, any non zero value in the DAC field will cause the SPEED output to be at least 10% level of feed voltage. Above that the response is more or less linear.

A zero value in the DAC field should always produce zero voltage at the output.

### 6.11.4 Motor Test -panel

The controls in this panel can be used to test and find out the maximum acceleration and velocity for an axis.

When you click the 'TEST' button EazyCNC will perform a test movement on that motor/axis and report the accuracy if you have a reference switch installed.

If you don't have a reference switch then you will need to use a Dial Test Indicator or some such to measure the accuracy.

#### Pre-requisites

*Note that this test can damage your machine if not performed carefully and as intended.*

The test run requires free travel of 25 mm or one inch plus the distance is takes to accelerate from the start velocity to the top velocity and back.

To use this test you must have the reference switch located at least 30 mm away from the negative end of the axis movement.

If you don't have that 30 mm spare travel you run the risk of hitting the end of the axis movement range on the return leg of the test run. It is acceptable to temporarily move the reference switch to a suitable location for this test or use a temporary switch if you so desire.

If you don't have a reference switch then you need to ensure that you start the test from a position on the axis from which there is enough free travel on both sides of the starting position.

#### The Test run

When you click the 'TEST button the test movement is performed as follows.

First EazyCNC 'homes' the axis/motor ie it moves slowly towards the reference switch and then just out of it.

EazyCNC makes an internal note of this step position.

Next the axis/motor starts to move into the positive (or negative, depending on which direction you have selected) axis direction and accelerates at the given acceleration until the motor reaches the given test speed. The movement continues for 25 mm (about 1 inch) and then it decelerates back to stand still.

Then the axis is homed again and a note of the step position at which the reference switch is detected is noted down.

The difference between the two home positions noted down is reported as the accuracy or repeatability of the movement at the given acceleration and movement velocity.

A positive value indicates that steps were lost on the way out i.e. during acceleration or high speed movement. A negative value indicates that steps were lost on the way back i.e. slow movement; this should not really ever occur.

A small non zero value is acceptable or even expected as it is unlikely that the system can be absolutely accurate all things considered, but repeated tests at given acceleration and velocity should show consistently similar values.

To guarantee that the test is valid for machining conditions the acceleration is performed step wise at the current machine Update Period just as it will when take place when executing G-codes.

Therefore you need to remember that if you change the update period it is good to re-run the test for each axis, especially if you are running close to the speed and acceleration limits of your system.

#### Top Velocity -entry field

This is the velocity you want to test for so keep adjusting this and retesting until you have maxed out your system.

#### Acceleration -entry field

This is the acceleration you want to test for so keep adjusting this and retesting until you have maxed out your system.

Direction -popup menu

This selects weather the test movement direction will be along the positive direction or negative direction from home/ref position.

This is the acceleration you want to test for so keep adjusting this and retesting until you have maxed out your system.

# Chapter 7

# Operating Your CNC Machine

This chapter tells you all how to operate EazyCNC to machine stuff with G-code.

G-code is covered in more detail is Chapter 9, but since the sole purpose of EazyCNC is G-code interpretation most functionality is related to G-code in some way I cover that in this chapter.

## 7.1 Using Keyboard and Joystick

Most of the EazyCNC functions needed to operate your CNC machine can be activated from the keyboard with single key from the computer's keyboard.

A Joystick or Gamepad can also be used to active most of the functions.

To extend the number of functions accessible using the Joystick/Gamepad buttons EazyCNC supports a software 'Shift Lock' -button, Figure 7.1. If this button is activate then the Joystick/Keyboard generates another set of key presses onto which functions can be attached.

See Section 6.9 to learn how to view and change both keyboard and joystick shortcuts.

The default keyboard / shortcuts assignments are given in Table 7.1.

Figure 7.1: The Shift Lock -button

Table 7.1: Keyboard and Joystick Shortcuts

| Key | Function |
|---|---|
| R | RUN |
| H | HOLD |
| S | STOP |
| Space | Toggle RUN/HOLD |
| C | Toggle Coolant ON/OFF |
| T | Toggle Spindle ON/OFF |
| Cursor LEFT | Jog -X |
| Cursor RIGHT | Jog +X |
| Cursor DOWN | Jog -Y |
| Cursor UP | Jog +Y |
| Stick LEFT | Jog -X |
| Stick RIGHT | Jog +X |
| Stick DOWN | Jog -Y |
| Stick UP | Jog +Y |
| Button 1 | RUN |
| Button 2 | HOLD |
| Button 3 | STOP |
| Alt-Cmd S | Save Configuration (Mac OS X) |
| Alt-Ctrl S | Save Configuration (Windows/Linux) |
| F12 | Toggle Java console on/off |

## 7.2 Using Manual Pulse Generator (MPG) / CNC Pendant

Manual Pulse Generator or Pendant is a remote control for a CNC machine.

Very often the computer is placed further away from the milling machine or plasma cutter to protect it from the chips, sparks, dust and other grime.

Also the keyboard is not the most convenient means to jog the milling table around or manually position the plasma torch.

Hence the by now ancient invention of the MPG, so named because in the primordial times people were talking about pulse generation for the servo motors that controlled the axis movements.

EazyCNC supports a small selection of commercially available MPG pendants.

Manufactures of these MPG pendants do not release information about how to interface their products to any old CNC software. This seems counter productive to their commercial success but it is what it is. So to interface them one is reduced to searching the interweb for info and acquiring and reverse engineering these pendants.

It is important to realise that this means that EazyCNC author has no control over how the pendant works and cannot guarantee the functionality or safety to any degree. Also this makes implementing some trivial functionality impossible because the manufacturers firmware does not allow it, a case in point is adding custom messages to pendant displays or making the display more sensical.

It is perfectly possible that if you buy one of the supported pendants that it does not work the same as described here because the manufacturer has changed the firmware or something. This may not be likely but it can happen.

Caveat emptor.

Please note that these commercial pendants have labels in the keys for the functions that the manufactures have intended. These labels / functions do not map very well to EazyCNC functionality in every case.

Importantly if you see a label on the keypad or rotary knob that does NOT imply that the function is available with EazyCNC using that pendant or if it is that it functions as it would with the CNC software that the manufacturer supports.

Instead you should take the view that the keypad is an empty keyboard on to which you can assign any functions you like.

Out of the box EazyCNC configures the keys on the keypad and functions to the 'best' available functionality in EazyCNC.

However you are not limited to these, all the keypad keys can be configured in the Mach Setup / Shortcuts screen, see section 6.9,

Note that for each key it is possible to associate two functions, one function for a short press of the key and another one for long press of the key. To do that simply press the key on pendant for shortly or for a 'long' time when assigning the shortcut.

## 7.3 Simulation versus Cutting Metal

EazyCNC can be in one of two operating modes, simulation mode or machining mode.

The simulation mode is handy for training or experimentation and for G-code verification before you actually machine anything. Because no TOAD4 motor controller is used or needed in the simulation mode you can install and run the EazyCNC software in any computer and use it in the comfort of your study or office.

The simulation mode attempts to be step-accurate i.e. simulate the exact movements the stepper motors will make and it runs in real-time (if your computer is fast enough) meaning that simulation takes as long as the actual machining will take. As this may take a long time you may speed things up when simulating by using a faster feedrate (the F-word G-code) but this will affect the small details of the cutting paths so the path generated at high speed is not 100% same as the lower speed path.

You control the operating mode with SIMU and MACH buttons at the bottom of the screen, Figure 7.2, the current operating mode is show with the button lit.

To toggle a mode on or off click the corresponding button.

*Note that the stepper motor drivers are enabled/disabled when you press the MACH button. When the motor drivers are disabled they may not hold their exact position and thus accuracy may be lost if you exit MACH mode even temporarily before you have finished machining your part.*


Figure 7.2: The operating mode control and indicator buttons

## 7.4 Status Display

At top of the panel titled 'Machine Controls' at the bottom of the screen, Figure 7.3, there is a status/error display.


Figure 7.3: The status/error display

This display shows the general status of the system or a helpful error message in case of an error.

Most errors are transient in nature because they are the results of an operator (that's you) mistake, such as pressing the right button at the wrong time or entering an invalid value into an entry field, thus the error messages as just shown briefly accompanied with a beep sound.

If you are not able to read the error/message quickly enough you can press and hold the '?' button down to re-call the last error message and read it at leisure.

During machining it is possible that the G-code contains messages to the operator, via the '(MSG, operator message)' -mechanism, and those messages are also displayed in this display.

## 7.5 Interactive Execution of G-code

The status display line doubles as an interactive G-code execution entry field. To execute G-code interactively click on the status line, type in G-code and hit ENTER-key.

This feature should be used with extreme caution as it very easy to enter G-code commands that damage the machine or the workpiece.

Even more care and forethought needs to be exercised if the interactive execution is used in the middle of machining when the system is in the HOLD-state, because the the interactive commands may violate the expectations built into the G-code.

For example : it is possible to turn off the spindle interactively with the 'M5' code in which case continuing with the machining is likely to result in a broken milling cutter when the machining is continued and the spindle is not running.

## 7.6 G-code display

On the main screen the top right-hand quarter of the screen is occupied by the G-code display and the associated controls, Figure 7.4.

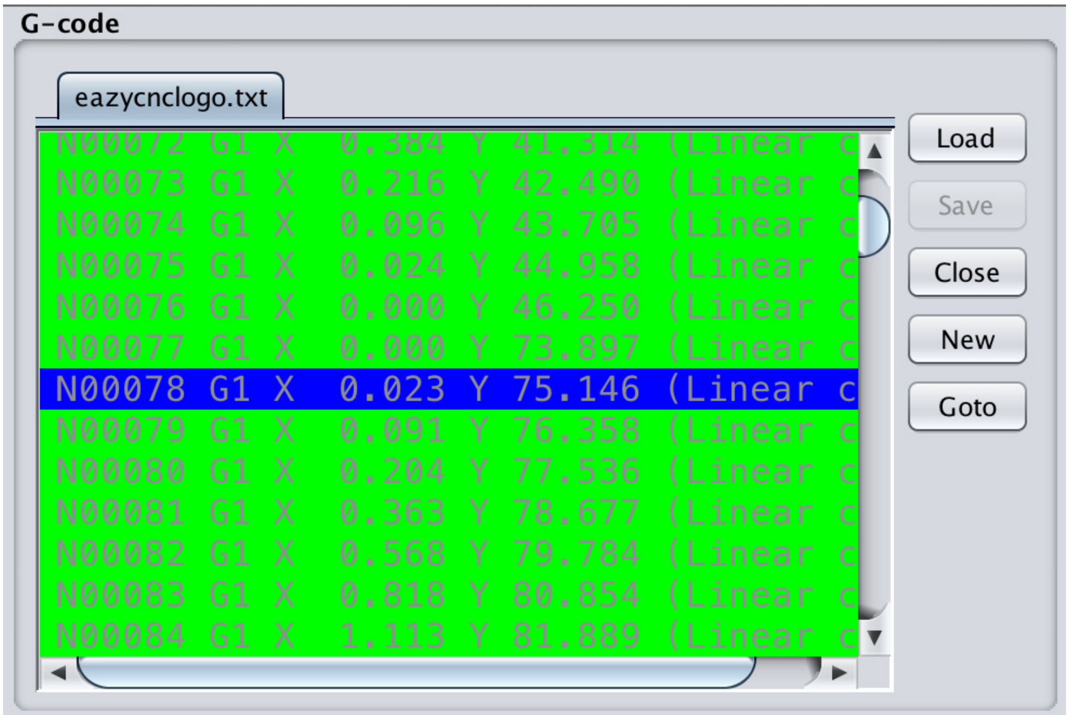



Figure 7.4: The G-code editor and display panel

The current G-code file is displayed there. Only one G-code program can be open at any given time, but if it calls subroutines in another G-code file it is possible that multiple 'tabs' or files are displayed there.

The next G-code line to be executed is highlighted in blue. If there is an error in the G-code then the line containing the error is highlighted in red and the associated error message is displayed in the status/error display.

The background of the G-code display turns green if the system is in the RUN or HOLD state, indicating that the system is either machining or ready to continue machining.

### 7.6.1 Loading G-code for execution

In order to execute a G-code program you need to load it into the EazyCNC first, to do that, click 'Open' button and select the G-code file to load. G-code files are pure text (7-bit ASCII) and so often have a '.txt' extension but that is not mandatory, some people and systems use '.nc' .

### 7.6.2 The Goto -button

G-code programs are meant to be executed from the beginning to the end. However it is possible to start from somewhere else, for example if you need to continue execution after a blackout, even if that is not recommended.

To start the machining from somewhere else than the current line highlighted in blue, click on any line and then click the 'Goto' button, this will move the highlight to that line and machine will enter the 'HOLD' -state.

The caveat is that because you are skipping G-codes when you 'Goto' a specific line in the program you maybe breaking assumptions the G-code generator, be it a person or a CAM software, made when the code was written and this may have consequences.

For example : a plasma cutter is typically programmed to start the cut with a piercing burn outside the actual part outline and if you skip that the arc may not be established and the metal not be pierced or the piercing may leave its mark on the part.

### 7.6.3 Editing G-code

It is also possible to edit and even create G-code files directly in EazyCNC, but this is not recommended.

*EazyCNC is not a general purpose text or G-code editor, any time you edit the code EazyCNC will go through the whole code and re-calculate the tool path and this can be slow, especially if the G-code file is very long, as it very well can be.*

To edit the code ensure that EazyCNC is in the stopped state (you cannot edit the code while machining) and just click on line you want to edit, most standard basic text editing facilities work as you would expect.

To create a new G-code file from scratch click the 'New' button.

An asterisk, '*', after a file name indicates that the file contains unsaved changes, to save the changes to the actual file click the 'Save' button.

## 7.7 Toolpath display

On the main screen the top lefthand quarter of the screen is occupied by the toolpath display and the associated controls, Figure 7.5.

In this panel the toolpath that the currently loaded G-code represents is illustrated in a three dimensional view.

The movement limits of the axes are displayed as white cube and a visual reference to assess the toolpath is display as colored grids, see Figure 7.6.

Note that the white cube is only displayed if you have enabled the axis limits, see section section 6.6.2.

If the white cube blinks, then the EazyCNC has not been HOMEed by pressing the HOME buttons for each axis and thus the EazyCNC does not know where in the real physical world the axes are which means that the limits are not reliable at the moment. Press HOME on each axis to remedy the situation.

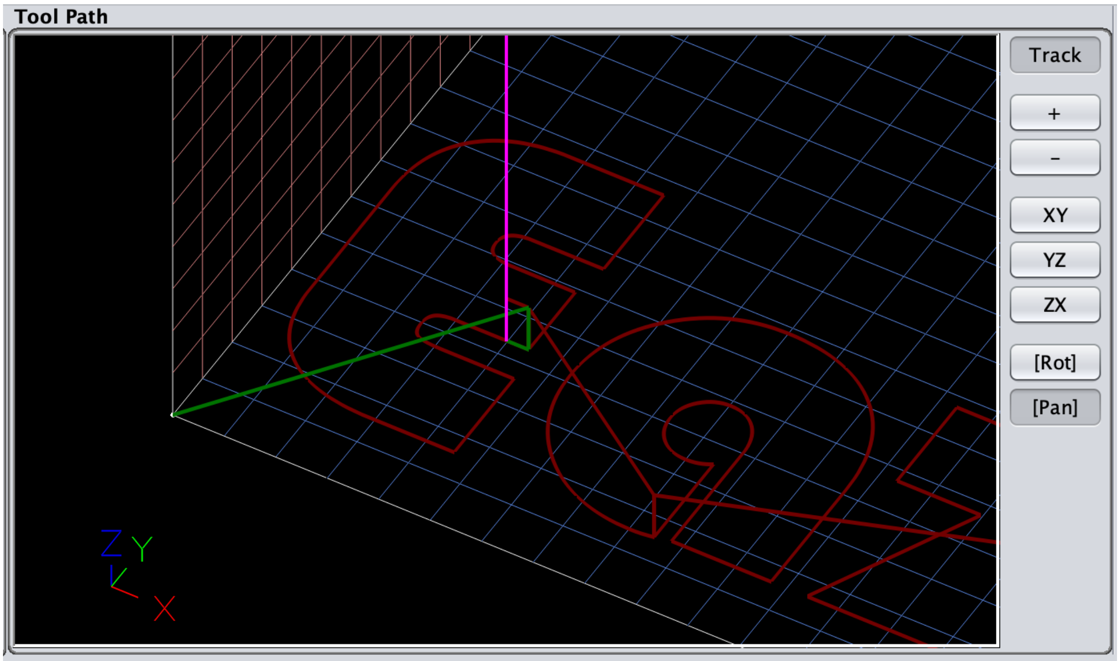



Figure 7.5: The toolpath display panel

The grids are always displayed in the coordinates defined by the current work offsets. This is convenient because it allows you to visually reference the toolpath against the grids which is the purpose. You can change the grid colors, see section 6.4.9.

However if you change the current work offset (see section 7.19)or you have multiple work offsets in use in your G-code you have to be careful before you jump into conclusion that something is wrong with your G-code or the grids.

The cutting tool is illustrated as a purple cylinder and the actual tool path is displayed as series of connected red and green lines of different intensity and line width. The colors can be changed, see section 6.4.8.

The display is updated in real-time as the machining progresses. The toolpath machined or cut so far is displayed in green and the yet to be cut toolpath is displayed in red. You can change the colors, see section 6.4.6.

For the already machined toolpath, in green, the width of the line corresponds to the width of the tool used to cut that path if the spindle was turned 'on' during the machining, otherwise the toolpath is shown as a thin line. The colors can be changed, see section 6.4.6.

Weather the spindle is 'on' or 'off' for a given toolpath segment is indicated by the intensity of the color of the line, spindle 'on' is illustrated with bright red or green and spindle 'off' is illustrated with a darker shade.

When the G-code program is executing or running you can see, if you look carefully, that the green toolpath appears to precede the purple tool, this actually accurately represents the fact that the movement commands are queued in the motor controller waiting to be executed.

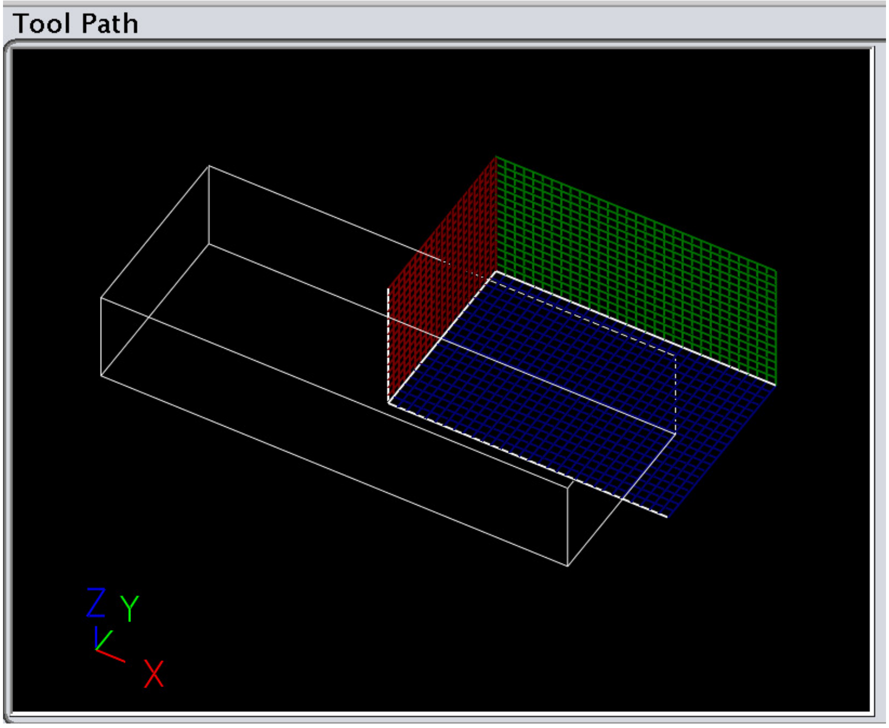



Figure 7.6: The toolpath panel with grids and limits but no toolpath

### 7.7.1 Controlling the toolpath display

If the 'Track' button is not active you can move around the three dimensional virtual space and inspect different parts of the toolpath.

If the [rot]-button is activated dragging in the tool path panel rotates the view, if the [Pan]-button is activated then dragging moves the view.

If the 'Track' button is active the display is always automatically cantered at the tooltip and follows it as the tool moves. You can toggle the tracking on and off by clicking the button repeatedly.

If you hold down the 'ALT' key on your keyboard while you drag with the mouse you can rotate the view without activating the [Rot]-button.

If one of the 'XY', 'YZ' or 'XZ' buttons is active then the rotation is disabled and the view is forced to be towards the corresponding coordinate plane along the 'plane normal' axis. For example : if you have the XY button selected then the view is as if you were looking down from the top of the positive Z-axis towards the XY-plane.

Note that you can de-select any of the 'Track', 'XY', 'YZ' or 'XZ' by just re-clicking on it.

You can use the '+' and '-' buttons to zoom in and out to see more or less of the toolpath or you can use the mouse scroll wheel. On Operating systems that support touch screen 'pinching' that should also work for zooming.

## 7.8 Coordinate displays aka DROs

The coordinate displays, Figure 7.7, also known as Digital Readouts or DROs, show the tool position in the currently active coordinate system in the currently selected units.

The coordinate system can be quite complex but for most practical purpose you can just think that the DROs just show the position EazyCNC thinks the tool tip is at the moment. If the DROs says that the tool X-axis is at position 100 and the G0 -code tells it to move to X150 then the tool will travel 50 units to the right, no matter where it physically actually was.

By clicking at the DRO and typing in a number you can change the displayed value. This will not move the tool but will of course change how subsequent G-code coordinates are interpreted.

You can also zero the DRO by clicking at the 'ZERO' button.

If you press the 'HOME' button and you have a home/reference switch installed for that axis EazyCNC will drive the axis to the home switch position and then zero the DRO automatically. This establishes a repeatable absolute origin for that axis.

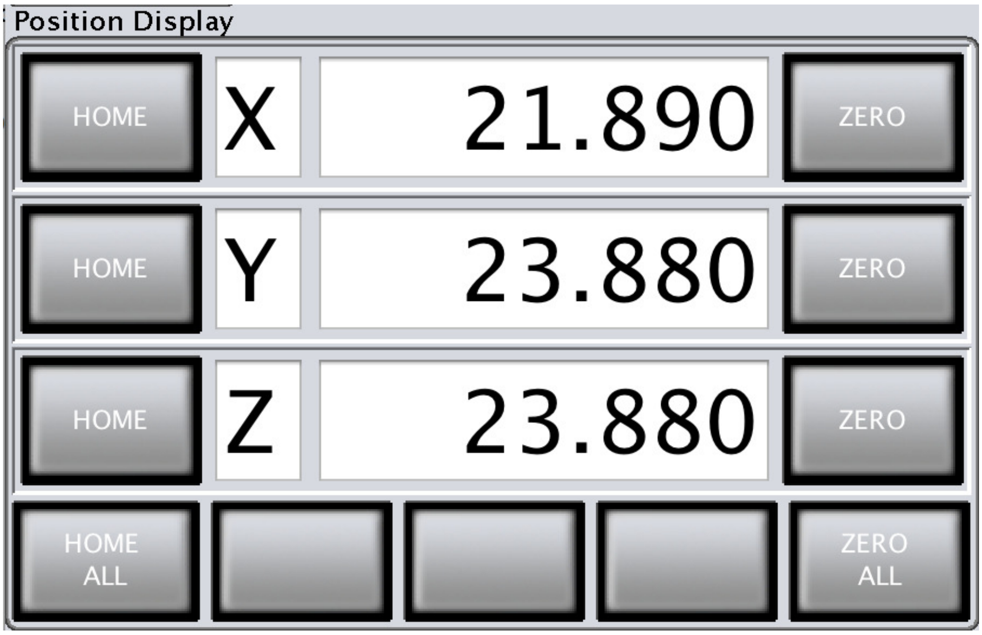



Figure 7.7: The Digital Readouts

## 7.9 Jogging

The Jog buttons, Figure 7.8 can be used to drive or move the motors/axes 'manually'. If you need to move the axes you should use the jog buttons and not any handles your machine possibly has for moving the axes, as if you use the manual handles EazyCNC has no way of knowing that the tool has moved and thus coordinate positions used by the system will be wrong.

The buttons to the right of the jog arrow buttons set the jogging mode i.e. they control how the jogging is performed.

Never turn off the power to TOAD4 to move the axes manually, always use one of the jog modes provided.

See 6.6.4 for details how to setup the speeds used in jogging.

### 7.9.1 MODE++ -button

The current jogging mode is display right above this button.

By clicking this button you can change the mode.

There are six jog modes plus a manual mode which is engaged with the 'MANUAL' button.

### 7.9.2 FAST and SLOW jog modes

Two of the modes, FAST and SLOW, are used to move the axes as long as you press corresponding jog arrow button.

If the FAST mode is selected then pressing and holding a jog button will first move that axis slowly and then rapidly accelerate to the full jog speed. Pressing a jog button briefly will always move the axis a predetermined, minimal distance.

The 'slow' speed is specified in the 'Crawl veloc.' entry field for that axis in the Axis Setup -screen. The 'fast' speed is specified in the 'Jog veloc.' entry field and the acceleration is controlled by the 'Accel.' entry field. See Section6.6.4 and Section 6.6.4.

If the SLOW mode is selected then pressing a jog button will move that axis slowly a as long as the button is pressed.

The 'slow' speed is specified in the 'Crawl veloc.' entry field for that axis in the Axis Setup -screen. See Section 6.6.4.

A short press of the SLOW-button will move the axis the amount specified in the 'Min. crawl' entry field for that axis in the Axis Setup -screen. See Section 6.6.4.

### 7.9.3 Step jog modes x0,x1,x10,x100 and x1000

There are four step modes, displayed as 'x1000', 'x100', 'x10', 'x1' and 'x0'.

In these modes the axis will advance a single step for each press of the jog arrow button.

The size of the step depends on the mode which is named as the number of 1/1000th of the basic unit that you have set your system up with. For example : if you have set up your system to use 'inch' then step size in step mode 'x100' moves the axis 100/1000th of an inch or 0.1 inch. Like wise if you use 'mm' then step mode 'x10' then causes the step size to be 10/1000th of millimeter or 0.01 mm.

Step mode 'x0' is special and causes the step size to be the smallest possible given your stepper motor setup i.e. the size is one step in this mode.

If you read carefully you noticed that I said there are four step modes but I listed five. That is no mistake. In 'inch' mode the step mode 'x1000' is skipped and in 'mm' mode the 'x1' mode is skipped as they do not make sense the other being too large and the other being too small.

### 7.9.4 MANUAL -button

If this button is selected then the motor drivers in TOAD4 are disable i.e. the motor currents are turned off and the axes can be moved manually with the hand wheels of the machine.

This mode of jogging is intended to be used before or after machining to allow the operator to adjust the axes. This is necessary for example when using a feeler to touch a work piece before setting the coordinates.

When the machine axes are moved with the hand wheels EazyCNC has no way of knowing how much the axis has moved and thus the coordinates will be off/erroneous unless you reset (ZERO) them after moving the axes manually.

Note that when the motor drivers are turned off the axes are free to move and thus even small forces, perhaps even internal tensions of the machine or tool/workpiece contact may move the axes a small amount and thus ruin the accuracy of the coordinates.

So think twice before pressing this button.

Important, never ever turn off the TOAD4 to 'free' the axes if you need to move them manually, always use the MANUAL jog mode if you want to use the hand wheels to move the axes.

Worth noting is the if you exit the MACH mode then the motor drivers are also disabled freeing the axis. Note that exiting EazyCNC does not have this effect.

### 7.9.5 SPECIAL -button

If this button is selected then the jog buttons are hidden and a set of user defined 'function keys' are displayed. See Section /refsec:user-functions.

To re-display the jog buttons click any of the jog mode buttons above.

### 7.9.6 SAFE Z -button

Pressing this button will cause the Z axis to automatically move to the absolute coordinate defined in the Z Axis Setup Safe Z entry field.


Figure 7.8: The Jog control buttons

## 7.10 Finding your bearings i.e. coordinates

Before you start machining you need to establish the correct relationship between the virtual part described by the coordinates in your G-code file and the work piece you are about to cut.

It is important to remember that EazyCNC does not really know where the tool is, it just keeps track of the position relaying on an initial known position. Further even though EazyCNC can keep track of the tool position it knows next to nothing about the workpiece location.

There are different ways to calibrate all three (the EazyCNC coordinate system, the actual physical axis positions and the workpiece location) in relation to each other, but most of the time this would be an overkill.

The relationship between the physical axis position and the coordinate system only really matter for two things: if you enable the axis limits and if you need to continue machining after a power down.

In the first instance, if EazyCNC does not have the correct relationship between the physical axis position and its internal coordinate system then it will allow you to drive the axis bang against one end of the movement and will unnecessarily limit your movements at the other end.

In the second instance if you lose power, a fuse blows or the job takes so long that you can't do it all at one go, unless the system can be returned to the same physical axis coordinate system relationship you will not be able to continue machining accurately after the system has been powered off.

The relationship between the EazyCNC coordinate system and the workpiece really only matters if you need to continue machining, see above, or you need to remove and remount the workpiece during the course of machining, work on existing part or the work piece is very close in size to the finished part.

## 7.11 The easy and lazy way

The easiest way to setup the coordinate systems is as follows. Just bolt down the workpiece. Jog the axes to ensure that you can reach all the parts to be machined without hitting the axes physical limits. Then jog the tool to an approximate location whose coordinates you know and set the DROs to those coordinates.

If you create your G-codes so that the finished parts smallest coordinates are at 0,0,0 i.e the part extends to the right, forward and up of the origin, then all you have to do after mounting your workpiece is to drive the tool to the bottom left corner on top of the workpiece, press 'ZERO' buttons to zero X and Y and type in the height of the work piece in the Z axis DRO and you are good to go!

For a simple system like a plasma cutter where the motors are not strong enough to cause any damage even if you hit ends of the movement above is very adequate, no home/reference or limit switches, no calibration, no nothing.

For milling you actual you want to drive the tool to the bottom left corner of the workpiece so that the tool is well within the workpiece so that there is good margin for cutting. And you also want to use set the Z-axis DRO to value slightly less than the height of the workpiece so that the cutter will not foul the table.

## 7.12 Going pro

If you need or want to align EazyCNC's coordinate system and the physical axes positions the easiest way is to have the home/reference switches installed. With those all you have to do after powering up the system is to press the 'HOME' buttons and the system will drive the axis to the switches and reset the coordinate positions.

For that to work accurately the switches need to perform repeatably so optical gates are the way to go. On the other hand optical switches need to be protected from chips and dust.

As the axes will travel at slow speed during 'homing' it is a good idea to have the reference switch in the middle of the movement range instead of at the end as this will cut down the homing time.

It is also possible to just carefully run the axes to end stops and zero the DROs or type in a know value for that position.

Above takes care of the EazyCNC coordinate system and physical machine axis relationship.

To 'calibrate' the workpiece to machine relationship you can use what ever mechanically accurate method you want. Simplest being using known sized gauge blocks and edge finder.

Or if you equip your system with a touch probe you can use that to measure the workpiece location.

## 7.13 Adjusting Feed Rate

Feedrate is controlled by the `F`-word in your G-code programs but can be override with the feed override controls illustrated in Figure 7.9.

The idea is that while the G-code programmer designs at what feedrate the part should be cut and programs that into the code it sometimes it is necessary to adjust or fine tune that while machining.

You can turn the override on and off by clicking he 'ON' button and you can adjust the override with the '+' and '-' buttons.

When the override is off the display shows the current feedrate as set by the last F-word executed, when the override is off the display shows the overridden value.

The override can be expressed either as a percentage of the F-word value or as units/minute. To change that click on the button to the right of the display field.

You can adjust the override with the '+' and '-' keys or you can type in a new value. Adjusting or setting the override value will turn on the override.

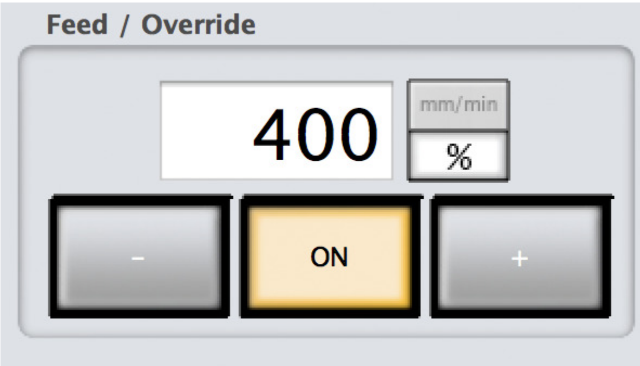



Figure 7.9: The Feed Override controls

## 7.14 Controlling the Spindle

Spindle is mainly controlled by your G-code program using S-word for setting the speed, M3 to turn the spindle 'on' clockwise, M4 to turning the spindle 'on' counter clockwise and M5 to turning the spindle 'off'.

You can manually control or override the G-code program commands with the spindle control illustrated in Figure 7.10, but as soon the G-code execution reaches next spindle control command that command will take over.

When the override is off the display shows the current spindle speed as set by the last S-word executed, when the override is off the display shows the overridden value.

You can adjust the override with the '+' and '-' keys or you can type in a new value. Adjusting or setting the override value will turn on the override.

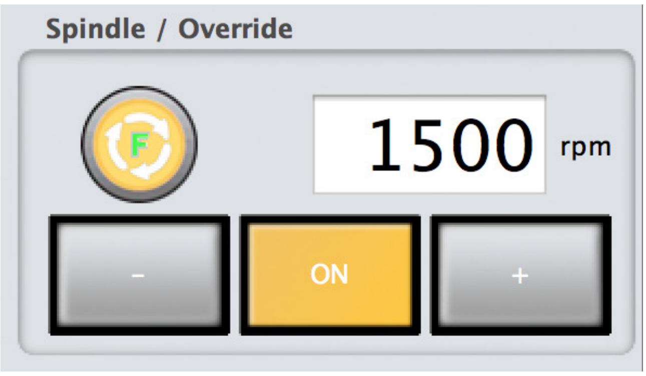



Figure 7.10: The Spindle controls

## 7.15 Machining!

Finally!

### 7.15.1 Starting the machining

Executing G-code or machining with G-code is basically very simple.

You set the operation mode, simulation or machining, with the 'SIMU' or 'MACH' button.

You load your G-code program with the 'Open' button.

Then you start the execution with the 'RUN' button, Figure 7.11.


Figure 7.11: The G-code execution control buttons

This will cause EazyCNC to interpret the G-codes line by line and control the motors and axis accordingly.

### 7.15.2 Pausing the machining

If you need to temporarily pause the execution, for example to clean up some swarf from the work area, you can press the 'HOLD' button.

If Auto SPINDLE OFF or Auto COOLANT OFF checkbox in the Options screen has been ticked then the spindle and/or coolant are turned off when enter the HOLD state.

Note that pausing the machining may slightly change the actual machined toolpath because pausing changes the speed of movements and this in turn may affect how corners are cut.

To continue machining you press the 'RUN' button.

Note that if the spindle is off when you press 'RUN' EazyCNC will issue a warning in the status/error display and beep and NOT commence running. This is to prevent cutter breakage which is sure to happen if you start machining without the spindle running.

If you get the beep/warning turn on the spindle and press 'RUN' again. Regardless if the spindle is on or off a second press of 'RUN' will always commence running.

You can't move the tool 'manually' with the jog controls while the G-code is executing, but you can move it when the system is in the paused or 'HOLD' state. If you move the tool 'manually' EazyCNC will, when you hit the 'RUN button, automatically return the tool the position where it was before you jogged it.

In the hold state the spindle and coolant will keep running if they were 'on' when you paused the machining. You can turn them on/off manually with the respective control buttons, but they will NOT be automatically restored to their original state when you resume machining.

EazyCNC may also enter the hold state when it encounters one of the pause codes `M0`, `M1` or `M60`.

When EazyCNC encounters the `M1` code it only enters the hold state if the M1-pause switch is activated, Figure 7.12.


Figure 7.12: The M1-pause switch

### 7.15.3 Stepping and Reversing

Sometimes, especially when simulating and examining G-code, you may want to execute the G-code one line at a time. To do that active the 'STEP' button, Figure 7.13.


Figure 7.13: The step and reverse execution control buttons

When the 'STEP' button is active the execution of G-code will automatically enter the pause or 'HOLD' state after executing every line of G-code.

An other use for the 'STEP' feature is in combination with the 'REVERSE' button. In general you can't run G-code backwards because you can't un-machine material back to the work piece or un-flow the coolant, but sometimes you want re-run some cut or G-code line because for example the cutter broke or you had a flameout in plasma cutting in the middle of the movement.

If something like that happens you press the 'HOLD' button to pause the machining. Then you active the 'STEP' and 'REVERSE' buttons and press the 'RUN' button to run backwards as required to get to a position before where your the tool bit broke or the arc flamed out. Don't forget to deactivate 'STEP' and 'REVERSE' afterwards.

Next you stop the spindle to change the cutter and restart the spindle.

Remember to install a kill switch for the spindle/torch and always use it before touching the spindle!

Once you have the cutter changed you can continue machining by hitting the 'RUN' button.

### 7.15.4 Stopping

The machining will of course automatically stop when it reaches the end of the G-code file or if it encounters a stop code, M2 or M30.

If you hit the 'STOP' button the machining will enter the stop prematurely.

When the system enters the STOP state G-code is re-wound to the beginning, so that next time you hit 'RUN' it will execute from the beginning. In addition if the Auto SPINDLE OFF and/or Auto COOLANT OFF checkbox in the Options screen has been ticked then the spindle and/or coolant are turned off.

## 7.16 Setting up and managing the coordinate systems

This section discusses the coordinate systems in detail.

### 7.16.1 Coordinate axes

When standing in front of the machine and looking into the machine X-axis runs from left to right, Y axis from front to back and Z axis from bottom to top. This is a well established convention in CNC world. See Figure 7.14 which illustrates the coordinate axes.

The origin of the machine coordinate system is where ever your home/ref switches are positioned or if you don't have a home/ref switch installed and enabled for an axis then origin is where ever that axis motor is when you press the 'HOME' switch. Note that the machine coordinates are not really relevant except when setting up the limits to protect your machine. The machine coordinates are not used in G-code.

The physical origin of the coordinate axis is not really relevant most of the time as you need to 'calibrate' or set your work coordinate system with respect to the work piece anyway as explained in the next section.

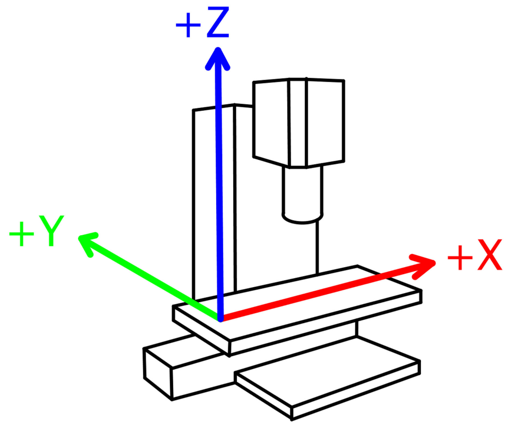



Figure 7.14: The Coordinate axes

## 7.17 XYZ versus ABC axes

EazyCNC supports up to six axes. These are named as X, Y and Z for the obvious cartesian coordinates and A, B and C for remaining rotational axes.

As far as EazyCNC is concerned all axes perform exactly the same, whether it is linear or rotational axes. Only the unit handling is different, for XYZ the units are either in 'mm' or 'inch' depending on G20/G21 mode but for ABC axes the unit is always 'deg' i.e. degrees.

If you use the ABC axes then the toolpath display is not correct as EazyCNC ignores the rotations when it draws the toolpath. In order for EazyCNC to display the toolpath correctly it would have to know the exact physical configuration of your tilting rotary tables or what have you. At this point in time this is not supported in EazyCNC.

If you command both XYZ and ABC axes to move at the same time then all the movements are perfomed in sync and complete at the same time, however feedrate calculations are probably wrong as the actual physical feedrate depends on the radius at which the cutting tool is relative to the rotating axis and there is no facility in the G-codes to specify that radius.

Therefore it is recommended that you only rotate one axis at a time and calculate the correct F-word for the move manually.

Further, the path tolerance and tool compensation do not make any sense with the rotational axes so you must turn them off manually in your G-code before you make moves involving the rotational axes.

## 7.18 Coordinates in G-codes

G-codes allow you to apply number of 'transformations' to be applied to the coordinates in the G-code file before they are used to position the tool in the coordinate system.

For convenience you can view and change all of them in the the Coordinates screen, see Figure 7.15

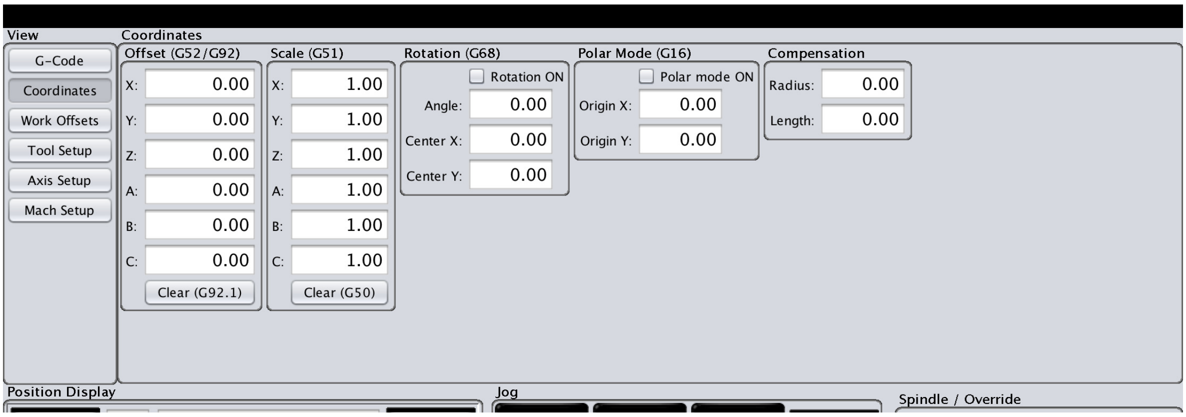



Figure 7.15: The Coordinates screen.

But you really should not change them interactively on the screen, these settings belong to the G-code file, except perhaps the work offsets.

For details how these work see the chapter 9.

## 7.19 Work/Fixture Coordinate System/Offsets

Work coordinate system, also know as fixture coordinate system or offsets, is the main thing that relates the axes physical positions to the coordinates displayed in the DROs or the G-codes coordinates.

EazyCNC supports 255 work coordinate systems, you are unlikely to ever need more than a few, most people use only one.

Mathematically a work coordinate system is simply a number for each axis that is added to the DRO or G-code axis value to convert them to the machine axis position, so basically they offset your coordinates, hence the name.

Changing the coordinate system never moves the tool or axes but it will of course change how much or to where commands move the axes after the coordinate system is changed.

The purpose of the work/fixture coordinate systems is to allow you to set up multiple workpieces and 'calibrate' a separate coordinate system for each and then machine them at one go instead of machining them one by one. Using multiple coordinate system is hardly ever worth the effort in a hobby installation, so I suggested you use coordinate system number one and stick with it.

You manage the coordinate systems with the Work Offsets panel, see Figure 7.16.

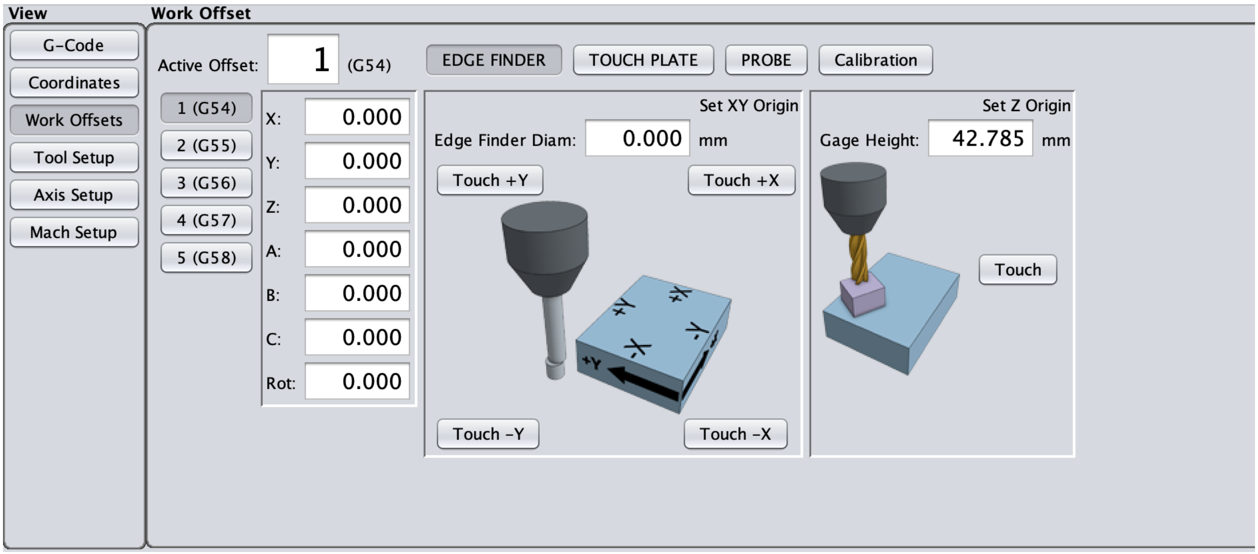



Figure 7.16: The Work Offsets screen

## 7.20 Selecting the active coordinate system / work offsets

At any given time one of the coordinate systems is in effect.

To switch from current coordinate system to another click on one of the buttons labeled with '1 (G54)' etc. or type in the number of the coordinate system you want to use into the entry field labeled 'Active Offset'. The coordinate systems are numbered from 1 to 255.

You can also, of course, change the active coordinate system in you G-code. To remind you of this and make the connection in your mind between the coordinate systems and the G-codes the corresponding G-code is shown on the button and to the right of the 'Active Offset' entry field.

## 7.21 Changing offsets/setting up the coordinates

One way to set up the coordinate system was already described in Section 7.10, that is you drive the axis to a known or desired position and set coordinates in the DROs either manually or automatically.

Using this method you effectively teach the system where the origin coordinate system i.e. the coordinates 0,0,0 are.

If you do that while you are in the 'Work Offsets' screen you will see how the offsets in the 'X:','Y:'... entry fields change.

If you feel comfortable working with the offsets directly you can just type them into the entry fields.

A third way to set offsets and to specify the coordinate/axis origin is by 'touching' the work piece or touch plate with an edge finder, electric probe or cutter in the spindle.

## 7.22 Setting the X/Y work offsets via touching

This is illustrated and done with the controls in the 'Set XY-origin' panel. This feature assumes that you want to set XY origin to edge of the work piece, typically the left/front edge of it.

Usually a special edge finder probe or wobbler tool is held in the chuck while touching because it gives much better sensitivity and accuracy to the touching and does not mark the work piece and neither it is likely to break like cutter would.

To use the touch feature simply Jog the axis to a work piece edge, ensure that the probe diameter is correctly entered into the 'Probe Diameter:' entry field and click on one of the 'Touch' buttons to indicate the edge you have the driven the probe to. EazyCNC then calculates and sets the offsets so that the coordinate origin for that axis is at the indicated edge of the work piece.

## 7.23 Setting the Z work offset via touching

This is illustrated and done with the controls in the 'Set Z-origin' panel. This feature assumes that you want to set Z origin to top milling table.

Remember that physically changing the tool or cutter requires you to re-set the Z-axis via any of the methods described here, unless you have a pre-settable toolholder system and keep the tool table up to date.

As setting the Z-coordinate origin via touching is dependent on the tool or cutter length mounted in the chuck you need to ensure that the correct tool is currently selected and set up as described in the following Section 7.27.

Touching in the Z-direction is usually done with the tool or cutter in the chuck. To use the touch feature you will need a gauge piece about 10 mm or half an inch thick.

Start by jogging down the Z-axis close to the milling table taking care not to actually hit the table with the cutter. Closer than the thickness of the gauge piece is close enough.

Next jog *up*, slowly and carefully, until your gauge piece just fits between the cutter and the table.

Now enter the thickness of your gauge piece into the 'Gauge Thickness:' field and click 'Touch'.

EazyCNC then calculates and sets the Z offset so that the coordinate origin for that axis is at the top of the workpiece.

Note that it is much safer from the tool breakage point of view to jog up i.e. first jog close to the milling table without the gauge piece in place, then job up a little bit at a time until you can just slide the gauge piece between the table and the cutter.

## 7.24 Setting the work offsets using an electric probe

If your system is equipped with an electronic touch probe you can use it by activating the PROBE button.

As a safety feature if the PROBE feature is enabled then the Touch buttons to changed to look like in Figure 7.17

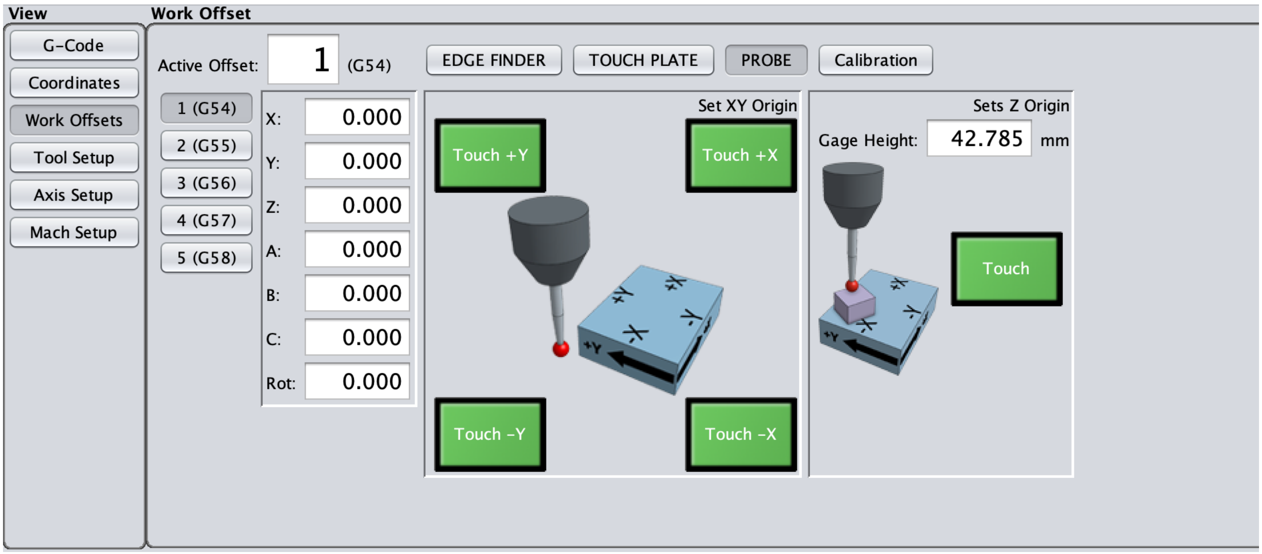



Figure 7.17: The Work Offsets screen when the PROBE button is activated

First you need to jog the tool close enough to the edge in question of the workpiece using the jog controls.

Pressing one those green Touch buttons will then cause the machine to make a small (10 mm 1/2 inch) movement on the axis in question. The movement stops as soon as the electronic probe trips and the offset for the corresponding axis is set just like it would be if you had used the manual mode an edge finder or gauge block and pressed the Touch button.

After the probe trips it retracts to its original position.

If the probe fails to trip then an error is displayed in the status line and no retraction is performed and the Work Offsets are not affected.

## 7.25 Probe calibration

Before you can use the probe you need to calibrate the probe tip in the probe calibration screen by activating Calibration-button, see Figure 7.18

For calibration you need a smooth bore ring gauge.

To calibrate place the ring gauge on vice jaws on the milling table, mount the probe in to the spindle and enter the gauge inner diameter into the Gage Diameter entry field.

Then jog the axis until the probe tip is approximately inside the ring in all three axes and the press the CALIBRATE button.

EazyCNC will the move the probe to touch the gauge ring from all directions, after which it will ask you to manually turn the spindle 180° and the EazyCNC repeats the movements.

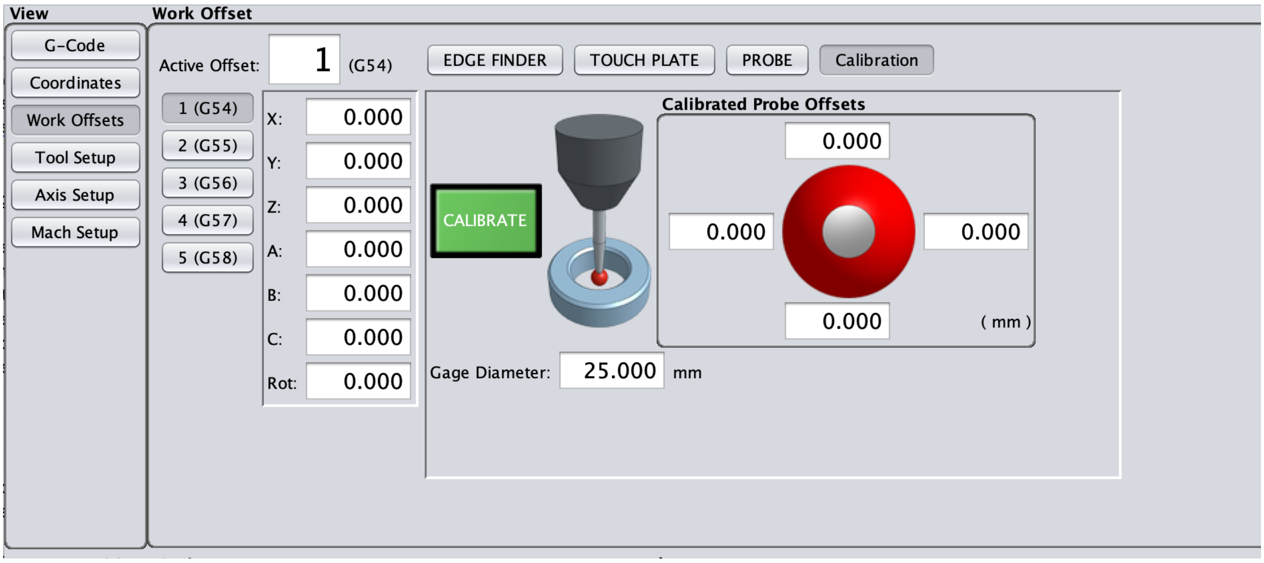



Figure 7.18: The Work Offsets screen when the PROBE button is activated

Once the all that is completed EazyCNC will calculate values for the Offsets.

If you have good quality probe all the values should be almost equal.

When you mount the probe it should always be mounted in the same orientation otherwise the calibration will not work as intended.

If you don't have a gauge ring and you trust your probe to be rotationally invariant or centric then you can skip the calibration steps and just manually enter half of the tip diameter to each of the offset entry fields.

Don't forget to SAVE your settings, other wise you will need to redo your calibration next time you use the probe.

## 7.26 Setting the works offsets using an electric touch plate

Figure 7.19

First a few words on touch plate which I guess is less familiar to many people.

This is a plate that has two lips on the under side of two edges that allow you to place it on top of the work piece so that the plate aligns with the work piece. The plate is usually galvanically (i.e. electrically) isolated from the work pieces so that the cutter touching the plate can be electrically detected. See

Figure 7.20

To use the plate you first mount the cutting tool to the spindle and then you place the plate on top of the work piece taking care to push it against a corner of the work piece.

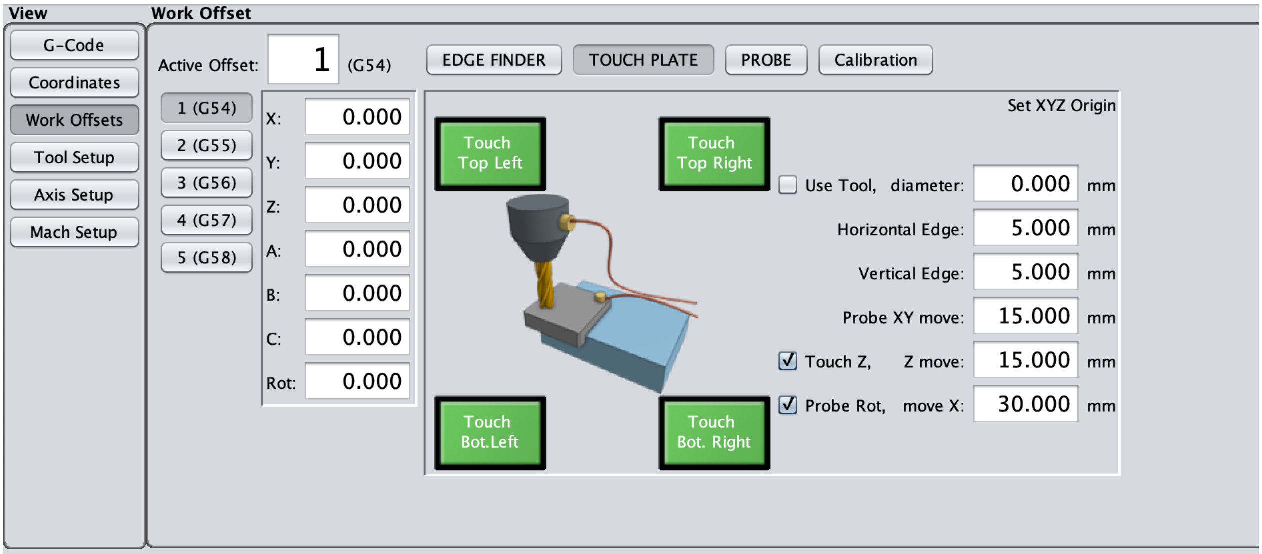



Figure 7.19: The Work Offsets screen when the TOUCH PLATE button is activated

Next you attach (using magnet usually) a wire from the probe input to the spindle. The touch plate is connected via another wire to the other probe input connector.

In this way when the cutter touches the plate the circuit is closed and a touch is detected.

The nomenclature and meaning of explanations in following maybe not be obvious, for clarification refer to Figure 7.21 and Figure 7.22 . First you mount the cutting tool into the spindle and enter the diameter of the cutter diameter field. Also make sure the 'Use Tool' checkbox is selected.

Next jog the tool so that the tool it is above and inside the touch plate perimeter when viewed from the top, as illustrated in Figure 7.20.

This tool position is the starting position.

To use the calibration you need to know the widths of the two underside lips of the touch plate because they together with the cutter diameter determine are needed to calculate the offsets when the cutter touches the plate.

Now enter those widths to the 'Horizontal Edge' and 'Vertical Edge' fields. Also enter the cutter diameter to the 'diameter' field.

Next you need to determine how much the tool needs to move both in X and Y axis from the starting position to clear the touch plate so that when the tool moves down it will not touch the plate. Enter that distance to the 'Probe XY move' field.

If you want the EazyCNC to set the coordinate system rotation select the 'Probe Rot.' checkbox. When this checkbox is selected EazyCNC will make an additional probing movement along the horizontal (x) edge of the touch plate. You need to decide that distance and enter it into the 'move X' entry field. The longer the distance the more accurate the measurement will be but obviously the move must not bring the tool outside the edge of the touch plate.

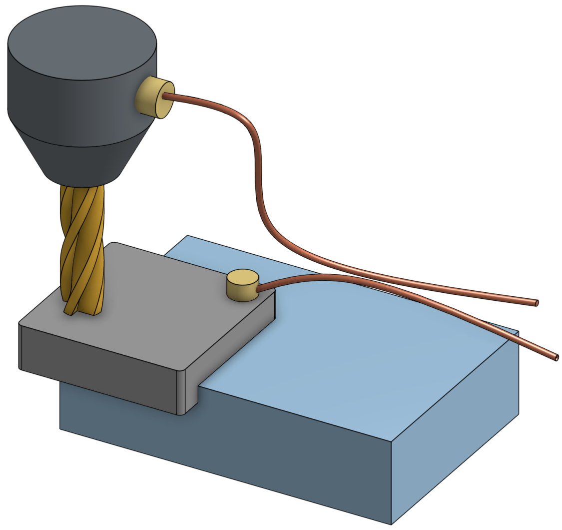


Figure 7.20: Touch plate usage

If you want to set the Z axis work offset as well as the XY offsets, make sure the 'Touch Z' checkbox is selected and enter, to the 'Z move' field, a distance that is long enough to bring the tool down from the starting position to touch the work piece.

To initiate the probing click on the green button that corresponds to the corner of the workpiece where you placed the touch plate, normally bottom left.

When a confirmation dialog the tool will probe first in the Z direction from the initial position and then sequentially from the #1 and #2 positions and finally from #3 position(if probe rotation is selected).

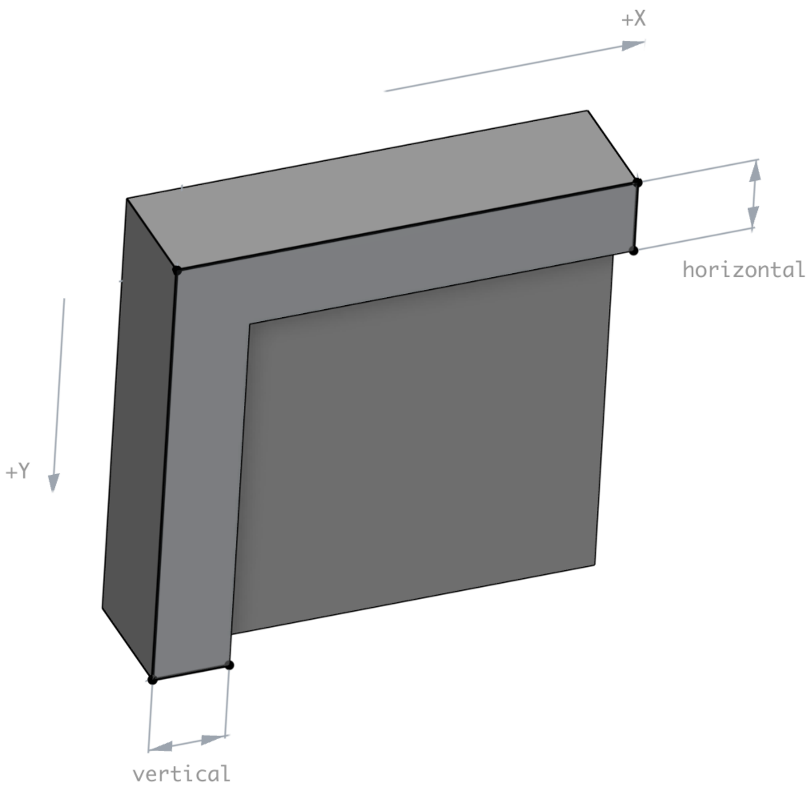



Figure 7.21: Touch plate underside

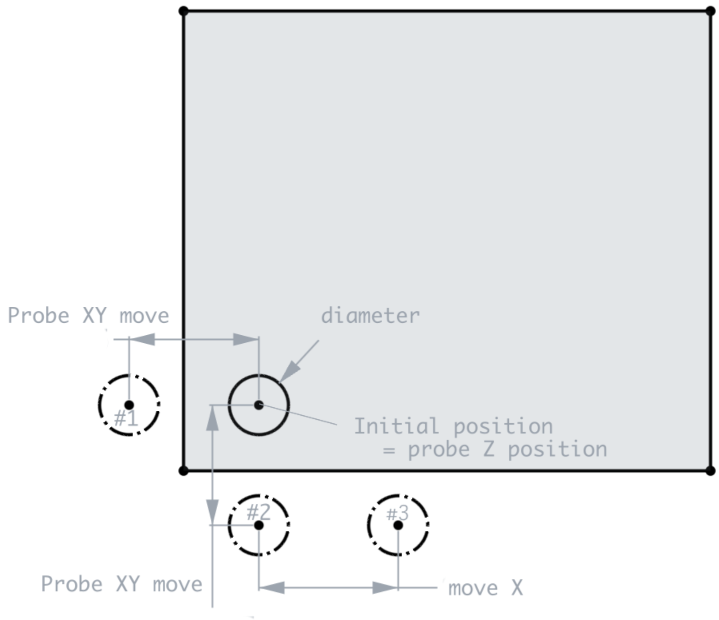



Figure 7.22: Probing positions in XY plane

## 7.27 Setting up and managing tool information

Certain features of EazyCNC rely on correct information of the tool i.e. the cutter diameter and length. To help you to manage this EazyCNC maintains a table of tool information for up to 256 tools.

You manage the tool table with the Tool Setup screen, see Figure 7.23

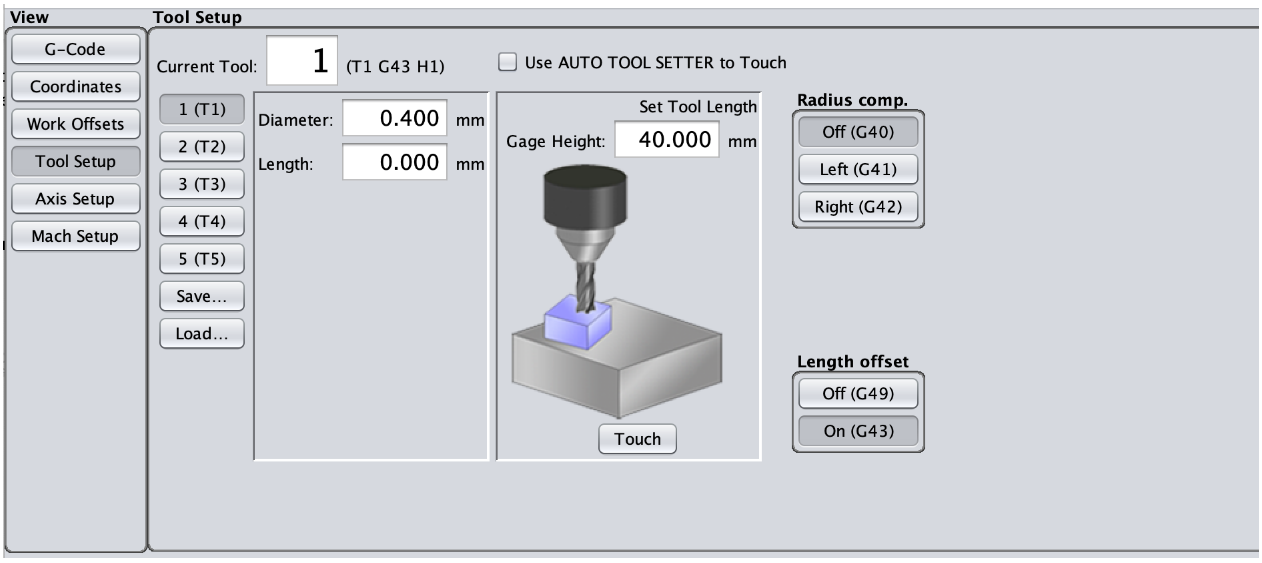



Figure 7.23: The Tool Setup screen.

The tool dimension are used in movement limit calculations and if they are not correct you may either move bang against the end of an axis movement or EazyCNC may not allow you to move the tool. This applies to both manual jogging and running G-code programs.

The tool diameter is also used in the cutter compensation calculations, and it can take advantage of the tool diameter information in the tool table or you can manage that in the G-code.

Keeping the tool information in the tool table has the advantage that the tool information can be kept up to date without touching the G-code files as cutters wear or are changed, but for some types of machines like plasma cutters where the width of the cut depends on the feed rate it maybe better managed in the G-code files themselves.

To effectively use the tool table you need a repeatable toolholder system that allows you to repeatably change the tool without affecting the cutter position i.e. a system that allows you to take a tool out and put it back in later and which keeps the tool at the same height in relation to the mill spindle.

If you don't have a toolholder like that you can just set the cutter manually, hopefully with a jig or something, to a correct depth in the chuck and update the tool length in the table manually. With this method, as the length info is probably not accurate, you need to re-set the Z-axis coordinate system every time you change the tool.

You may also decide that keeping the tool length information correct in the tool table is too much work in which case you can just set the tool length to zero in the tool table, re-set the Z-coordinates as needed and disable the Z-axis limits, see Section 6.6.2.

You also don't have to maintain the tool diameter as this can be handled in the G-code. If you find keeping the tool diameter correct in the tool table too much work then just set it zero and disable the XY-axis limits.

### 7.27.1 Setting the current tool

To set which tool in the tool table is in use, shown and manipulated on the screen click on one the buttons labeled with '1 (T1)' etc or type in a number you want to use into the entry field labeled 'Current Tool'. The tools are numbered from 1 to 256.

You can also set the current tool in you G-code program with the `T`' word. To remind you of this and make the connection in your mind between the tool table entries and the G-codes the corresponding T-word is shown on the button and to the right of the 'Current Tool' entry field.

Note that EazyCNC relies on you to keep the physical world and the software in sync. Just changing the tool number in you G-code program with the '`T` is not enough, you actually have to physically change the tool. To be able to do that you need to stop the spindle and pause the G-code execution by adding following sequence to you program.

```
M5                    ; Spindle off
T 5                   ; Select tool 5
M0 (MSG Load tool 5)  ; Pause with a message to the operator
```

When EazyCNC encounters above sequence it sets the current tool to 5 and pauses i.e. enters the 'HOLD' state, allowing you to change the tool after which you hit 'RUN'.

Don't forget to wait for the spindle to stop and use the kill switch before touching the spindle!

If you change the tool length or diameter in the tool table when you change the tool those updated length and diameter values are *not* automatically used. Instead you need to click one of the G40,G41,G42,G43 and G49 buttons as needed if you update the tool table during machining.

### 7.27.2 Managing the tool diameter and length

To set or examine a tool table entry for a tool first set it as the current tool and then view or type in the tool diameter and length in the 'Diameter:' and 'Length:' fields.

You can also update the tooltable in your G-code program with `G10 L1 Ptoolno Z radius X length` command; you could for example have a G-code program that updates all of your tool table.

### 7.27.3 Setting the tool length via touching

This is illustrated and done with the controls in the 'Set Tool Length' panel.

This feature expects that you have set your Z-coordinate system correctly, which may sound sound a bit odd as setting the Z-coordinates system via touching requires that the tool height is correct!

The way around this problem is designate one tool, (usually number 1) as the reference too and enter the lenght of that tool manually. The actual 'length' entered for that tool is immaterial as the other tool lengths will just be relative to that length. So before setting the other tool lengths select tool 1, put the reference tool into the spindle/chuchk and using a gauge block set the Z axis to zero as explained in the Section Section 7.23

Don't forget that all this is hardly worth the trouble if you do not have a re-settable toolholder, otherwise you can just always re-set the Z-axis and be done with it.

To set the tool length via touching you need a gauge piece of known thickness. The thickness is not important but about 10 mm or half an inch is good.

First enter the correct tool number into the 'Current Tool' entry field and then mount the corresponding cutter into the spindle. Then jog down the Z-axis close to the milling table taking care not to actually hit the table with the cutter. Closer than the thickness of the gauge piece is close enough.

Next jog *up*, slowly and carefully, until your gauge piece just fits between the cutter and the table.

Now enter the thickness of your gauge piece into the 'Gauge Thickness:' field and click 'Touch'.

EazyCNC then calculates and sets the Tool Length.

### 7.27.4 Using an Automatic Tool Setter

If your system is equiped with an electronic Automatic Toolsetter you can enable the automatic touch feature by checking the 'Use AUTO TOOL SETTER to Touch' check box.

As a safety feature if the 'Use AUTO TOOL SETTER to Touch' feature is enabled then the Touch button is changed to look like in Figure 7.24

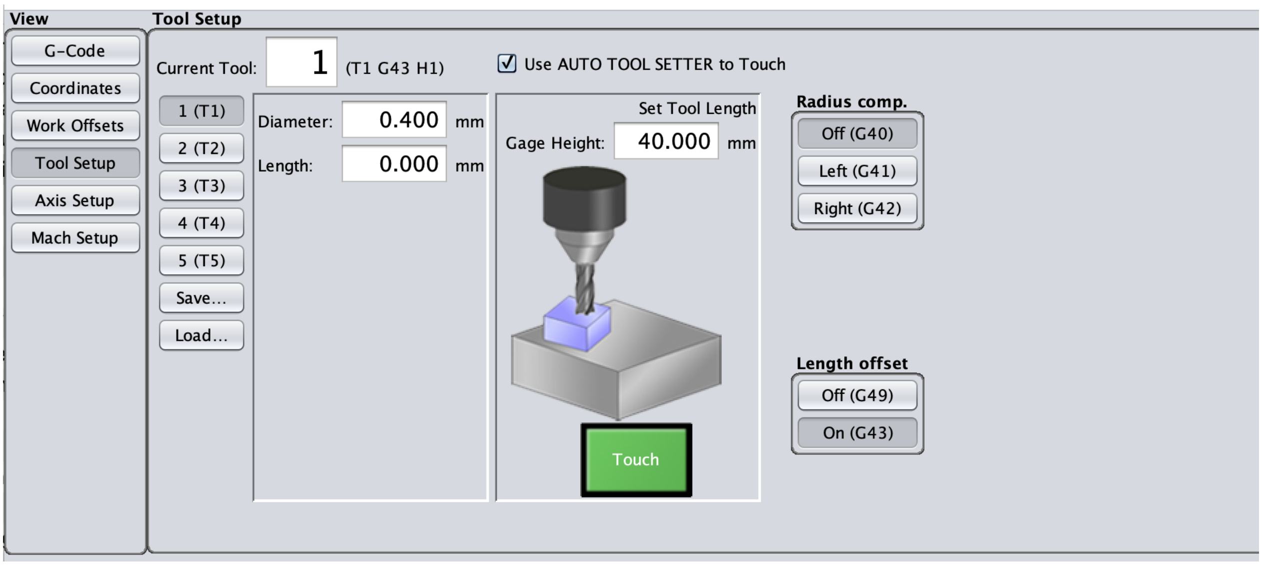



Figure 7.24: The Tool Setup screen when the 'Use PROBE to Touch' is enabled.

First you need to jog the tool close enough to top of the gauge block using the jog controls.

Pressing the green Touch button will then cause the machine to make a small (10 mm 1/2 inch) movement on Z axis. The movement stops as soon as the electronic Automatic Tool Setter trips and the length for the corresponding tool is set just like it would be if you had used the manual mode pressed the Touch button after sneanking up to the gauge block with the jog controls.

After the setter trips the tool retract to its original position.

If the setter fails to trip then an error is displayed in the status line and no retraction is performed and the Tool Length is not affected.

### 7.27.5 Editing tool setup in a spreadsheet

Some people find it more convenient to edit the tool table in a spreadsheet or text editor program.

For this purpose this screen contains the Load and Save buttons that allow you to import / export the tool settings (which are stored in the EazyCNC configuration file) to/from a text file.

The details of the text file format can be setup in the Export/Import settings screen, see section ??.

#### Save... -button

Clicking this button allows you to save the whole tool table to a text file you specify in the dialog the file dialog that appears when you click the button.

#### Load... -button

Clicking this button allows you to load the whole tool table from a text file you specify in the dialog the file dialog that appears when you click the button.

## 7.28 User Functions

EazyCNC allows you to define the function of some of the buttons in the user interface.

These buttons are located immediately below the axis DROs and in the jog buttons area. To access the later you need to press the SPECIAL key.

To use a button you will first need to associate a text file with that button.

To do that you press and hold that button until a file dialog appears and then you can browse/point to the file you want to associate with that button.

You can associate several kinds of files with a button. Which type of execution EazyCNC tries to perform when you press the button depends on the file extension of that file.

Don't forget to press SAVE to make your changes permanent!

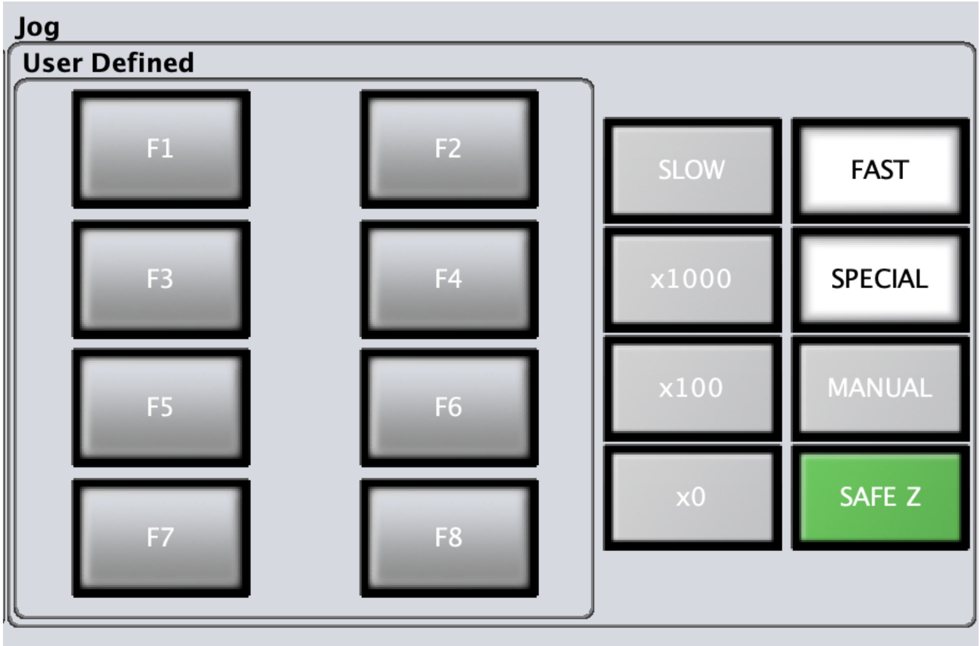



Figure 7.25: The User Function buttons

If the file name ends with '.java' then EazyCNC tries to compile that file and execute the resulting bytecode in the Java Virtual Machine.

If the file name ends with '.BAT' then EazyCNC asks the operating system (Windows) to execute that file as if you had typed that name at the DOSBox.

If the file name ends with '.sh' then EazyCNC asks the operating system (MacOS/Linux) to execute that file as if you had typed that name at the terminal window.

If none of the above applies then EazyCNC tries to execute that file as a G-code file i.e. behind the scenes it does the same as if you pressed LOAD and RUN for that file.

All of the above file formats are basically just text files and support adding comments them.

When you associate a file with a button EazyCNC scans the file for comments and recognizes a few special comments that have no significance to the execution of that file but which can and should be used to configure the button.

EazyCNC recognizes lines that begin as follows as comments:

';' for use in G-code files,

'#' for use in shell scripts,

'REM' for use in .BAT files and

'//' for use in .java files.

If a comment line begins with 'button = sometext' then EazyCNC will use that 'sometext' as the text to be displayed inside that button. If you want to have two or more lines of text in the button you can use the '
n' marker to marke the line break in the text.

If a comment line begins with 'key = keyname' then EazyCNC will associate that button with that keyboard key and execute that file when you pres that keyboard key. Joystick and MPG pendant keys can be used as well.

### 7.28.1 Built-in User Functions

EazyCNC has number of built-in user functions.

You can find them in ' /EazyCNC/plugins/builtin/com/eazycnc/builtin' if you want to use them as example or starting point for your own functions. Note that you should not edit them as EazyCNC will overwrite your changes on every launch of the application. Instead make a copy of any of them with a different file name and assign that file to a button.

As an example the ZERO ALL button is implemented as a G-code text file stored into the file 'ZeroAll.txt' :

```
; button = ZERO\nALL
G54
G10 L20 P1 X0 Y0 Z0 A0 B0 C0
```

and the HOME ALL button is defined by the following Java code stored in the file RefAllHome.java:

```
// button = HOME\nALL

package com.eazycnc.builtin;
import com.eazycnc.plugin.*;
import com.eazycnc.uwk.widgets.*;

public class RefAllHome extends Plugin {
    @Override
    public void onAction(Button button) {
        EazyCNC.doHomeAction(button,0,1,2);
    }
}
```

Note that ZERO all zeros all six axes X, Y, Z, A, B and C but HOME ALL homes only X, Y and Z axes.

# Chapter 8

# Cutter compensation

G-codes specify the cutter or tool movement with coordinates. These coordinates specify the position of the spindle of the milling machine. Because the cutter of the milling has a non negligible size they do *not* specify the size and shape of the part you are cutting. It is the job a CAM software post processors software to turn CAD models into toolpaths that accommodate for the diameter and length of the cutter.

However G-codes specify and EazyCNC supports limited automatic cutter length and diameter compensation so that you can use the *coordinates of the part* to be machined instead of the coordinates of the spindle.

These can be useful with hand written G-code programs and with some special machine configurations such as plasma cutters and when working mainly in plane i.e. two dimensions.

## 8.1 Tool length compensation

Tool length compensation is rather simple to understand, basically it just moves the machine higher by the length of the tool. So when the G-codes specify for example `Z 200` and the current tool length compensation is turned on (with `G43` code) and the tool length for the current tool (set with the `T` word) is 50 the actually Z coordinate the machine moves to is 250.

## 8.2 Cutter diameter compensation

Cutter diameter compensation is not difficult to understand but there are more details to consider.

Basically if you use the diameter compensation you put the coordinates of the part your are machining in to your G-code program and when you turn on the compensation you tell EazyCNC on which side of the path you want the tool travel. The side of course depends on weather you are cutting an opening or an outline of the path and weather you are traveling the path clockwise or counter clockwise.

The cutter compensation cannot change the past, what is cut is cut, so when you turn on the compensation (`G41` for and `G42`) for right) the compensation will only have an effect the next move.

Therefore you need pay attention to the first move when you are planning your tool path and you need to pre-position (usually with G0 code) the tool outside of the part by at least half the tool diameter. Also consider the direction this first move-in cut will make, ideally this should be in the same direction as the first actual part outline to be cut, but for inside holes or openings this is not always possible.

### 8.2.1 Cutter compensation example - cutting part's outline

To get a more concrete picture of above please study the following example G-code and the Figure 8.1 which illustrates various aspects of the diameter compensation process.

```
N1 G40      ; make sure diameter compensation is off
N2 G0 X2 Y1 ; move to our starting position
N3 G42 P0.4 ; turn on compensation to the right for 0.8 unit tool diameter
N4 G1 X3 Y2 ; first move, from starting point to part perimeter
N5 G1 X8 Y2 ; cut the lower edge
N6 G1 X8 Y6 ; cut the right side
N7 G1 X3 Y3 ; cut the slanted top edge
N8 G1 X3 Y2 ; and finally back to where we started cutting the left edge
N9 G40      ; compensation is off
```

Referring to the Figure 8.1 the black line illustrates the uncompensated coordinate path i.e. the coordinates in the G-code file. The green lines illustrates the path of the center of the tool and the compensated coordinates. And finally the red line illustrates the left edge of the groove the cutter actually cuts, i.e. this is the actual outline of the part we are making here, so any deviation between the black and red line (except for the initial move-in movement) is a indication of badly constructed G-code program.

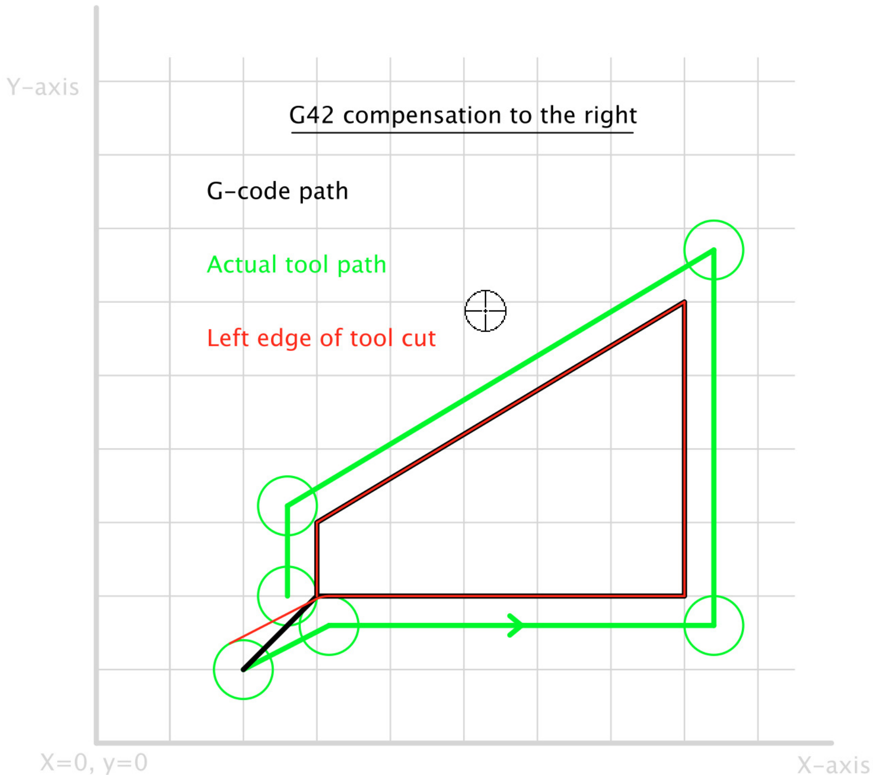



Figure 8.1: G-code path versus compensated cutter path

Overall the part outline and the cut path seem to match but if you look carefully in the blown up detail of the beginning of the cut in Figure 8.2 you see that cutting the path as specified would leave a semi circular concave notch to the part and besides, at least in theory, the part would be left hanging by a thread as the loop is not completed. These problems come from the first two actual moves (lines N2,N4 and N4) which are not parallel.

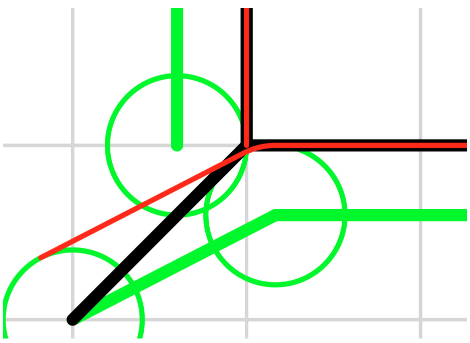


Figure 8.2: Cutter compensation detail

### 8.2.2 Cutter compensation example - cutting holes and openings

To illustrate cutting a hole or cutout to match the above part we can just set compensation to the left and move the starting point to the inside of the path. Following G-code does just that, the actual path is identical to the previous example only the compensation side and starting point are different. The starting point was deliberately chosen badly and the cutter is way oversize relative to the path dimensions to exaggerate the issues you may encounter when designing your paths.

```
N1 G40       ; make sure diameter compensation is off
N2 G0 X5 Y3 ; move to our starting position
N3 G41 P0.4 ; turn on compensation to the left for 0.8 unit tool diameter
N4 G1 X3 Y2 ; first move, from starting point to part perimeter
N5 G1 X8 Y2 ; cut the lower edge
N6 G1 X8 Y6 ; cut the right side
N7 G1 X3 Y3 ; cut the slanted top edge
N8 G1 X3 Y2 ; and finally back to where we started cutting the left edge
N9 G40       ; compensation is off
```

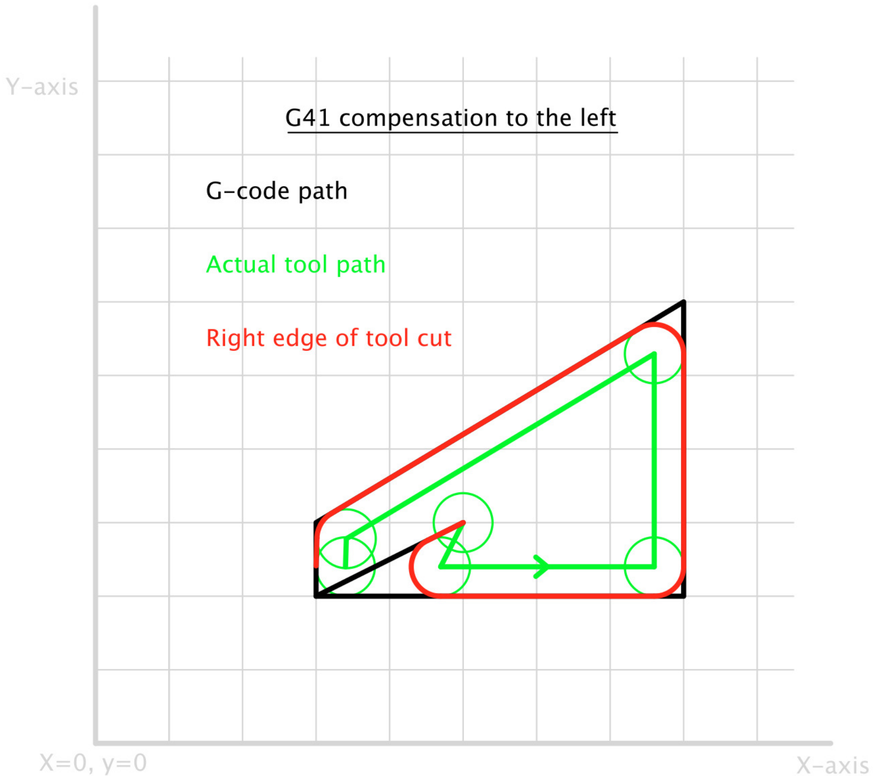



Figure 8.3: G-code path versus compensated cutter path

# Chapter 9

# G-code reference

G-code is ancient by computer standards, you can trace it's roots back to MIT labs in the 1950s. Wikipedia describes G-code as "a language in which people tell computerised machine tools what to make and how to make it".

Today this is not actually a good description. Rather today G-code is the standard by which computer aided design and manufacturing programs, CAD/CAM software, communicate the machining instructions to CNC machines.

There is a EIA standard from year 1980 called RS274D that sort of defines the G-codes but over the years manufactures have extended and interpreted it differently so that today there are many dialects.

EazyCNC tries to support a common subset of what the two most common hobby CNC controller programs, EMC2 and Mach3 support.

G-code should not be written by hand, instead a CAD/CAM program should be used to generate it. Not only is this a lot faster but it also reduces the chances of errors and improves the quality of machining as the CAD/CAM program has a much better understanding of the machining process as a whole than the CNC machine ever can.

CAM software actually produces a very simple, 'pidgin English' kind of G-code which is rather universal in order to be avoid the differences between machines. Different interpretation for some of the G-codes can be configure in the Mach Setup screen, see Section 6.8.

Therefore it is not really important to learn to write G-code, but it is handy to be able to read it to some extent and this is what this chapter tries teach you to do.

## 9.1 The Basics

G-codes programs are text files made of lines of commands for the CNC machine. The file names often have the file extension '.nc' but this is by no means required, you can use '.txt' as well.

EazyCNC reads the program line by line executing the commands.

The commands are made of single letter 'words' possibly followed by number, for example 'G0' or 'X1000'.

The words can be written in upper or lower case letters. Any number of 'space' characters can be placed anywhere on each line to improve readability; they are ignored and make no difference to the execution.

In this document a hashtag, '#' after a word is used to denote a word followed by a value, i.e. number or expression. These are optional arguments or parameters to the G-code in question. For example : 'G0 X# Y# Z#' indicates that you can write 'G0 X10 Y20'. Weather parameters are optional or not is explained in the text. A asterisk '*' is used to denote a compulsory or required word.

Comments to the human reader i.e. text that EazyCNC will not try to interpret as command can be placed on lines if preceded with a semicolon ';' or enclosed in parentheses, '(' and ')'.

Everything on after a semicolon ';' on a line is also treated as a comment.

The order of words on a line makes no difference.

No word or G-code can appear more than once on any given line.

Some G-codes are actually M-codes!

Line numbers are optional and effectively ignored but if you want you can use line numbers. Line numbers are expressed as the 'L' word followed by up to five digits.

### 9.1.1 Operator Messages

Everything after the comma ',' in a comment that starts with '(MSG,'' is display as message to the operator in the status display, Figure Figure 7.3.

### 9.1.2 Debug Messages

Everything after the comma ',' in a comment that starts with '(PRINT,'' is output to the Java console. This can be very handy when debugging G-code programs. When outputting text to the console #number -references in the text are replaced with value of the parameter with that number.

For example :

```
(PRINT,value of parameter 123 is #123)
```

will output the value of parameter 123 into the console.

## 9.2 Numbers, Expressions and Parameters

### 9.2.1 Numbers

Numbers in commands are expressed as one or more digits, optionally followed by a decimal point and one or more digits. A number can optionally be preceded with a plus or minus sign.

It is also acceptable to leave out a zero preceding the decimal point.

Some valid examples:

`7 3.1415 -2.0 .1`

### 9.2.2 Expressions

Where ever a number can be used an arithmetic enclosed expression in brackets, '[' and ']', can be used.

For example : the commands

`X100`

and

`X [ 50 + 70 ]`

are equivalent.

#### Arithmetic operators

Nine common arithmetic operations are supported.

The operations are dived to three groups and denoted as follows. Operations in a higher precedence group are executed before those with lower precedence. Operators in on the same precedence level are executed from left to right. This pretty much follows the common arithmetic expression calculation order.

The highest precedence group only has the exponentiation or power operator '**'.

The second highest precedence contains the multiplication, division and modulus operators '*', '/' and '`MOD`' respectively.

The lowest precedence is for addition, substraction, i.e. '+' and '-', and the logical operators '`AND`', '`OR`' and '`XOR`' aka exclusive or.

The logical operators work on numbers interpreting any non zero number as logical one or true value.

The order of operations can be altered by enclosing the sub expression in square brackets ('[' ']'), do not use parentheses ('(' ')').

#### Arithmetic functions

Arithmetic expressions support a number of common arithmetic functions. See table Table 9.1.

Table 9.1: Mathematical functions

| Operator | Function |
|---|---|
| ABS | absolute value |
| ROUND | round to nearest integer |
| FIX | round down to previous integer |
| FUP | round up to next integer |
| COS | cosine function |
| SIN | sine function |
| TAN | tangent function |
| ACOS | arcus cosine function |
| ASIN | arcus sine function |
| SQRT | square root function |
| EXP | e to the 'power of' function |
| LN | natural logarithm function |

A function is expressed as the name of the function followed by the function argument enclosed in brackets, '[' and ']'.

For example : calculate the square root of two you would write:

```
SQRT[2]
```

Note that for some weird historical reason one function, ATAN, has an exceptional format, to write the equivalent of

$arctan(123/456)$

you need to write

```
ATAN[123]/[456]
```

### 9.2.3 Parameters

Parameters are what most programming languages would call variables i.e. they are storage locations for values which can be used in place of numbers in expression.

Instead of names the parameters are referred to by numbers preceded with a '#'#sign.

For example :

```
#5324
```

Some parameters have a special meaning, others you are free to use as you wish to make your code more useful.

You can for example create G-code programs that is parametrized to cut different sized parts by just changing a parameter or two.

To change or set a parameter you follow the parameter name with the equal sign '='. For example :

```
#5324 = 12.34
```

It is possible to both set and refer to the value of a parameter, even several times, on the same line. The parameter values are only assigned or set once the whole line has been executed. If a parameter is assigned several times on a line the last one will take effect.

So for example after executing following lines:

```
#22=5 #22=[#22+1] #23=#22 #22=[#22-1]
```

the parameter 22 has value 4 and parameter #23 has value 5.

Note that the number following the '#'#sign can itself be an expression which makes for some interesting possibilities, you can for example use it to index parameter to simulate what most program languages would call arrays.

## 9.3 G-codes and M-codes

### 9.3.1 Length Units, G20,G21 codes

Coordinates i.e. positions or distances in G-codes are expressed in either millimetres or inches.

The G20 code tells EazyCNC that following coordinates and lengths are in inches, G21 tells it that they are in millimeters.

It is recommended that no coordinates or length are specified on the same line as G20 or G21 code is used.

### 9.3.2 Coordinate Axes

G-code specifies the movement of the machining tool using orthogonal Cartesian coordinates. (There is also polar coordinate mode, see G-codes G15 and G16.)

The coordinate system follows the right-hand rule with the positive X-axis pointing to right, Y-axis away from the operator and Z-xis pointing up, so values increase from left to right, front to back, and bottom to top, as illustrated in Figure 9.1.

Figure 9.1: Coordinate axes of a 3-axis CNC System

### 9.3.3 Setting the length units – G20,G21

G-codes specify the length and any related physical quantities like feedrate in either metric or imperial units. For the metric system the basic unit is one millimeter and for the imperial system it is one inch. It is possible but not recommended to mix the units in a single G-code program, but it is perfectly feasible machine 'metric' G-code programs in a machine configure for imperial units as long as the program contains the proper G-code to set the length units.

The `G20` sets the imperial (inch) units mode and `G21` sets the metric (mm) units mode.

It is an error to have botn `G20` and `G21` on the same line.

### 9.3.4 Feedrate – F-word

The `F`-word sets the current feedrate for all the other movement commands than `G0` which is always performed at maximum speed.

Depending on the length units mode in effect the `F`-word value specifies the feedrate either in inches per minute or millimeters per minute.

It is an error if the feedrate is not a positive number.

### 9.3.5 Spindle speed – S-word

The `S`-word value sets the current spindle speed in rotations per minute (rpm).

It is ok to specify a larger rpm value than what is set up for the system, see Section ??, but of course the spindle can't run faster that it's maximum speed.

Setting the spindle speed does *not* turn on the spindle, you need to program `M3` or `M4` to turn on the spindle.

Note that the speed change can take several seconds to take effect so a dwell or `G4` code should be programmed right after a spindle speed change.

Also note that specifying zero speed does not guarantee that the spindle stops, on the contrary it is most likely to remain turning at some low speed. To stop the spindle use the `M5` command.

Remember never to touch the spindle unless the kill switch has been applied!

It is an error if the value is not a positive number or zero.

### 9.3.6 Spindle on/off – M3,M4,M5 codes

The `M3` code turns on the spindle in the clockwise a.k.a. forward direction direction.

The `M4` code turns on the spindle in the counter clockwise a.k.a. reverse direction direction.

Note that reverse running the spindle can be dangerous if the chuck (in a lathe) is screw mounted and of course trying to machine with reverse running cutter will only damage the cutter and ruin the workpiece.

The `M5` code turns off the spindle.

Note that the turning the spindle on/off can take several seconds to actually take place so a dwell or `G4` code should be programmed right after commanding the spindle on or off.

Also note that depending on your spindle drive system abruptly changing directions can damage the equipment or be dangerous.

It is an error if more than one of `M4`,`M5` or `M6` is programmed on the same line.

### 9.3.7 Coolant on/off – M7,M8,M9 codes

The `M7` (mist cooling) and `M8` (flood cooling) codes both turn on the coolant and mist/flood aspect of the codes is ignored.

The `M9` code turns off cooling.

Note that when turning coolant on it can take several seconds for the coolant pump to react so you may want to program a dwell or `G4` right after turning the coolant on.

It is an error if more than one of `M7`,`M8` or `M9` is programmed on the same line.

### 9.3.8 Select a tool – T-word

The `T`-word designates a tooltable entry as the current tool. Note that this alone does *not* do anything else.

To physically change the tool you need to program a pause `M0` and change the tool yourself and to use the tool length you need to use the `G43` command and to use the tool diameter info you need to program `G40` or`G41` as needed.

It is an error if the value is not a positive or is larger than the number of tools supported by EazyCNC.

### 9.3.9 Dwelling – G44-code

The `G4`-word causes EazyCNC to wait for the specified time before executing the next G-code program line, this is useful for example after turning on or changing the spindle speed.

The wait time is specified with the `P`-word in either seconds or milliseconds depending on which G-code interpretation options has been selected, see Section 6.8. If a G-code program seems to hang for a long time then probably that setting for the `P`-word interpretation is wrong.

It is an error if the value is not a positive or zero.

### 9.3.10 Coordinates/moving axes – XYZABC -words

Coordinates in G-code programs are expressed as an axis letter, see followed by the coordinate position either as a number or as an expression enclosed in brackets.

The presence of an axis letter in a G-code line is an implicit command for the named axis to move.

The movement is carried out in the current motion mode either `G0`,`G1`,`G2` or `G3`. This allows for more compact representation of the toolpath as most of a G-code program will be just a long list of coordinates to move the tool.

Axis letter names the axis that will move, valid letters are '`X`','`Y`','`Z`','`A`','`B`','`C`'.

If an axis does not need to move on a given line it is not necessary to specify the axis and its position at all.

The axis movements are co-ordinated so that the movements start and stop at the same time and the axis speeds are such that the tool will move at the specified rate.

Here is an example that will cause X and Y axis to move to the position 120,140 in current motion mode:

```
X 120.0 Y [100.0+40.0]
```

It is an error if the same axis word appears twice any given line.

Note that actual axis movements are subject to machine limits acceleration limits and if the best speed mode (`G64`) is enabled then the movements may 'cut corners'.

Also note that the coordinates specified with the axis words maybe scaled, offset and event rotated, see XXX

### 9.3.11 Motion mode – G0,G1,G2 and G3 codes

As describe above specifying coordinates with axis words causes EazyCNC to move the axis according to the current motion mode.

The `G0`,`G1`,`G2` and `G3` codes are used to specify the current motion mode, which will stay in effect until and other motion mode is specified.

If the motion mode is explicitly specified with one of the above G-codes, then at least one axis word needs to specified on the same line.

It is an error to specify more than one motion mode in a single line.

Is is an error to specify a motion mode without any axis words.

### 9.3.12 Rapid positioning – G0 code

Rapid positioning will move the specified axes in co-ordinated fashion as fast as possible i.e. at the Max Velocity set up in the Mach Setup, see YYY.

`G0` is used to rapidly position the tool for the beginning of a cut and is not meant for machining.

### 9.3.13 Linear interpolation – G1 code

For some archaic reason the machining G-codes are called interpolations.

`G1` is used to tell EazyCNC to move the tool at the current feedrate to the given position. This is the 'work horse' mode of all G-codes, most machining will take place in this mode.

### 9.3.14 Clockwise Arc interpolation – G2 code

`G2` is used to tell EazyCNC to move the tool at the current feedrate from its current position to the given position following a circular arc path clockwise on the active plane as set by the `G17` (XY-plane), `G18` (XZ-plane) or `G19` (YZ-plane).

Clockwise means as if you were looking down at the arc on the active plane from the third axis i.e. if your are cutting in the XY plane and looking down at it from the positive Z axis.

The end point of the arc is specified with the `'X'`,`'Y'` `'Z'` words. It is acceptable to leave out axes which do not need to move, for example typically if you are cutting an arc in the XY plane then you don't specify the `'Z'` coordinate.

If a movement in the direction perpendicular to the current plane is specified then the tool will actually follow a three dimensional helical path.

The curvature of the arc is specified either by giving the centre point of the arc or by specifying it's radius.

#### Specifying arc using center point

`G2 X# Y# Z# I# J# K#` specifies the arc curvature using the radius method.

The center point is specified with the `'I'` and `'J'` and `'K'` words for the coordinates in the X,Y and Z planes respectively. The the IJK words specify the center coordinates relative to the starting point of the arc or as coordinates in the current coordinate system.

Which interpretation is used depends on the machine setup see Section 6.8.1 or G-codes G90.1 and G91.1 see Section 9.4.18.

If your toolpath preview shows large arcs that don't make sense then it is likely that the interpretation mode of the IJK words is wrong.

It is acceptable to omit any of the any but not all of the end point words (XYZ) and center point words (IJK) in which case the last specified word values are used.

Note that by specifying the start,end and center point of the arc you are over specifying the arc, this may result in an error message if the distance from the start point to the center differs too much from the distance from the end point to the center.

#### Specifying arc using radius

`G2 X# Y# Z# R#` specifies the arc curvature using the radius method.

If the center point method of specifying the arc curvature is not used then radius of the arc must be specified with the `'R'` word. The start and end points alone do not specify an unambiguous arc, mathematically for any two points and radius there are two arcs that connect the end points, one arc is less than 180°and the other is larger.

If the radius specified with the `'R'` word is positive then this is interpreted as the arc that turn less than 180°and if the radius is negative then it is interpreted to mean the arc that makes the longer turn and the absolute value of the `'R'` word is used as the radius.

Do not try to cut full or nearly full circles with this method as this makes the start and endpoints very nearly the same which means that any roundoff error in the calculations has large effect in the internal arc center point calculation.

### 9.3.15 Counter Clockwise Arc interpolation – G3 code

The `G3` command performs the same as `G2` but the arc is 'drawn' counter clockwise.

### 9.3.16 Perform probing move – G31

The probing command `'G31'` works the same as the `'G1'` in other words it programs a linear movement at current feedrate, except that this command must always be explicitly specified on a line and does not 'carry over' from line to line like the modal `'G1'`.

If the probe input becomes activated during the movement the movement is stopped as soon as feasible without losing any steps and the parameters DRO values at which the probe became active are copied to the parameters 2000-2005.

Note that it is important that the feedrate is low enough because the probe movement will overshoot the position at which the probe trips/triggers by an amount that depends on the feedrate and it is equally important that the probe design allows for the overshoot, otherwise the probe and/or workpiece can be damaged.

### 9.3.17 Pause Machining – M0,M1

The command `'M0'` pauses the execution of G-codes and puts the system into the HOLD-state.

The 'M1' works the same way except it only pauses if the M1-switch on the user interface is activated, see Figure 7.12.

### 9.3.18 Stop Machining – M2

The command 'M2' stops the execution of G-codes and puts the system into the STOP-state.

## 9.4 Coordinate systems

As said in Section 9.3.10, the coordinates are specified using the axis words 'X','Y','Z', 'A','B','C' and 'I','J','K' for the arc centers in G2 and G3 commands.

This is not the end of the story though.

G-codes provide number of ways to transform the axis word values before they become final physical positions of the CNC machine axes.

This section provides the nitty gritty details.

Table 9.2 lists, in order of application, all the coordinate transformations every axis word value goes through.

If the absolute mode G53 is active then none of the offsets or the rotation is applied.

Table 9.2: Coordinate transformations

| |
|---|
| Apply G51 Scaling |
| If in G91 mode interpret words as incremental coordinates |
| If in G16 mode interpret words as polar coordinates |
| Apply G52 temporary coordinate offsets |
| Apply G68 rotation |
| Apply G44 tool length offset |
| Apply G54 work/fixture offsets |

### 9.4.1 Scaling – G50,G51 codes

G50 turns off scaling and sets scale factors for all axis words to 1.

G51 X Y# Z# A# B# C# turns on scaling and specifies the scaling factor for the given axes. The scale factors for axes that are not specified remain at their previous values.

The purpose of scaling is to scale the part being cut and thus it is the first operation to be applied to coordinates and so it will not affect position of the part on the workpiece which is set by the fixture offsets nor does it affect the tool length or temporary offsets.

For example : following will scale a 2D part design to three times it's original size.

```
G50       ; ensure all scaling is off and set to 1.0
G51 X3 Y3 ; scale 2D design by three
```

It is an error to have both G50 and G51 on the same line.

### 9.4.2 Incremental mode – G90,G91 codes

G91 turns on incremental coordinate mode. In incremental model all axis words are interpreted relative to the current to position tool position.

For example : following

```
G91      ; turn on incremental mode
G1 X100 ; mill hundred units to the right
Y50      ; hundred units forward
X-100    ; back to start X coordinate
Y-50     ; and we are where we started from
```

defines a rectangular toolpath 100 units wide by 50 units high to the right and front of current tool position.

G90 turns off incremental coordinate mode

It is an error to have both G90 and G91 on the same line.

### 9.4.3 Polar coordinate mode – G15,G16 codes

G16 turns on the polar coordinate mode and sets the current tool position as the origin of the polar coordinate system. In polar coordinate mode the 'X' word is interpreted to mean the polar coordinate distance relative to the polar coordinate origin and 'Y' word is interpreted as the angle (in degrees) of the polar coordinate.

In incremental mode 'X' and Y' are interpreted relative to the previous distance and angle, not the current Cartesian X,Y position.

The following example defines a hexagonal tool path where each side of the hexagon is 10 units long.

```
G16      ; turn on polar mode
G91      ; turn on incremental mode
X0 Y0    ; 'reset' distance and angle for incremental mode
X10 Y60 ; Cut the 1st side
X10 Y60 ; ... and 2nd side
X10 Y60 ; ... and 3rd
X10 Y60 ; ... 4th
X10 Y60 ; ... 5th
X10 Y60 ; 6th side and back to where we stared from
```

Following example the defines a pentagonal tool path with radius of 20 units.

```
G16      ; turn on polar mode
G90      ; turn off incremental mode
X20 Y72  ;
X20 Y144 ;
X20 Y216 ;
X20 Y288 ;
X20 Y360 ;
```

`G15` turns off the polar coordinate mode.

It is an error to have both `G15` and `G16` on the same line.

### 9.4.4 Temporary coordinate system offsets – G52

The `G52 X# Y# Z# A# B# C#` code sets the temporary coordinate offsets for the given axes to the given values. The offsets for axes that are not specified remain at their previous values.

### 9.4.5 Temporary coordinate system offsets – G92,G92.1,G92.2,G92.3 codes

All these codes are legacy features that are best left unused.

As all *G52* and *G92* codes all use the same mechanism mixing them requires extreme care, one more reason to leave all this well alone!

`G92 X# Y# Z# A# B# C#` sets the temporary coordinate offsets for the given axes so that the current tool position has the given coordinates.

`G92.1` saves the temporary offsets to G-code parameters 5211..5216.

`G92.2` clears the temporary offsets.

`G92.3` restores the temporary offsets from the parameters 5211..5216.

### 9.4.6 Coordinate system rotation – G68,G69 codes

`G68 A# B# I# R#` sets the coordinate system rotation.

The '`A`' and '`B`' specify in the local coordinate system the X,Y coordinates around which the coordinate system is rotated.

The 'R' word specifies the rotation in degrees, positive values giving counter clockwise rotation when viewed down from positive Z-axis i.e. looking down at the work piece.

If the 'I' word is present then the 'R' word value is treated as an increment to the current rotation, the value of 'I' word is ignored.

Rotation is only available in the XY-plane. Rotation can only be turned on if the active plane is XY (G17).

G69 turns off the coordinate system rotation.

It is an error if 'A', 'A' or 'R' is not specified.

It is an error to have both G68 and G69 on the same line.

### 9.4.7 Active plane – G17,G18,G19 codes

The arc interpolation i.e. arc cutting ('G2' and 'G3') works by calculating a circular path in the active plane which is one of the main coordinate planes XY,XZ or YZ plane.

The 'G17' sets the active plane to XY-plane.

The 'G18' sets the active plane to XZ-plane.

The 'G17' sets the active plane to YZ-plane.

It is an error to use more than one of G17,G18 or G19 on the same line.

It is an error to program G18 or G19 if the coordinate system rotation G( is on.

### 9.4.8 Tool length compensation – G43,G44,G49 codes

G43 H# sets the tool length compensation based on the tool number specified with the 'H' word and the tool length of set up for that tool in the Tool Setup panel.

G44 H# works the same as G43 but it expect that the length values in tool set up are negative, this should not be used and is provided for compatibility only.

A 'H' value of 0 can be used to turn off tool length compensation and is equivalent to the G49 command.

It is an error 'H' word is missing, is not an integer, is negative or larger than the number of tools EazyCNC supports.

It is an error more than one of G43,G44 and G49 on the same line.

`G49` turns the tool length compensation off by setting it to 0.0 value.

### 9.4.9 Work/fixture offsets – G54,G55,G56,G57,G58,G59 codes

Any of the six first work/fixture coordinates systems/offsets can be selected with the `G54...G59` codes. The `G54` selects the first i.e. number one coordinate system specified in the Work Offsets panel.

`G59 P*` selects the work/fixture coordinate system specified with the 'P' word. The 'P' word is optional and if not given it behaves as described above.

It is an error the'P' word is given with `G59`, is not an integer, is smaller than 1 or larger than the number of work/fixture coordinate systems EazyCNC supports.

### 9.4.10 Absolute coordinates – G53 code

If a line that causes linear interpolation (i.e. implicit or explicit `G0` or `G1` command) contains the `G53` command then all the coordinates on that line are treated as absolute coordinates without applying any offsets or rotations. Scaling and cutter radius compensation are applied regardless if they are enabled.

### 9.4.11 Cutter radius compensation – G40,G41,G4 codes

EazyCNC can adjust the programmed tool path automatically to compensate for the non-negligible width of the cutter. While this works the cutter compensation is best carried in the CAM software that you should be using to create the G-code program.

When you use the G-code cutter radius compensation you program your tool paths as if you were cutting the outline of the part with an infinitely thin cutter i.e. the XYZ coordinates specify the outline of the part to be cut.

Of course the real cutter has a substantial size so you need to tell EazyCNC what is the cutter diameter and on which side the part outline the cutter should cut.

Use `G41` to indicate that the cutter should stay to left of the toolpath, use this if you are cutting the outline of a part clockwise (or if you are cutting a hole counter clockwise).

Use `G42` to indicate that the cutter should stay to the right.

Left and right are defined as if you were riding on the cutter and facing the direction of the travel.

Note that when you turn on the cutter compensation it will only affect the next movement of the tool ie the tool does *not* move from where it is when you turn on the compensation, rather the next position will be offset by the tool radius and the next move will then move from the current position to that compensated position. Therefore you should always plan a move-in movement when you turn on the compensation.

To turn off the compensation use `G40`.

There are several ways you can specify the tool/cutter radius. Following shows them for G41 but G42 works just the same, only the compensation is taken to the right.

To turn on the compensation to the left and use the tool/cutter diameter stored in the tool table for the currently selected tool (as specified with the T -word) use G41.

To turn on the compensation to the left and use the tool/cutter diameter stored in the tool table use G41 D#, where the D word specifies the number of the tool in the tool table.

To explicitly set the compensation to the left with given amount of compensation use G41 P#, where the P word specifies the *radius* of the cutter.

### 9.4.12 Feedrate mode – G93,G94,G95 codes

The EazyCNC does *not* support the inverse time feedrate mode G93 *nor* the units per ref feedrate mode G95.

The G94 code programs the feed per minute mode in which axes movements are carried out so that the that tool moves at the rate specified with the F-word inches or millimeters per minute depending on which length unit mode, G20 or G21, is in effect.

It is an error to use G93 or G95 code.

G40*nor*

### 9.4.13 Feedrate override on/off – M48,M49 codes

The feed override that the machine operator can adjust during machining, see Section 7.13, can be turned on or off in the G-code program, but programming it in G-code does not prevent the operator form turning it on and off again.

The idea is that the G-code programmer knows the best what at what feedrate the part should be cut but sometimes it is necessary to adjust or fine tune that while machining.

The actually override percentage *cannot* be set with G-codes.

The M48 code turns the feed override on.

The M49 code turns the feed override off.

It is an error to have both M48 and M49 codes on the same line.

### 9.4.14 Tool change – M6 code

EazyCNC does not support the M6 tool change command but ignores it.

### 9.4.15 Tool length compensation – G43,G44,G49 codes

The length compensation basically works by just offsetting the Z-axis value so that there will be room for the specified tool length between the spindle/chuck and the work piece.

To turn off the tool length compensation program `G49`.

To turn on and set the tool length compensations from the length stored in the tool table use `G43 H#`, where the `H`-word specifies the number of the tool in the tool table. If the `H`-word is zero i.e. specifies no tool number then compensation is set to zero and effectively turned off, in fact this equivalent to programming `G49`.

The `G44` works the same as `G43` but expects that the length values in the tool table are negative. Since you can't enter negative values this is provided for compatibility with existing practice only.

It is an error if the `H`-word value is negative or larger than the number of tool supported.

It is an error to have more than one of these G-codes on the same line.

### 9.4.16 Path mode – G61,G61.1,G64 codes

In any CNC system there are basically two options how the system tries to follow the specified tool path. Either the system tries to obey the specified coordinates i.e. position or the specified speed. You can't have both at the same time, think about it: if you need to move from point A to point B at a given speed you would need infinite acceleration and deceleration at the beginning and end of travel.

The path mode along with the machine limits, Section 6.7, determine how EazyCNC calculates the actual tool path.

The set exact path mode program `G61`, in this mode the path follows the specified path as closely as possible which results in the axes velocities coming to a complete stop at the end of movement. This is fine when milling but slows down the machining, especially if a lot of small cuts are used. When cutting with a torch stopping at the end of the movements causes local 'burn outs' so this is not an acceptable mode for plasma cutting.

The set best speed mode program `G64`, in this mode the path tries still to follow the specified path but is allowed to deviate from it by as much as the 'Path tolerance', Section 6.7.3, allows trading accuracy for speed. Because of limited path lookahead many small cuts in a row still result in a severely limited speed, so when programming tool paths for plasma try to avoid small movements.

For all practical purposes `G61.1` performs the same as `G61` and this is supported for compatibility only.

It is an error to have more than one of these commands in the same line.

### 9.4.17 Incremental XYZ mode – G90,G91 codes

The axis words, '`X`','`Y`', '`Z`' etc can be interpreted either as coordinates or as a change of coordinates relative to the previous coordinates.

To treat coordinates as 'absolute' positions in the local/current coordinate system program `G90`. This is the usual way to specify coordinates in G-code programs.

To turn on the incremental interpretation program `G91`, in this mode the axis word values are treated as increments to the previous axis word values.

It is an error to have both of these codes in the same line.

### 9.4.18 Incremental IJK mode – G90.1,G91.1 codes

Interpretation of the IJK values in the arc interpolation codes `G2` and `G3` can be either absolute or incremental.

To set the absolute interpretation use `G90.1`, in this mode the IJK words specify the center coordinates in the local/current coordinate system.

To set the incremental interpretation use `G91.1`, in this mode the IJK words specify the center coordinates relative to the starting point of the arc.

Incorrect settings of this mode will usually result in large and incorrectly oriented arcs in the toolpath display.

You can also set this mode in the Mach Setup screen, see Section 6.8.1.

It is an error to have both of these codes in the same line.

### 9.4.19 Set tool table – G10 L1 code

It is possible change tool table entries with G-code commands. This makes it possible to maintain different tool sets.

`G10 L1 P# A# Z# X#` sets the tool table entry for tool number specified by the `P`-word. The `Z`-word sets the tool height and the `X`-word sets the tool *radius*; the `A`-word, tool tip radius, is ignored but accepted for compatibility reasons.

The `A`,`Z` and `X` -words are all optional and it is ok to leave any or all of them out.

It is an error if the P word is missing, smaller than one or larger than the number of tools EazyCNC supports.

It is an error if the L word is missing.

### 9.4.20 Set work/fixture offsets – G10 L2 code

It is possible to set the work/fixture offsets in G-code. This makes it possible to maintain multiple different jig set ups easily.

`G10 L2 P# X# Y# Z# A# B# C#` sets the work/fixture offsets for the fixture number specified with the `P` word. The axis words `X`,`Y` etc specify the corresponding offsets. It is ok not specify all or none of the axis and those that are not specified are left untouched.

It is an error if the P word is missing, smaller than one or larger than the number of work/fixture offsets EazyCNC supports.

It is an error if the L word is missing.

## 9.5 Canned Drilling Cycles

Canned drilling cycles (G73,G81,G82,G83 and G85) all work more or less the same with detail variations to help with real world drilling issues such chip breaking etc.

The canned cycles are modal i.e. once a command is given it causes repeated execution of the cycle for each line that specifies an X/Y position until the mode is cancelled with `'G80'` or some other modal movement command.

In the following in the interest of simplicity canned cycles are described as working in the XY plane with the drilling happening Z directions. However they work equally well in any other plane (`G17,G18,G19`).

Note that in the following desctiption when I talk about 'depth' it really refers to a Z-position, not depth as a you as a machining would define the depth of a hole.

All canned drilling commands accept an optional repeat count specified with the L-word.

At first sight this seem silly, why would you drill the same hole multiple times? Of course you don't but when combined with incremental distance mode (`'G91'`) it allows simple commands to produce reqular lines or arcs of holes.

If you plan to use the incremental distance mode note that the first hole is drilled after the increment has been applied so it does not end up where you position the tool.

Interweb has a lot of great illustrations about canned cycles so I will leave a more graphical presentation to more capable hands.

It is an error if tool radius compensation is on when a canned cycle command is given.

It is an error if rectraction level R is greater than initial Z-level.

It is an error if hole bottom level is higher than the rectraction plane R.

Below is a simplified description how the different canned cycles differ from each other.

G81 - Drilling

This is the mother of all drilling cycles, a single peck move down to the desired depth at feedrate and rapid retract.

G83 - Peck Drilling

Same as G81 but makes a multiple increasingly deeper pecks at feedrate to drill a deep hole with full rapid retraction between pecks to clear the hole from chips.

G73 - High Speed Peck Drilling

Same as G83 but to speed things up the retraction is shorter as the retraction is for just breaking the chip.

G82 - Spot Facing

Same as G81 drilling but the tool dwells at the bottom of the hole for a specified time before retracting.

G85 - Boring

Same as G81 drilling but the retraction as well as the drilling is done at feedrate rather than as a rapid motion.

### 9.5.1 Cancel Canned Cycle – G80 code

The command `'G80'` cancels canned cycle modal mode.

### 9.5.2 Canned Cycle Return level – G98,G99 codes

After each hole the drill returns to either to the Z-position it was before the cycle started or to the position set by the R-word.

Use `'G98'` to cause the tool to return to the original Z-position after the cycle.

Use `'G99'` to cause the tool to return to level set by the R-word.

### 9.5.3 High Speed Peck Drilling – G73 code

`G73 X# Y# Z# R* Q* F# L#` command is used to drill a hole to depth set by the Z-word in location specified with the X and Y -words with pecks of distance set by the Q-word and rectracting the drill between pecks to depth set by the R-word.

Pecking is performed at feedrate set by the F-word, retraction happens at maximum velocity.

An optional repeat count can be given with the L-word.

### 9.5.4 Drilling – G81 code

`G81 X# Y# Z# R* F# L#` command is used to drill a hole to depth set by the Z-word in location specified with the X and Y -words with single peck and then retract.

Pecking is performed at feedrate set by the F-word, retraction alway happens at maximum velocity.

An optional repeat count can be given with the L-word.

### 9.5.5 Spot Facing – G82 code

`G82 X# Y# Z# R* F# P# L#` command is used to drill a hole to depth set by the Z-word in location specified with the X and Y -words with single peck and then retract.

Pecking is performed at feedrate set by the F-word, retraction always happens at maximum velocity.

The drill will dwell at bottom of the hole a for time set by the P-word if given.

An optional repeat count can be given with the L-word.

### 9.5.6 Peck Drilling – G83 code

`G83 X# Y# Z# R* Q* F# L#` command is used to drill a deep hole to depth set by the Z-word in location specified with the X and Y -words with pecks of distance set by the Q-word and rectracting the drill fully between pecks to depth set by the R-word.

Pecking is performed at feedrate set by the F-word, retraction always happens at maximum velocity.

An optional repeat count can be given with the L-word.

### 9.5.7 Boring – G85 code

`G85 X# Y# Z# R* Q* F# L#` command is used to drill a hole to depth set by the Z-word in location specified with the X and Y -words with single move and then retract.

Both drilling and retraction is performed at feedrate set by the F-word.

An optional repeat count can be given with the L-word.

## 9.6 Using subroutines – M98/M99

### 9.6.1 Call subroutine – M98 code

It is possible to create subroutines in a G-code program.

A subroutine is any continuous sequence of G-code lines that ends with the `M99` G-code.

The first line of subroutine should contain the `O`-word, to give the subroutine an id-number. When calling the subroutine with the `M98` code a `P` word with the same id-number must be used.

Typically subroutines are placed at the end of a G-code file after a M2, M30 or M99 code so that the normal G-code execution does not reach them.

Subroutines can be nested.

A subroutine can be in the same file (recommend) where the calling code is or in a separate file.

An optional repeat count can be specified with the `Q` or `L` words.

To call a subroutine use one of the following forms.

`M98 P1234` calls a subroutine that starts with `O1234` in the same file as the calling M98 code.

`M98 P1234 (example.txt)` calls a subroutine that starts with `O1234` in a separate file called 'example.txt' .

`M98 P2000 L10` calls ten times a subroutine that starts with `O2000` in the same file as the calling M98 code.

It is an error if the P word is missing.

It is an error if the both L and Q words are used.

It is an error if the L or Q word specifies a count smaller than one.

It is an error if

### 9.6.2 End of subroutine – M99 code

`M99` returns G-code execution to the place whence the subroutine was called from.

It is an error this code is executed when no subroutine (M98) has been called.

## .1 Appendices

# Appendix A

# Revision History

EazyCNC 2.0.41 Revision 1

Manual updated for EazyCNC 2.0.41

EazyCNC 2.0.36x1 Revision 1

Following subjects have been updated/added to reflect the changes in the software.

Jogging using the new step jog modes Using electronic probe to set Work Offsets Using auto tool setter to set Tool Length Using an Manual Pulse Generator / CNC Pendant List of supported G-codes (Thank you Erik!) WHB04 Manual Pulse Generator / Pendant reference WHB04B Manual Pulse Generator /Pendant reference

EazyCNC 2.0.18-X Revision 1

Added description of canned cycles. Added description of user function.

EazyCNC 2.0.0.X Revision 1

Updated too many things to list here - mainly related to the Axis Setup and Mach Setup screens.

–

EazyCNC 0.0.0.26a Revision 2

Added this revision history to this manual.

In section improved the description of problems associated with the virtual serial port.

Added section 6.1.1 on how to set up the permission for the virtual serial port in Linux.

In section 7.7 added references to the corresponding UI setup sections.

Add description of the ABCXYZ button

Add description of the missing jog button

# Appendix B

# Supported G-codes

Table B.1: A simple longtable example

| Code | Description | Modal | Section | Page |
|---|---|---|---|---|
| G0 | Rapid position | Y | 9.3.12 | 105 |
| G1 | Linear move | Y | 9.3.13 | 105 |
| G2 | Clockwise Arc | Y | 9.3.14 | 105 |
| G3 | Counter clockwise Arc | Y | 9.3.15 | 106 |
| G4 | Dwelling | N | 9.3.9 | 103 |
| G9 | Exact stop | N | - | - |
| G10 L1 | Set Tool table | N | 9.4.19 | 114 |
| G10 L2 | Set work/fixture offsets | N | 9.4.20 | 114 |
| G12 | Clockwise Circle | Y | - | - |
| G13 | Counter Clockwise Arc | Y | - | - |
| G15 | Turn off Polar mode | Y | 9.4.3 | 108 |
| G16 | Turn on polar mode | Y | 9.4.3 | 108 |
| G17 | Set active plane to XY | Y | 9.4.7 | 110 |
| G18 | Set active plane to XZ | Y | 9.4.7 | 110 |
| G19 | set active plane to YZ | Y | 9.4.7 | 110 |
| G20 | Set imperial(inch) mode | Y | 9.3.3 | 102 |
| G21 | Set metric (mm) mode | Y | 9.3.3 | 102 |
| G28 | Return zero | N | - | - |
| G30 | 2nd, 3rd, 4th Zero Return | N | - | - |
| G31 | Perform probing move | N | 9.3.16 | 106 |
| G32 | Treading | N | - | - |
| G38.2 | Probing | N | - | - |
| G40 | Turn radius compensation Off | Y | 9.4.11 | 111 |
| G41 | Radius compensation to left | Y | 9.4.11 | 111 |
| G42 | Radius compensation to right | Y | 9.4.11 | 111 |
| G43 | Set Tool length compensation | Y | 9.4.15 | 113 |
| G44 | Set Tool length compensation | Y | 9.4.15 | 113 |
| G49 | Cancel Tool length compensation | Y | 9.4.15 | 113 |
| G50 | Turn OFF scaling | Y | 9.4.1 | 107 |
| G51 | Turn ON scaling | Y | 9.4.1 | 107 |
| G52 | Temporary system Offset | Y | 9.4.4 | 109 |
| G53 | Absolute coordinates | N | 9.4.10 | 111 |

*Continued on next page*

Table B.1 – *Continued from previous page*

| Code | Description | Modal | Section | Page |
|---|---|---|---|---|
| G54 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G55 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G56 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G57 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G58 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G59 | Work/Fixtures offsets | Y | 9.4.9 | 111 |
| G60 | Unidirection approach | N | - | - |
| G61 | Exact stop mode | Y | 9.4.16 | 113 |
| G61.1 | Path mode | Y | 9.4.16 | 113 |
| G64 | Constant velocity mode | Y | 9.4.16 | 113 |
| G65 | Macro call | N | - | - |
| G66 | Macro modal call | Y | - | - |
| G67 | Cancel modal macro call | Y | - | - |
| G68 | Set Coord. system rotation | Y | 9.4.6 | 109 |
| G69 | Turn off coord. system rotation | Y | 9.4.6 | 109 |
| G73 | High speed peck drilling | Y | 9.5.3 | 116 |
| G74 | Reverse tapping | Y | - | - |
| G76 | Fine boring | Y | - | - |
| G80 | Cancel Canned Cycle | Y | 9.5.1 | 116 |
| G81 | Driling | Y | 9.5.4 | 116 |
| G82 | Spot facing | Y | 9.5.5 | 117 |
| G83 | Deeep Hole Peck drilling | Y | 9.5.6 | 117 |
| G84 | Tapping | Y | - | - |
| G85 | Boring,retract at feed, spindle on | Y | 9.5.7 | 117 |
| G86 | Boring, retract at rapid, spindle off | Y | - | - |
| G87 | Back boring | Y | - | - |
| G88 | Boring, manual retract | Y | - | - |
| G89 | Boring ,dwell,retract at feed, spindle on | Y | - | - |
| G90 | Incremental coord. mode OFF | Y | 9.4.17 | 113 |
| G91 | Incremental coord. Mode ON | Y | 9.4.17 | 113 |
| G90.1 | Absolute/Incremental IJK mode | Y | 9.4.18 | 114 |
| G91.1 | Absolute/Incremental IJK mode | Y | 9.4.18 | 114 |
| G92 | Local Coord. System setting | Y | 9.4.5 | 109 |
| G92.1 | Temporary systems offsets | Y | 9.4.5 | 109 |
| G92.2 | Clear Temporary systems offsets | Y | 9.4.5 | 109 |
| G92.3 | Restore Temporary systems offsets | Y | 9.4.5 | 109 |
| G93 | Inverse time feed | Y | - | - |
| G94 | Feed per minute | Y | 9.4.12 | 112 |
| G95 | Feed per revolution | Y | - | - |
| G96 | Constant surface speed | Y | - | - |
| G97 | Constant RPM | Y | - | - |
| G98 | Canned cycle, initial return point | Y | 9.5.2 | 116 |
| G99 | Canned cycles R-point return | Y | 9.5.2 | 116 |
| M0 | Mandatory Program stop | | 9.3.17 | 106 |
| M1 | Optional Program stop | | 9.3.17 | 106 |
| M2 | Program end | | 9.3.18 | 107 |
| M6 | Tool change (Not used) | | - | - |
| M7 | Mist cooling | | 9.3.7 | 103 |
| M8 | Flood cooling | | 9.3.7 | 103 |

*Continued on next page*

Table B.1 – *Continued from previous page*

| Code | Description | Modal | Section | Page |
|---|---|---|---|---|
| M9 | Stop cooling | | 9.3.7 | 103 |
| M48 | Feed override on | | 9.4.13 | 112 |
| M49 | Feed override Off | | 9.4.13 | 112 |
| M98 | Call subroutine | | 9.6.1 | 117 |
| M99 | End of subroutine | | 9.6.2 | 118 |

Appendix C

# Supported MPG pendants

## C.1 XHC WHB04 Pendant/MPG

EazyCNC supports a commercially available MPG named WHB04, see Figure C.1. This is a wireless model, a sister model called HB04 with wired USB connection is also available and it is supposed to be 100% compatible but this has not been tested.

The WHB04 pendant is not particularly excellent, the wheel sometimes misses pulses, the display update speed is nothing to write home about and the wireless link exhibits a glitch every now and then. But it is what we have at the moment.

Note worthy is also that WHB04 has a battery saving mode that kicks in in a about thirty seconds and stops the display from updating unless you wake the pendant up by pressing a button, turning a knob or moving the wheel.

Having said all that, it is still a useful device.


Figure C.1: WHB04 Pendant/MPG

### C.1.1 WHB04 Controls

Figure C.2 shows a close up of the WHB keypad.

The two main controls of the MPG are the small axis selector knob you can see in the closeup and the large pulse wheel.

In general the axis selector selects which axis or other parameter is affected by the pulse wheel or buttons on the keypad.

For some functions setting the selector knob to 'Off' will make those functions apply to all axes at once.

### C.1.2 WHB04 Display

Figure C.3 shows a close up of the WHB04 display.


Figure C.2: WHB03 Keypad

The display has DROs for three axes at a time. If the axis selector is in X,Y or Z position then the corresponding axis values are displayed in the DROs. If the selector is in A position then axis positions of the A, B and C axis are displayed.

The DROs in the 'MC' column are not used (to avoid confusion) and always display'0.000'

The DROs in the 'WC' column display the same DRO values as on the EazyCNC computer screen.

Note that you can NOT see more decimals on the pendant DRO than on the computer screen eventhough the pendant always displays three digits.

Next to the WC text to the right an 'inch' or 'mm' sign is show and reflects weather the DRO values and wheel operate in inch or mm mode. This is controlled by the units selected for the DROs in EazyCNC.

Figure C.3: WHB04 Display

### C.1.3 WHB04 Wheel

Every 'click' of the wheel moves the selected axis one step to the wheel direction. The speed at which the mill table,head or plasma torch moves is relative to how fast you turn the wheel.

In principle the wheel allows absolute (well incremental really) control of the axis position and speed. In practice the wheel sometimes misses pulses so turning the wheel ten clicks may only result in nine steps, always confirm by looking at the DROs.

Because it is possible to turn the wheel faster than the axis can move there is a windup prevention that prevents the wheel position from advancing too much beyond where the axis has advanced.

Because of the long chain of hardware and software that connects the pulse wheel to the axis stepper motor, the feel of control you get with the wheel is far from perfect. However it does allow you to control the axis position rather swiftly and accurately.

The 'step size' i.e. how much every wheel click moves milling table is displayed to the left of the 'WC' text. It is expressed as number of 1/1000th of the units selected for the DROs. Very much like the jogging step selection but separate from that.

In 'mm' mode the displayed text and step size are as follows:

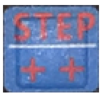
| | |
|---|---|
| x1000 | 1.000 mm |
| x100 | 0.100 mm |
| x10 | 0.010 mm |
| x0 | 1 stepper step |

In 'inch' mode the displayed text and step size are as follows:

| | |
|---|---|
| x100 | 0.100 inch |
| x10 | 0.010 inch |
| x1 | 0.001 inch |
| x0 | 1 stepper step |

You change the step size with the 'STEP++' key, see below.

### C.1.4 Step++ -key




Pressing this key shortly will advance the step size to the next *smaller* size. After 'x0' the step size reverts to the largest steps size ('x1000' for 'mm mode and 'x100' for 'inch' mode).

A long press (over 1 seconds) will revert the step size tolargest steps size.

Whenever the largest step size is selected a long 'beep' sound is emitted from the computer to alert the user.

### C.1.5 Probe XY -key

Pressing this key has the same effect as pressing the -X or -Y Touch button in the Work Offsets screen when the 'Use PROBE to Touch' feature is enabled.

In other word this will perform a short probing move on the selected axis and as soon as the probe trips the movement stops and rectracts and the selected axis work offset is set to the negative half value of the Probe Diameter parameter in the Work Offsets screen.

Thus this is a handy way to perform edge finding using a probe and the MPG.

Use the axis selector knog to select either X or Y axis. You can only probe on the left/front side of the work piece with this key.

### C.1.6 'Probe Z' -key


Pressing this key has the same effect as pressing the Touch button in the Work Offsets / Set Z origin screen when the 'Use PROBE to Touch' feature is enabled.

In other word this will perform a short probing move on the Z axis axis and as soon as the probe trips the movement stops and rectracts and the Z axis work offset is set to the Gage Height parameter in the Work Offsets screen.

Thus this is a handy way to perform 'zero' the Z-axis with the probe.

### C.1.7 Spindle -key


Pressing this key has the same effect as pressing the SPINDLE button on the computer screen.

In other words it toggles the spindle on and off.

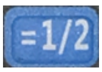
### C.1.8 Start/Pause -key


This key has the same effect as pressing alternatively the RUN and HOLD buttons on the computer screen.

### C.1.9 Stop -Key


This key has the same effect as pressing the STOP button on the computer screen.

### C.1.10 '=1/2' -key




Pressing this key has the same effect as pressing the Touch -X or Touch -Y button in the Work Offsets screen when the 'Use PROBE to Touch' feature is NOT enabled.

In other words the selected axis Work Offset is set to the negative half value of the Probe Diameter parameter in the Work Offsets screen.

Thus this is a handy way to perform edge finding manually with an edge finder and the MPG.

Use the axis selector knog to select either X or Y axis. You can only Touch on the left/front side of the work piece with this key.

### C.1.11 'Goto Origin' -key

Pressing this key will cause the X and Y axis to jog to the zero position of the DROs.

### C.1.12 '=0' -key


Pressing this key has the same effect as pressing the ZERO button in the selected axis.

Use the axis selector knob to select the affected axis, or set the axis selector knob to 'Off' to ZERO all axis.

### C.1.13 Safe Z -key


Pressing this key will cause the Z-axis to move to the Safe Z value set in the Axis Setup / Axis Z.

### C.1.14 Reset -key


Pressing this has the same effect as pressing HOME button on the computer screen.

Use the axis selector knob to select the affected axis, or set the axis selector knob to 'Off' to HOME ALL axis.

A long press will force HOME ALL action regardless of the axis selector knob position.

### C.1.15 Rewind -key


Pressing this key will change the currently selected tool to the next tool number, in other words this has the same effect as L-word in G-code. Note that it does not actually cause any tool change but does effect the tool offset, if it is enabled with G43 code.

Long press will reset the current tool to tool number 1.

Whenever the tool number1 is selected a long 'beep' sound is emitted from the computer to alert the user.

This key is mainly intended to help setting up tool length without touching the computer keyboard.

## C.2 XHC WHB04B Pendant / MPG


Figure C.4: WHB04B Pendant / MPG

EazyCNC supports a commercially available MPG named WHB04B (note the B at then end of the type code), see Figure C.4. This actually comes in two varieties, WHB04B-4 for use with four axis and WHB04B-6 for use with six axis.

Out of the box this pendant (or rather EazyCNC) support a much smaller set of functionality than for example the WHB04. This is because the pendant has a smaller number of buttons and even smaller number of usefully labeled buttons.

### C.2.1 WHB04B Controls

The three main controls of the MPG are the axis selector knob, step size selector knob and the large pulse wheel.

In general the axis selector selects which axis is affected by the pulse wheel or buttons on the keypad.

The step size selector change how fast the wheel moves the axis.

The keypad is used to activate EazyCNC functions.

### C.2.2 Display

Figure C.5 shows a close up of the WHB04B display.

The display has DROs for three axes at a time. If the axis selector is in X,Y or Z position then the corresponding axis values are displayed in the DROs. If the selector is in A position then axis positions of the A, B and C axis are displayed.

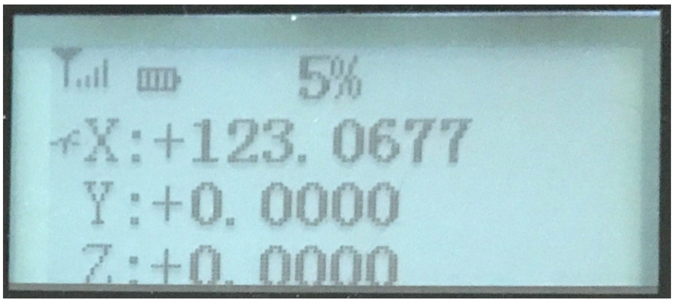



Figure C.5: WHB04B Display

An asterisk ('*') in front of an axis letter indicates that that axis has is currently selected by the Axis Selector knob.

The DROs display the same DRO values as on the EazyCNC computer screen.

Note that you can NOT see more decimals on the pendant DRO than on the computer screen eventhough the pendant always displays four decimal digits.

Note that if the Axis Selector is in the OFF position then EazyCNC is not able to update the display at all.

On the top row alternatively eithe step selector percentage is displayed or feed ('F') or spindle ('S') is displayed. The percentage makes little sense but can be used as a reminder of the selected wheel step size.

To display the feed or spindle speed you need to press the Feed +/- or Spindle +/- key. If the corrosponding feed or spindle speed is not currently displaying when you press the key then the display will switch and the key is ignored, i.e. it will not change the feed or spindle speed.

When the pendant is powered up you may see the text 'RESET' on the top row. To get rid of that you have to press any button on the key pad and rotate the step selector knob. This is makes no sense but it is what it is and EazyCNC cannot do anything about that.

### C.2.3 Keypad

Figure C.6 shows a close up of the WHB04B keypad.

The keypad has 15 keys plus the orange 'Fn' key which alter the functions of those keys that have an orange label on them. To activate the 'orange' function you need to hold the 'Fn' key down while pressing the intended key.

Note that the functions associated with the keys are not fixed and you can change them in the Mach Setup / Shortcuts screen, see section 6.9,

For inspiration of functions you could assign to the keys I suggest reading the WHB04 appendix which lists number of useful functions that are available out-of-the-box with that pendant.


Figure C.6: WHB04B Keypad


Figure C.7: WHB04B Axis Selector

### C.2.4 Axis Selector

### C.2.5 WHB04 Wheel

Every 'click' of the wheel moves the selected axis one step to the wheel direction. The speed at which the mill table,head or plasma torch moves is relative to how fast you turn the wheel.

In principle the wheel allows absolute (well incremental really) control of the axis position and speed. In practice the wheel sometimes misses pulses so turning the wheel ten clicks may only result in nine steps, always confirm by looking at the DROs.

Because it is possible to turn the wheel faster than the axis can move there is a windup prevention that prevents the wheel position from advancing too much beyond where the axis has advanced.

Because of the long chain of hardware and software that connects the pulse wheel to the axis stepper motor, the feel of control you get with the wheel is far from perfect. However it does allow you to control the axis position rather swiftly and accurately.

The step size that is used when you turn the wheel depends on the Step Selector, see below.

### C.2.6 Step Selector

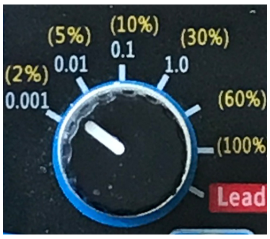



Figure C.8: WHB04B Step Selector

The Step Selector has some confusing labeling, you should go by the labels in white.

Depending on weather you are working with inches or millimeters (see Mach Setup / Screen / Units) the wheel click/step size varies as shown in the table below.

| Selector position | Units = mm | Units = inch |
|---:|---|---|
| 0.001 / 2% | 0.001 mm | 0.001" (0.0254 mm) |
| 0.01 / 5% | 0.01 mm | 0.001" (0.254 mm) |
| 0.1 / 10% | 0.1 mm | 0.001" (2.54 mm) |
| 1.0 / 30% | 1.0 mm | minimal |
| LEAD% | minimal | minimal |

In above 'minimal' indicates smallest possible step, for example if in your Axis Setup / Step/mm you have 400 step/mm then the minimal step size is 1/400 mm i.e. 0.0025 mm.

Note that if theoretical step size as per above table results in a step that is smaller than 'minimal' then the minimal step is selected.

For safety the "1.0" selector position does not result in one inch step size in inch mode.

### C.2.7 Reset -Key


Pressing this has the same effect as pressing HOME button on the computer screen.

Use the axis selector knob to select the affected axis, or set the axis selector knob to 'Off' to HOME ALL axis.

A long press will force HOME ALL action regardless of the axis selector knob position.

### C.2.8 Stop -Key


This key has the same effect as pressing the STOP button on the computer screen.

### C.2.9 Start / Run -key


This key has the same effect as pressing alternatively the RUN and HOLD buttons on the computer screen.

### C.2.10 Macro-1 [Feed+]-key


If the 'F' is not displayed on the pendant display then the key press is ignored and the display changes to display the feed speed.

When pressed together with the 'Fn' key this button has the same effect as pressing the '+' key in the Feed/Override panel on the screen.

When pressed without the 'Fn' key this key has no function unless you assign one to it.

### C.2.11 Macro-2 [Feed-]-key


If the 'F' is not displayed on the pendant display then the key press is ignored and the display changes to display the feed speed.

When pressed together with the 'Fn' key this button has the same effect as pressing the '-' key in the Feed/Override panel on the screen.

When pressed without the 'Fn' key this key has no function unless you assign one to it.

### C.2.12 Macro-3 [Spindle+] -key


If the 'S' is not displayed on the pendant display then the key press is ignored and the display changes to display the spindle speed.

When pressed together with the 'Fn' key this button has the same effect as pressing the '+' key in the Spindle panel on the screen.

When pressed without the 'Fn' key this key has no function unless you assign one to it.

### C.2.13 Macro-4 [Spindle-]-key


If the 'S' is not displayed on the pendant display then the key press is ignored and the display changes to display the spindle speed.

When pressed together with the 'Fn' key this button has the same effect as pressing the '-' key in the Spindle panel on the screen.

When pressed without the 'Fn' key this key has no function unless you assign one to it.

### C.2.14 Macro-5 [M-HOME]-key


This key has no function unless you assign one to it.

### C.2.15 Macro-6 [Safe-Z]-key


Pressing this key will cause the Z-axis to move to the Safe Z value set in the Axis Setup / Axis Z.

### C.2.16 Macro-7 [W-HOME]-key


Pressing this key will cause the X and Y axis to jog to the zero position of the DROs.

### C.2.17 Macro-8 [S-ON/OFF]-key


Pressing this key has the same effect as pressing the SPINDLE button on the computer screen.

In other words it toggles the spindle on and off.

### C.2.18 Fn -key


This key together with some other is used to evoke the secondary function of that key. To use that press and hold the 'Fn' key down while pressing the other key.

### C.2.19 Macro-9 [Probe-Z]-key


Pressing this key has the same effect as pressing the Touch button in the Work Offsets / Set Z origin screen when the 'Use PROBE to Touch' feature is enabled.

In other word this will perform a short probing move on the Z axis axis and as soon as the probe trips the movement stops and rectracts and the Z axis work offset is set to the Gage Height parameter in the Work Offsets screen.

Thus this is a handy way to perform 'zero' the Z-axis with the probe.

### C.2.20 Macro-10


This key has no function unless you assign one to it.

### C.2.21 Continuous -key


This key has no function unless you assign one to it.

### C.2.22 Step -key


This key has no function unless you assign one to it.

*** The End ***