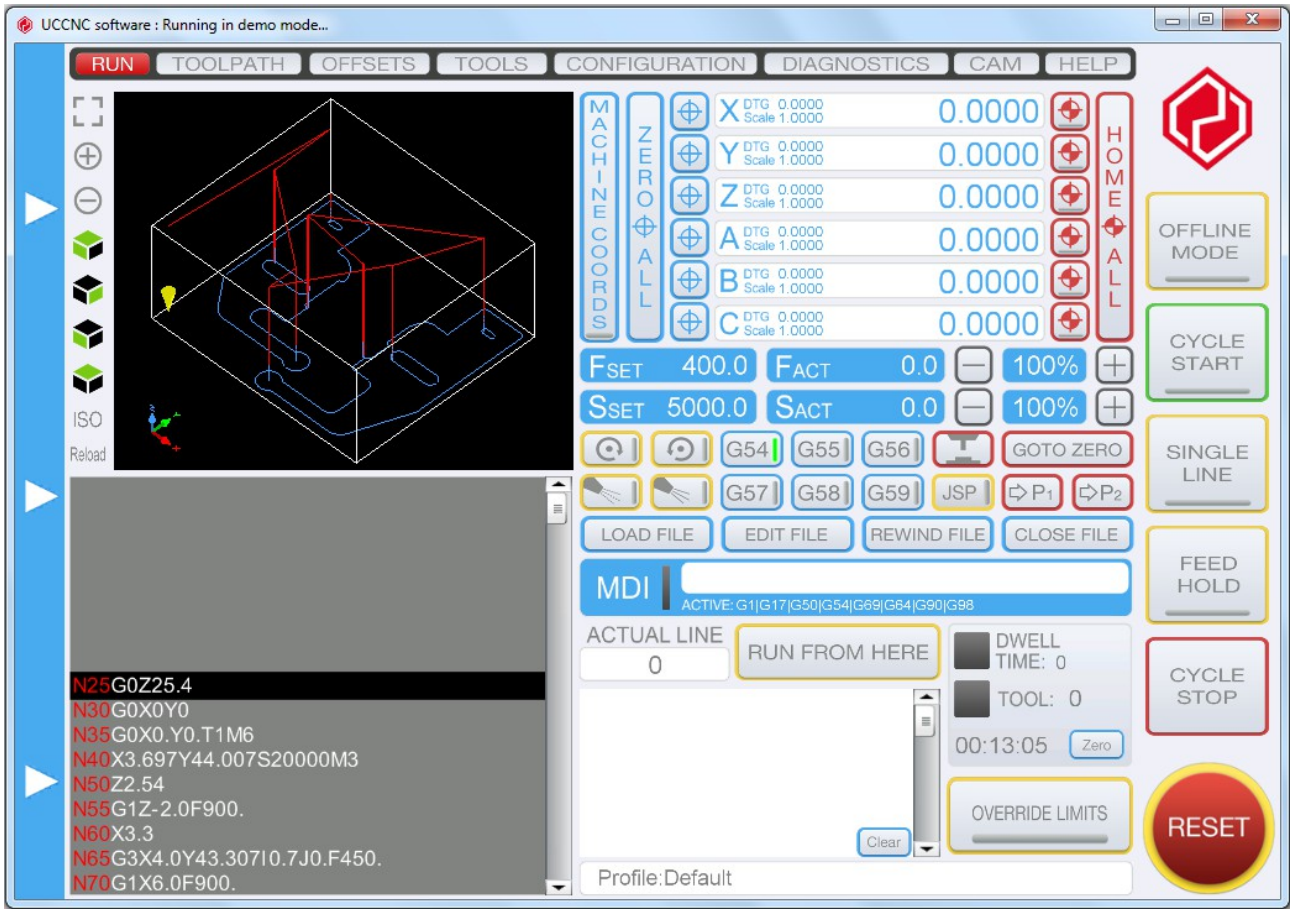
# **UCCNC software** installation and user's guide




Version of this software manual: 1.0046
Software version: 1.2111

Supported motion controllers:

- UC100
- UC300 – 5LPT
- UC300 – M44
- UC300 - 5441
- UC400ETH
- AXBB
- UC300ETH – 5LPT
- UC300ETH – M44
- UC300ETH - 5441
- UC300ETH - M45
- UC300ETH - UB1

## Contents

# 1 .Installation of the software

## 1.1 .Introduction

For first of all we thank you for your interest in our software product and reading this user's guide. This first section of the manual will describe the basic purpose of this software and what hardware you will need and may need to you use it.

The UCCNC is a machine control software. It uses and external hardware to generate signals to produce coordinated motion on upto 6 machine axis. The external hardware is a motion controller device which is currently our UC100 or UC300 or UC400ETH motion controller.

This software connects to the external motion controller via a single USB port connection or ethernet connection (depends on the motion controller model) of a personal computer (PC) and via a software application interface (API) which is built into the software. The installer of the UCCNC software includes and completes all the tasks to make it possible to use this software on your computer. The device drivers are included in the installer too.

The UC100 motion controller and this software is not capable to run motors and machine axis directly, it is capable only to produce the required signals to run a machine like a CNC mill which requires programmed coordinated motions. The possibility and the selection on the external motor controller is nearly endless and there are lots of options and choices on the stepper and servo motor controller market. We also offering a few options and most of the 3rd party hardware available on the market today will also work. To get a more clear idea about how this software works and what else electronics you may need to get your machine running please read this user's guide further and make your own resource on the internet, there are lots of useful informations and articles available

about motion and motor control electronics.

## 1.2 .Safety notes

Moving objects like machine axis and machine parts can be dangerous and could cause personal injury or even death. Please keep this always in mind and operate your machine carefully keeping all safety standards and regulations first.

## 1.3 .Hardware requirements

This software requires Microsoft Windows XP, 7 , 8, 8.1 or 10 Operating System running on an x86 or x64 desktop or laptop or tablet computer.

The minimal hardware requirements for the computer are as follows:

- CPU frequency: minimum 1.8 GHz (duo or dual core is recommended.)
- Graphics card: OpenGL 1.3 or higher compatible
- RAM: minimum 1GB for XP and 2GB for all other supported OS.
- Hard drive space: minimum 16GB

The above are the recommended minimal values, however the software may run on PCs with lower recources, but we do not advice to do so.
If large G-code files with the hundreds of thousands or millions of code lines count are run then the requirements may be higher.
We suggest to always try the software even in demo mode on the planned to be used computer with the largest and most complex planned to be run g-code files to see the performance and to see if the computer fits the software requirements.

## 1.4 .Software prerequirements

The software targets the .NET framework 4.0. The .NET framework 4.0 is Microsoft's runtime environment for applications (like the UCCNC) developed in Visual Studio.
Starting from .NET 4.0 the frameworks are backwards compatible which means that installing a higher .NET framework version than 4.0 (for example .NET 4.5) will also run the UCCNC software.

On Windows XP operating system the .NET framework 4 can be installed from a file which is available on Microsoft's website: https://www.microsoft.com/en-us/download/confirmation.aspx?id=17718 ,but the Operating System must have the SP3 service pack installed, without that the .NET framework 4.0 can't be installed.

On Windows 7. the .NET framework 4 can be installed from file which is available on Microsoft's website: https://www.microsoft.com/en-us/download/confirmation.aspx?id=17718 ,but the Operating System must have the SP1 service pack installed, without that the .NET framework 4.0 can't be installed.

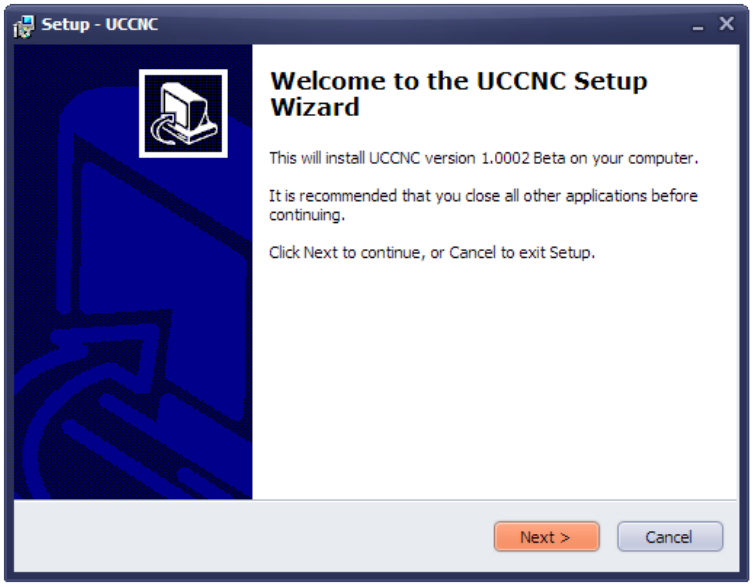
On Windows 8.,8.1 and 10 the .NET 4 framework is part of the operating system, installation is not required.

If the .Net framework is not installed, the UCCNC software installer will not run, it will drop an error message on startup and the installer will exit.

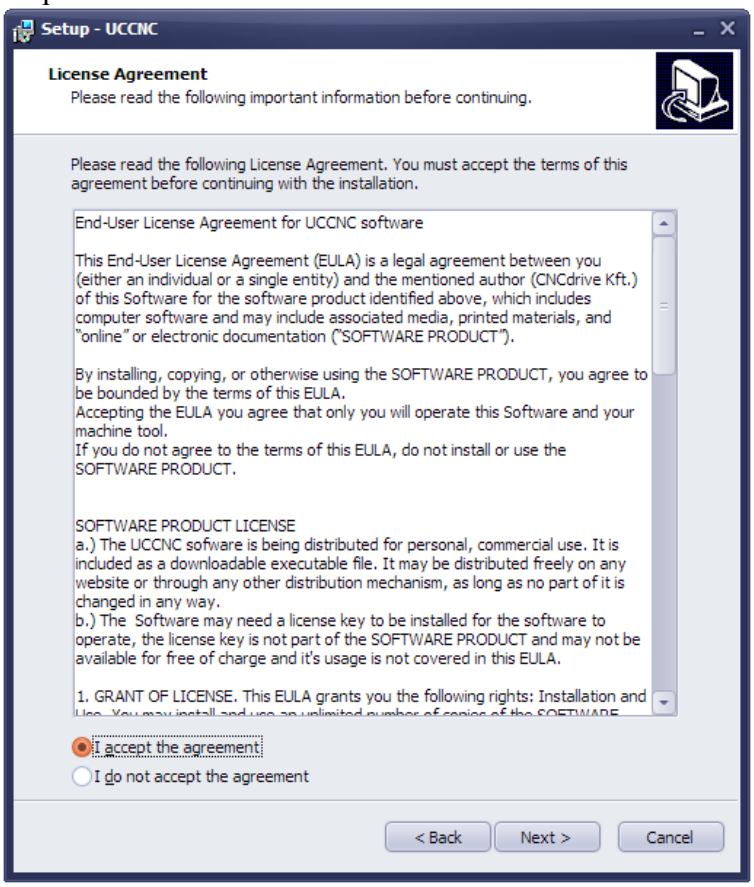
## 1.5 .Download and install

The UCCNC software comes in a single .exe executable installer package, the uptodate version is always available as a download on our website:
http://www.cncdrive.com/UCCNC/download_UCCNC.php

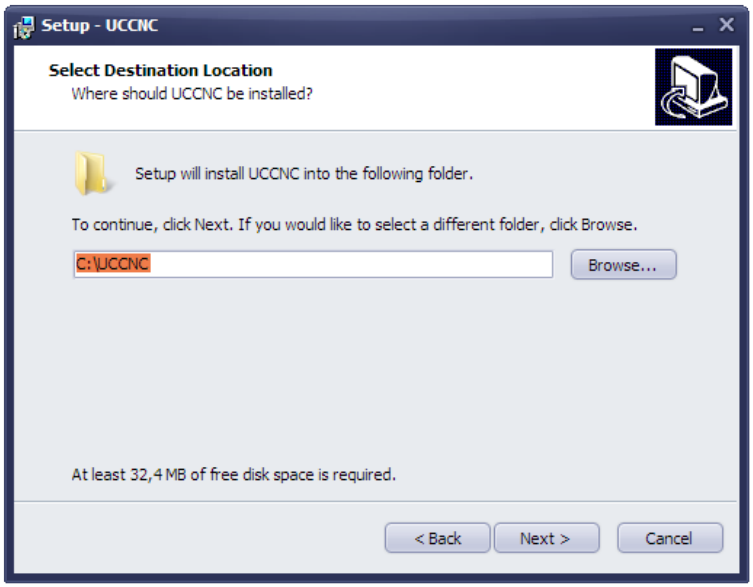
Download the setup.exe file from the above link, save it to your harddrive and double click the file to start the installation process.
When running the setup file the following screen will pop up:



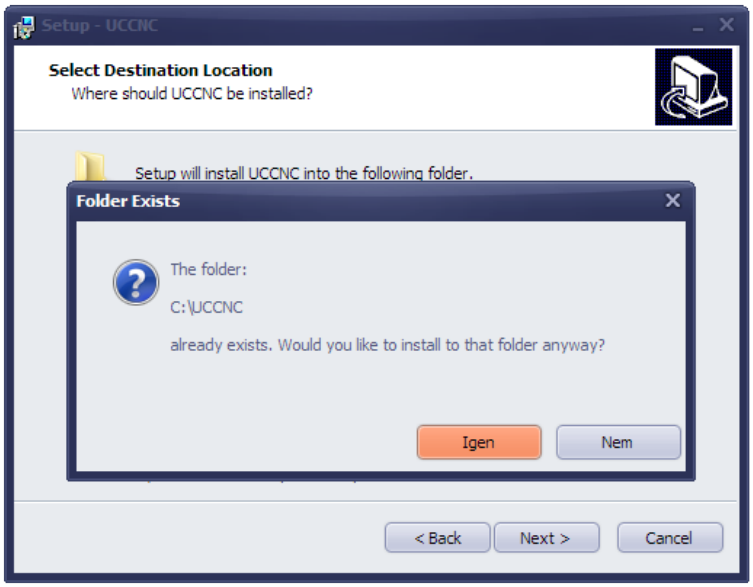

Press the next button to proceed the installation.



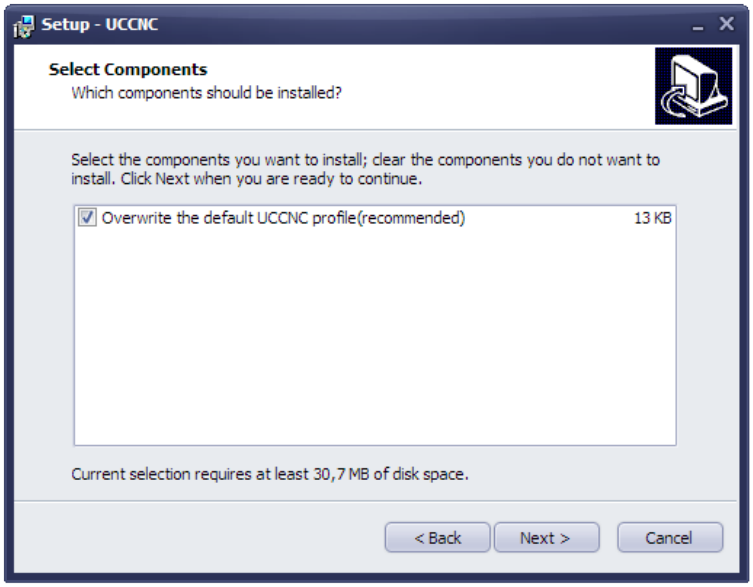
The End-User License Agreement for the software will appear and you should read the EULA carefully and if you agree the terms select the "I accept the agreement" radiobutton and press the next key. If you do not accept the EULA then press the cancel button and in this case the installation will terminate.



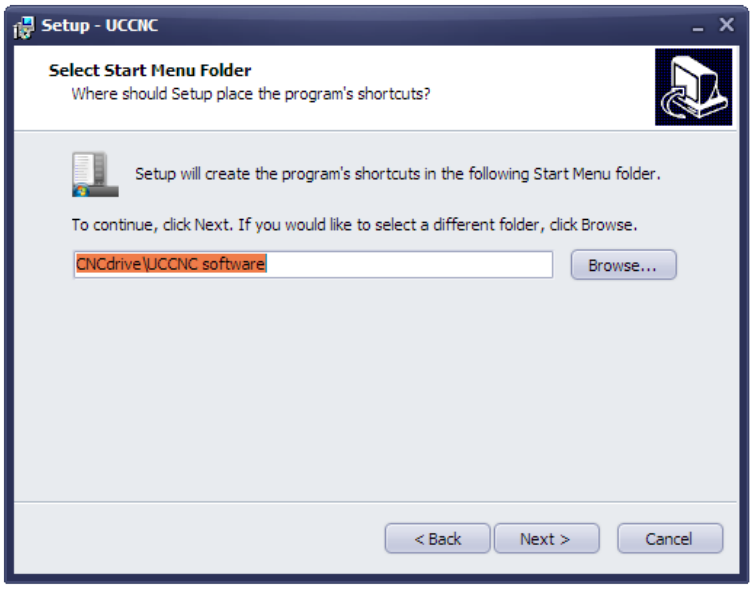

On this screen the installation folder path can be set, The default installation location is your main harddrive which is mostly the C: drive and the \UCCNC folder. The path can be changed with overwriting the path string on this window. Before pressing the next button make sure that there is enough disk space available on the harddrive used for the installation, the required disk space is indicated on this window. When these are done press the next button.



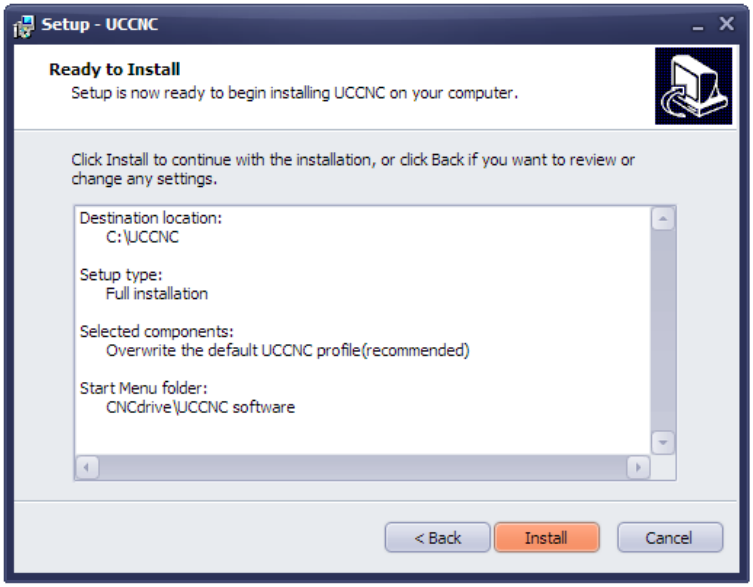

If the software was installed previously to the same directory then the installer will ask for permission to overwrite the previously installed software version and the files.
Select yet to proceed the installation.



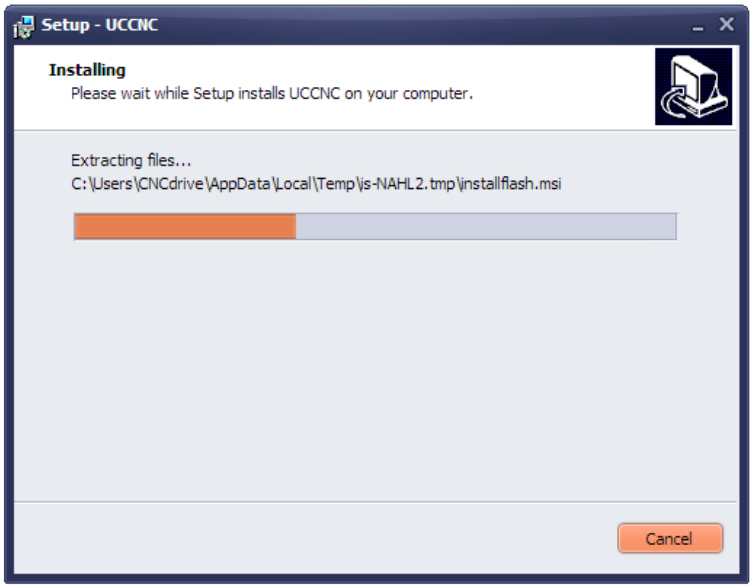

There is an option to overwrite or keep the previously installed default profile file. The default profile file contains all the settings of the screen, machine settings etc.
It is strongly recommended to select this option.

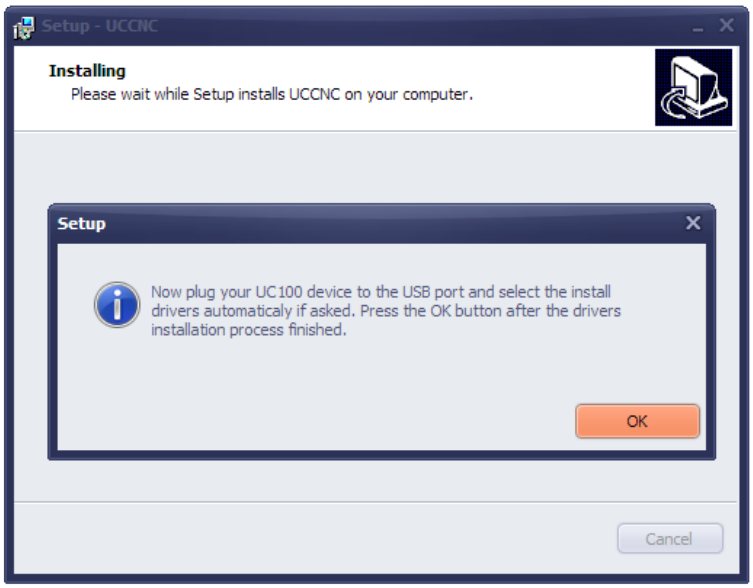



The installer will ask under what section name to create a shortcut icon for the software in the startup menu in Windows. The default is CNCdrive\UCCNC software. Press next to proceed the installation.



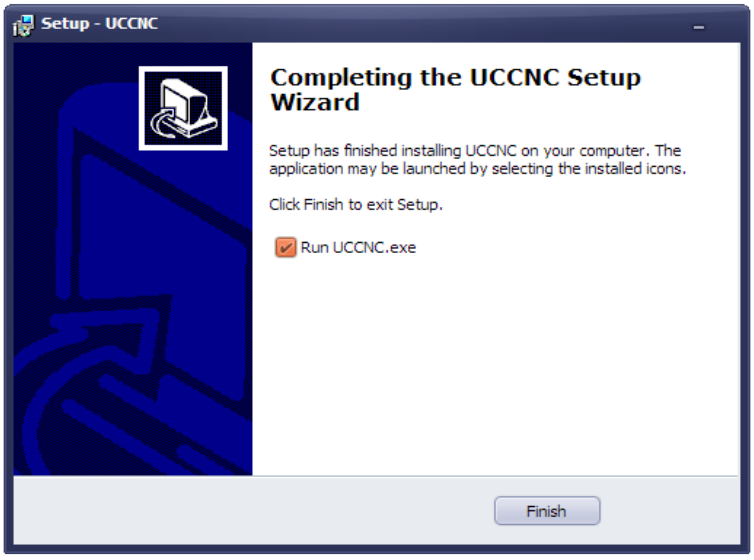

Finally before the actual installation process starts a summary screen will appear where you can check the settings you made. This is the last time the installation can be rolled back with pressing the back button.
To confirm the settings and start the installation process press next.






The installation starts and the installer extracting and copying all files and registry entries as nessessary. Wait while the process finishes, it may take a few minutes.
At the end of the installation process you will be asked to plug the UC100 device to the computer's USB port, do so.

Because the USB drivers were preinstalled by the installer Windows will automaticly assign the USB drivers to the device once it gets plugged to the PC's USB port.
Wait while the process finishes and press the next button.




The installation got finished and selecting the "Run UCCNC.exe" option will start the software when pressing the Finish button. Press the finish button to exit the installer.

## 1.6 .Licensing.

The software requires a license key to fully function. The license key is an encrypted file released by the author and owner of the software: CNCdrive Kft. company. One license key is valid and can be used with one UC100 device only. The license keys are personalised and assigned to a UC100, UC300, UC300ETH, UC400ETH or other future motion controller device's serial number which makes the software function with that specific device only.

The motion controller device's serial number and the license key's serial number must match, the license key with one serial number and the motion controller device with a different serial number will not work together.
A separate license key should be purchased for each motion controller devices for the UCCNC software to work with them. It also must be noted that the software can connect to and can run with one motion controller device one time. The license key file is delivered in e-mail only, no physical media (CD/DVD) is supplied.

To install the license key, simply copy the license key file to the UCCNC installation folder and run the software. The software automaticaly senses the license key in the installation directory and will connect to the device.

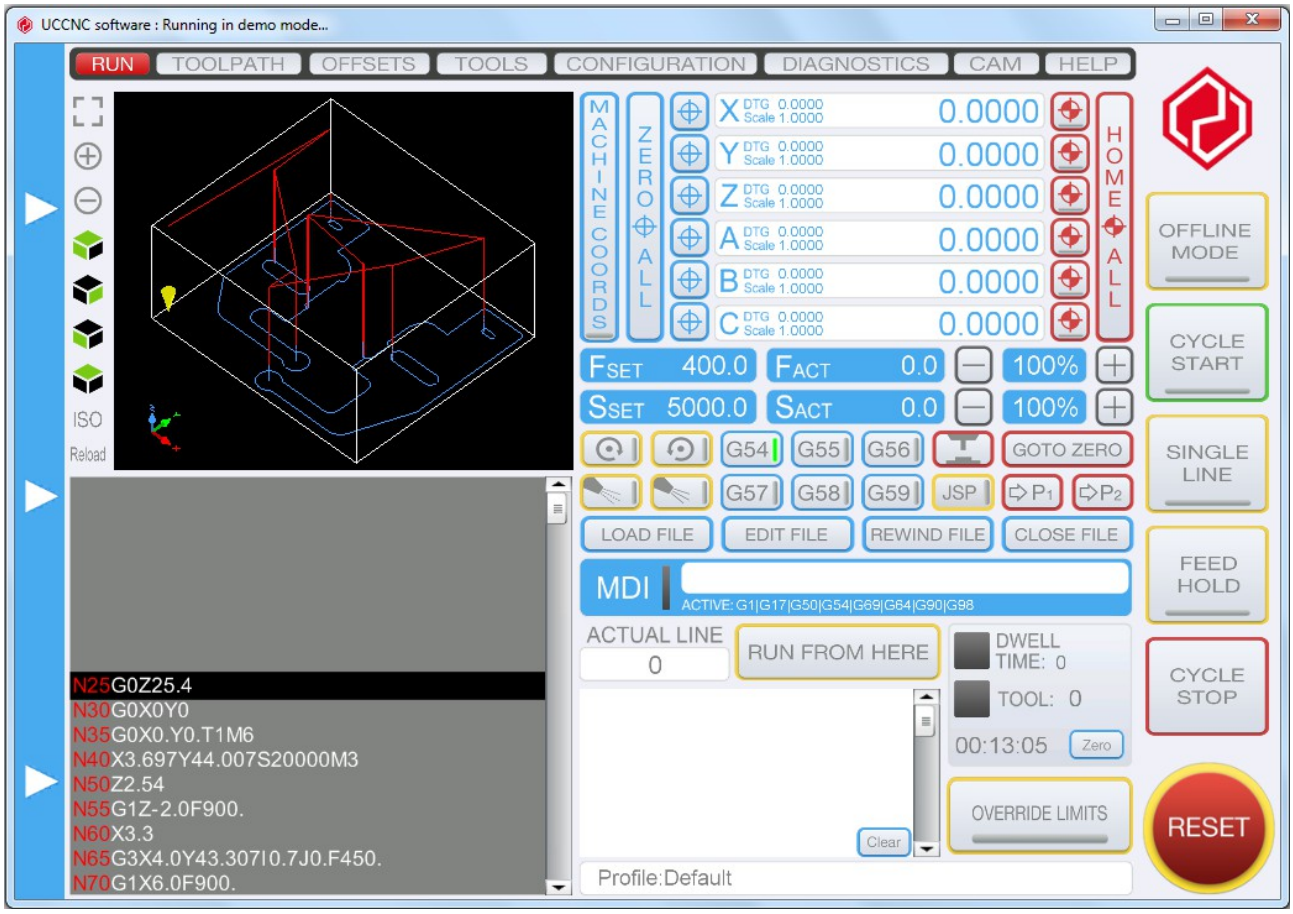
Without a license key the software runs in demo mode when there is no physical connection is made to the external motion controller, the software emulates the motion, but signals are not generated by the device.

Without the motion controller device connected or installed to the computer the software will start in demo mode. As mentioned previously in this case the software emulates the motion, but no signals are generated by the device, all motions happening on the screen is just simulation.

The possiblity of using the software in demo mode without a software license key and without buying a motion controller provides the ability for you to try and test the software without the need of any prior purchase. We recommend that you test and make sure that the software fits your needs before purchasing a software license key.

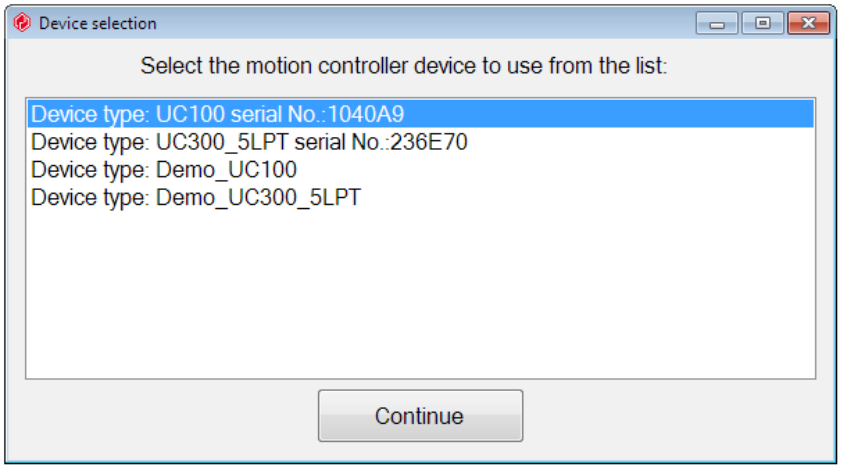
Software license keys because of the nature of this kind of product are not returnable and are not refundable.

The following printscreen picture shows the software running in demo mode, please note the " Running in demo mode..." text in the header of the window.




## 1.7 .The first run, first steps.

After the installation process and licensing the software it is time to run and setup the software to work with your machine.
To run the software double click the UCCNC icon on the desktop or in the start menu.
If only a single UC100 or UC300-5LPT device is connected and if the software is licensed for that device then the software will startup.
If no devices are connected or if more then one devices are connected or if one device is only connected but it is not licensed for the software then the following device selection window will popup prior to the software startup:




The device selection window lists all the available motion control devices which are actually cunnected to the PC and also it lists the available demo modes to run the software in. The devices

can be selected with clicking and then with pressing the Continue button the software will startup and will connect to the selected device in case that device is licensed. If the selected device is not licensed then the software will run in demo mode simulating the selected device.
Selecting any of the demo modes will also simulate the device.

### 1.7.1 .Command line arguments

Command line arguments can be used to influence how the software starts.
The arguments may be written to the shortcut icon properties target field after the path of the UCCNC.exe file.
The available command line arguments are as follows:

- /n : Using this switch in the argument will only allow the software to start one instance. If one instance is already running, the software will show an error message and will close.
- /p profilename : Using this switch forces the software to load a specific machine profile file.
- /s serialnumber : Using this switch forces the software to connect to a motion controller with the specified serial number. If the motion controller is not connected (not available) or if it is already in use then the software will show an error message and will close.
- /d devicetype : Using this switch forces the software to always run in demo mode only, no matter if any motion controller devices are connected. The devicetype parameter can have one of the following values:
  UC100, UC300_5LPT, UC300_ISOBOB, UC300_M44, UC300_5441, UC300_HI, UC300_LOW, UC300ETH_5LPT, UC300ETH_5441, UC300ETH_HI, UC300ETH_LOW, UC300ETH_M44, UC300ETH_M45, UC300ETH_ISOBOB, UC300ETH_UB1, UC400ETH, AXBB

And example for using the command line arguments:

C:/UCCNC/UCCNC.exe /p myprofile /s 60CC4D

This comamnd will load the myprofile profile file (myprofile.pro) and will connect only to the device with the 60CC4D serial number.

## 2 .The graphical user interface

The graphical user interface (GUI) is based on OpenGL technology.
All screen components like the buttons, labels, TAB layers are customisable, the look, format of these elements are defined in a screen set script file. The UCCNC software loads the screen set script on startup. The name of the screen set file is defined in the Profile file in the "mainscreenfilename" key entry. Changing this key entry value will read a different .ssf screenset file. The user could write their own screen set file customizing the GUI and it's components.

The screen can be dinamicly resized on the fly and the form elements resizing accordingly, this makes the usage ideal on monitors with any screen resolutions.

### 2.1 .The default screenset

The default screenset is the screen set file (.ssf) which the software installs by default.
This screenset file contains components and graphics for a 6-axis machine controller.
Printscreens of this screenset are shown in this documentation.

## 2.2 .Screen elements

### 2.2.1 .Tab pages

Tab pages are similar to the standard Windows TAB control pages. These elements are like layers on the screen. Clicking the label text of the TAB page changes the screen view and showing the components on the selected TAB layer. TAB layers can have a hierarchy where any TAB layer can have child TAB layers and all TAB layers have a parent layer. The basic layer is layer 1. which is the main and lowest layer on the screen, in other words this layer is the UCCNC window itself.
Any number of tab pages with any labels can be placed onto the screen when making a custom screen set file. The way of placing TAB controls on the screen will be further discussed in a separate documentation which will be about writing own screen sets.

The TAB pages on the default screen set are:

- Run

This page is the main page of the screen, it contains the buttons to load, edit, run, close a G-code file. It also contains buttons to switch the spidle, coolant, mist on/off, select the offset coordinate system. It contains a 3D toolpath viewer and a G-code viewer. On the top of the screen the 6 axis position DROs and actual and set feedrate, spindle set rotational speed and spindle actual rotational speed DROs are taking place. An MDI (manual data input field) is placed on the middle of the screen, this component allows a manual G-code input via the keyboard.

- Toolpath

This page is to get a more clear view of the toolpath loaded into software. The page contains a large, high resolution 3D toolpath viewer and buttons to navigate, zoom and to have different viewing angles of the toolpath.

- Offsets

This page contains the offset coordinate system parameters. The offset coordinate systems are on 6 sub TAB pages and are G54, G55, G56, G57, G58, G59 respectively. The actual offset coordinate system can be selected on the Run page. The offset values on the actually selected offset system is applied to the coordinate DROs of the machine and the name of the actual offset system is indicated in a label on this screen for example "Active fixture: G54" means that the G54 coordinate system is selected. Currently work offset is available for all the 6 axis and in addition a tool offset is available for the Z-axis. The offset numeric values are one by one all editable on the screen.

The current position which is the actual position of the machine can be offset with a single button press. Also the work offset can be cleared with one button and the tool offset can be cleared in a similar way with a single button press.

- Tools

This page contains the tool offsets for the Z-axis, because as previously described tool offsets are currently only available for the Z-axis only.
There are in total 96 pieces of tool offsets are available marked Tool#1 to Tool#96 on the screen. All tools can have it's own tool length offset value. All tools numeric values are editable on the screen and the values can be saved to the profile file.

- Configuration

This page is the most complex tab page, it contains several sub TAB pages and this page has all the software configuration parameters. The sub tab pages and their purpose will be discussed in the followings.

- Axis setup (sub tab page)

This page has sub pages for all 6-axis settings X, Y, Z, A, B, C axis respectively.
The axis settings can be made on these tab pages.
There is also a sub tab page for the Spindle settings.

- I/O setup (sub tab page)

This page contains the input and output pins definitions, settings for the E-stop, charge pump and other signals which are not exact relations with the axis and the spindle.

- General settings (sub tab page)

This page contains the parameters for the USB communication and some machine behaviour.

- Appearance (sub tab page)

This page contains settings for the toolpath viewer screen colors.

- Importers (sub tab page)

This page contains data import function(s).

- Profiles (sub tab page)

This page contains the machine profile setup, profiles can be loaded, created, deleted.

- Diagnostics tab page

This page shows datas and feedback about the current job and the machine properties, like I/O and functions logic states.

- Help tap page

This page lists the supported G and M and other codes with basic descriptions. Also the motion controller device parameters and the license key validity can be read here.

### 2.2.2 .Buttons

Buttons are one of the most important screen elements, these are like Windows buttons, pressing them execute an action. All buttons have a function to call inside the UCCNC software. All these functions have a pre-defined identify number. When creating a screen set file the buttons can be defined to call one of these functions by adding the function number to the button by a parameter. The function numbers of the different buttons and how to add them to a custom screen will be defined in a separated documentation which will be about writing a custom screen set file.

To read more about the buttons controls read the Buttons_by_number.htm document.

### 2.2.3 .Labels

Labels are used to show texts on the screen and also they are used to input values from the user. Labels can be static labels which are not updateable, their contant is constant and always the same. Static labels does not have a reference number since they never need an update of their value, their content is static, always the same. The text content of these buttons are defined in the screenset script and uploaded to the screen when the software load the screenset.
There are also output and input type dynamic labels. Dynamic labels are updated by the software mostly in sequence or in case an event is happening. Dynamic labels all have a reference/function number. There are input and output type dynamic labels. The output type labels are to view only text while the input type labels are used to input datas from the user via the screen.

To read more about the labels controls read the Labels_by_number.htm document.

### 2.2.4 .LEDs

LED controls are used to show the state of Boolean variables states to the user. Leds can have 2 states on/off and the actual state of each LEDs is updated by the software automaticly.
Every LED function have a function number which means the variable the LED will indicate.

To read more about the LED controls read the LEDs_by_number.htm document.

### 2.2.5 .Toolpath viewer

The toolpath viewer is a special control which is used to show, visualise the path the machine will execute for the loaded G-code program. Each tab page can have one toolpath viewer control.
There are several associated buttons functions which can show the toolpath from different view points, angles, zoom the contents, zoom in and out etc. To read more about these read the Buttons_by_number.htm document and look for the associated buttons controls.

### 2.2.6 .Jog control

The jog control panel is located at the right side of the window and is used to move the machine is manual, "jog" mode.

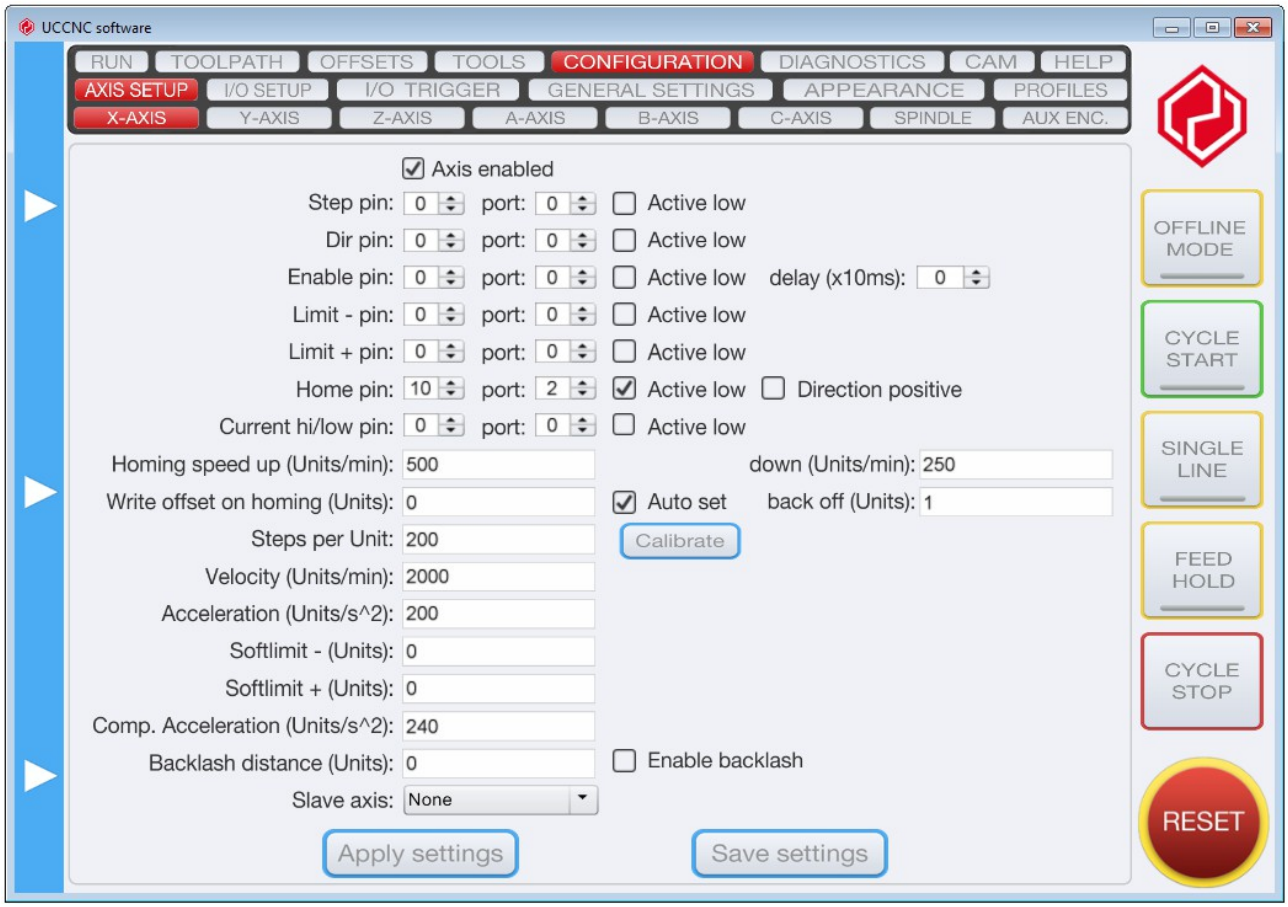
## 2.3 .Virtual mouse control

There can be machine controls where no physical mouse or trackball is installed. In these system it is still possible to control the mouse cursor with keyboard keys using the virtual mouse function.
To enable the virtual mouse control press the TAB keyboard key and this will enable the virtual mouse control mode. The indication of the virtual mouse control is enabled is that the mouse pointer changes from the arrow to a cross icon. In virtual mouse control mode the mouse cursor can be moved left/rigth and up/down with the keyboad arrow keys and the left mouse click can be emulated with pressing the enter key on the keyboard.
The cursor can be moved with high speed if the Shift keyboard key is pressed and is held down while the cursor is commanded to move with the arrow keys.
To disable the virtual mouse control press the TAB keyboard key again and the mouse cursor will change back to the arrow icon and the function is then disabled.

# 3 .Setting up the software for the machine (Configuration)

## 3.1 .Axis setup

The machine axis settings can be configured in the axis setup.
The axis setup page contains 6 sub tab pages for the 6 available axis.
All axis has identical settings, except that the X,Y and Z axis can have a slave axis, so these axis have an additional setting for the slave axis selection.
Each axis has the following settings:




**Step port & pin:** Which is the physical output pin on the motion controller device for the step signal. The signal can be configured active low or active high selecting or unselecting a checkbox next to the control.
The active high means that the rising edge (0->5Volts) is the active edge of the step signal and active low means that the falling edge (5->0Volts) is the active edge of the step signal.

**Dir port & pin:** Which is the physical output pin on the motion controller device for the step signal. The signal can be configured active low or active high selecting or unselecting a checkbox next to the field. Changing the active polarity of the signal reverses the axis running direction.

**Enable port & pin:** Which is the physical output pin on the motion controller device for the enable signal. The signal can be configured active low or active high selecting or unselecting a checkbox next to the field.
The enable signal goes active when the axis is enabled with the "Axis enabled" checkbox marked and when the software is out of reset. Otherwise the enable signal is inactive.

**Enable delay:** This parameter delays the enable signal with the set amount of time to activate.
The delay is the set value times 10 milliseconds. The range is 0-255 which means 0-2.55 seconds

time interval. This parameter is currently only implemented in our ethernet motion controllers. Our USB controllers do not have this parameter and functionality.

**Limit – and + port & pins:** These are the physical input pins on the motion controller device for the limit switches.
Limit switches are used to limit the workspace, limit the length of an axis or in other words to not allow the machine to run over a certain position on each axis direction.
When the limit switch is activated the UCCNC software triggers the reset signal and stops the motion. The signal can be configured active low or active high polarity. The correct polarity depends on if the limit switch or sensor inputs a low (0V) or high (5V) signal when activated.
One negative and one positive side of limit pins can be defined.
The limit pins may be the same for more than one axis. Any combination of the limit pins for any axis is allowed. To disable a limit input pin, write 0 value to the field.

**Home port & pin:** Which is the physical input pin on the motion controller device for the Home of Reference point switch or sensor. The machine axis can be homed and pick up the reference point using this switch or sensor. The signal can be configured active low or active high polarity. The correct polarity depends on if the Home switch or sensor inputs a low (0V) or high (5V) signal when activated.
The home direction to the positive or to the negative side of the axis can be configured with the "Direction positive" checkbox.
The home pin may be the same pin as a limit pin, if using the same sensor for the home and limit functions. If the home and limit pins are the same then the pin will be treated as a limit pin all the time except when a homing sequence for that axis is in execution. In this case the lmit function is disabled and the pin is treated as if it was a home switch only. When the homing sequence is finished, the pin is again treated as a limit input pin.

**Current high/low port & pin:** This is a physical output pin which can be used as an enable or a current reduction signal for the external motor control electronics connected to the system. The pin can be inverted checking the active low checkbox next to the textfield. The signal is active when a motion or dwell is ongoing and activates a few milliseconds before the motion starts and deactivates a few milliseconds after the motion stopped. The few milliseconds delays are to let the stepper motor driver adjust the current in time to the full or to the holding current level.

**Homing speed up:** Which is a numeric value and sets the feedrate of the motion when homing the axis and when the axis is moving to the home sensor.
The Homing feedrate can be equal or lower than the Velocity parameter of the axis.
The parameter is defined in Units per minute.

**Homing speed down:** Which is a numeric value and sets the feedrate of the motion when homing the axis and when the axis is moving down from the home sensor.
The Homing feedrate can be equal or lower than the Velocity parameter of the axis.
The parameter is defined in Units per minute.

**Write offset on Homing:** Which is a numeric value and sets the coordinate to write to the axis position DRO when a homing action is executed and succeded.
The parameter is defined in Units.
The homing position DRO value can be autoset with checking the checkbox next to the field.
If the autoset checkbox is not set then the axis will remain it's coordinate when a homing sequence is successfully executed. If the autoset checkbox is not set then the set value will be written to the axis position DRO when a homing sequence is successfully executed.

**Back off:** Which is a numeric value and is a movement distance the machine axis will make after

the homing off. If the parameter is set to 0 then no movement will happen after the homing.
The movement can be to the negative or positive direction depending on the sign of the value.
The final position after the back off movement is registered as the home position.

**Steps per Units:** Which is a numeric value and means the output pulses per one Unit of mechanical travel on the machine axis. The correct value depends on the resolution of the motor and motor drive in case a stepper motor and drive is used or on the encoder resolution if a servo motor and drive is used to drive the axis.
**Velocity:** Which is a numeric value and sets the maximum feedrate on the axis. The axis cannot run on higher feedrate then the set value. In G0 (rapid linear motion) code execution the motion is executed with this feedrate value.
The value of this parameter is in Units/minute.

**Acceleration:** Which is a numeric value and sets the maximum acceleration value of the axis. The axis cannot accelerate faster than the set value.
The value of this parameter is in Units/sec/sec.

**Softlimit - :** Which is a numeric value and is the negative direction software position limit. The axis cannot run to a lower coordinate than the set value. Note: The software limits can be enabled or disabled on the general config tab page with checking or unchecking the enable softlimits checkbox.
The value of this parameter is in Units.

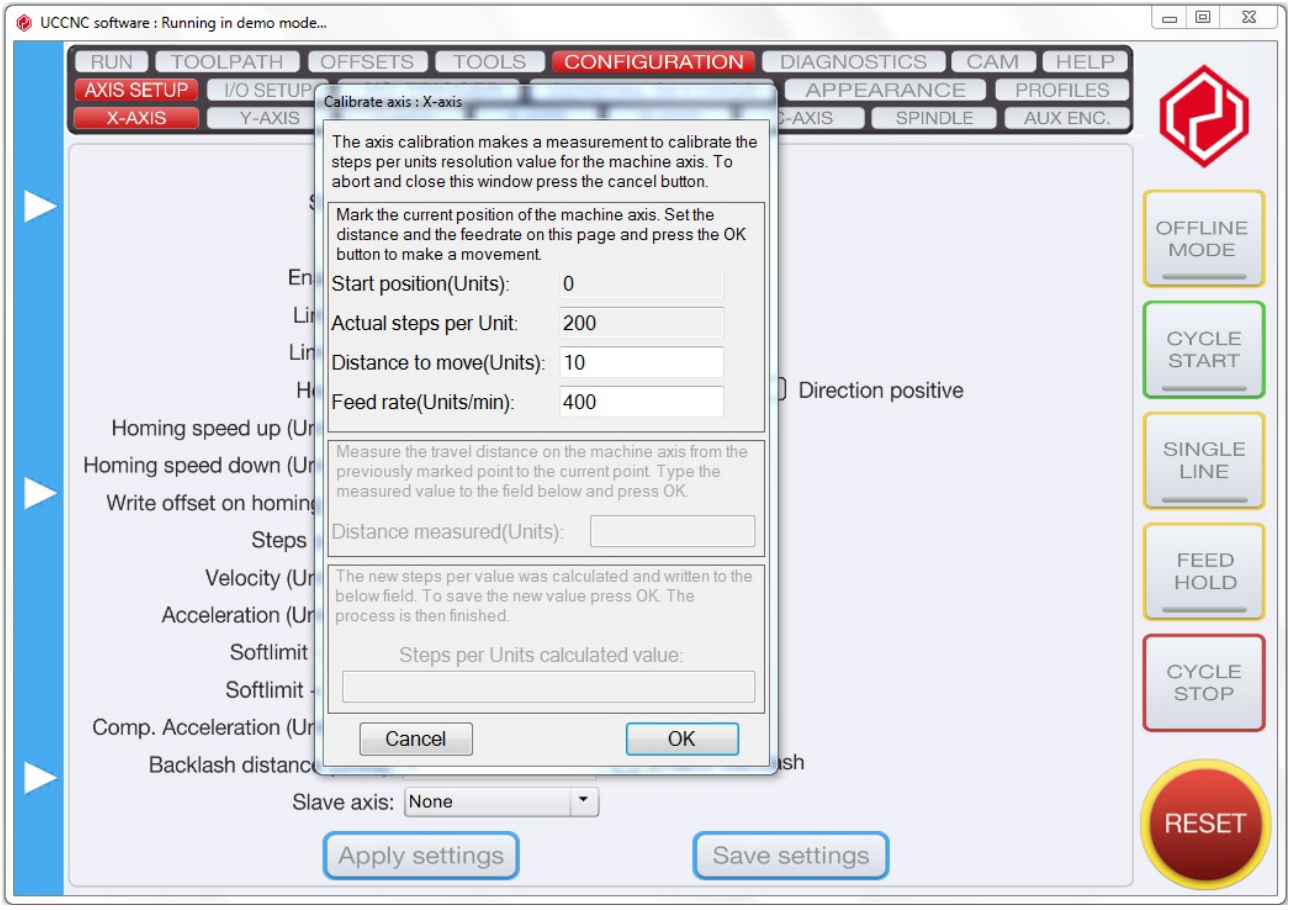
**Softlimit + :** Which is a numeric value and is the positive direction software position limit. The axis cannot run to a higher coordinate than the set value. Note: The software limits can be enabled or disabled on the general config tab page with checking or unchecking the 'enable softlimits' checkbox.
The value of this parameter is in Units.

**Backlash distance:** Which is a numeric value and sets the backlash distance to compensate on the axis. The backlash compensation can be enabled or disabled with checking or unchecking the 'enable backlash' checkbox next to the textfield.
The value of this parameter is in Units.

**Compensation acceleration:** Which is a numeric value and defines the acceleration of the axis when compensating the backlash and also used when rigid tapping (G33.1 and G33.2) and when thread cutting (G33 and G76). This value has to be set at least 20% higher than the acceleration parameter of the axis.
The value of this parameter is in Units/sec/sec.

**Slave axis:** This setting is only available for the X, Y and Z axis. These axis can have a slave axis attached to them. The slave axis moves together with the master axis. The slave axis can be the A, B or the C axis. The slave axis functionality is useful for example in router machines where 2 separated motors running the same axis on the 2 sides of the gantry axis. These motors can setup as a master and slave axis, so the two axis will run together in syncron.

**Calibrate button:** Each axis setup page has a calibrate button. Use this button to calibrate the resolution (steps per value) on the axis. Pressing this button will open an axis calibration window which looks the following:




The calibration is built from a 3 steps sequence. Press the OK button to process the sequences.
The first step of the sequence moves the axis to the set distance with the set feedrate using the currently set steps per unit value. The next step is to measure the distance on the machine axis between the movement startpoint and the movement endpoint and type the measured distance value back to the screen. In the third seqeunce the software calculates the new steps per value based on the previous and the measured input data. The new steps per value is shown on the page and again press OK to save the new value and to end the calibration process.
A new calibration process can be started with pressing the OK button again, or the process can be terminated and the window can be closed pressing the cancel button any time.

In addition to the above listed settings the A, B and C axes have 2 more settings, because these axes can be set to function as rotary axes instead of linear and the A axis can work as a tangential rotary cutter knife.

**Rotary axis:** When this option is set then the axis is not calculated into the feedrate of the movement vector, but only the linear axis are calculated for the feedrate. The rotary axes are only following the linear axes. The upper limitation for the linear axis feedrate will be still the maximum velocity of the rotary axis. In other words the rotary axis will not run faster than it's velocity setting and this might limit the feedrate of the linear movement.

**Roll over on 360°:** When this option is set then the axis angle range is between 0-360° and the axis position counter over rolls when it moves from the 0° to 360° or from the 360°to 0°.
It is important to note that when this option is set then the movement direction is always to the shorter path, clockwise or counterclockwise.
For example: The A axis is at position 90 and a G0 A10 is programmed. In this case the movement happens to A90, A89, A88 … A10. The other direction would be the A90, A91, A92 … A359, A0,

A1 … A10. Ofcourse the second path is longer and therefor the first solution is selected for the movement, because that path is shorter.

The following image shows the rotary axis options for the A, B and C axes:


The A axis can also work as a tangential cutter knife when the A axis rotation angle is automatically calculated and rotated to the XY plane movement vector direction. This way a motorised rotary knife can be used to cut foil or sticker sheets for example.
There are 3 parameters for this option:

**Enable tangential knife:** This checkbox enables the tangential knife control.

**Moveback feed (Units/min):** When the knife put back to the cutting height after a pull-up then the feedrate setup in this parameter is used for that Z axis downwards movement. If the value is set to 0 then rapid feedrate is used.

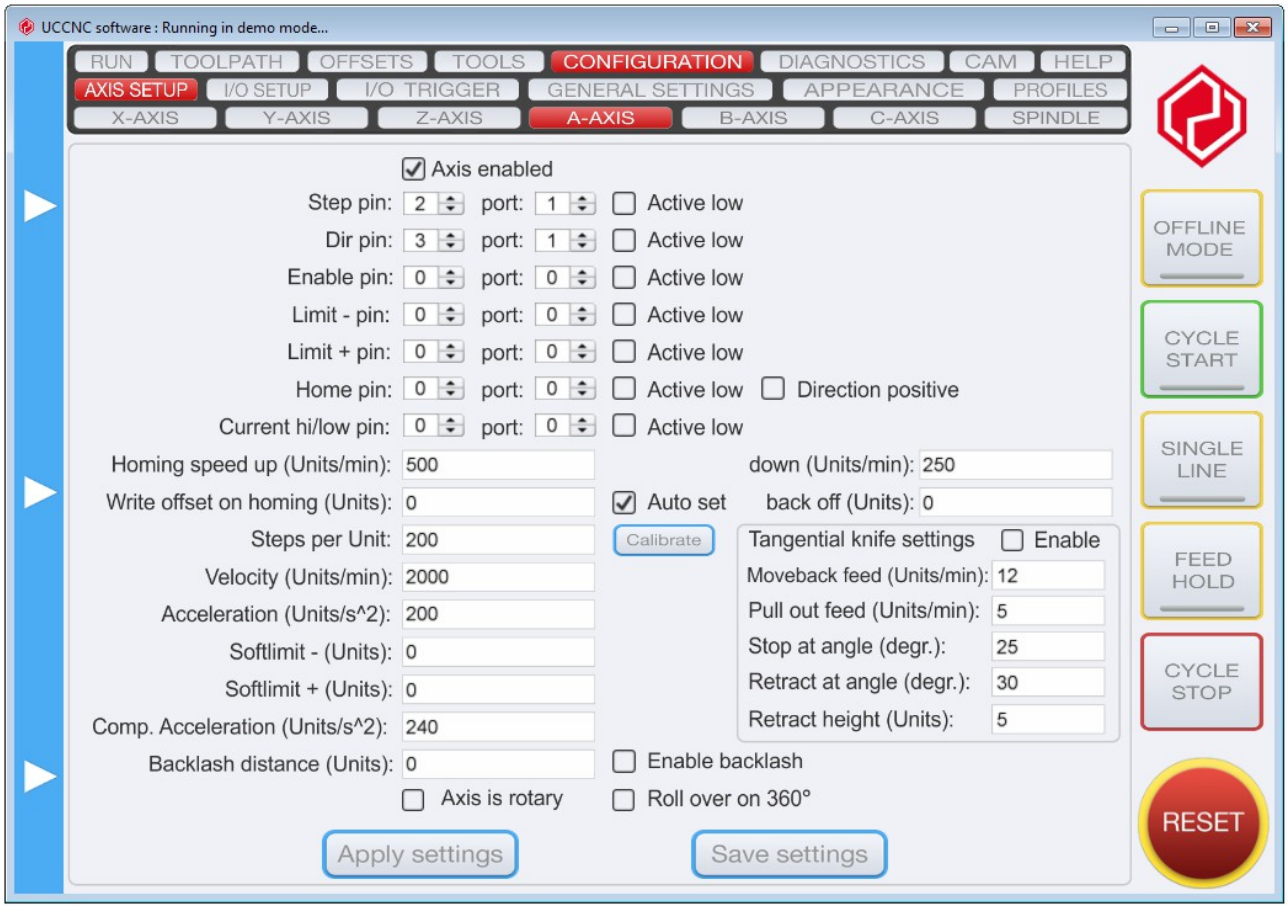
**Pull out feed (Units/min):** The retract feedrate of the automatic pullout of the knife (Z-axis.)

**Stop at angle (degree):** If the XY plane angle angle between the 2 connecting path vectors is larger than this value then the movement deccelerates to the connection point and the software rotates the knife to the new angle. The meaning for this feature is to safe the edge of the knife. The parameter value range is 0 to 180 degrees.

**Retract at angle (degree):** If the XY plane angle between the 2 connecting path vectors is larger than this value then the movement deccelerates to the connection point and the Z axis lifts the knife up to the retract height, the knife is rotated to the new angle and put back to the cutting height and

then it continues the motion. The meaning for this feature is to safe the edge of the knife. The parameter value range is 0 to 180 degrees.

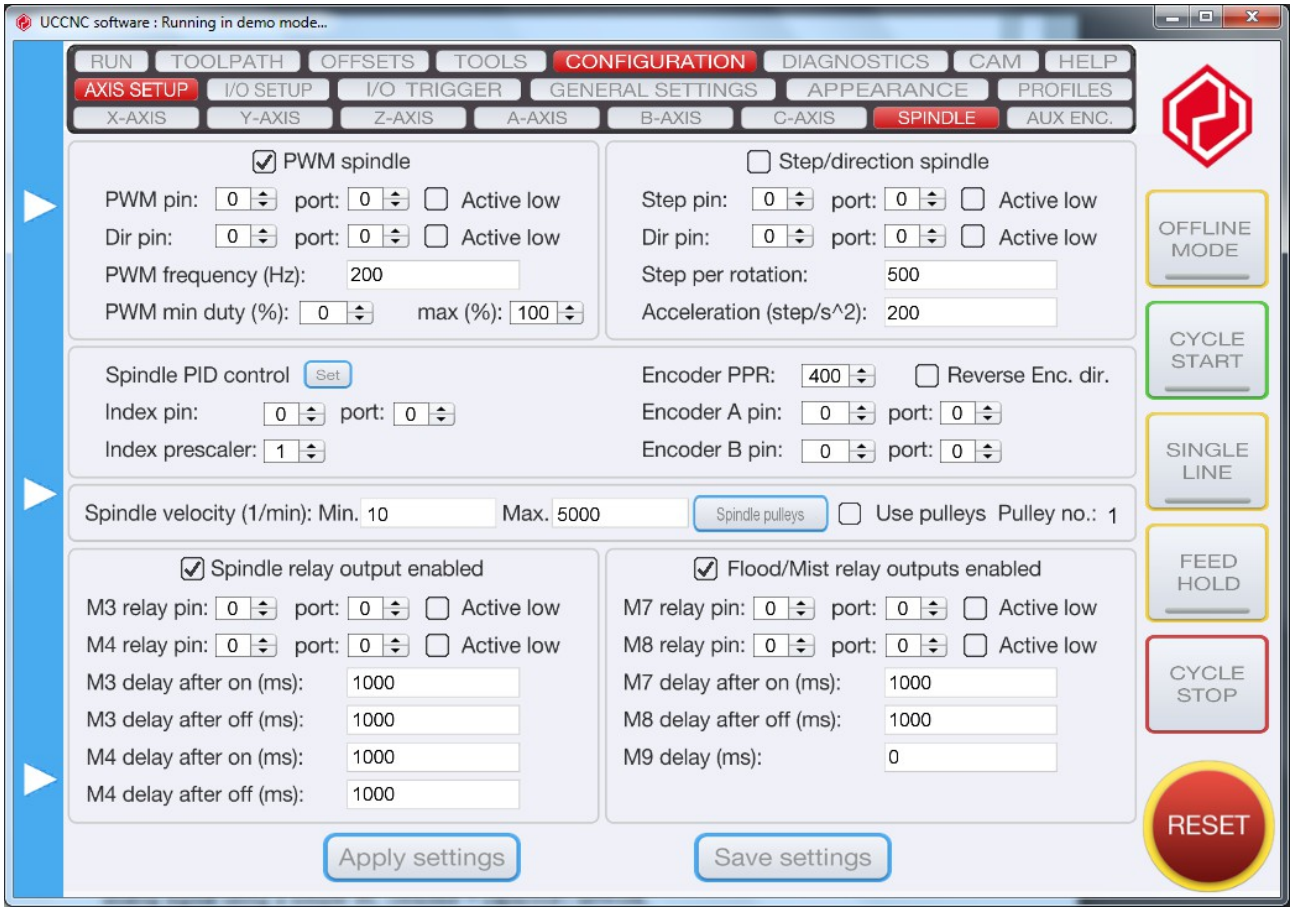
**Retract height (units):** This parameter sets the height of the Z axis retraction when the movement vectors connection angle is greater than the setup retract angle. The retract height is an incremental value which means that for example the retract hight parameter value is 10 and the cutting height is at -2 units then the axis is lifted to the 8 units height. The parameter has to have a non negative value.
The following image shows the tangential knife options for the A axis:




## 3.2 .Spindle setup

The spindle setup tab page contains settings for the main spindle of the machine.
The spindle can be a spindle with PWM (analog input) control or a step/dircetion control spindle.
Separate relays can be also setup to switch the spindle on/off.
The mist and flood coolant control parameters setup are also located on this tab page, because these are in close relation to the spindle control.

The spindle setup tab page has the following parameters:




Settings for a PWM control type spindle:

**PWM spindle check box:** Selecting this checkbox sets the spindle to PWM control mode. In PWM control the spindle control signal is a PWM (pulse width modulation) signal and the PWM duty cycle is changed proportionally with the programmed spindle speed.
The PWM spindle control mode is useful for analog input spindle electronics forexample for Variable Frequncy Drives with analog input. The PWM signal can be filtered, smoothed to an analog signal using a simple RC (resistor + capacitor) network.

**PWM port & pin:** This is the physical output pin number for the spindle control PWM signal, the signal will appear on the selected pin. The signal can be inverted setting the 'active low' checkbox next to the textfield.

**Dir port & pin:** This is a physical output pin number of the spindle direction signal. The running direction of the spindle depends on if an M3 (Clockwise rotation) or M4 (Counter clockwise rotation) is programmed. The direction signal changes according to the programmed running direction. The signal can be active low or high. Changing the direction active state inverts the signal polarity.

**PWM frequency:** This parameter sets the base frequency of the PWM signal, in other words it sets how long one PWM cycle will last. The time perion of the PWM cycle equals 1/PWM frequency.
The value of this parameter is in Hertz.

**PWM min duty (%):** This parameter sets the minimum duty cycle of the spindle speed PWM signal. The PWM duty cycle will go to this level when the lowest spindle speed is set.

**PWM max duty (%):** This parameter sets the maximum duty cycle of the spindle speed PWM signal. The PWM duty cycle will go to this level when the highest spindle speed is set.

Settings for a step/direction control type spindle:

**Step/direction spindle checkbox:** Selecting this checkbox sets the spindle to step and direction control mode. In step and direction control the spindle control signal is a step pulse train just like it is on the machine axis. This control mode is useful for position control spindles like servo motor spindles.

**Step port & pin:** This is the physical output pin number for the spindle control step signal, the signal will appear on the selected pin. The signal can be inverted setting the 'active low' checkbox next to the textfield.

**Dir port & pin:** This is a physical output pin number of the spindle direction signal. The running direction of the spindle depends on if an M3 (Clockwise rotation) or M4 (Counter clockwise rotation) is programmed. The direction signal changes according to the programmed running direction. The signal can be active low or high. Changing the direction active state inverts the signal polarity.

**Steps per rotation:** This is a numeric value which represents the number of pulses the controller should send out on the step signal to produce one full motor rotation.

**Acceleration:** This is a numeric value and sets the maximum acceleration for the spindle.
The spindle cannot accelerate faster than the set value.
The value of this parameter is in Rotation/sec/sec.

The following settings are shared between and valid for both a PWM and step/direction type spindle:

**Minimum velocity:** This is a numeric value and defines the programmable minimum rotational speed for the spindle motor. The spindle can be programmed with the 'S' parameter. If a lower than the minimum set value is programmed then the minimum setup value is used.

**Maximum velocity:** This is a numeric value and defines the programmable maximum rotational speed for the spindle motor. The spindle can be programmed with the 'S' parameter. If a higher than the maximum set value is programmed then the maximum setup value is used.

**Index port & pin:** This is a physical input pin for the index input signal. The index input signal is used to measure the spindle rotational speed and to feed it back to the actual spindle speed DRO.
The index pin is also the index channel of the spindle feedback encoder for thread cutting.

**Index prescaler:** This is a numeric value and is used to divide the frequency of the spindle rotational speed. In other words this should be set to a value of 1 if a single slot spindle speed sensor is used and a a value of 2 if a dual slot spindle speed sensor is user and so on.

**Encoder PPR:** This is a numeric value and defines the pulse per revolution of the incremental encoder which is used to feedback the spindle position in sycnronous thread cutting applications.

**Reverse enc.dir:** This parameter changes the encoder count direction. If the encoder pins were accidentally connected up in reverse then no need to exchange them or to change the A with B pin, just change this parameter to change the count direction of the encoder.

**Encoder A port & pin:** This is the physical input pin for the encoder A channel.

**Encoder B port & pin:** This is the physical input pin for the encoder B channel.

**Spindle relay outputs enabled checkbox:** Selecting this checkbox enables the M3 (rotation CW) and M4 (rotation CCW) relay outputs. These output activates when an M3 or an M4 code is programmed and the outputs can be used to switch the spindle on/off. Only one of these outputs can be active one time. Both outputs can bedeactivated with programming an M5 command.

**M3 spindle relay port & pin:** This is a physical output pin for the spindle M3 (CW) relay. The pin can be inverted checking the active low checkbox next to the textfield. The pin is active when a M3 command is programmed and inactive after a M5 command is programmed.

**M4 spindle relay port & pin:** This is a physical output pin for the spindle M4 (CCW) relay. The pin can be inverted checking the active low checkbox next to the textfield. The pin is active when a M4 command is programmed and inactive after a M5 command is programmed.

**M3 delay after on:** This is a numeric value. The software dwells for the set amount of time in milliseconds when a spindle M3 command is programmed and after the spindle output goes active.

**M3 delay after off:** This is a numeric value. The software dwells for the set amount of time in milliseconds if the spindle M3 command was active and when a spindle M5 command is programmed and after the spindle output goes inactive.

**M4 delay after on:** This is a numeric value. The software dwells for the set amount of time in milliseconds when a spindle M4 command is programmed and after the spindle output goes active.

**M4 delay after off:** This is a numeric value. The software dwells for the set amount of time in milliseconds if the spindle M4 command was active and when a spindle M5 command is programmed and after the spindle output goes inactive.

**Flood/Mist relay outputs enabled checkbox:** Selecting this checkbox enables the M7 (Mist) and M8 (Flood) relay outputs. These output activates when an M7 or an M8 code is programmed and the outputs can be used to switch the Mist and Flood coolants. Both outputs can be activated the same time and both outputs can be deactivated with the M9 command.

**M7 mist relay port & pin:** This is a physical output pin for the spindle M7 (Mist) relay. The pin can be inverted checking the active low checkbox next to the textfield. The pin is active when a M7 command is programmed and inactive after a M9 command is programmed.

**M8 flood relay port & pin:** This is a physical output pin for the spindle M8 (Flood) relay. The pin can be inverted checking the active low checkbox next to the textfield. The pin is active when a M8 command is programmed and inactive after a M9 command is programmed.

**M7 delay after on:** This is a numeric value. The software dwells for the set amount of time in milliseconds when a M7 command is programmed and after the Mist output goes active.

**M8 delay after on:** This is a numeric value. The software dwells for the set amount of time in milliseconds when a M8 command is programmed and after the Flood output goes active.

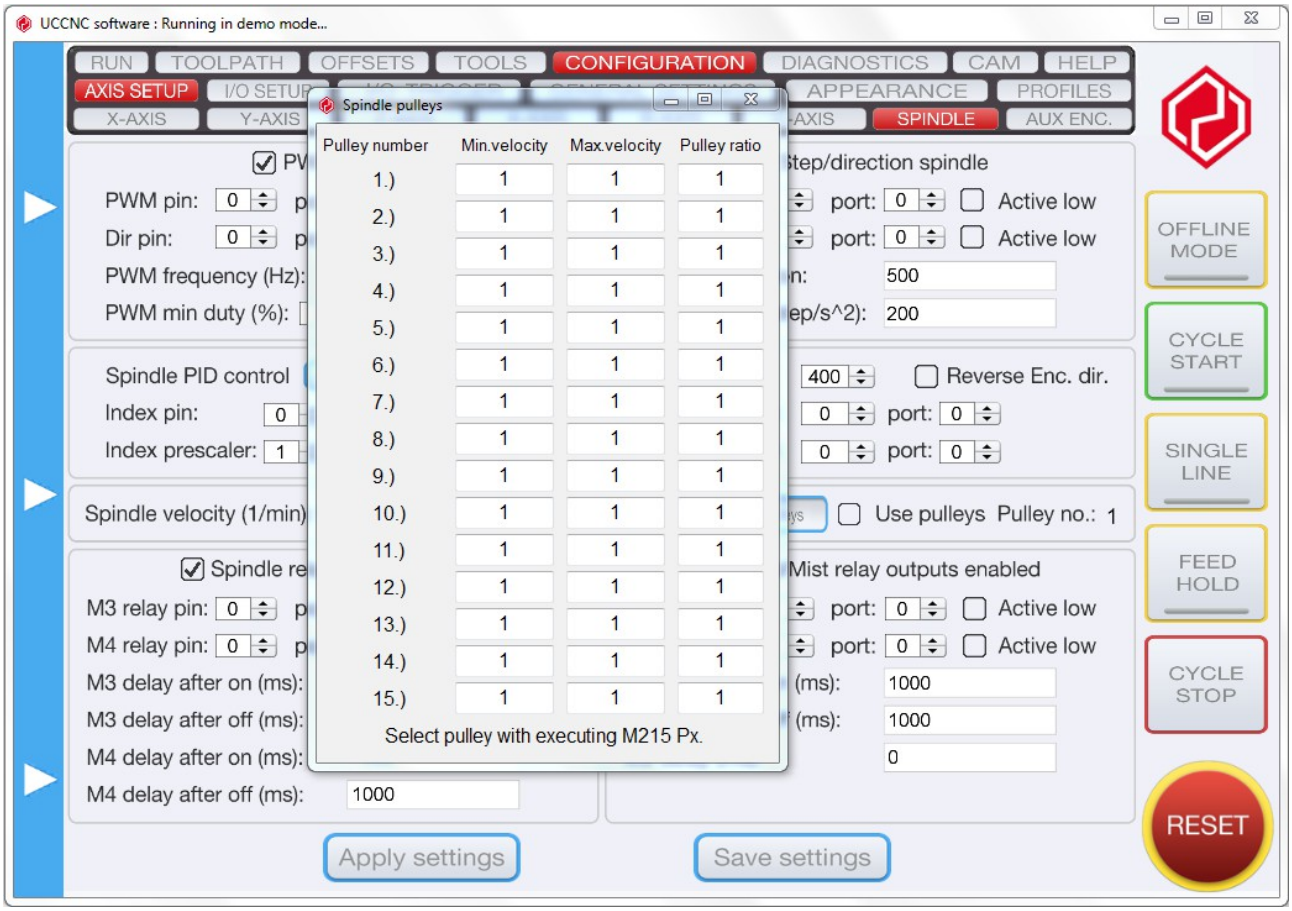
**M9 delay:** This is a numeric value. The software dwells for the set amount of time in milliseconds if the M7 and/or the M8 command was active and after these outputs goes inactive.

### 3.2.1 .Spindle pulleys

The spindle pulleys settings can be useful for machines which have pulleys on between the spindle and the driving motor. These pulleys with different mechanical ratios can vary the speed range of the spindle and also the spindle feedback speed if the spindle Index sensor which measures the speed is placed on the motor and not on the spindle.
The spindle pulleys settings can be accessed with pressing the Spindle pulleys button on the Spindle tab page and can be enabled with checking the Use pulleys checkbox next to the button.
On the spindle pulleys Window there are 15 available slots to setup with different spindle speed ranges and ratios.
When the use pulleys checkbox is set then the pulley number can be selected with the M215 Px command, where the P parameter means the pulley number to be used, for example M215 P1 selects pulley number 1.
The minimum and maximum speed value for the pulley defines what value the software will accept when programming the spindle speed with the S word.
The software automatically adjusts the spindle speed range to the setup PWM minimum and maximum values when a PWM spindle is selected and it automatically adjusts the speed range to the spindle step frequency if a Step/dir spindle is configured.
The ratio parameter in the spindle pulleys settings scales the feedback with multiplying the measured spindle speed with the ratio value before showing the measured value in the Sact Spindle speed DRO.

If the use pulleys checkbox is not set then the pulley window settings are not used and the spindle speed range used will be the value range setup on the main Spindle settings tab page and the ratio used for the feedback will be value 1.

The following page shows the Spindle pulleys settings window:



### 3.2.2 .Spindle PID controller

The spindle PID controller function is is only available with our ethernet motion controllers (UC400ETH and UC300ETH), this functionality is not available with our USB controllers (UC100 and UC300).
The function is accessible with pressing the Set button next to the Spindle PID control text.

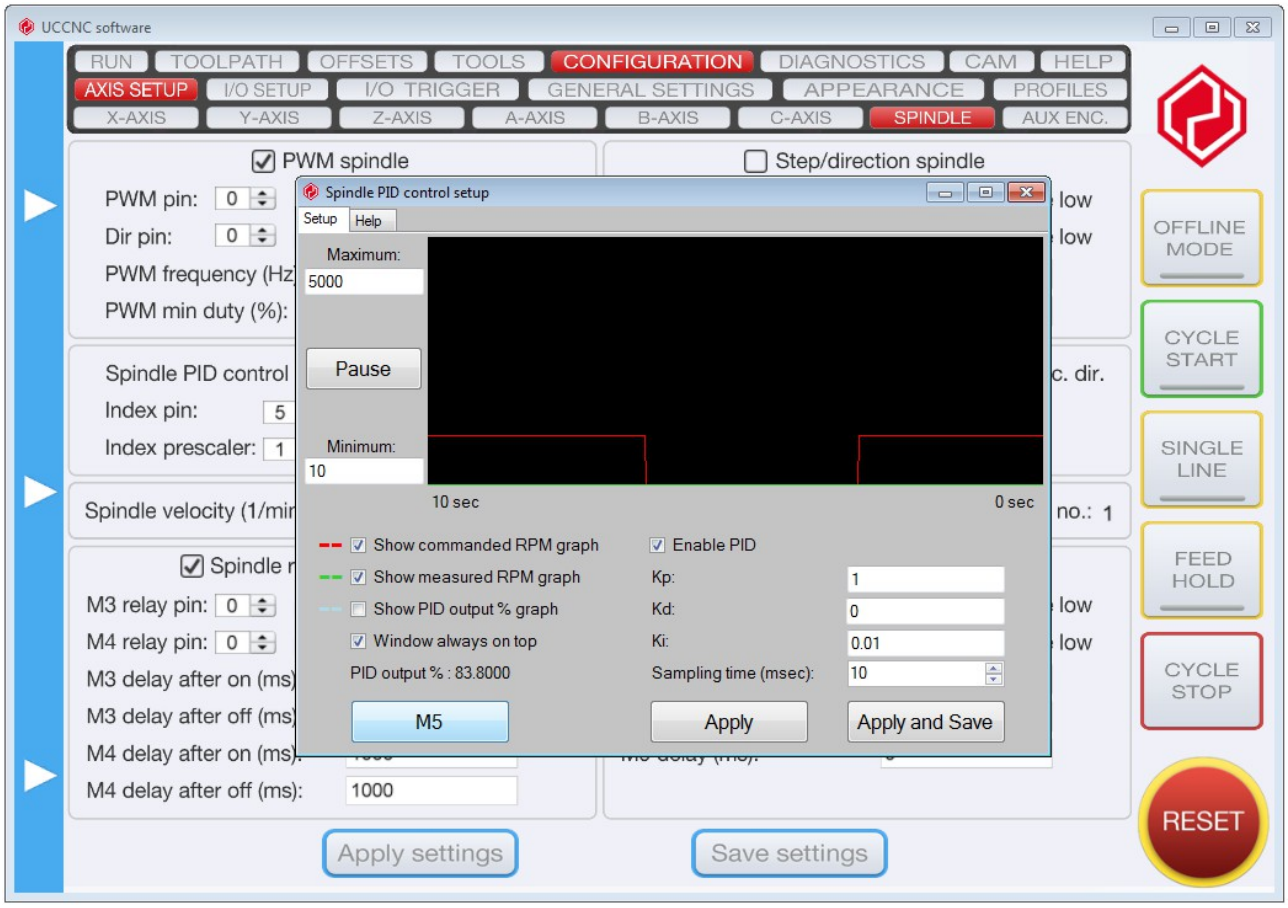
The PID controller makes the spindle control closed loop.
If the spindle speed drops then the controller adjusts the output signal to compensate and control the spindle speed to keep the programmed spindle speed value.
The controller takes the programmed S parameter (Sset) as the control signal and the spindle speed measurement (Sact) as the feedback signal.
A speed error signal is calculated using the Error = Sset - Sact.
The error signal is fed to the PID controller input and a calculation is made to get an output signal which signal controls the spindle motor.
The output signal is the setup PWM and/or analog output which are defined in the spindle setup.

For the spindle PID control to work the followings are required:

- The spindle speed has to be controllable with the output PWM or analog output signal.
- An incremental encoder with A and B channels has to be connected and measure the spindle speed and the encoder resolution has to be configured.
(Note: Single Index channel feedback is currently not supported, only an incremental encoder can be used.)

The PID parameters are the followings:
Kp: Proportional Gain
Kd: Differential Gain
Ki: Integrap Gain
Sampling time: The repeat time interval in milliseconds of the PID calculation.

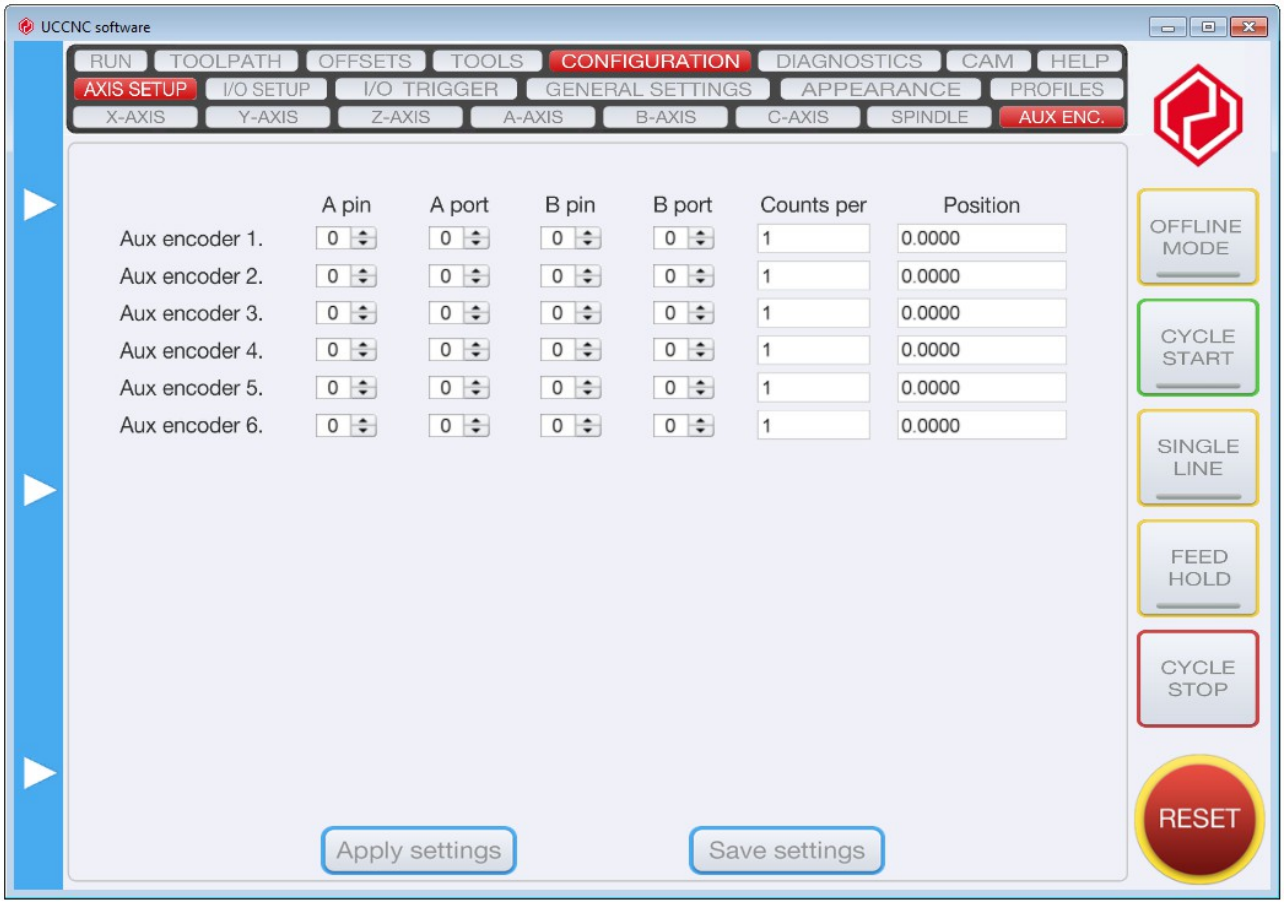
The following printscreen shows the spindle PID controller setup window:




## 3.3 .Auxiliary encoders

The auxiliary encoders menu and function is only available with our ethernet motion controllers (UC400ETH and UC300ETH), this functionality is not available with our USB controllers (UC100 and UC300).
The auxiliary encoders page contains 6 pieces of freely configurable incremental encoder counters.
The encoders must be incremental type ones with A and B channel output signals.
The encoders' positions can be read and written through the screenset with reading and writing the Textfields of the encoder coordinates. The Textfields can be read and written by GUI input and via macros and plugins.
With the 'Counts per' parameter the counts per the encoder ticks can be defined, so the encoder counting can be scaled up or down as required.
The aux encoders can't be used to close the position loop of the axes, they could be used to verify position or for other custom purposes which requires encoder counters.

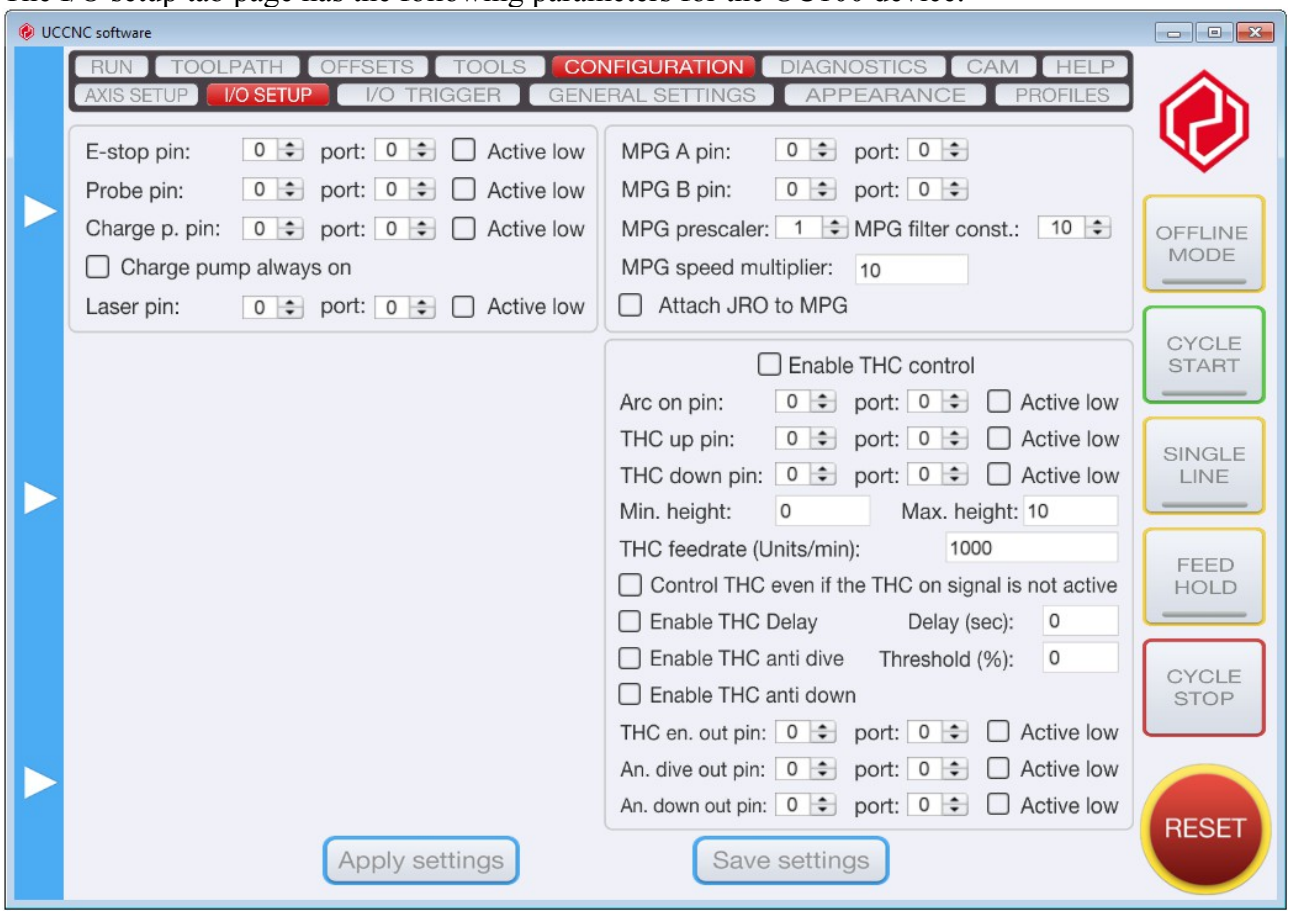
The following printscreen shows the auxiliary encoders tab page:



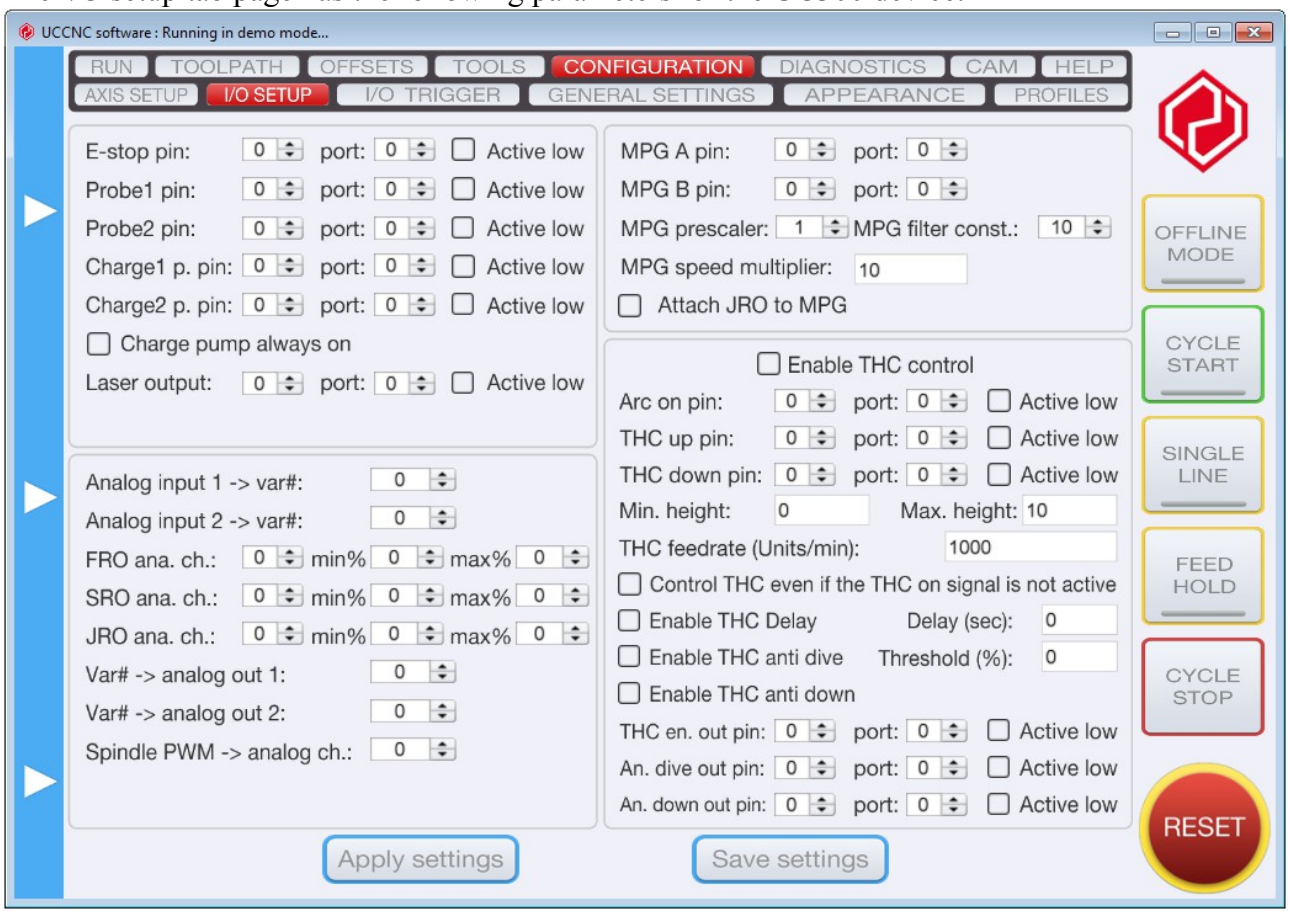

## 3.4 .I/O setup

The I/O setup tab page contains settings for in and output functions which are not in exact relation to the axis. These are the E-stop, probe, index inputs, charge pump output settings.

The I/O setup tab page has the following parameters for the UC100 device:




The I/O setup tab page has the following parameters for the UC300 device:



**E-stop port & pin:** This is a physical input pin for the e-stop button input signal. The pin can be inverted checking the active low checkbox next to the textfield. The machine goes to reset when this input signal is active.

**Probe port & pin:** This is a physical input pin for the straight probe input signal. The pin can be inverted checking the active low checkbox next to the textfield. The straight probe can be programmed with a G31 code.

**Charge pump port & pin:** This is a physical output pin for the charge pump safety output signal. The charge pump safety is a PWM signal. The pin can be inverted checking the active low checkbox next to the textfield. The PWM has a fixed 12.50kHz carrier frequency.
If the 'charge pump always on' checkbox is set then the PWM signal is always on when the UCCNC software is running. If the 'charge pump always on" checkbox is not set then the PWM signal is only active if the software is not in reset and inactive if in reset.
The charge pump signal can be used to enable an external electronics only if the UCCNC software is loaded or if it is not in reset making the operation of the electronics safe.

**The following parameters are available for the UC300 only, because the UC100 has no analog signals:**

**Analog input 1-> #var:** This setting makes the software to load the analog input 1. channel's read value to the internal variable number set in this field. The software continously reading the analog input and loading it into the variable. The value of the analog input is represented on 16bits, so the value can be between 0 and 65535, however the analog channel of the UC300's DSP has 12 bits resolution only, therefor the read value is multiplied to 16bits and transferred to the software.
To disable this function write 0 to this field.
The variables can be used in G-code execution or to read out the value in MDI command: ? #variable number.

**Analog input 2-> #var:** This setting makes the software to load the analog input 2. channel's read value to the internal variable number set in this field. The software continously reading the analog input and loading it into the variable. The value of the analog input is represented on 16bits, so the value can be between 0 and 65535, however the analog channel of the UC300's DSP has 12 bits resolution only, therefor the read value is multiplied to 16bits and transferred to the software.
To disable this function write 0 to this field.
The variables can be used in G-code execution or to read out the value in MDI command: ? #variable number.

**FRO analog channel:** This setting attaches the feedrate override (FRO) to the selected analog input channel. Using this function the FRO can be set with an external potentiometer connected to the analog input. The min.% value means the FRO percentage, which is set when 0Volts applied to the analog input and the max.% value sets the maximum FRO percentage, which is when the analog input has the maximum 10Volts input Voltage. Setting the min% to 0 value and turning the potentiometer down to 0Volts sets the feedrate override to 0% which causes a feedhold till the potentiometer is turned up again to higher than 0Volts.
Setting the min.% value lower than the max.% value reverses the function, in this case the 10Volts input Voltage applies the min% value for the FRO and the 0Volts input applies the max.% value.
This reversible operation makes it possible to use the potentiometer even if it was connected backwards to the power supply Voltage.
**SRO analog channel:** This setting attaches the spindle rate override (SRO) to the selected analog input channel. Using this function the SRO can be set with an external potentiometer connected to the analog input. The min.% value means the SRO percentage, which is set when 0Volts applied to the analog input and the max.% value sets the maximum SRO percentage, which is when the analog

input has the maximum 10Volts input Voltage.
Setting the min.% value lower than the max.% value reverses the function, in this case the 10Volts input Voltage applies the min% value for the SRO and the 0Volts input applies the max.% value.
This reversible operation makes it possible to use the potentiometer even if it was connected backwards to the power supply Voltage.

**JRO analog channel:** This setting attaches the jog rate override (JRO) to the selected analog input channel. Using this function the JRO can be set with an external potentiometer connected to the analog input. The min.% value means the minimum JRO percentage, which is set when 0Volts applied to the analog input and the max.% value sets the maximum SRO percentage, which is when the analog input has the maximum 10Volts input Voltage.
Setting the min.% value lower than the max.% value reverses the function, in this case the 10Volts input Voltage applies the min% value for the SRO and the 0Volts input applies the max.% value.
This reversible operation makes it possible to use the potentiometer even if it was connected backwards to the power supply Voltage.

It is possible to set more than one override function to a single analog channel, for example the FRO and SRO can be set to the same channel and then when rotating the potentiometer both the feedrate override and the spindle rate override will change.
It must be noted that when the FRO, SRO or the JRO is attached to an analog input then the screen buttons to adjust the values of these will not function, because then the value is derived from the analog input.

To disable any of these functions set the channel to 0 value.

**Var# → analog output 1.** This setting makes the software to load the set variable's value to the analog 1. output channel. The valid value ranges are 0 to 65535 (16bits). Writing 0 to the variable causes the analog output to be 0 Volts and writing 65535 to the variable will make the analog output to go to 10Volts.
Variables can be programmed in code execution or via MDI command.
For example to make the variable #1 to get a value of 1000, command #1 = 1000
To disable this function set the variable number to 0.

**Var# → analog output 2.:** This setting makes the software to load the set variable's value to the analog 2. output channel. The valid value ranges are 0 to 65535 (16bits). Writing 0 to the variable causes the analog output to be 0 Volts and writing 65535 to the variable will make the analog output to go to 10Volts.
Variables can be programmed in code execution or via MDI command.
For example to make the variable #1 to get a value of 1000, command #1 = 1000
To disable this function set the variable number to 0.

**Spindle PWM → analog channel.:** This setting attaches the PWM spindle control PWM to the analog output. The analog output channel will have a value of the spindle PWM duty cycle.
For example if the minimum PWM is running at 50% duty cycle then the analog channel will have a 5Volts output which is the 50% of the 10Volts maximum.
It must be noted that this setting overrides the other settings for the analog outputs which means that if this function is enabled then other functions cannot write to the selected analog channel.
To disable any of these functions set the channel to 0 value.

**MPG A port & pin:** This is a physical input, the pin of the A input channel of an externally connected manual pulse generator (MPG).
The MPG should produce incremental quadrate A and B channel signals.

**MPG B port & pin:** This is a physical input, the pin of the B input channel of an externally connected manual pulse generator (MPG).
The MPG should produce incremental quadrate A and B channel signals.

**MPG prescaler:** This is a numeric value and sets the number of encoder ticks when the axis will move when the MPG hendwheel is rotated. Setting the value to 1 value will move the axis for every encoder ticks.

**MPG filter constant:** This is a numeric value and with this setting the smoothness of the axis motion for the MPG jog can be set. The higher value means a larger time constant and therefor a smoother motion even if the operator hand movement is not smooth, however the larger the value the slower the MPG will react. It is adviced to find the optimum where the MPG motion is smooth enough and where the reaction time is still not too slow.

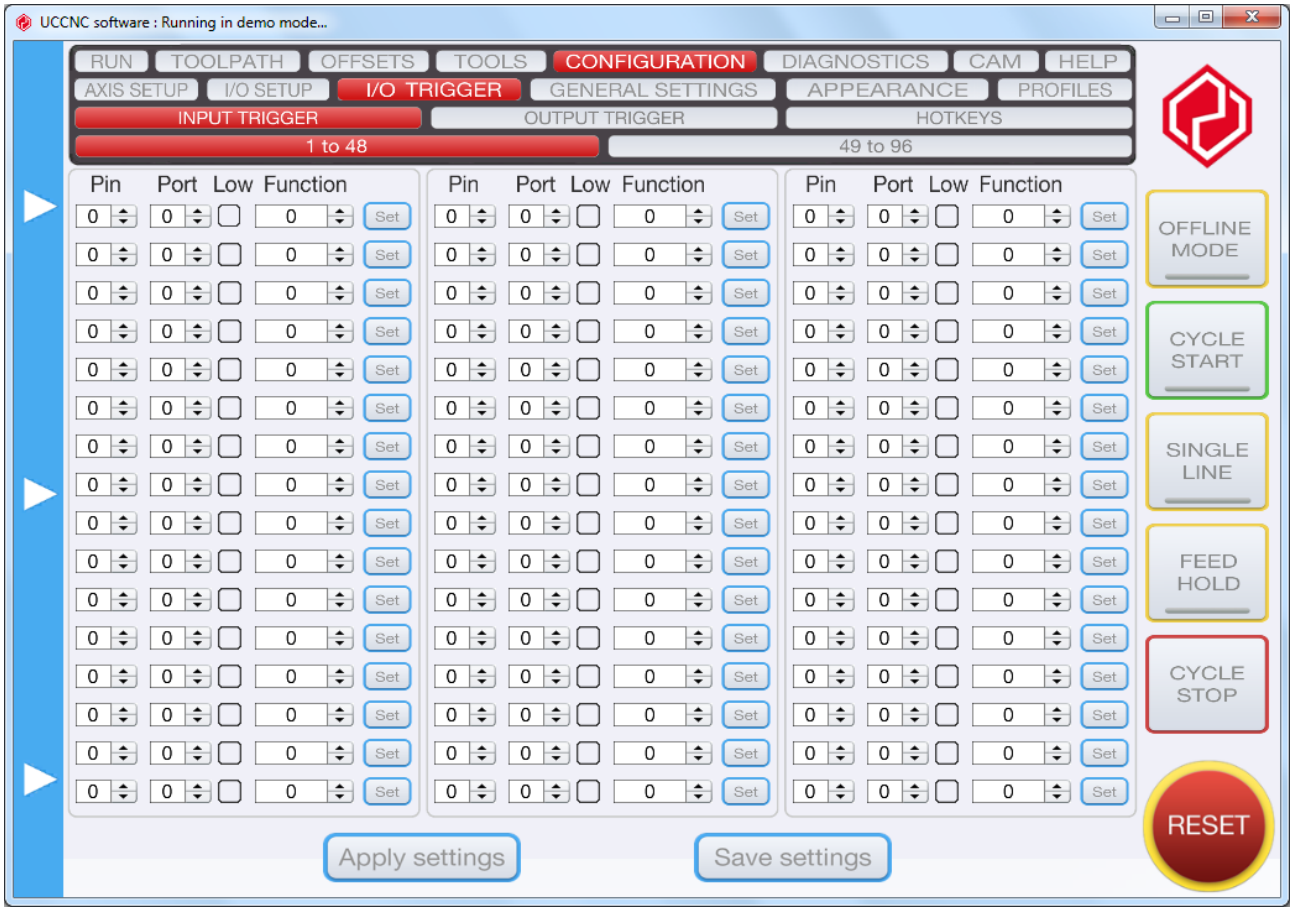
**MPG speed multiplier:** This is a numeric value and it sets how fast the axis will travel for the MPG wheel rotation. Using a higher value will run the machine fast and long for even small rotation on the encoder wheel. The optimal value for this parameter depends on several factors, like the encoder resolution and the machine type. For metal working machines it is mostly more practical to use a low value to let the machine move slower and shorter distances for the MPG rotation to let the operator position the machine very precisely.
For large scale wood routers it is mostly practical to set this value higher to let the axis move the long distances faster.

## 3.5 .IO trigger

The IO trigger function contains 3 sub pages which all makes the user interaction with the software easier.

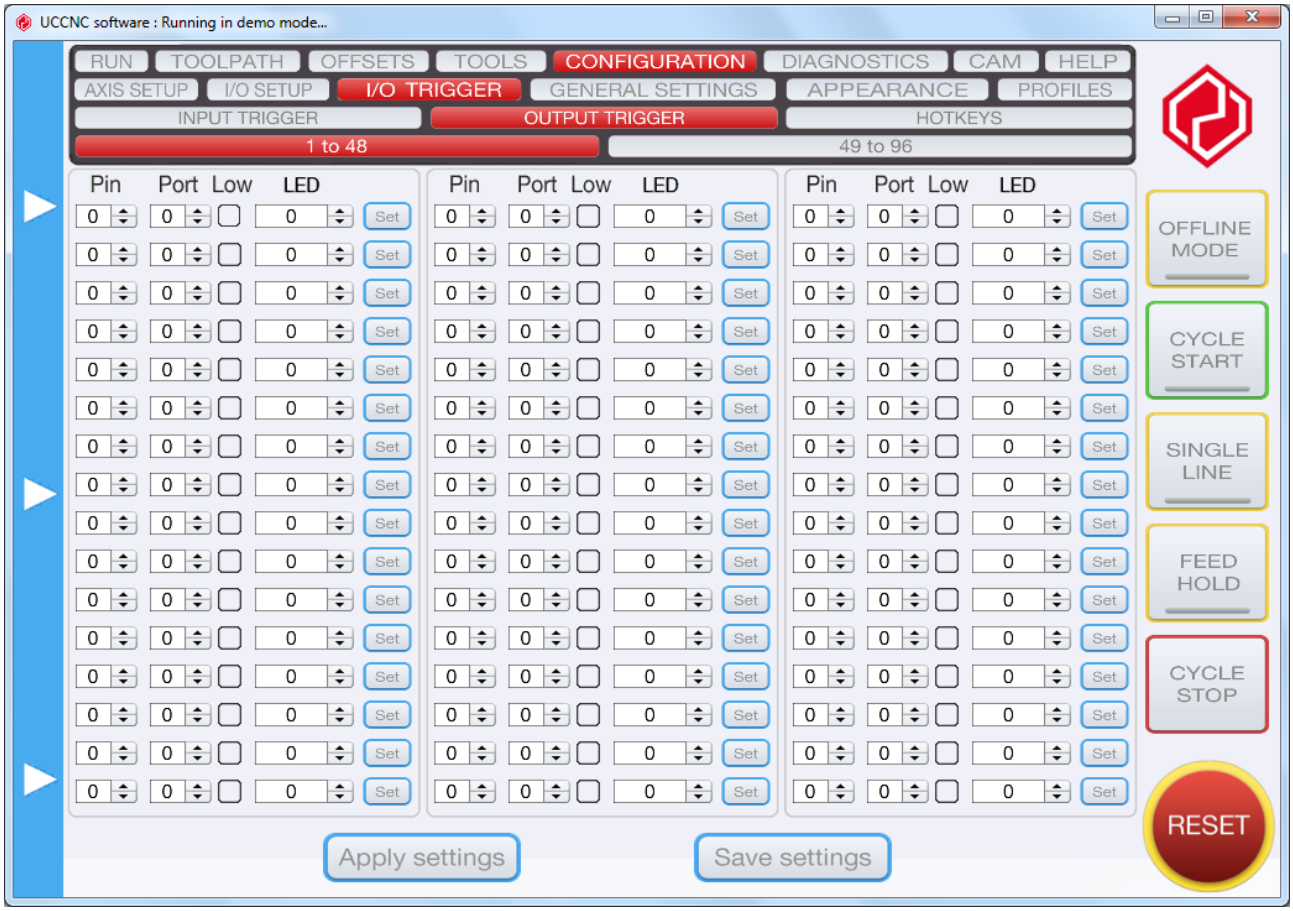
### 3.5.1 . Input trigger

The input trigger functionality makes is easy to attach external pushbuttons or switches to call internal functions of the software.
The input port pins can be attached to the UCCNC function numbers. When the input is triggered the UCCNC runs the selected function. If the "Low" setting is marked then the input triggers on the falling edge (when the input goes from 5Volts to 0Volts) otherwise it triggers on the rising edge (when the input goes from 0Volts to 5Volts) of the signal. All function numbers are listed in the documetation subfolder of the UCCNC installation, in the Buttons_by_number.htm document.
An example is shown on the below printscreen picture, where Pin13. On the Port1. Is attached to the 128. function which is the Cycle Start function of the UCCNC software.
The signal is set active low which means that if the input goes from 5V to 0V then the Cycle start function is called by the software and if a g-code is loaded then it will start running.
Calling the 128. function does basicly the same as if the Cycle start button was pressed on the screen.



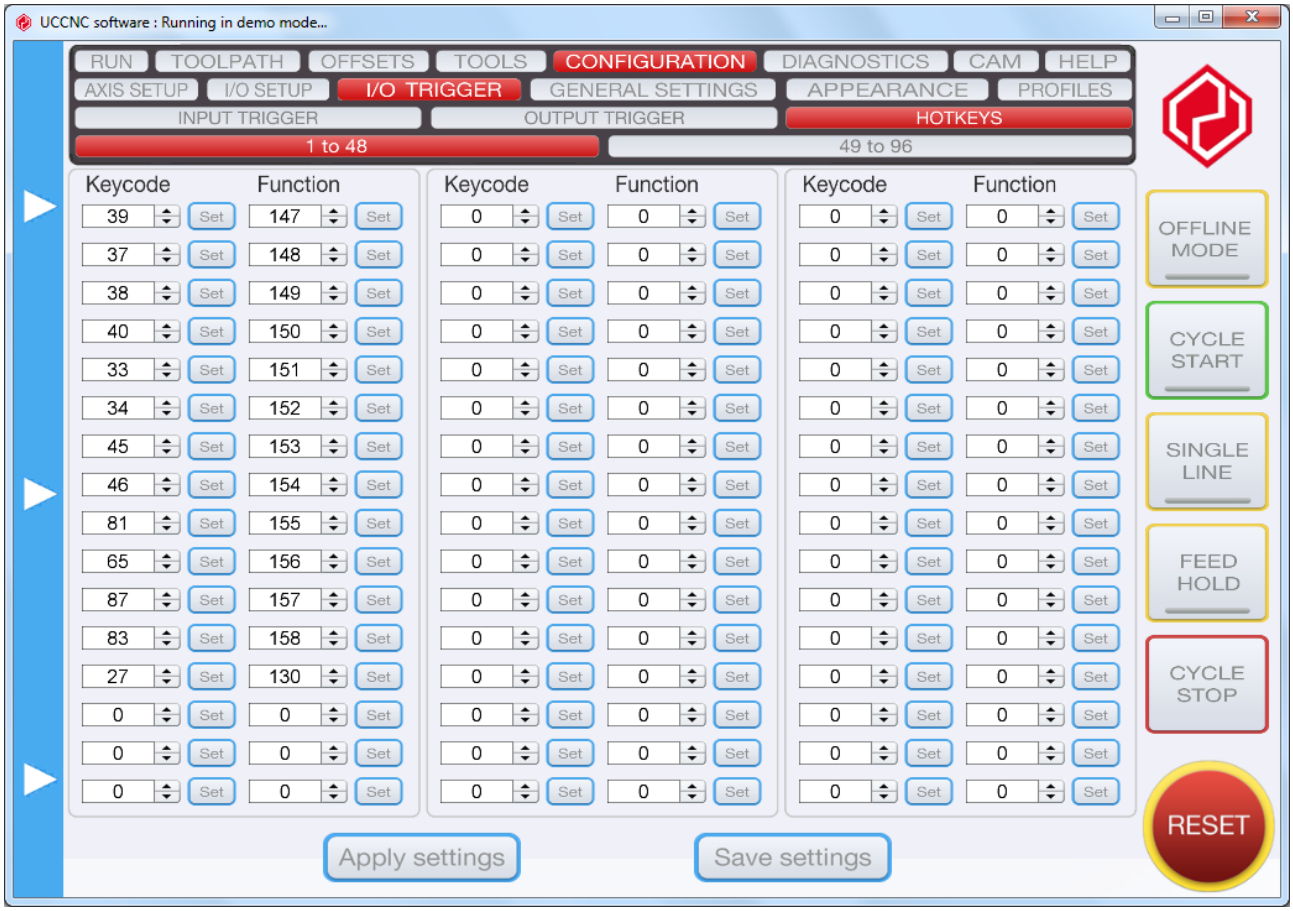

### 3.5.2 .Output trigger

The output trigger function attaches LED codes to physical output pins of the motion control device. The actual logic state of the LED is read and is reflected to the selected output pin. If the 'low' field is set then the signal gets inverted before the output.




### 3.5.3 .Hotkeys

To Hotkeys function attaches keyboard key triggers to UCCNC function calls.



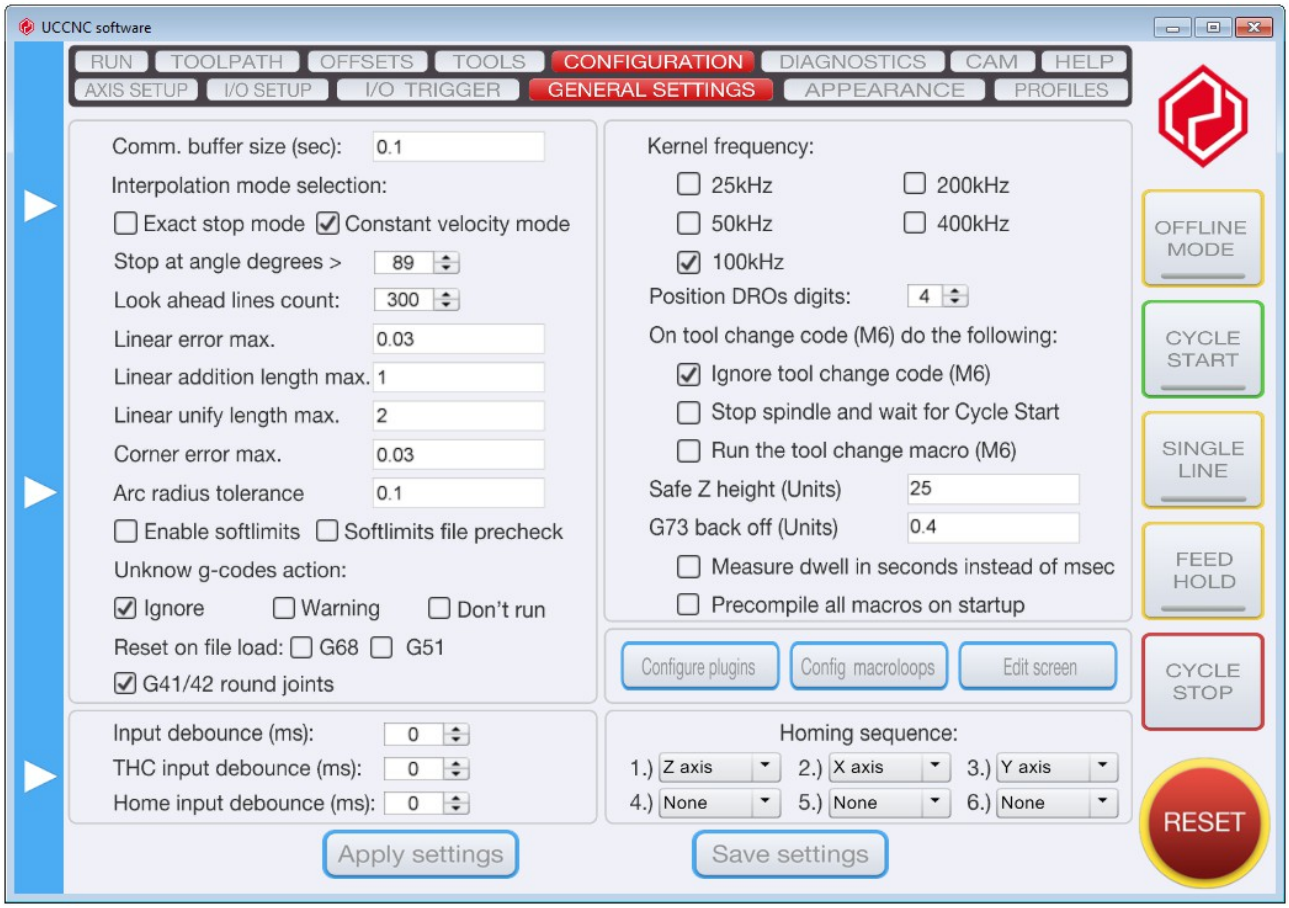

## 3.6 .General settings

This tab page contains settings for the machine behaviour.

The settings available on this page are the following:

**Communication buffer size:** This parameter sets the length of the USB communication buffer. The parameter is in seconds. The shorter the buffer is the faster the machine will react to button presses. The minimum set value is 0.05seconds(50milliseconds) and the longest is 0.5seconds(500mseconds).
Because the motion core of the software is running in a special high priority loop in our testings we found that 0.1seconds (100milliseconds) buffer length is enough in all cases and we recommend to leave this parameter at this value, but we left the ability to change this setting for the most flexibility of the software.

**Interpolation mode selection:** This parameter selects how the trajectory planner executes the motion. There are two modes, the exact stop mode and the constant velocity mode.
These two different modes can be also selected executing the G61.1 (exact stop mode) and G64 (constant velocity mode). When these codes are executed the selection on the screen changes accordingly. When a G64 code gets executed the parameters for this command are read from this screen settings.



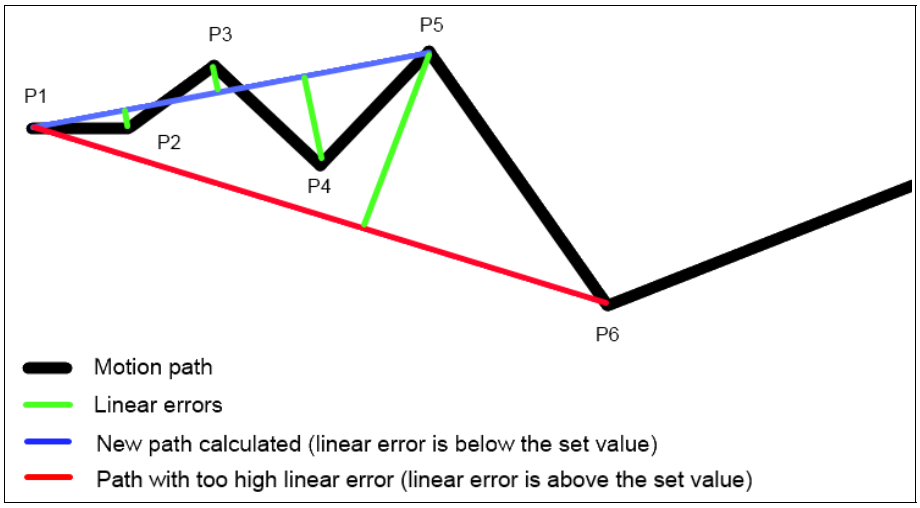

**Exact stop mode:** This is a trajectory planning mode. In this mode the trajectory planner follows the motion path positions exactly. The machine axis accelerates at the beginning of each segment and deccelerates at the end of each segment. This interpolation mode is adviced to set if close tolerance parts are manufactured. The position error is always zero with this interpolation mode.

**Constant velocity mode:** This is a trajectory planning mode. In this mode the trajectory planner is interpolating to the set feedrate, in other words the trajectory is planned looking ahead in the motion path and trying to keep the feedrate constant on the set feedrate.
For this mode to work some additional parameters needs to be defined which are discussed in the followings. The constant velocity mode is adviced to be used when high speed machining is nessessary for example 3D artworks are machined. This mode can also be used for close tolerance parts machining, but the tolerance parameters for this mode must be set correctly to force the motion planner to not make higher position errors than what is allowed for the workpiece.
There is always a tradeoff between machining speed and precision. The higher the allowed machining errors are set the faster the job will be finished, because the motion planner has more space for optimising the motion, however the less the workpiece precision might be.

**Stop at angle degrees:** This parameter works only if the Constant velocity mode is selected for the mode of interpolation. The parameter is in degrees and means a physical angle.
When set in constant velocity mode the motion planner looks ahead in the code it will execute which means it sees the code it will execute in the near future. The stop angle parameter defines that if a higher connection angle on one line segment to the other segment happens then stop the looking ahead at that point and only interpolate the previous segments. The looking ahead will start again after the point of that too high connection angle. Setting this parameter lower can reduce the rounding error on between line segments with high connection angle or in other words on sharp edges.

**Look ahead lines count:** This parameter works in constant velocity mode only and defines how far the motion planner looks ahead in the code, how many line segments in maximum should it precheck. Setting the parameter to a high value makes a better constant velocity interpolation especially when the motion path is built from short segments and when a long lookahead is nessessary to see even a short distance in advance. The higher the look ahead lines are set ofcourse the higher the comuputer's CPU and memory usage will be because it takes CPU time and memory to look ahead. The default setting is 200 lines which works pretty good in most cases.

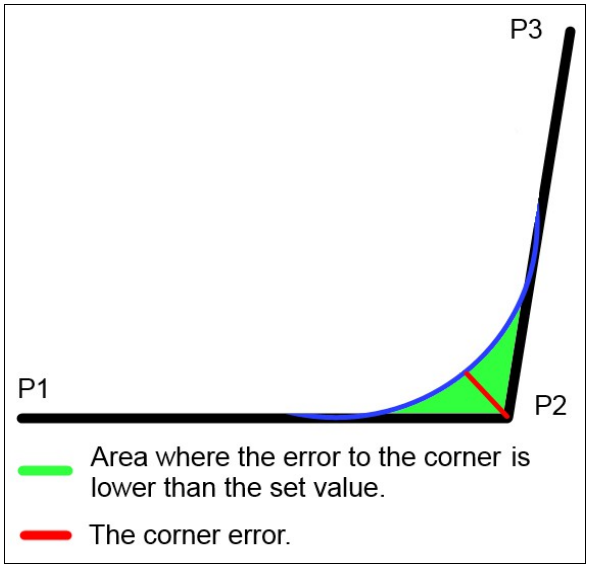
**Linear error max.:** This parameter sets the maximal linear error for the interpolator and works in constant velocity mode only. The higher the parameter is set in most cases the faster the job will finish, because the interpolator will shorten the motion path as much as possible with the linear error setting.
The following diagram shows an example path and demomstrates how the linear error checking works. The diagram is a simplified view, because at each new point in the path all the previous non executed path points are checked. (these are not drawn).
On the diagram the blue colored path will be a new path and the red colored path shows the path where the linear error is higher then the set value, this path gets ignored and the blue path gets executed. The linear error calculation is then starts from the point P5 again.
The parameter is in Units.




**Linear addition length:** This parameter is in direct conjunction with the "Linear error max." parameter and defines the maximal vector length which the interpreter can add to the unified path when the machine is moving in constant velocity mode and when the next line segments fits other parameters. This parameter is to limit the position error when interpolating long lines, in other words these long lines (which can likely be sides of parts) can be excluded from the look-ahead controlled unify process, keeping these lines precise even if the machine is moved in constant velocity mode. This parameter has a meaning and work only in constant velocity interpolation mode.

**Linear Unify length:** This parameter is similar to the "Linear addition length", but it limits the length of the unified line segment. In other words the unified vector cannot be longer than defined in this parameter. The purpose of this setting is the same as the purpose of the "Linear addition length" parameter. This parameter has a meaning and work only in constant velocity interpolation mode.

**Corners error max.:** This parameter sets the maximal interpolation error at the connection between two path points and works in constant velocity mode only.




The motion path is calculated in a way that the new faster path contains the error at these connection points lower than the set value. In the following graph an example connection and new path calculation is demonstrated. The green area is where the corner can be rounded. The machine can move anywhere in the green marked area and the fastest possible path will be selected by the motion planner. The parameter is in Units.

**Arc radius tolerance:** This parameter sets the tolerance of the arcs radius which arcs are programmed in radius (R) mode. When an arc is programmed with the radius (R) parameter then it is possible to set the radius too small to define a center point between the start and the end points of the arcs. If the radius is smaller than the half of the distance between the start and the end points then the arc cannot be defined mathematically. CAM programs often not defining the radius precise enough (rounding) and so the radius can become too small which would produce an arc error. If the software sees that the radius is too small to define the center point and therefor to create the arc then the software will calculate the midpoint between the start and the end points and will measure the center point distance to them.
The software will only give an arc error if the calculated center point is more far from the start and the end points then the radius plus the arc tolerance parameter. If the calculated center point is closer then no arc error will be given and the arc will be created with the calculated center point.

**Enable softlimits:** Setting this checkbox enables the software position limit on all axis. If this checkbox is not set then the softlimits are disabled. If the checkbox is set the software limits gets enabled and carry the value set on the axis setup page.

**Unknown g-codes:** The loaded g-code program may contain g-codes which are unknown, not supported by the UCCNC software. This setting allows the user to configure the software what to do when any unknown g-codes are found in the g-code program. There are 3 warning levels:

1.) **"Ignore"**, selecting this mode the software will ignore the unknown g-codes and will run without giving any warnings.
2.) **"Warning"**, selecting this mode the software will give a warning message asking the operator what to do, to run the code or not.
3.) **"Don't run"**, selecting this mode the software will give a warning message and will not allow to run the g-code program.

**Reset on file load G68 and G51:** If these checkboxes are checked cancels the G68 coordinate system rotation and resets the G51 scale factors to 1 prior to loading a new g-code files.

**G41/G42 round joints:** If this checkbox is checked then in the G41 and G42 tool radius compensation routine the walkaround on high angled vector connections on the path happens with circular shape instead of square.

**Input debounce:** This parameter filters the input signals with a low pass (RC) filter algorithm.
The value of the parameter sets the input filter time constant. If the parameter is 0 then no filtering is applied. The parameter can have a range of 0-255 milliseconds.
When the input signal changes then the internal state of the signal only changes if the signal's new state is valid for the set amount of time. The filter is useful when dealing with noise spikes on the input pins. Noise spikes shorter than the input filter time constant are filtered out.
The Input debounce parameter is for the Limit inputs and for the Reset input. Signals which require fast state changes like the encoder inputs are not debounced, this filter is not used on them.
This parameter is currently only implemented in our ethernet motion controllers. Our USB controllers do not have this parameter and functionality.

**THC input debounce:** This parameter works the same as the input debounce parameter, but it applies on the THC input signals.
This parameter is currently only implemented in our ethernet motion controllers. Our USB controllers do not have this parameter and functionality.

**Home input debounce:** This parameter works the same as the input debounce parameter, but it applies on the Home input signals.
This parameter is currently only implemented in our ethernet motion controllers. Our USB controllers do not have this parameter and functionality.

**Homing sequence:** The order of the homing sewuence can be set here. There are 6 slots where the different axis names can be selected. When the Home all command gets executed the axis get homed in the order selected here. The first axis in the first slot gets homed first and the axis in the sixth slot gets homed last. If not all home slots needed then the value of "none" can be selected then those slots will be unused.

**Kernel frequency:** The maximum frequency what the UC100 device should output can be defined with this setting. The selectable options are 100kHz, 50kHz and 25kHz.
This parameter defines how many pulses per second the controller should produce. For example with the 100kHz setting the maximum steps per second is 100 000 which means a 10microseconds total time period for each pulse. With the lower frequencies the signal period is longer which makes the setting better suite of drives built with low performance (slow) optocouplers.

**Position DROs digits:** This parameter sets the decimal places to show in DROs. The value can be 0 to 6 decimal places. The numbers shown are then rounded on the screen view, but they are not rounded when used for the motion, in other words the numbers rounding only happens on the screen.

**On tool change code (M6) do the following:** This setting defines how to handle the M6 tool change codes. There are 3 options to select from:
1.) Ignore tool change code(M6)
Selecting this mode will ignore all M6 codes in the program execution and also if placed into the MDI. The M6 codes are simply skipped. This mode should be selected if the machine has no automatic toolchanger and if the tool change macro means nothing for the machine.
2.) Stop spindle and wait for cycle start
Selecting this mode will stop the motion at any M6 codes and the software will wait for the operator to press the Cycle start button. When the Cycle start button got pressed the code execution continues.

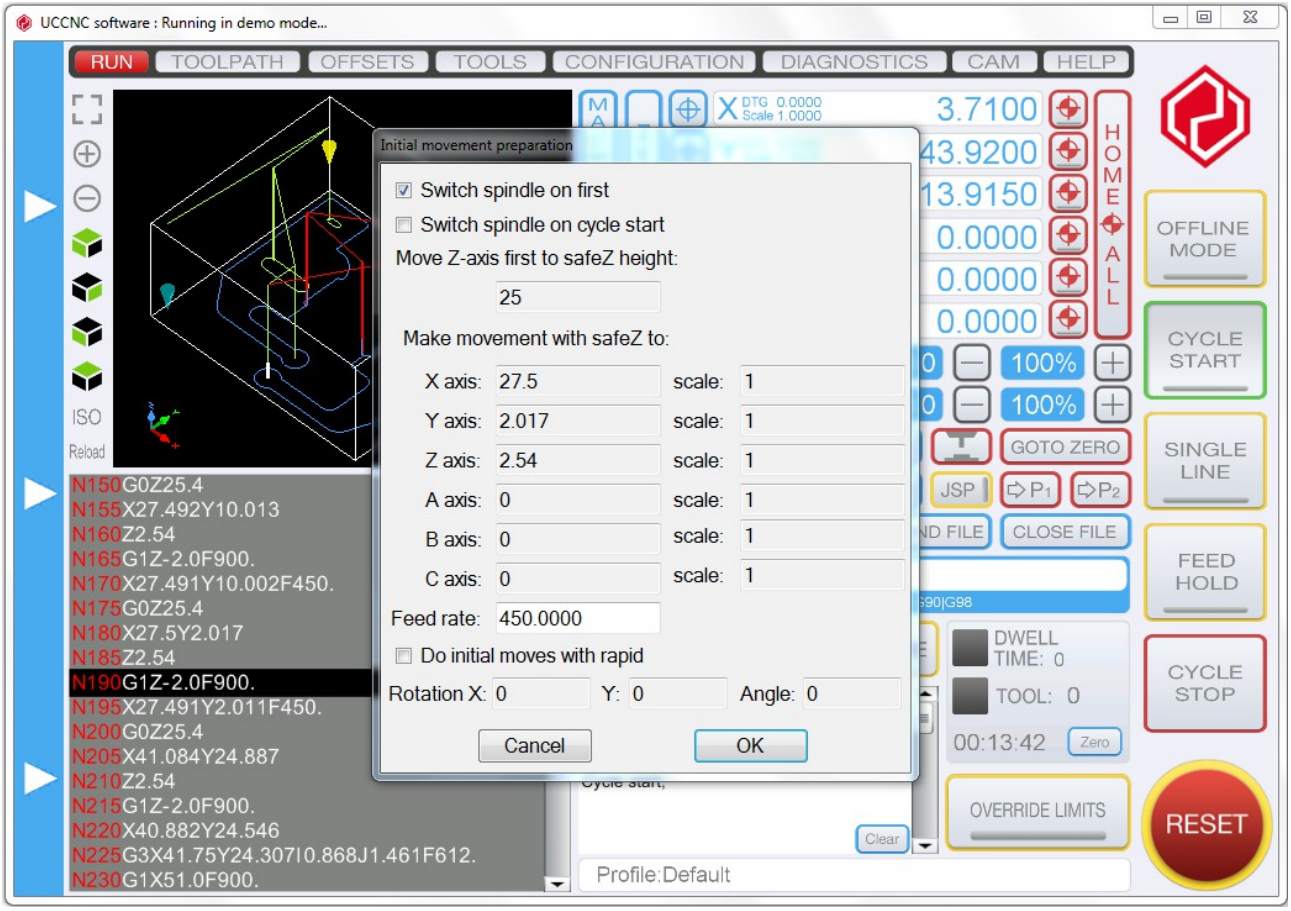
3.) Run the tool change macro(m6)
Selecting this mode will read and execute the M6 macro. The M6 macro is a text file and is located in the Profiles\Macro_Name of profile\ folder, where the "Name of profile" is the profile name the machine is running. The M6 macro as other macros can contain even complex code to execute a complete tool change sequence. For more informations about writing macro codes please read point 5. in this manual and the Macroing_capability_detailed.pdf document.

**Safe Z height:** This parameter defines the Z-axis position which is high enough to safely travel of the X and Y axis with no risk of a tool collision. This setting is used when the G-code program is stopped and the code execution pointer is changed by the user and the "run from here" button was pressed. In this case an initial movement is produced moving the machine to safe Z height first and moving the machine to the next XY coordinate point.

**G73 back off:** This parameter sets the distance to back the tool off when a G73 peck drilling g-code is executed.

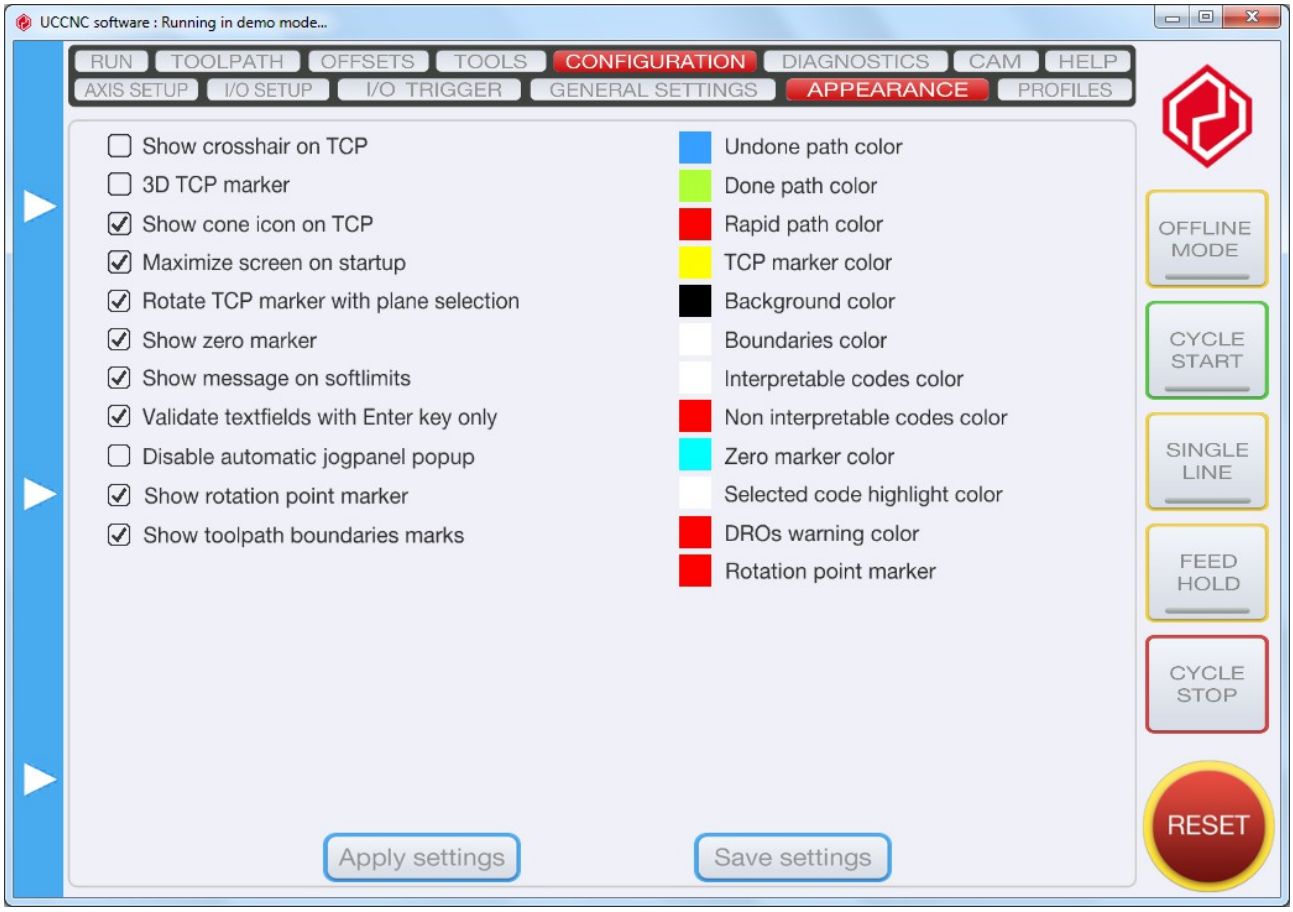
**Measure dwell in seconds instead of msec:** By default the time for dwell is measured in milliseconds. Ticking this option will measure the time in seconds, so executing a G4 dwell command will dwell for the P parameter defined amount of seconds instead of milliseconds.

**Precompile all macros on startup:** If this option is checked then all of the text macros in the profile folder are complied when the software starts up. The advantage of the macros being precompiled is that there is no delay when a precompiled macro is executed, because the macro is already in an executable code form and then the software only needs to execute the macro executable code. If this option is not set then the macros are compiled when they run the first time and the macro runs only after the compilation time which is in the 50-200mseconds for most computers. On the second and next macro runs the macro is already compiled and will run without a compilation delay.
If this option is set or not set in both cases the macros are always recompiled if the user changes the macro text. The software checks for the macro text change on every macro calls and recompiles the macro if the macro code text was changed.
Ofcourse this parameter only influances the text type macros which macros are coded in the profiles/macros folder. This parameter has no effect on the built in macros, for example on the M10/M11 fast laser control macros, because these macros are built into the software and do not need to be compiled on runtime.




## 3.7 .Appearance

The appearance screen contains settings about the look of the toolpath viewer screen and the G-code viewer screen.




The colors of the background, different paths type, tool center point viewer etc. can be set here. The following items color can be set:

**Show crosshair on TCP:** If this checkbox is set then the a crosshair is shown on the TCP marker.

**3D TCP marker color checkbox:** If this checkbox is set then the TCP markers move always at the top of the bounderies box and the Z-axis movement is shown with a straight line from the XY point to the Z actual height. If this checkbox is unchecked then the TCP markers move always on the actual Z height.

**Show cone icon on TCP:** If this checkbox is set then a cone icon marks the TCP.

**Maximize screen on startup:** Setting this checkbox makes the software to start in fullscreen mode.

**Rotate TCP marker with plane selection:** If this checkbox is set then the TCP marker orientation changes according to the selected plane G17, G18 or G19.

**Show zero marker:** If this checkbox is set then a cone icon is shown on the 0,0,0 zero coordinate.

**Zero marker color:** This is the color of the zero marker cone icon.

**Show message on softlimits:** If this checkbox is set then a popup window is shown when the softlimits coordinate is reached and the axis has to stop, otherwise only a status message is shown.

**Validate textfields with enter key only:** If this checkbox is set then pressing an Enter keyboard key is required to validate the new input values in textfields. If the checkbox is not set then leaving the checkbox will always validate the new value.

**Disable automatic jogpanel popup:** If this checkbox is checked then the jog control panel does not pop up automatically when the mouse pointer enters the panel's screen area, but a mouse click is required on the panel for it to pop up.

**Show rotation point marker:** If this checkbox is checked then the G68 rotation point is shown with a cone icon when the G68 code is active.

**Show toolpath boundaries marks:** If this checkbox is set then the boundaries of the toolpath is sorrounded with a frame.

**Undone path color:** This is the color of the toolpath which was yet not executed.

**Done path color:** This is the color of the toolpath which was already executed.

**Rapid path color:** This is the color of all rapid (G0) movements in the toolpath.

**TCP marker color:** This is the tool center point marker's color, which is a cross built from 2 lines and the 2 lines connects at the tool-center-point.

**Background color:** This is the color of the toolpath viewer's background.

**Boundaries color:** This is the color of the toolpath boundary box.

**Interpretable codes color:** This is the color of the G and M and other codes which the G-code interpreter understands and which codes are executable by the interpreter.

**Non interpretable codes color:** This is the color of the G and M and other codes which the G-code interpreter does not understand and which codes are not executable by the interpreter.

**Zero marker color:** This is the color of the zero point marker cone on the toolpath.

**Selected code highlight color:** This is the color of the current selected movement on the toolpath.

**DROs warning color:** This is the text color of the DROs when they are in warning mode. For example if the G68 rotation is active then the X and Y current position DROs are drawn with this color to make the user aware that the coordinates shown are in the rotated coordinate system.

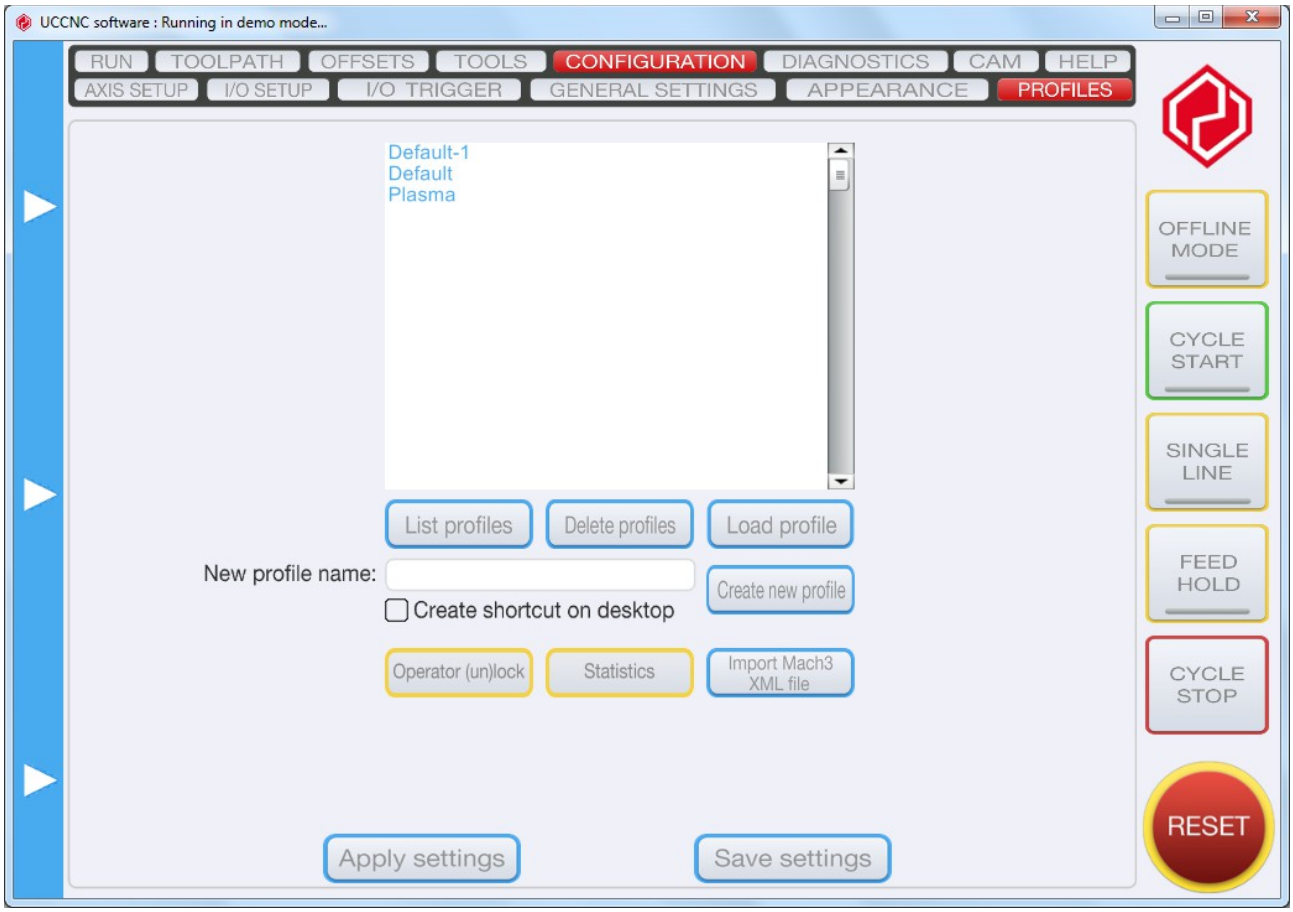
**Rotation marker color:** This is the color of the rotation point marker cone on the toolpath.

## 3.8 .Importers

The importers tab page contains settings import functions. Currently a Mach3 .xml setup file importer is available. Pressing the "Import Mach3 XML file" button a file dialog box appears and a Mach3 setup file can be imported to the UCCNC software. Loading a Mach3 .xml file imports all the settings to the UCCNC Configuration screen. The settings can be applied and/or saved pressing the "Apply settings" and "**Save** settings" buttons.

## 3.9 .Profiles

The profiles tab page contains the setup for the machine profiles.




Each profile can have different machine setup which means that one installation of the UCCNC software can be used for more than one machine not simultaneously. The Profile files are located at the UCCNC installation folder and the Profiles subfolder, the default file path is C:\\UCCNC\Profiles\ , but the installation directory can be different as it is changeable on the software installation.
The Profile files are named as the name of the profile and has a .pro file extension. Each profile is stored in a separated file. The .pro files are plain text files with key entries, the files are therefor editable manually too with a notepad, but we recommend not to edit it this way, because a mistype can cause the key to be unreadable for the software.
For each profile there is also a folder named Macro_profilename, where the profilename is the name of the profile. This folder contains the macro files for that profile.

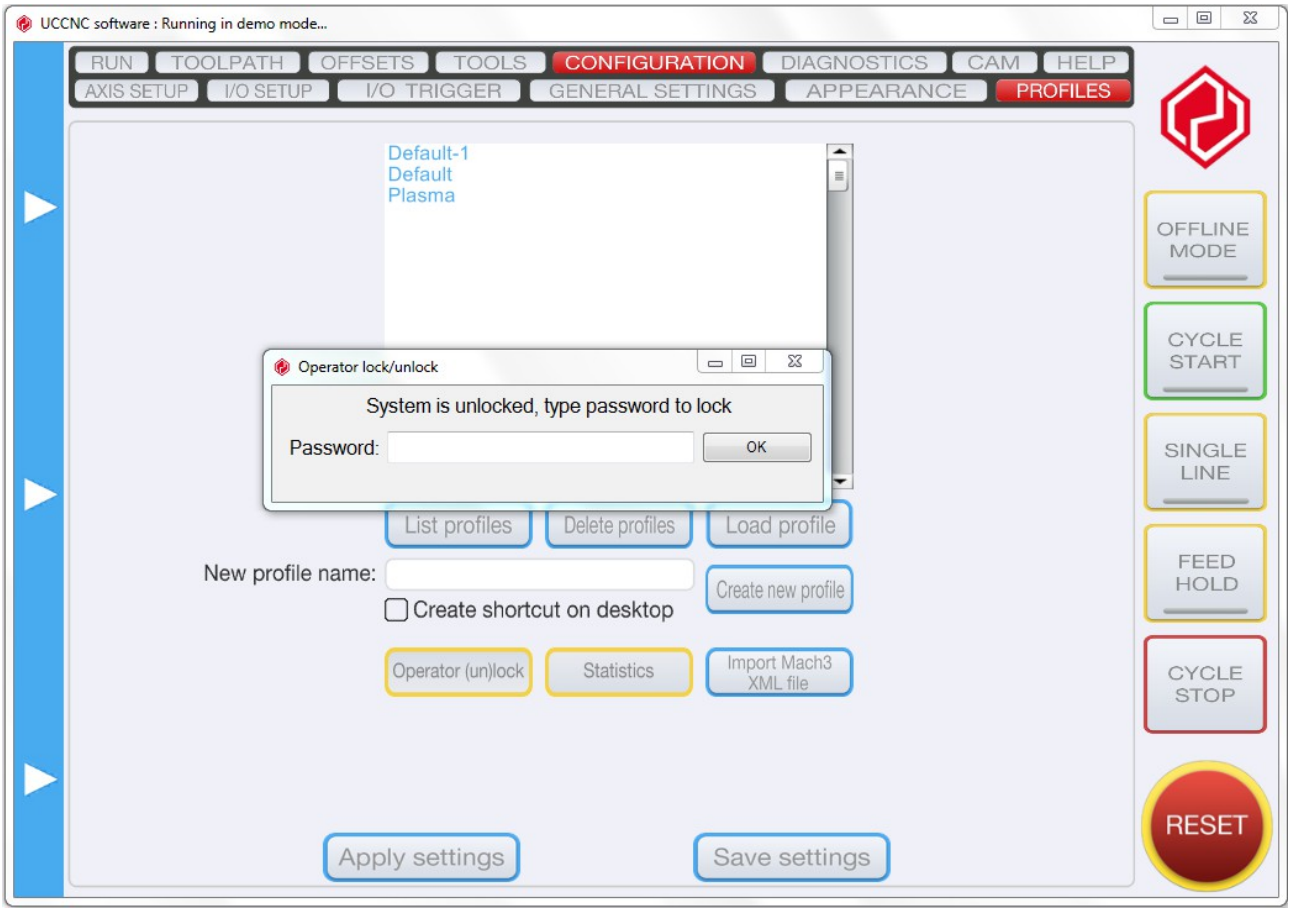
On the profiles tab page all profiles can be listed, new keys can be created, already made profiles can be deleted and existing keys can be loaded.
The following picture shows the profile tab screen:

Pressing the "list profiles" button lists all of the available profile names in the above listbox. To select a profile click the name of the profile in the listbox.
After selecting a profile clicking the "delete profile" button will delete that profile. Clicking the load profile will load that profile.
To create a new profile put the new profile's name into the textbox left next to the create profile button and press the create profile button. If the "Create shortcut on desktop" checkbox was selected then a shortcut icon with the profile name will be also created and will be placed on the desktop.

Double clicking that icon on the desktop will run the UCCNC software loading this profile automaticaly.
After creating a new profile a new .pro file gets created and placed to the UCCNC\Profiles folder. Also a new macro folder is getting created with the default macro files inside.

### 3.9.1 .Operator (un)lock

To lock the setting of the current profile press the Operator (un)lock button and type in a password. If a password is set then the settings can't be changed, because the Apply settings and Save settings buttons will not allow the new settings to be applied and saved, but the software will ask for the password. If the correct password is typed then the profile gets unlocked and it will again allow applying and saving the settings.
The following image shows the Operator (un)lock window:




### 3.9.2 .Statistics

The statistics window gives some details about the machine usage, so the operator can more easily plan the machine maintance work cycles.
The left side of the window lists the total run distance of every axis. The distance is in Units. The next column shows the total run distance of each axes with active spindle.

The times column shows the following informations:

- Total time, which elapsed with the software profile execution.
- Session time, which elapsed since the software profile was last time started.
- M3 time, which elapsed with the spindle on in the M3 (CW) direction.

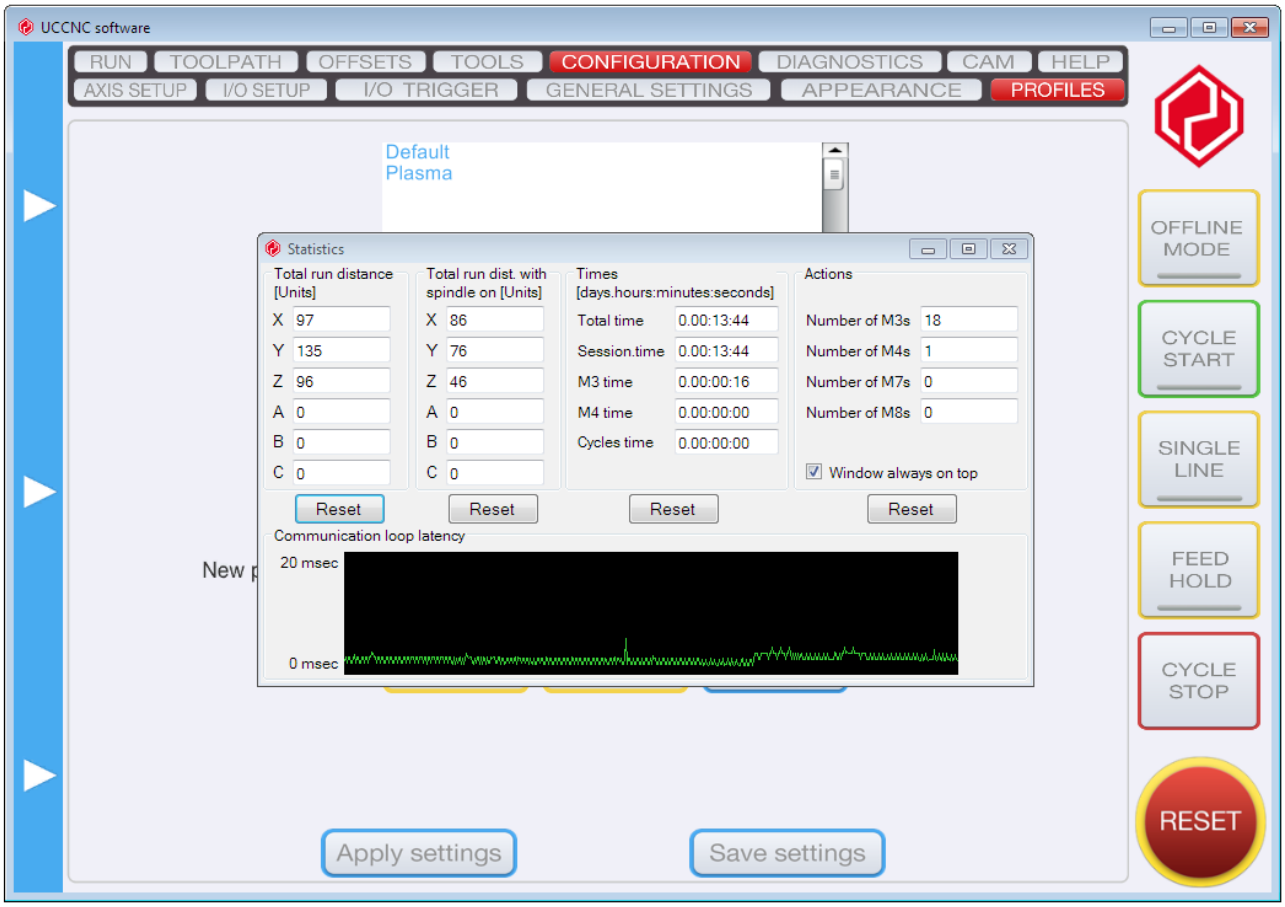
- M4 time, which elapsed with the spindle on in the M4 (CCW) direction.
- Cycles time, which is the total elapsed time with the Cycle running.

The Actions column shows the total switch times of the spindle M3 and M4 and the Mist and Coolant M7 and M8.

To reset the datas to zero press the Reset button under the columns.

On the bottom of the Window a graph is shown with the communication loop latency.
The graph is continuously updating and scrolling on the screen.
The graph shows a 0-20 milliseconds interval range of the communication loop.
The lower the values the better the latency of the loop is and the more the computer is sufficient to run the software.
If the graph values are all high is an indication that the computer might have not enough power to run the software without issues.

The following printscreen shows the Statistics window:



## 3.10 .Offsets

The offsets tab page contains all position offset related settings.
Currently the work offsets and a tool offset is available.
The following picture shows the Offsets page:


The offsets page contains 6 sub tab pages, these are the G54, G55, G56, G57, G58 and G59 offset pages respectively. These sub tab pages are identical and each of them is a different fixture.
The fixtures can be selected with programming G54 to G59 in the G-code program or in the MDI.

There are four columns on each fixture table, the first one is the Current coordinates which is the Machine coordinates substracted with the Work offsets. The Machine coordinates which is the absolute coordinate system's actual coordinates, the Work offset coordinates which are separately setable for each offset coordinate systems and the Current coordinates are offset with these values. The last column is the Tool offset which is the tool length compensation, this value is common for all fixtures and currently it is available for the Z-axis only.

The current position can be offset with a single button press by pressing the "Offset current position" button. Pressing this button offsets all the 6 axis with the actual fixture position.
All offsets can be cleared by pressing the "Clear work offset" button. Clearing the offset will change the current coordinates, the cleared offst values are added to the Current coordinates.

The tool offset can be cleared by pressing the "Clear tool offset" button. Clearing the tool offset adds the tool offset to the Current coordinates.

All the Current coordinates, Work offset and the Tool offset are editable on the screen.

## 3.11 .Tools

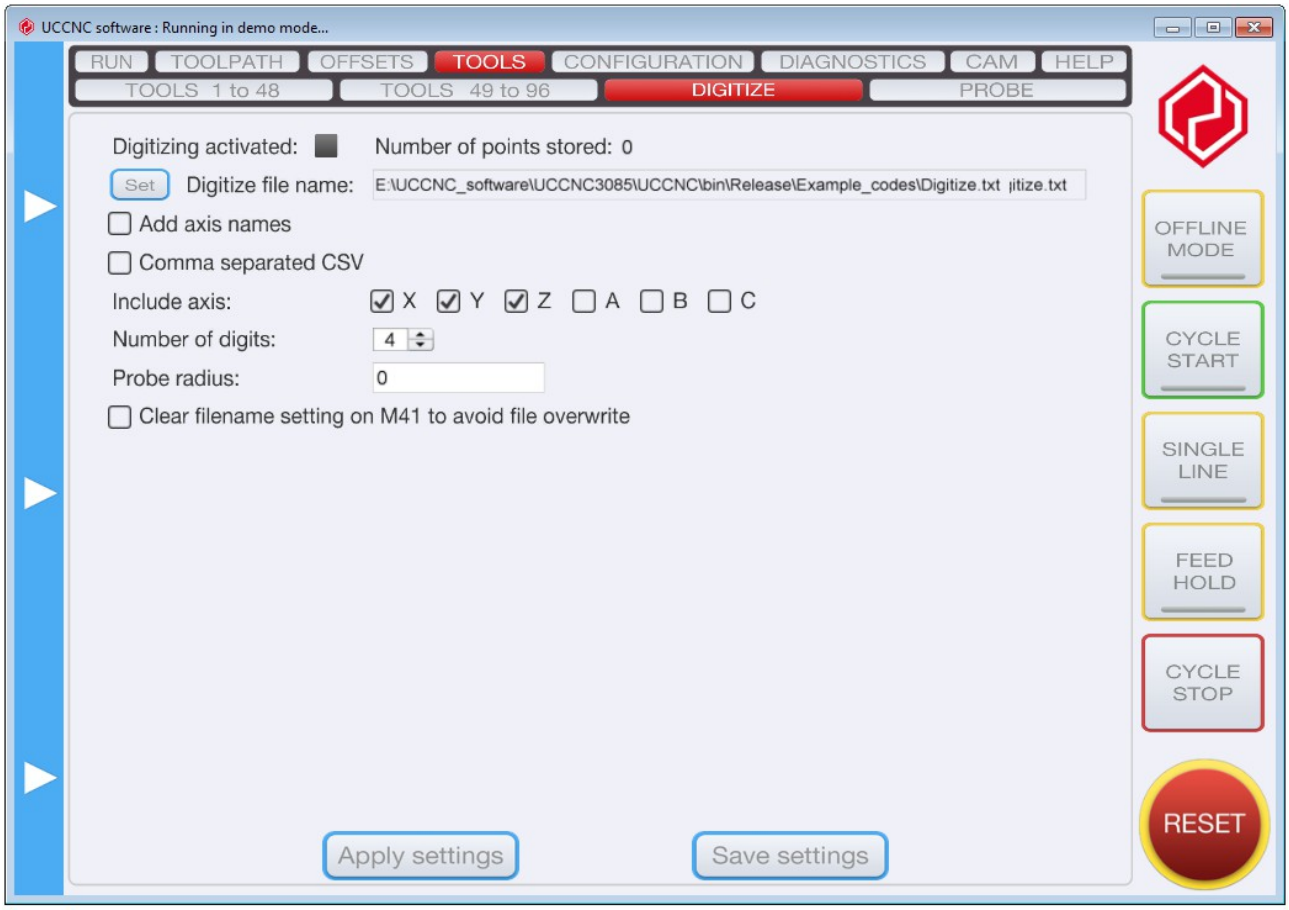
The tools page contains a tools table where there are 96 tool slots listed on 2 tab pages. The different tools length and diameters can be programmed on the screen. The different tools can be selected programming G43 H"tool number" or G44 H"tool number" in program or via the MDI. When programming the G43 or G44, the tool length of the selected tool will be offset to the Z-axis. The tool offset can be cleared programming G49 in program or via the MDI. The tool radius compensation can be programmed with the G40/G41/G42 codes.
The following picture shows the Tools page:


The tools page also contains the digitize settings on the digitize tab sub page.

The Digitize activated LED shows when the digitizing function is active. The number of points currently being stored in the digitize variable array is also shown. The digitizing can be activated with the M40 command and can be deactivated with the M41 command.

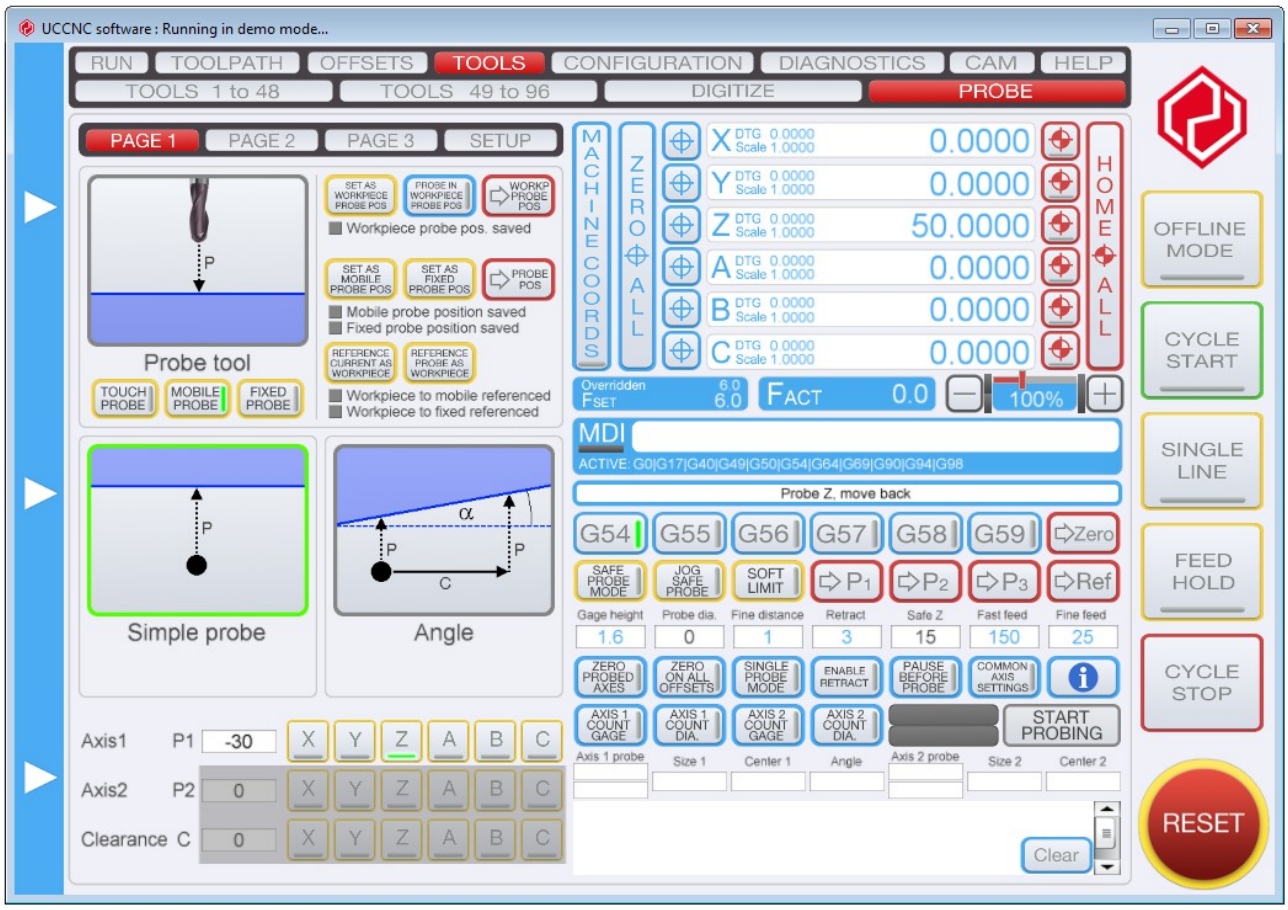
The Digitizing file name textfield shows the name of the file where the digitized points will be saved. A new upto 6 dimensions point is added to the points variable array when executing a G31 probing command. All points are saved to the file when the M41 command is called and then the points array is cleared.
Which axes coordinates are saved to the file can be also selected with the Include axis checkboxes and if the axis names has to be written before the coordinate numbers.
The Comma separated CSV options places a comma character between the coordinates.
The Clear filename setting on M41 to avoid file overwrite option clears the filename string when the M41 macro is called, so the filename has to be set again, this helps the user to not overwrite the already saved file with the next digitizing sequence.
If the probe radius is not 0 then the probed point coordinates are compensated with the radius value

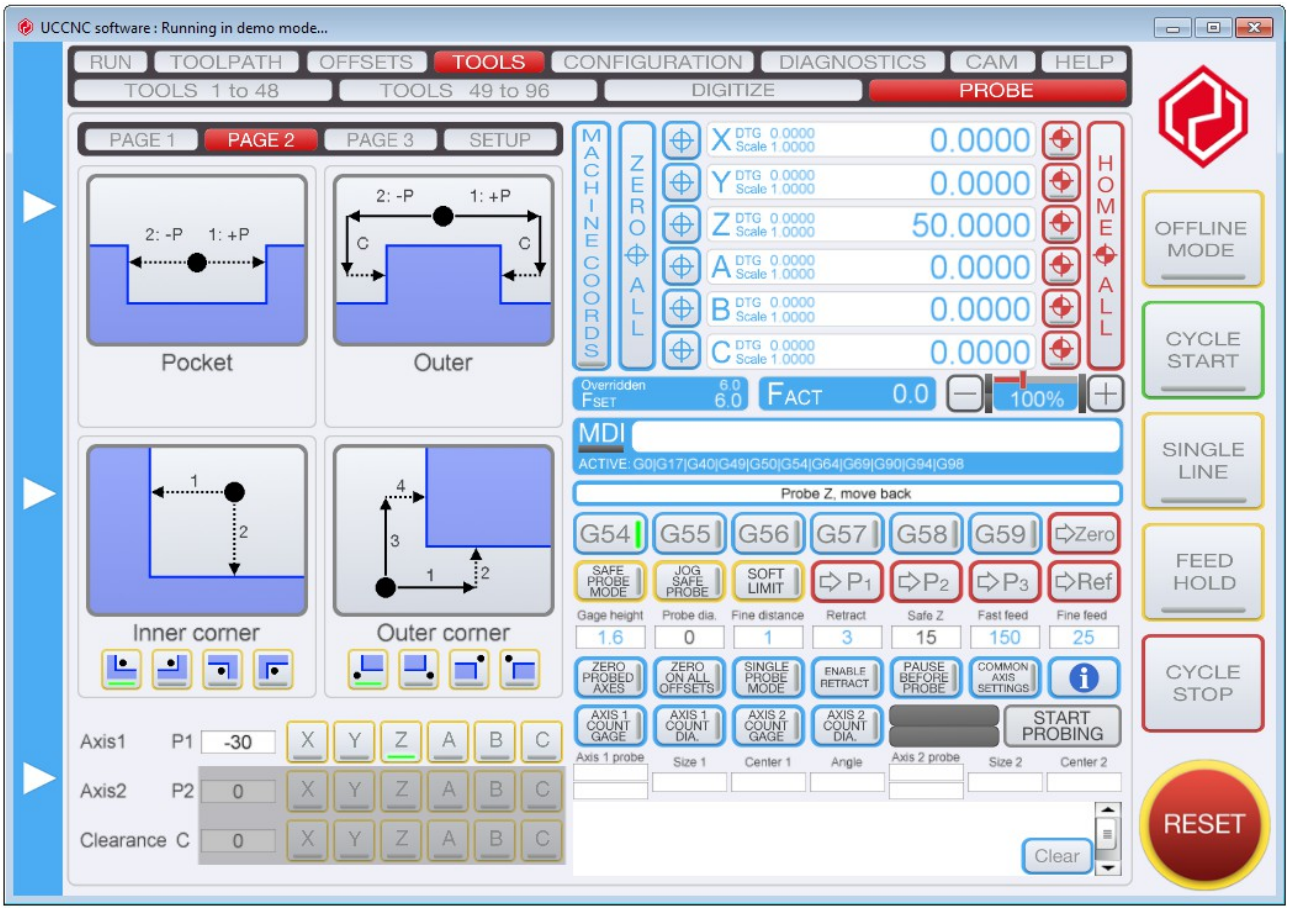
on the XYABC axes.
The following image shows the digitize settings tab page:

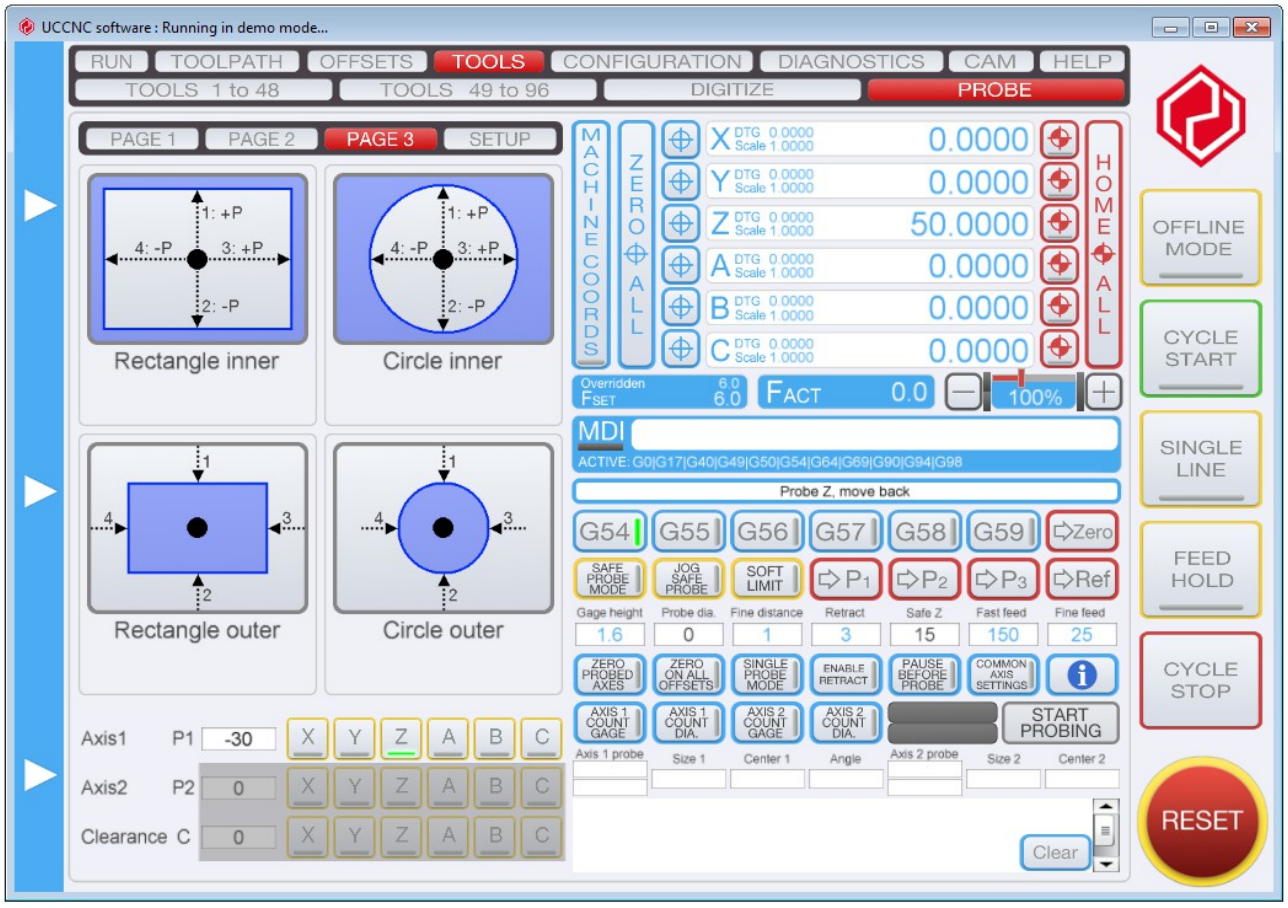


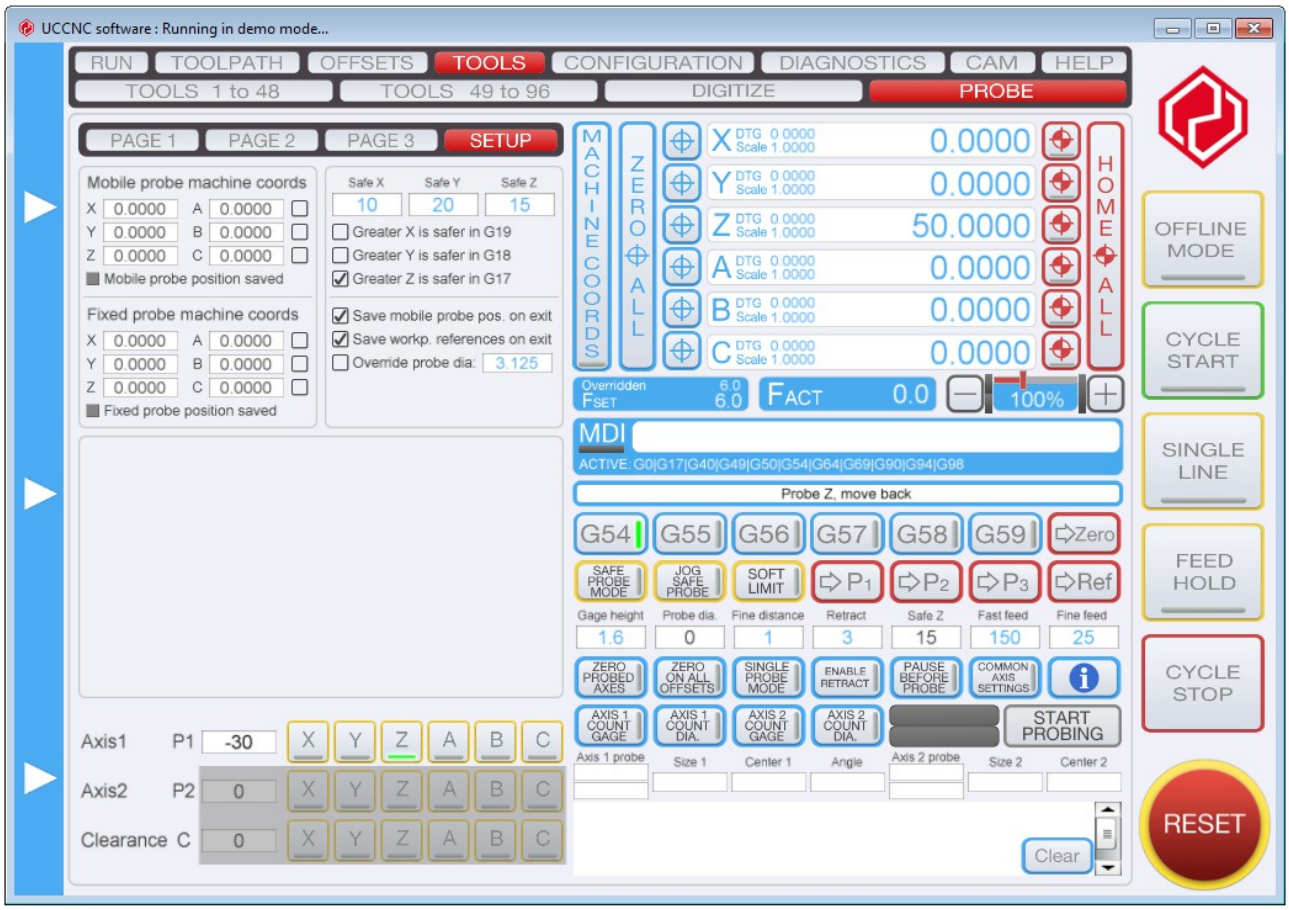

The last sub-page on the Tools tab page is the probe page which contains different probing routines and functions. The probe functions are documentated separately in the UCCNC/Documentation/Probe_help.mht document file. The same documentation can be opened with the information (I) button on the probing pages. To run the probe page routines the probing plugin has to be enabled in the Plugins configuration on the General settings page of software.

The following images show the probe tab pages:




UCCNC software : Running in demo mode...
RUN
TOOLPATH
OFFSETS
TOOLS
CONFIGURATION
DIAGNOSTICS
CAM
HELP
TOOLS 1 to 48
TOOLS 49 to 96
DIGITIZE
PROBE
PAGE 1
PAGE 2
PAGE 3
SETUP
Rectangle inner
Circle inner
Rectangle outer
Circle outer
Axis1 P1 -30
Axis2 P2 0
Clearance C 0
MACHINE COORDS
ZERO ALL
HOME ALL
Overridden FSET 6.0 6.0
FACT 0.0
100%
MDI
ACTIVE: G0|G17|G40|G49|G50|G54|G64|G69|G90|G94|G98
Probe Z, move back
G54 G55 G56 G57 G58 G59 Zero
SAFE PROBE MODE
JOG SAFE PROBE
SOFT LIMIT
P1 P2 P3 Ref
Gage height 1.6
Probe dia. 0
Fine distance 1
Retract 3
Safe Z 15
Fast feed 150
Fine feed 25
ZERO PROBED AXES
ZERO ON ALL OFFSETS
SINGLE PROBE MODE
ENABLE RETRACT
PAUSE BEFORE PROBE
COMMON AXIS SETTINGS
AXIS 1 COUNT GAGE
AXIS 1 COUNT DIA.
AXIS 2 COUNT GAGE
AXIS 2 COUNT DIA.
START PROBING
Axis 1 probe
Size 1
Center 1
Angle
Axis 2 probe
Size 2
Center 2
Clear
OFFLINE MODE
CYCLE START
SINGLE LINE
FEED HOLD
CYCLE STOP
RESET


UCCNC software : Running in demo mode...
RUN
TOOLPATH
OFFSETS
TOOLS
CONFIGURATION
DIAGNOSTICS
CAM
HELP
TOOLS 1 to 48
TOOLS 49 to 96
DIGITIZE
PROBE
PAGE 1
PAGE 2
PAGE 3
SETUP
Mobile probe machine coords
Mobile probe position saved
Fixed probe machine coords
Fixed probe position saved
Safe X 10
Safe Y 20
Safe Z 15
Greater X is safer in G19
Greater Y is safer in G18
Greater Z is safer in G17
Save mobile probe pos. on exit
Save workp. references on exit
Override probe dia: 3.125
Axis1 P1 -30
Axis2 P2 0
Clearance C 0
MACHINE COORDS
ZERO ALL
HOME ALL
Overridden FSET 6.0 6.0
FACT 0.0
100%
MDI
ACTIVE: G0|G17|G40|G49|G50|G54|G64|G69|G90|G94|G98
Probe Z, move back
G54 G55 G56 G57 G58 G59 Zero
SAFE PROBE MODE
JOG SAFE PROBE
SOFT LIMIT
P1 P2 P3 Ref
Gage height 1.6
Probe dia. 0
Fine distance 1
Retract 3
Safe Z 15
Fast feed 150
Fine feed 25
ZERO PROBED AXES
ZERO ON ALL OFFSETS
SINGLE PROBE MODE
ENABLE RETRACT
PAUSE BEFORE PROBE
COMMON AXIS SETTINGS
AXIS 1 COUNT GAGE
AXIS 1 COUNT DIA.
AXIS 2 COUNT GAGE
AXIS 2 COUNT DIA.
START PROBING
Axis 1 probe
Size 1
Center 1
Angle
Axis 2 probe
Size 2
Center 2
Clear
OFFLINE MODE
CYCLE START
SINGLE LINE
FEED HOLD
CYCLE STOP
RESET

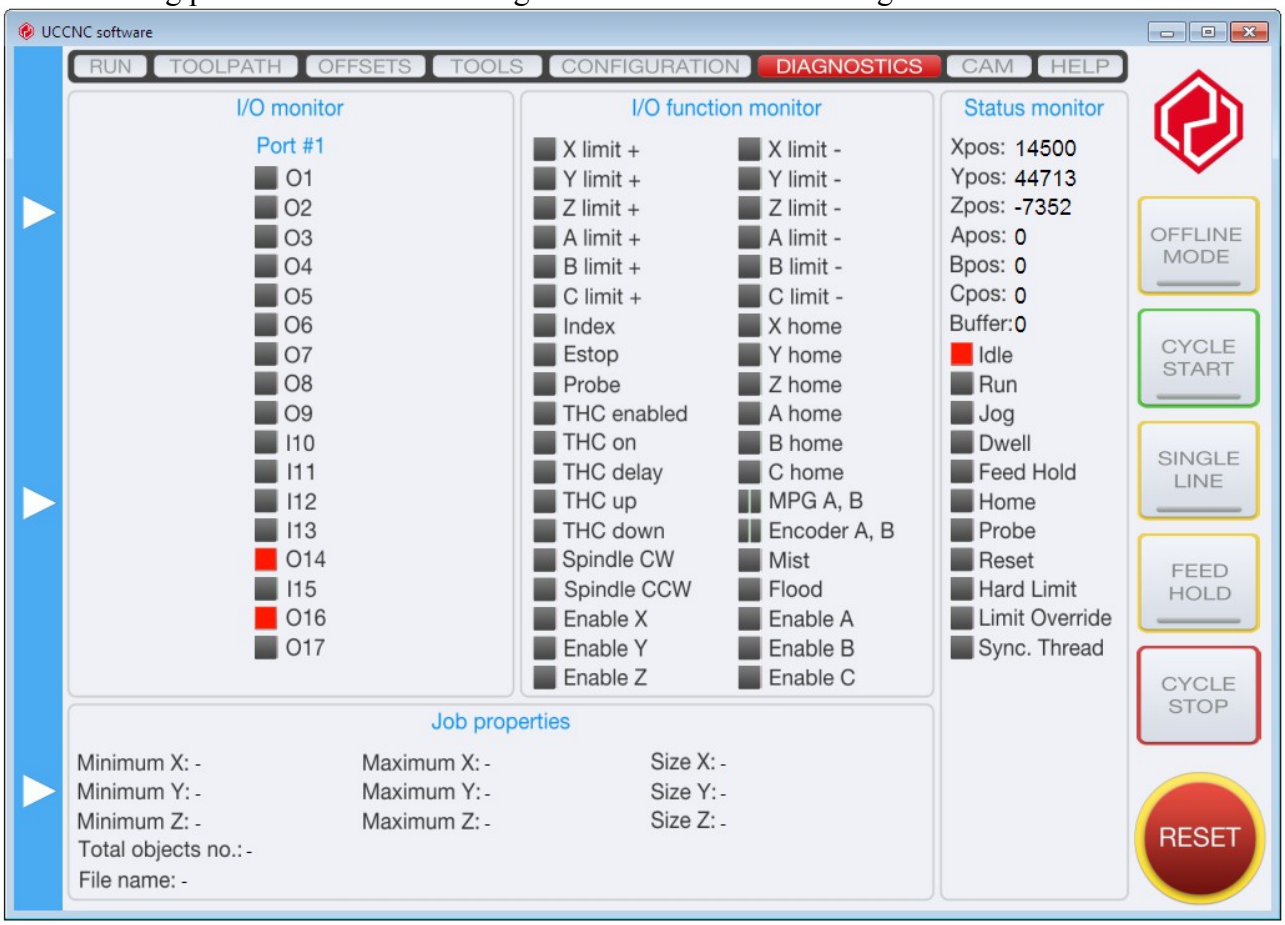
## 3.12 .Diagnostics

The diagnostics page contains visual feedback of the inputs and outputs and functions different state. The feedback values are shown with virtual LEDs on the screen.
The LEDs are sectioned into 3 columns, these columns are the following:

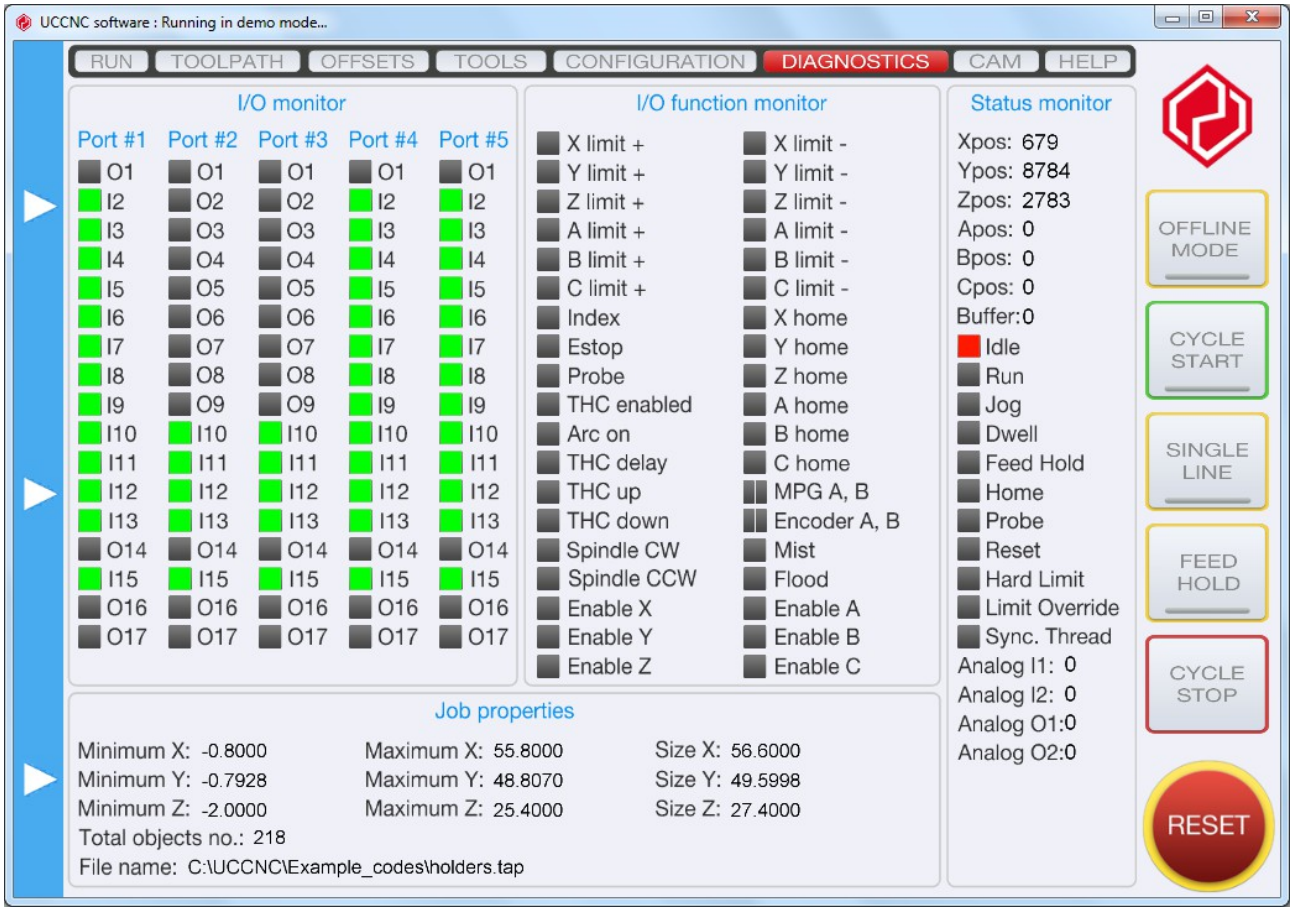
**I/O monitor column:** This culomn shows one LED for each input and each output pins on the UC100 device. The LED states correspond to the logic level of the in and outputs. If the virtual LED is grey means that the in or output is low. The input type LEDs are green colored and the output type LEDs are red colored when logic high.

**Status monitor column:** This cokumn shows the the most important flags in the UC100, there are virtual LEDs for the current running status of the controller, the status of the jog, dwell, backlash compensation, Homing, probing actions. Also it is showed if the machine is on any hardware limit switches and if the limit override function is active. Finally the spindle and the coolant on/off states are listed. On the right side of the same culomn the step counter DROs are shown, this counter shows the internal step counter values in the UC100 device, these counters have integer values and should not be mixed with the position DROs. On the bottom the actual fill of the data buffer is shown.

At the very bottom of this screen the loaded G-code (job) properties are shown. The G-code job properties lists the X, Y and Z axis limits, or in other words the minimal and maximal coordinates where the machine will move when executing the code. And the size of the part is also shown in the X, Y and Z dimensions. The "total objects number" lists the number of the interpreted objects to execute in the loaded G-code program. At the bottom the path for the loaded file is listed.
The following printscreen shows the diagnostics screen when running the software with the UC100:



The  diagnostics screen when running the software with the UC300-5LPT:




## 3.13 .Porting the settings to a different computer

It can be sometimes nessessary to copy the settings of the machine or to just port them to a different computer. Also it is always a good idea to keep a backup copy of the settings files once the machine was setup. In the UCCNC software porting the settings is easy, because all the settings are located in an external text file. This text file is located in the UCCNC installation directory /Profiles subfolder. The profile file name is always the name of the profile with a .pro extension. For example the Default named profile file is named Default.pro. Saving this file saves all of the machine settings and copying this file to the same folder on another installation of the UCCNC will copy the profile to that computer.
There are some other important files which are the macro files and these files are compiled and called when M (macro) codes are executed. Each profile has it's own macro directory and this directory is always named as "Macro_Name of profile" where the "Name of profile" is the profile's name. This folder also needs to be ported if macro macro calls will be used.

To make the profile ported installation complete and easy to use for the user a shortcut icon can be created to load the profile directly when double clicked on the Windows desktop or an old shortcut icon can be copied and edited. To let the shortcut icon load the UCCNC software, edit it's properties and set the target value to point the UCCNC .exe file's location which is by default the C:\UCCNC\UCCNC.exe, however the path may be different, because it can be changed at the software installation. Also set the /p switch for the shortcut icon and write in the name of the profile which the shortcut has to load when clicked. In summary, the shortcut icon's target icon to load the profile named testprofile1 should look the following: C:\UCCNC\UCCNC.exe /p testprofile
When double clicking the shortcut icon it runs the UCCNC.exe software and will pass the /p parameter with the "testprofile" value which will force the UCCNC software to load the profile

named testprofile.

# 4 .The code interpreter

The code interpreter can read the G-code program when opened from file and also it can read G-code commands if entered into the MDI. The interpreter first reads the code then compiles it to and internal structure list which it can execute as movements, I/O manipulations or whatever the code has to do when executed.

When the code is loaded from a G-code file the interpreter marks the codes it understood and which it can execute with a different color than the codes it cannot understand and can't execute.
This colored code text is shown in the g-code viewer control on the screen.
In default the codes which can be executed are marked with a white color and the ones which the interpreter can't understand and are not executable are marked with a red color by default, but these colors are configurable on the appearance settings page.

## 4.1 .Supported codes

The supported codes are the codes which the code interpreter understands and which it can execute. In the following sections these codes are listed with some descriptions.
When developing we followed the industry standard RS274 format and used the same codes which codes this standard implements.

### 4.1.1 .G-codes

In this section of the documentation the currently supported G-codes are listed, these are the codes which the code interpreter understands. The list of the supported G-codes will change, we planned to add support for more codes in the future, but the currently supported codes already covers the needs of the avarage machining jobs.

Most of the g-codes are in modal groups. In one modal group only one code can be and has to be active the same time. The following table lists the modal groups implemented in the UCCNC software:

| **Name and number of modal group** | **Member G words** |
|---|---|
| Non-modal codes (Group 0) | G4, G10, G28, G28.1, G31, G33, G33.1, G33.2, G76, G52, G53, G92, G92.1 |
| Motion (Group 1) | G0, G1, G2, G3, G73, G80, G81, G82, G83, G84, G85, G86, G89 |
| Plane selection (Group 2) | G17, G18, G19 |
| Distance Mode (Group 3) | G90, G91 |
| Feed Rate Mode (Group 5) | G93, G94 |
| Cutter Diameter Compensation (Group 7) | G40, G41, G42 |
| Tool Length Offset (Group 8) | G43, G44, G49 |
| Canned Cycles Return Mode (Group 10) | G98, G99 |
| Coordinate System (Group 12) | G54, G55, G56, G57, G58, G59 |
| Control Mode (Group 13) | G61, G61.1, G64 |

| | |
|---|---|
| Scale (Group 16) | G50, G51 |
| Coordinate system rotation (Group 17) | G68, G69 |

The supported G-codes are the following:

**Rapid Linear Motion : G0**

For rapid linear motion, program G0 X… Y… Z… A… B… C…, where all the axis words are optional. The G0 is optional if the current motion mode is G0.
If absolute distance mode is selected (G90) then this command will produce coordinated linear motion to the destination point at the maximal possible feed rate.
If relative distance mode is selected (G91) then this command will produce coordinated linear motion to the current point plus the programmed point coordinates. In other words the new point will be an increment on the current point.
It is expected that cutting will not take place when a G0 command is executing.
If all axis words are ommited then no motion will happen, but the active motion mode changes to G0. The axis coordinates

**Linear Motion at Feed Rate : G1**

For linear motion at feed rate (for cutting or not), program G1 X… Y… Z… A… B… C…, where all the axis words are optional. The G1 is optional if the current motion mode is G1.
If absolute distance mode is selected (G90) then this command will produce coordinated linear motion with the programmed feedrate to the destination point at the maximal possible feed rate.
If relative distance mode is selected (G91) then this command will produce coordinated linear motion with the programmed feedrate to the current point plus the programmed point coordinates. In other words the new point will be an increment on the current point.
If all axis words are ommited then no motion will happen, but the active motion mode changes to G1.

**Arc at Feed Rate : G2 and G3**

A circular or helical arc is specified using either G2 (clockwise arc) or G3 (counterclockwise arc).

- The arc lies on the XY plane if the G17 plane is in selection.
- The arc lies on the XZ plane if the G18 plane is in selection.
- The arc lies on the YZ plane if the G19 plane is in selection.

A 3rd axis offset can be also programmed, in this case the 3rd axis will do a linear motion and will start to move the programmed distance when the arc interpolation starts and finishes the movement when the arc interpolation ends. If the 3rd axis offset is programmed then the path will be a helix instead of an arc.
There are two ways to define the arc, one way is the center mode and the other is the radius mode.
In the center mode the I and J and K parameters defines the center of the circle.
In the radius mode the R parameter defines the radius of the circle.

To interpolate an arc or helix in the center format, program G2 (or G3) X... Y... Z... I... J... K... , where X and Y and Z are the endpoint coordinates and

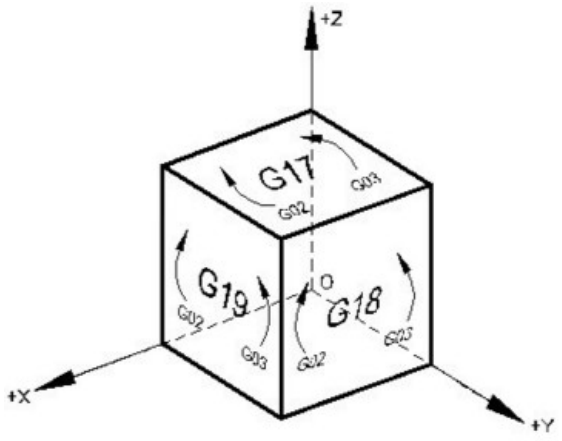
- if the G17 plane is selected then I and J specifies the center of the arc.
- if the G18 plane is selected then I and K specifies the center of the arc.
- if the G19 plane is selected then J and K specifies the center of the arc.

To interpolate an arc or helix in the radius format, program G2 (or G3) X... Y... Z... R... , where X

and Y and Z are the endpoint coordinates of the arc and R specifies the radius of the arc. A positive radius indicates that the arc turns through 180 degrees or less, while a negative radius indicates a turn of 180 degrees to less than 360 degrees.

In both formats the axis coordinates mean the endpoint of the arc or helix if the absolute distance mode is selected (G90). And in relative distance mode the endpoint is the current point plus the programmed coordinates.

The following image shows the directions of the arcs on the different planes:




**Dwell : G4**

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

**Set tool lengt in tool table: G10 L1**

To set the tool length value program G10 L1 P … Z … R ..., where the where the P number must evaluate to an integer in the range 1 to 96 (corresponding to tool number 1 to 96.)
The programmed Z parameter will be the tool length for the programmed tool.
The programmed R parameter will be the tool radius of the programmed tool.
The tool length data is uploaded to the Tools table screen of the UCCNC software.
The tool radius data times 2 (D=R*2) is uploaded to the Tools table screen of the UCCNC software.
It is an error if:
• the P number does not evaluate to an integer in the range 1 to 96.

**Set Coordinate System Data : G10 L2**

To set the coordinate values for the origin of a coordinate system, program
G10 L2 P … X… Y… Z… A… B… C…, where the P number must evaluate to an integer in the range 1 to 6 (corresponding to G54 to G59) and all axis words are optional. The coordinates of the origin of the coordinate system specified by the P number are reset to the coordinate values given (in terms of the machine coordinate system). Only those coordinates for which an axis word is included on the line will be reset.

It is an error if:
• the P number does not evaluate to an integer in the range 1 to 6.
Example: G10 L2 P1 x 3.5 y 17.2 sets the origin of the first coordinate system (the one selected by G54) to a point where X is 3.5 and Y is 17.2 (in machine coordinates). The Z coordinate of the origin (and the coordinates for any rotational axes) are whatever those coordinates of the origin were before the line was executed.

**Plane selection : G17, G18, G19**

To select the XY plane program G17.
To select the XZ plane program G18.
To select the YZ plane program G19.

Arcs will be interpolated on the selected plane. For more informations please read the description about G2, G3 in this documentation.

**Run to home position : G28**

To move the machine to the set home coordinates , program G28 X... Y... Z... A... B... C...,
where the parameters are the axis coordinates of an intermediate point.
When executing the G28 code and if any intermediate coordinates are programmed then the machine will first make a move to this point with rapid linear interpolation (G0) and from this point it will move to the set home coordinates again with G0 interpolation.
If intermediate coordinates are not programmed then the machine moves straight to the home position.

Warning: Only execute a G28 when the machine is homed.

**Home axis: G28.1**

To home the machine, program G28.1 X... Y... Z... A... B... C..., where the parameters are the axis coordinates of an intermediate point.
When executing the G28.1 code and if any intermediate coordinates are programmed then the machine will first make a move to this point with rapid linear interpolation (G0) and from this point it will start a homing sequence, will home the programmed axes with the feedrate setup in the homing configuration. The homing sequence will be in the order setup in the homing configuration.
If no axis words are programmed then the code will execute the same procedure as the Home all button does, it will home all the axes without any prior movement.

**Straight Probe : G31**

Program G31 X… Y… Z… A… B… C…to perform a straight probe operation.
Currently programming one axis word one time is allowed only.

In response to this command, the machine moves the controlled point (which should be at the end of the probe tip) in a straight line at the current feed rate toward the programmed point. If the probe trips the coordinates at the trigger point are written into the variables #5061 to # 5066 (axes coordinates X to C in order) and the axis stops with deceleration.

If the probe does not trip even after reaching the programmed coordinate, the axis stops and the probing action ends. In this case the variables #5061 to #5066 are set to 0 value.

In addition to the axes coordinates variables the variable #5060 is written to value 0 if the probe

input was triggered and therefor the operation ended with success. It is written to value 1 if the probe was not triggered and the probing ended with reaching the programmed endpoint and it is written to value 2 if the probe input was already active when the G31 command was issued and therefor the movement was aborted.

The probing command can be practically combined with other macro scripts, an example code is the M31 macro code which code can be find in the Macros library.

An extra function for probing is the software's ability to probe with jogging. If the probe input activates while the machine is jogging then the axes stop automatically and the current jog direction on the axes get disabled until the probe input deactivates. This protects the probe from getting damaged from over traveling the touch point. The machine operator can re-enable the disabled jogging directions with jogging the axes off the probe deactivating the probe input.

**Spindle Syncronised motion : G33**

Program G33 X... Z... K... Q... to perform a spindle syncronised motion, where the X parameter is the final position of the X-axis, the Z parameter is the final position of the Z-axis. The K parameter is the pitch per revolution and the Q parameter is the start angle in degrees.

For the spindle syncronised motion an incremental encoder with A, B and Index channels has to be installed onto the spindle and the spindle encoder has to be setup in the software.

When executing a G33 command, the motion controller will first waiting for the index channel signal from the spindle encoder and will syncronise the feedrate to the rotational speed.
The start angle (Q parameter) defines the angle between the encoder index signal and the motion controller measures the set angle after the index signal and starts the motion at this angle.
The G33 motion is always on a straight line just like with G1, but it is always on the XZ plane and the feedrate is continously syncronised to the spindle speed and taking the pitch (K) parameter into account. The syncronised motion always ends at the programmed X and Z coordinates.
If both the X and Z coordinates are programmed for the G33 command and if both coordinates differs from the starting coordinates then the thread will be cut on a cone on the XZ plane and the thread will then not be parallel to the Z not the X axis. The pitch is then calculated on axis which makes the longer movement.
If more G33 commands follows eachother and if the start angle is not programmed or only programmed for the first G33 command then the rest G33 commands will be syncronised with the previous G33 command which makes it possible to cut a continous thread even with changing pitch.

**Rigit tapping G33.1 and G33.2**

To execute a right hand (Leads in with M3) rigid tapping cycle program G33.1 Z... K... Q... and to program a left hand (Leads in with M4) rigid tapping cycle program G33.2 Z... K... Q..., where the Z parameter is the bottom of the tap along the Z-axis and the K is the thread pitch per revolution.
The Q parameter is optional and defines the peck depth.
If the Q parameter is not programmed then the tapping cycle is executed from a single pass.
If the Q parameter is programmed then a peck tapping is made with each cycle incements Q depth and a spindle syncronised tool pull off happens to the tapping command start position.
If the Q parameter value is not an integer divider of the full tap depth then the final tap depth will be still correct, in this case the last peck cycle depth incremenet will be shorter than the Q parameter value.

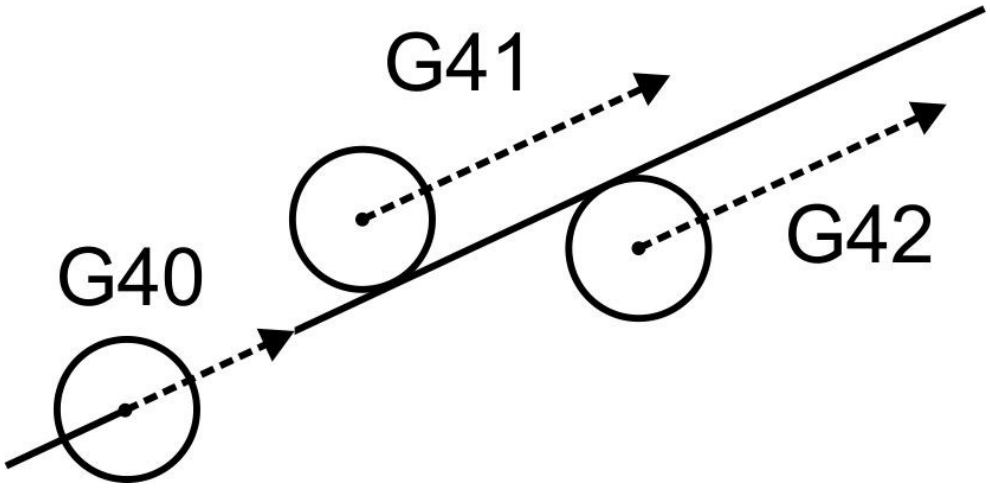
The difference between the G33.1 and G33.2 is the direction of the thread, the G33.1 starts the tapping with CW rotation of the spindle, with M3 command in action when the tool is leading into

the material and leads out with an M4 command. The G33.2 command starts the tapping with an M4 command when the tool is leading into the material and leads out with an M3 command.
Prior to a rigid tapping cycle the tool on the XY plane should be moved to above the drilled hole which needs to be tapped and the M3 or M4 comamnd must be active. (depending on which code was executed.)
A rigid tapping command executes the following cycle:

1.) The motion controller waits for the index signal of the spindle encoder and measures the rotational speed of the spindle and syncronises the Z-axis feedrate to the spindle speed with taking the thread pitch (K) parameter into account. The spindle should be already rotating to the correct direction, otherwise the tapping will not start and the software will wait for the spindle to turn.
2.) The Z-axis motion starts and the tool is moved down to the programmed Z position.
3.) At the bottom of the tap, at the programmed Z position the spindle direction is switched. If it is a G33.1 command then the M3 is switched to M4, or if it is a G33.2 command then the M4 command will switch to M3. This will reverse the spindle rotational direction. The syncron between the spindle and the Z-axis feedrate is kept.
4.) The tool is moved out from the material to the starting point where the motion controller removes the syncron between the spindle and the Z-axis feedrate. The Z-axis stops using the set decceleration. This means that the axis will overrun the start point a bit, because it needs to travel a distance to deccelerate from the current feedrate.
5.) The Z-axis is moved down to the start point with a rapid movement.

**Cutter radius compensation: G40, G41 and G42**

To turn the cutter radius compensation on from left, program G41 D...
To turn the cutter radius compensation on from the right, program G42 D...
The D parameter is the tool number in the tool table to be used for the cutter compensation.



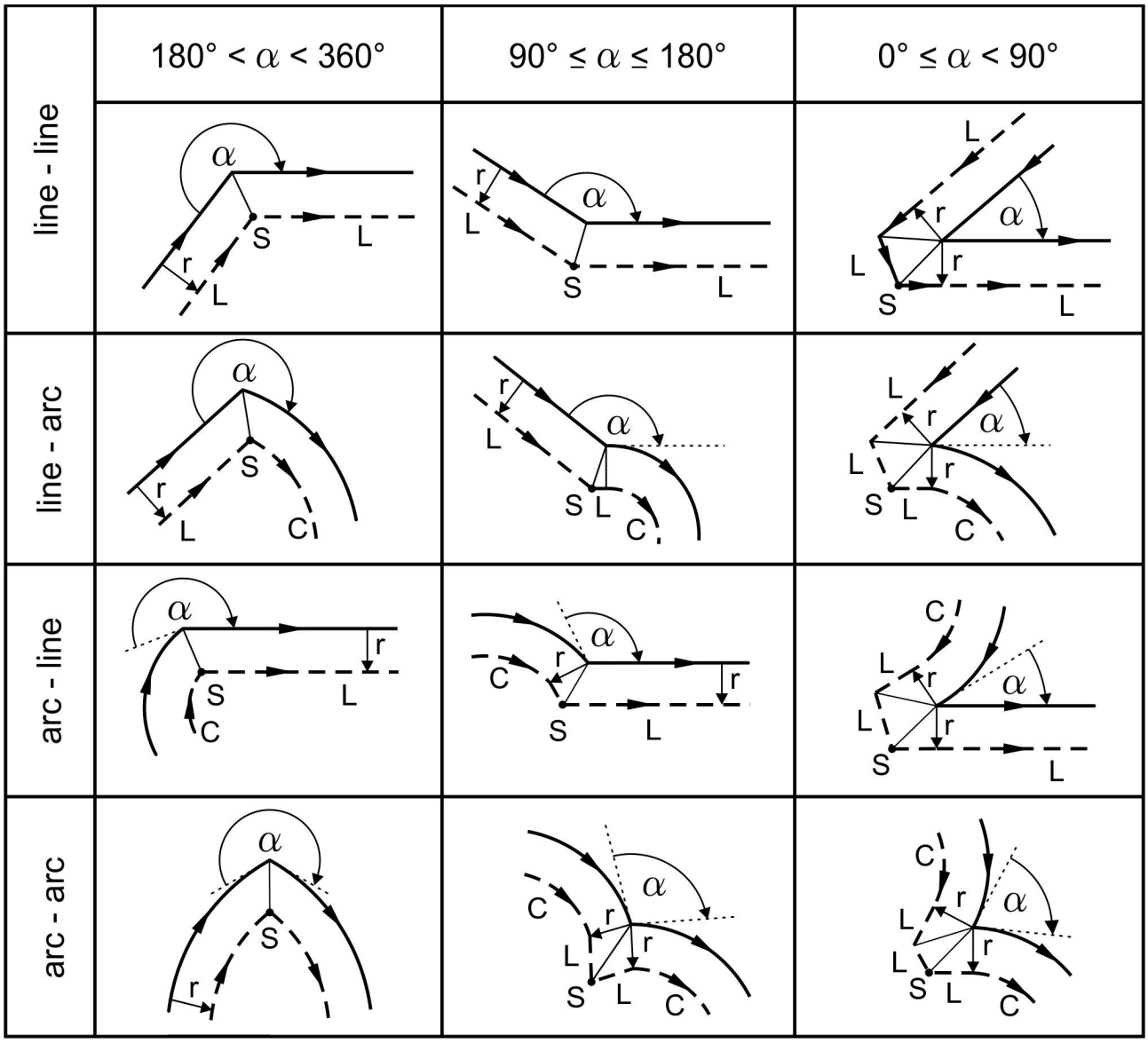

To turn the cutter radius compensation off, program G40.

When a G41 D... or G42 D... is programmed then the software starts performing the cutter radius compensation routine. The controller reads the tool diameter from the tooltable and uses the radius of the tool to offset the toolpath.
If the D parameter is not programmed then the radius of the currently selected tool is used for the cutter radius compensation.

The cutter radius compensation routine can be performed on the G17 (XY) plane only.

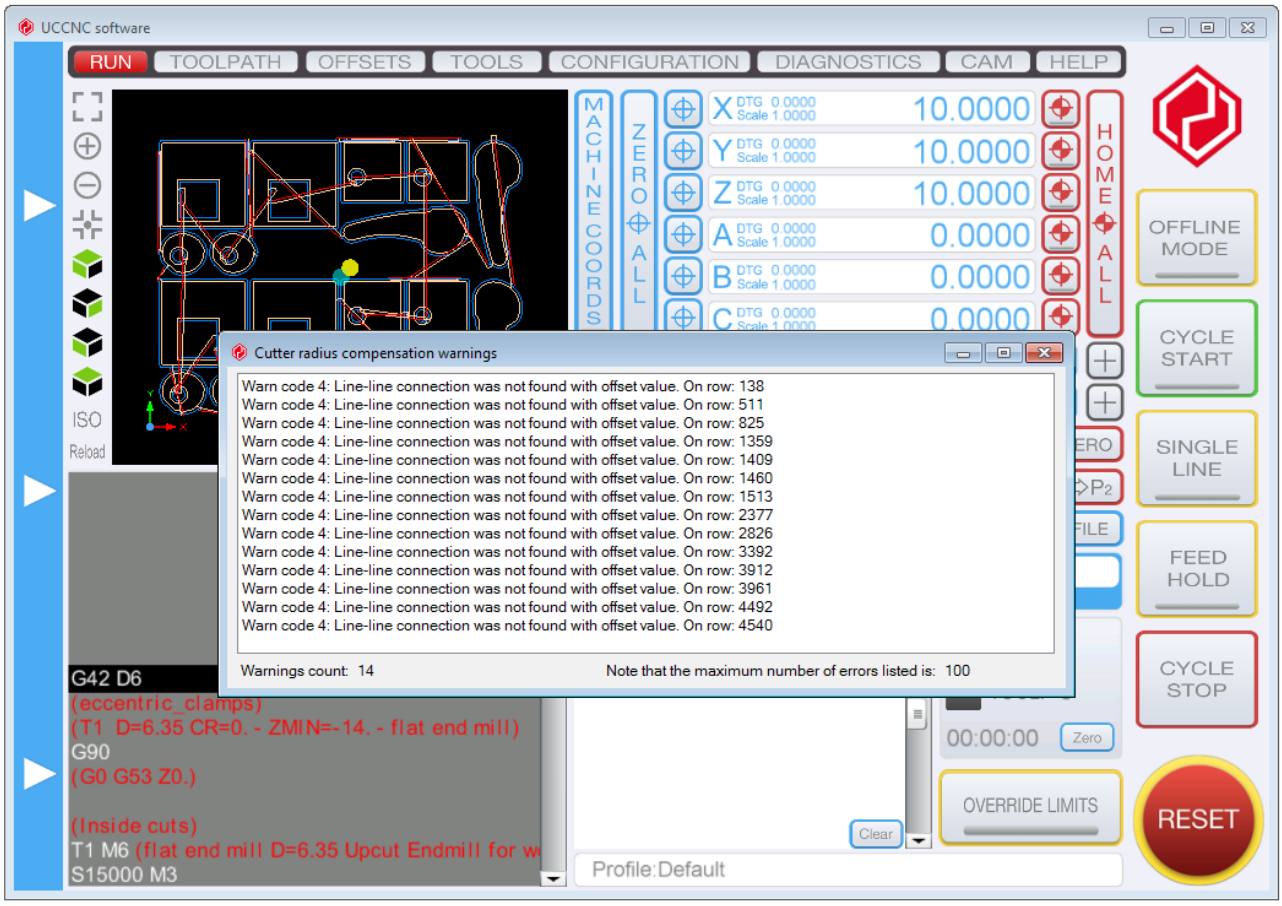
It is an error if the G18 or G19 plane is active and the cutter radius compensation is turned on. It is also an error if the cutter radius compensation is on and a G33 or G76 thread cut code is executed.
The following image shows how the cutter compensation is performed under different angles:
The symbols meanings are the followings:

r: The cutter radius value.
L: A line section.
C: An arc section.
S: The stop point when in single line execution.
Continous line: The programmed path.
Dashed line: The cutter radius offset path.




It is possible that no connection point is found with the set cutter radius. In this case the software gives a warning. The Cutter radius compensation warning list window appears when the g-code file which contains a G41 or G42 code is loaded and if there were any issues were found during the offset path calculation.
If the warnings are considered errors or not depends on the particular job and the user can check the warning lists and decide if the warning causes any problems for the job.

The warnings window can be enabled/disabled with the “Show R compensation analysis on file load” checkbox on the Configuration/Appearance tab page. When this option is disabled then the warning window is never shown.
The following printscreen image shows an example of the warnings:




The possible warning codes are as follows:

Warn code 1: Approach distance too low.
This warning happens when the tool is closer to the startpoint than the cutter radius.

Warn code 2: Approach angle too low, angle < 180°.
This warning happens when the angle of the starting point and the next 2 points is smaller than 180°, because this can cause an undercut when the tool approaches the toolpath.

Warn code 3: Approach line is missing. First movement is arc.
This warning happens if the first programmed movement is an arc. It is recommended that the first programmed item is a line.

Warn code 4: Line-line connection was not found with offset value.
This warning happens if there is no connection point between 2 path items. This can happen because of several reasons, for example the angle is too small compared to the used cutter radius, so the tool does not fit inside the two items.

Warn code 5: Arc radius is smaller than the compensation radius for inner retreat.
This warning happens when an arc's radius is smaller than the cutter radius for an inner retreat and so the tool does not fit inside the arc.

**Tool Length Offsets : G43, G44 and G49**

To use a tool length offset, program G43 H…, where the H number is the desired index in the tool table. The H number should be, but does not have to be, the same as the slot number of the tool currently in the spindle. It is OK for the H number to be zero, an offset value of zero will be used.
It is an error if:
• the H number is not an integer, is negative, or is larger than the number of carousel slots, which is currently set to a value of 20.
The G44 command is similar to the G43, but for negative tool length offsets.
To cancel the tool length offset compensation, program G49.
It is OK to program using the same offset already in use. It is also OK to program using no tool length offset if none is currently being used.

**Reset all scale factors to 1 : G50**

To reset the scale factor to value 1 on all XYZABC axes program G50.
The command has no parameters.

**Set scale factors : G51**

To scale any axis program G51 X… Y… Z… A… B… C… , where the axis parameters define the scaling factors and can be positive and negative numbers. In case of negative scale factors the axis gets mirrored.
All parameters are optional, but at least one parameter has to be defined.
The scaling happens on the axes around the 0 point in the current coordinate system.
All motion commands including linear and circular motion commands are scaled with the defined scale factor, except the G33, G33.1, G33.2 and G76 commands, these are not allowed to be scaled.

A simple scaled movement example:
G90
G0 X0
G51 X2
G1 X1

This code will move the X axis to position X=2*1 = 2, because the X axis was scaled to 2.

It is an error if:
- Different scale factor values are set for the 2 axes of an arc.
- Executing G33 thread cutting command when the X or Z axis is scaled.
- Executing G76 thread cutting command when the X or Z axis is scaled.
- Executing G33.1 or G33.2 rigid tapping command when Z axis is scaled.

**Temporary offset coordinate system : G52**

To offset the current point by a given positive or negative distance (without motion), program G52 X... Y... Z... A... B... C..., where the axis words contain the offsets you want to provide. All axis words are optional, except that at least one must be used. If an axis word is not used for a given axis, the coordinate on that axis of the current point is not changed.

It is an error if all axis words are omitted.

When G52 is executed, the origin of the currently active coordinate system moves by the coordinate values given.

The effect of G52 is cancelled by programming: G52 X0 Y0 etc.

The axis offsets are always specified in absolute values and effects all fixture coordinate systems.

**Move in Machine Coordinates : G53**

For linear motion to a point expressed in machine coordinates, program G1 G53 X… Y… Z… A… B… C… (or use G0 instead of G1), where all the axis words are optional, except that at least one must be used. The G0 or G1 is optional if it is the current motion mode. G53 is not modal and must be programmed on each line on which it is intended to be active. This will produce coordinated linear motion to the programmed point. If G1 is active, the speed of motion is the current feed rate (or slower if the machine will not go that fast). If G0 is active, the speed of motion is the current traverse rate (or slower if the machine will not go that fast).
It is an error if:
• G53 is used without G0 or G1 being active.

**Select Coordinate System : G54 to G59**

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

**Set Path Control Mode to exact stop : G61, G61.1**

Program G61 or G61.1 to put the machine into exact stop path control mode.

**Set Path Control Mode to constant velocity : G64**

Program G64 D... E... H... L... P... Q... to put the machine into constant velocity path control mode. All parameters are optional. If any parameters are left out (not programmed) then the software will get that parameter's value from the screen settings.
The parameter D is the Stop angle setting.
The parameter E is the Lookhead lines setting.
The parameter H is the Linear error setting.
The parameter L is the Linear addition length setting.
The parameter P is the Unify length setting.
The parameter Q is the Corner error setting.

**Rotate coordinate system: G68**

To rotate the coordinate system around a defined point on the XY plane program: G68 A... B... R..., where the A parameter is the X coordinate and the B parameter is the Y coordinate of the rotation center point and the R parameter is the rotation angle measured in degrees.
The A and B parameters are optional and if one parameter is missing then the current point of that axis is used for that rotation coordinate.
The rotation angle is programmed in degrees and a positive value means a counter-clockwise and a negative value makes a clockwise rotation.
When the rotation is enabled all of the motion commands are rotated, except the movements in the absolute coordinate system (G53).
The rotation is applicable only on the XY plane.

It is an error if:

- The rotation is applied when the active plane is not G17 (XY) or if the G18 or G19 plane is selected when the rotation is active.
- Executing G33 thread cutting commands when the rotation is active.
- Executing G76 thread cutting commands when the rotation is active.
-

**Reset coordinate system rotation: G69**

To reset the rotation of the coordinate system program G69.
The command has no parameters.

**Peck drilling cycle with set backoff distance: G73**

To execute a peck drilling cycle with a set back off distance, program G73 X... Y... Z... Q... R... , where the X, Y and the R parameters are optional and the Q parameter is modal.
The peck drilling cycle rapids to the XY coordinates and then to the R plane where it begins to feed down in increments of Q at a feed rate of F to a depth of Z. At the end of each Q increment the drill will stop and back off the distance set on the General configuration screen's G73 back off label value to break a possible stringer chip and then continue with he next increment of Q. As in all pecking cycles the programmer does not need to calculate the number of pecks and peck distance that will equal the exact Z Depth. The controller will calculate the distance of the last peck to coincide with the Z depth. At the end of the cycle the Z axis returns to the R plane if the R parameter was set or to the initial plane if the R parameter was not set.

**Threading cycle: G76**

To execute a threading cycle, program G76 P... Z... I... J... K... E... L... Q... H... , where the parameters are the following:

**P:** The pitch of the thread in distance units per spindle revolution.
**Z:** The final position of the thread along the Z-axis.
**I:** The thread peak position in units along the X-axis. This is the starting height of the thread and the material was mostly turned down to this size.
**J:** Cut depth per pass. The first cut will be J beyond I along the X-axis.
**K:** Full thread depth. The last cut will be this deep along the X-axis.
**E:** Taper length. Specifies the length of the taper(s). For a 45° taper program the E parameter equal to the K parameter.
**L:** Tapered ends. Specifies which end of the workpiece will get a taper. L0 means no taper, L1 means start taper, L2 means end taper, L3 means both start and end tapers.
**Q:** Slide in angle. Specifies an angle between the cuts. As the tool goes from one cut depth to the other the cut start is offset along the Z-axis with this angle measured from the X-axis. Programming a positive or negative angle will make one or the other side of the tool to remove more material.
**H:** Number of final cuts. To make any extra cuts on the final depth program this parameter other than 0. The value of the parameter will define the number of extra cuts on the final depth. If the parameter is not set the default value of 0 will be used causing a single final depth cut.

For the thread cutting cycle first move the tool to the starting point and then program the G76 with the required parameters. Executing the thread cutting code the tool will move to the thread peak position (I parameter) along the X axis in case no start taper is programmed. If a start taper is programmed then the tool will first move to the thread peak position minus the full thread depth. The motion controller will wait for the index signal input and will syncronise the axis feedrates to the spindle encoder. The thread cutting will start on the index signal and the feedrate is continously

syncronised to the spindle speed. If a start taper was programmed then the tool cuts the taper first and this motion is also syncronised to the spindle.
The thread cutting ends on the exact programmed Z position if no slide in angle was programmed (Q parameter) or if a slide in angle was programmed then the final Z position will be compensated with the slide in angle which can be in positive or negative direction depending on the sign of the Q parameter.
If an end taper was programmed then the tool will slide out from the material also in syncron making a taper to the end of the thread. The end taper angle will be the same as the start tapper angle, in other words the start and the end tappers will be symmetrical.
At the end of each thread cutting cycle the tool rapids out to the starting X coordinate first leaving the material and then to the start Z coordinate and begins a new cut cycle with a new cut depth.
The process continues until the final thread depth is reached and if the H parameter is set other then zero then the final cut depth cycle is repeated H times.

**Cancel canned cycle: G80**

To cancel the drilling cycle program G80. The command cancels the drilling cycle and the last programmed G0, G1, G2 or G3 modal motion mode gets enabled.

**Drilling cycle: G81**

To execute a drilling cycle program G81 X... Y... Z... R..., where the X and Y parameters are the coordinates of the drill on the XY plane, the Z parameter is the endpoint of the drill along the Z-axis and the R parameter is the retract plane.
The G81 code moves the tool to the XY coordinates with a single rapid movement where it starts moving down to the programmed Z coordinate along the Z-axis with the set feedrate.
When the Z endpoint gets reached the Z axis reverses and rapids out of the drill to the retract plane.

**Drilling cycle with dwell: G82**

To execute a drilling cycle with dwell program G82 X... Y... Z... R..., P... where the X and Y parameters are the coordinates of the drill on the XY plane, the Z parameter is the endpoint of the drill along the Z-axis, the R parameter is the retract plane and the P parameter is a dwell time in milliseconds.
The G82 is very similar to the G81 code, the only difference is that the software dwells at the bottom of the drill before it rapids the tool out.

**Peck drilling cycle with full backoff distance: G83**

To execute a peck drilling cycle with a full back off distance, program G83 X... Y... Z... Q... R... , where the X, Y and the R parameters are optional and the Q parameter is modal.
The G83 code is very similar to the G73, the only difference is that at each peck increments the controller returns to the R plane if the R plane was programmed or to the initial plane if the R parameter was not set. The purpose of this command is the same as of the G73, but because of the full retract at each increments can break spiral chips better.

**Rigid peck tapping cycle: G84**

To execute a rigid peck tapping cycle, program G84 X... Y... Z... K... Q... R... H...,
where the X, Y and the R and H parameters are optional and the Q and K parameters are modal.
The Z parameter is the bottom of the tap along the Z-axis and is modal and the K parameter is the thread pitch per revolution and is modal.
The XY parameters define the position of the drill.

The R parameter defines the retract plane.
The Q parameter is modal and defines the peck depth.
If the Q parameter is not programmed then the tapping cycle is executed from a single pass, except if the Q parameter was previously defined, then the previous Q value will be used, because the Q parameter is modal.
If the Q parameter is programmed then a peck tapping is made with each cycle incements Q depth and a spindle syncronised tool pull off happens to the R plane or to the initial plane depending on if the G98 or G99 code is active.
If the Q parameter value is not an integer divider of the full tap depth then the final tap depth will be still correct, in this case the last peck cycle depth increment will be shorter than the Q parameter value.
The H parameter is optional and if it is not defined or if it's value is 0 then it is a right hand tap otherwise it is a left hand tap.

The G84 cycle is only allowed to be programmed on the XY (G17) plane. It is an error if the G84 is programmed when the G18 or G19 plane is active.

**Boring cycle with feed out: G85**

The G85 cycle is very similar to the G81 cycle except that the retraction of the tool is with the current feedrate and not with rapid.

**Boring cycle with dwell and spindle stop/start: G86**

The G86 cycle is very similar to the G82 cycle except that the spindle is stopped at the bottom of the drill before the retraction of the tool and the spindle restarts rotation to the same direction it was spinning before stopped when the retraction finished.
In addition to the G82 code parameters the G86 code has an optional H parameter. If the H parameter is not programmed or if it's value is 0 then the retraction is executed with rapid otherwise it is executed with the programmed feedrate.

**Boring cycle with dwell and feed out: G89**

The G89 cycle is very similar to the G82 cycle except that the retraction of the tool is with the current feedrate and not with rapid.
**Absolute distance mode selection: G90**

To select absolute distance mode program G90. This will make coordinates in the executed commands to be interpreted as absolute position values.
The G90 command can be executed any time in code execution or via the MDI.

**Relative distance mode selection: G91**

To select relative distance mode program G91. This will make the coordinates in the execured commands to be interpreted as relative position values to the actual coordinates.
The G91 command can be executed any time in code execution or via the MDI.

**Temporary offset to programmed coordinates : G92**

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

It is an error if all axis words are omitted.
When G92 is executed, the origin of the currently active coordinate system moves. To do this, origin offsets are calculated so that the coordinates of the current point with respect to the moved origin are as specified on the line containing the G92. The offset for an axis is the amount the origin must be moved so that the coordinate of the controlled point on the axis has the specified value.

The axis offsets are always specified in absolute values and effects all fixture coordinate systems.
Because the G92 offset is temporary, unlike the other offsets the G92 offset coordinates are not saved when closing the software. The G92 offset coordinates are always 0 on software startup.

**Reset temporary offset coordinates : G92.1**

To reset the G92 offset coordinatets program G92.1.

**Inverse time feedrate mode: G93**

In inverse time feed rate mode the F word means that the move should be completed in [one divided by the F number] minutes. For example if the F number is 2, the move should be completed in half a minute.
When the inverse time feedrate mode is active, an F word must appear on every line which has a G1, G2, or G3. The inverse time feedrate mode does not effect the feedrate of the rapid (G0) movements.

**Units per minute feedrate mode: G94**

In units per minute feedrate mode the F word means that the axes should make the tool center point to travel the programmed amount of distance units per minute. For example F10 should make the tool center point to travel 10 units distance per minute.

**Canned cycle return level to initial plane : G98**

To make the canned cycles retract to the plane the depth axis was program G98.
The axis will return to the initial plane only if the initial plane is higher then the plane programmed by the R word, otherwise if it is lower then the axis will return to the R plane.

**Canned cycle return level to R plane : G99**

To make the canned cycles retract to the plane programmed with the R word, program G99.

### 4.1.2 .M-codes

Macro codes can be called from code execution or via the MDI control. To program a macro call Mx, where 'x' means a positive integer number and this number is the number of the macro to call.
Macros are text files and located in the folder of the machine profile.

There are default macros in the RS274 language and there are some other macros which we prepared as an example. These macros and their functions are described in this section of the documentation.

Macros can have parameters. The parameters are passed to the macro script file and can be used as predefined variable values inside the macro script file.
There are 2 ways to pass variables to the macro, one way is to pass upto 3 parameters programmable with the 'E', 'H' and 'Q' prefixes.

For example 'M300 E1.2 H3 Q4.5'.
The parameters are all optional and if a parameter is set it can be used inside the macro with the Evar, Hvar and Qvar variables. These variables are always defined and set when the macro is called. The default value of all these parameters are 'null', if the parameters does not exist when called then the parameters will have a 'null' value.

The other way to pass parameters to macros is to place the variables in {} brackets next to the macro call.
For example 'M300 {A1.2 B2.3 C3.4}'
Using the brackets for passing variables a much larger number of parameters can be passed, the available letters for the parameters are: A, B, C, D, E, H, I, J, K, L, N, O, P, Q, R, U, V, W, X, Y, Z.
In the macro file the variables can be refered and used as a member of the 'Allvars' structure, for example: Allvars.Avar , Allvars.Bvar.
If no variable is passed to the macro then the Allvars structure is not getting created it's value is 'null'. If a variable is not defined in the call code then that variable member will have a null value.
The macro writer should make a check whether a variable was passed or not before using the variable in the macro. The check can be done as the following example shows. In the example code the Allvars structure is checked if exists first and then if the X variable was defined and passed.

```
if(Allvars != null)
{
  if (Allvars.Xvar != null)
  {
    MessageBox.Show(exec.mainform, ""+Allvars.Xvar); //Print the X variable to a MessageBox
  }
}
```

The following are the default macros built into the UCCNC:

**Program stop: M0, M1, M60**
Program M0, M1 or M60 to stop the g-code program execution. The code execution can be continued with a new cycle start operation.
The M0, M1 and M60 codes function the same, but they activate different virtual LEDs so the user and plugins and macros can distinguish which code was activated.

**Program end: M2**
Program M2 to end the g-code program.
The M2 macro can be programmed in code execution only, in MDI it has no meaning.

**Turn spindle clockwise: M3**
Program M3 to start the spindle rotation in the clockwise direction.
The M3 macro can be programmed in code execution and via the MDI.

**Turn spindle counterclockwise: M4**
Program M4 to start the spindle rotation in the counterclockwise direction.
The M4 macro can be programmed in code execution and via the MDI.

**Stop spindle turning: M5**
Program M5 to stop the spindle rotation.
The M5 macro can be programmed in code execution and via the MDI.

**Tool change: M6**
To change a tool in the spindle from the tool currently in the spindle to the tool most recently

selected, program M6 which will call the M6 macro. The number of the tool to select can be programmed with the "T" keyword.
We supplied an example tool change M6 macro in the macro library of the Default profile.
Example usage: M6 T2 , will change the current tool to tool number 2.
The M6 macro calls can be overridden in the setup on the general settings tab page!

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

**Activate syncronous fast PWM (laser) output: M10**
To activate an output in syncron with the motion program M10 Q..., where the Q parameter is the duty cycle of the PWM output pin and the range is 0..255 for the 0 to 100% duty cycles. If the Q keyword is omitted then the maximum 255 value is used. The output port and pin number can be selected on the Configuration/I/O setup tab page. The output will be a PWM and the duty cycle of this signal will be set by the programmed Q parameter.
The M10 command can be used to control a laser device or any other device which requires fast and syncronous switching with the motion.

**Deactivate syncronous fast PWM (laser) output: M11**
To deactivate an output in syncron with the motion program M11
Executing this macro will switch off the fast syncronous laser output if it was previously switched on with an M10 Q... command.

**Activate syncronous fast digital output: M10.1 … M10.10**
To activate one of the 10 pcs outputs in syncron with the motion program M10.1 to M10.10.
The syncron outputs are digital (on/off).
These commands are supported by our ethernet controllers only.

**Activate syncronous fast digital output: M11.1 … M11.10**
To deactivate one of the 10 pcs output in syncron with the motion program M11.1 to M11.10.
These commands are supported by our ethernet controllers only.

**Stop program and rewind code: M30**
To stop the program and rewind the code to the first line, program M30.
Usually the M30 is placed at the very end of the G-code program.

**Start digitizing: M40**
To start the digitizing process and collect probed points execute M40.

**Stop digitizing: M41**
To stop the digitizing process and to save the probed points to file execute M41.

**Program rewind and continue running: M47**
To rewind the program and continue code execution from the first line, program M47.
Usually the M47 is placed at the very end of the G-code program.

**Enable the spindle speed and feedrate override controls: M48**
To enable the feedrate override (FRO) and spindle speed override (SRO), program M48.

**Disable the spindle speed and feedrate override controls: M49**
To disable the feedrate override (FRO) and spindle speed override (SRO), program M49.

**Enable/Disable the feedrate override control: M50**
M50 enables the feedrate override control (FRO).
M50 P0 disables the feedrate override control (FRO).

**Enable/Disable the spindle speed override control: M51**
M51 enables the spindle speed override control (SRO).
M51 P0 disables the spindle speed override control (SRO).

**Wait on input: M66**
To make the g-code cycle execution to wait on an input program M66 P... E... L... Q...

P specifies the digital input pin number
E specifies the digital input port number
L specifies the wait mode type.

Mode 0: IMMEDIATE : no waiting, the function returns immediately. The current value of the input is stored in parameter #5399
Mode 1: RISE - waits for the selected input to perform a rise event.
Mode 2: FALL - waits for the selected input to perform a fall event.
Mode 3: HIGH - waits for the selected input to go to the HIGH state.
Mode 4: LOW - waits for the selected input to go to the LOW state.

Q specifies the timeout in seconds for waiting. If the timeout is exceeded, the wait is interrupted and the variable #5399 will be holding the value -1. The Q value is ignored if the L-word is zero (IMMEDIATE mode).

**Subroutine call: M98**
To call a subroutine program: M98 P... L..., where the P parameter defines the number of the subroutine to call and the L number defines the number of times the subroutine needs to be called.
If the L parameter is not programmed then the default L value of 1 is used.
It is an error if the P parameter is not programmed.
A subroutine call can be placed inside another subroutine. The subroutine stack is 100 level deep.
If a subroutine call is deeper than the subroutine stack size then the stack will get full and a popup window will indicate this error and the code execution will be stopped. The stack full error is likely to happen if a subroutine is called from itself which will create an infinate loop and stack overflow.

**Return from subroutine: M99**
To return from a subroutine program M99.
This will return the line pointer to the code line the call was made.
It is an error if the M99 code was called without an M98 code was called, in this case the code interpreter will not know where to return the code line pointer and a popup window will indicate this error and the code execution will stop.

**Enable THC (Torch Height Control) control for plasma: M205**
To enable the THC control for a plasma cutter machine program M205.
The THC control will use the THC up and THC down and THC on signals to control the Z-axis position. This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Disable THC (Torch Height Control) control for plasma: M206**
To enable the THC control for a plasma cutter machine program M206.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Enable the THC (Torch Height Control) delay for plasma: M207**
To enable the THC delay for plasma cutter machine program M207.
The THC delay disables the THC inputs control to control the Z-axis for the set amount of time after the torch fires. This protects the machine from the Z-axis to be pushed downwards to the table because of the high Voltage generated while the arc is piercing the sheet.
The THC delay time can be set in the UCCNC setup.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Disable the THC (Torch Height Control) delay for plasma: M208**
To disable the THC delay for plasma cutter machine program M208.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Enable the THC (Torch Height Control) anti dive function for plasma: M209**
To enable the THC anti dive function for plasma cutter machine program M209.
The THC anti dive function runs directly in the motion controller. The function measures the feedrate of the movements and if the feedrate drops below the programmed value times the setup percentage then the downwards movement of the Z-axis controlled by the THC gets temporarily disabled. As soon as the feedrate rises back to the programmed rate times the set percentage the Z axis downwards control gets enabled again.
This function is to protect the Z-axis to be pushed downwards to the table when the feedrate drops and when a constant feedrate can't be maintained by the software for example if the code contains direction changes when the machine axes have to deccelerate and therefor the feedrate must be dropped. This cut feedrate reduction can cause the arc voltage to rise and the THC control will direct the Z-axis downwards. The anti dive function protects against this unwanted effect.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Disable the THC anti dive function for plasma: M210**
To disable the THC anti dive function for plasma cutter machine program M210.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Enable the THC (Torch Height Control) anti down function for plasma: M211**
To enable the THC anti down function for plasma cutter machine program M211.
The THC anti down function disables the downwards control of the THC.
This function is useful when the motion path contains movements which crosses already cut paths. For example the job is to cut an X shape. The cut is done with 2 linear movements, the 2 sides, 2 lines of the X shape. After the first half of the X got cut the second line has to cross the first already cut one, so the second cut line will have to cross the first at one point. When this happens there will be no material to cut for the torch for a short time and this can cause the arc voltage to quickly rise and this would control the Z-axis with the THC downwards to the table.
The M211 code can be inserted into the g-code program right before the movement to this problematic point, so the THC downwards movement will be disabled for a short period.
The THC upwards control will still be enabled, so if the sheet is bent is still not a problem as the torch can still move upwards. After the torch left the problematic point an M212 can be inserted into the g-code to disable the anti down function which will enable the THC downwards control again.
The M211/M212 can be inserted into the g-code program manually also, but in practise it is more useful to get the CAM software to do the job with adding this requirement to the post processor of the CAM software. Please note that this is possible only with CAM software which supports this in it's post processors.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Disable the THC (Torch Height Control) anti down function for plasma: M212**
To disable the THC anti down function for plasma cutter machine program M212.
This macro runs in syncron with the g-code without delays and without axes decceleration or stop.

**Enable the safe probe mode: M213**
To enable the safe probe mode program M213.
The M213 macro protects the probe sensor with generating a reset condition and hard-stopping the machine when this mode is active and if the probe is activated while the machine is in motion.
In practice this macro command can be executed for example in the M6 macro if the tool number to be used is a probing tool. This can prevent a probe damage if the operator accidentally cycle starts the g-code program or commanding the machine to move and hit the probe.

**Disable the safe probe mode: M214**
To disable the safe probe mode program M214.

**Select spindle pulley: M215 Px**
To select a spindle pulley program M215 Px, where the x is a number between 1 and 15.

**Constructor macro: M99998**
This macro is called on each software startup. Initial actions like user variables loading and startup settings could be made using and programming this macro.

**Destructor macro: M99999**
This macro is called on each software shutdown. User variables and DRO values could be saved or messages could be shown to the user etc. on the software shutdown with programming this macro.

### 4.1.3 .Other codes

**Set Feed Rate: F**
Program F... to set the feedrate.
Note that the Feedrate can be override in the 1-300% range using the + and – buttons in the feedrate override DRO.

**Set Spindle Speed: S**
To set the speed in revolutions per minute (rpm) of the spindle, program S… . The spindle will turn at that speed when it has been programmed to start turning. It is OK to program an S word whether the spindle is turning or not. If the speed override switch is enabled and not set at 100%, the speed will be different from what is programmed. It is OK to program S0; the spindle will not turn if that is done. It is an error if:
• the S number is negative.

Note that the Spindle speed can be override in the 1-300% range using the + and – buttons in the spindle speed override DRO.

**Subroutines: O**
The subroutines are labeled with the 'O' prefix.
Usually the subroutines are placed at the end of the program, after the M30 or M47 code.

**Load tool: T**
To load a tool program T... , for example T2.
Programming the T code loads the tool number into the next tool variable in memory and on the next M6 code programming the actual tool will be changed to the most recently programmed tool

number.

## 4.2 .Parametric programming

Instead of constants variables could be also used to define coordinates or feedrate, spindle speed etc. There are 1000 pieces of variables internal variables currently, named as #0 to #999.
When a number is programmed with the '#' suffix means that this paramater is not a constant, but a pointer to an internal variable.
The variables can get new value any time programming an equation in code execution or via MDI.
Mathematic equations can be also programmed to give value to variables.
For example to give a value of 1.23 to variable #2, code: '#2 = 1.23'.
Equations should always be in a new line, it is not allowed to place an equation as a parameter of an axis word or feedrate or spindle speed set code. A single variable can be programmed as a parameter standalone only.
The following code example shows how to use the variables:

#1 = 5 (Sets the value of variable #1 to 5)
#2 = 10 (Sets the value of variable #2 to 10)
G0 X#1 Y#2 Z1 (Moves the axis with rapid to coordinates X=5, Y=10, Z=1)
#2 = 1.5 (Sets the value of variable #2 to 1.5)
G1 X#1 Y#2 Z#2 (Moves the axis with set feedrate to coordinates X=5, Y=1.5, Z=1.5)
#3 = #1 + #2 (Sets the value of variable #3 to 6.5)
#4 = 100 (Sets the value of variable #4 to 100)
G1 X#3 F#4 (Moves the axis with set feedrate of 100 unit/min to coordinates X=6.5)

To program complex equations it is often required to use brakets. Because in the RS274NGC programming the round brackets '(' and ')' are used to defined comments for the equations the rectangular brackets '[' and ']' should be used. Also to define a comma for functions which has more than one parameters use the ';' character instead of the ',' comma.
For example: #1= [1 + #2]*3

And there are two available built in constant variables, these are the 'pi' and the 'e'.

### 4.2.1 .Available math operators and operations

In the equations the constants and internal variables can be mixed with mathematical operators and functions to calculate complex equations inline. The mathematic operators and the supported functions are listed below:

| Operator | Name of function | Short description |
|---|---|---|
| + | summation | Summarise numbers. |
| - | deduction | Deduct numbers. |
| * | multiplication | Multiply numbers. |
| / | division | Divide numbers. |
| % | division for remainder | Returns the remainder of a division. |
| ^ | power | Returns a specified number raised to the specified power. |
| ? | Get variable value | Shows variables value in a window.<br>For example ?#1 prints the value of #1 variable.<br>Note: This operator works in MDI input only. |

| | | |
|---|---|---|
| abs | absolute value | Returns the absolute value of a number. |
| acos | arc cosine | Returns the angle whose cosine is the specified number. |
| asin | arc sine | Returns the angle whose sine is the specified number. |
| atan | arc tangent | Returns the angle whose tangent is the specified number. |
| cosh | hiperbolic cosine | Returns the hyperbolic cosine of the specified angle. |
| exp | exponentiation | Returns e (the base of natural logarithms) raised to the specified power. |
| floor | floor | Returns the largest integer that's less than or equal to the specified number. |
| log | logarithm | Returns the natural (base e) logarithm of a specified number or the logarithm of a specified number in a specified base. |
| log10 | 10 base logarithm | Returns the base 10 logarithm of a specified number. |
| min | minimum | Returns the smaller of two numbers.<br>Example: min[1;2] gives an output of 1. |
| max | maximum | Returns the larger of two numbers.<br>Example: max[1;2] gives an output of 2. |
| pow | power | Returns a specified number raised to the specified power.<br>Exampe: pow[2;3] gives an output of 8. |
| rnd | round to 0 to 9 decimals | Rounds the input number to 0 to 9 decimal places.<br>Exampe: rnd[1.234;2] gives an output of 1.23. |
| sin | sine | Returns the sine of the specified angle. |
| sinh | hyperbolic sine | Returns the hyperbolic sine of the specified angle. |
| sqrt | square root | Returns the square root of a specified number. |
| tan | tangent | Returns the tangent of the specified angle. |
| tanh | hyperbolic tangent | Returns the hyperbolic tangent of the specified angle. |

## 4.3 .Using the software, executing codes, using functions

### 4.3.1 .Open, edit, close a G-code file

G-codes files are codes generated with a CAM software or simply written by hand using a notepad. These code files must have a format defined in the RS274 standard and also detailed in this user's guide. The UCCNC software can open, read and execute these files files.
To open a G-code program press the load file button in the software, this action will start a standard open-file dialog where the file can be selected and opened. When opening the file the G-code interpreter checks the file through, interpret the code written inside and places it into the G-code viewer control. Also the path for the file is calculated and showed in the Toolpath viewer control.

To rewind the file to the beginning (first line) press the rewind file button. To close the file press the close file button. Closing the file will unload the file from the G-code viewer and the Toolpath viewer and also the file will and the associated variables and structure will be unloaded from the computer's memory.

The file can be also edited to do that press the edit file button, this will open the file in notepad.exe software which is built into the Windows Operating System. When the file editing is finished simply close the file and the UCCNC software will automaticaly reload the file.

### 4.3.2 .Executing code from program

To execute, stop a program use the buttons on the right side of the screen. These buttons are visible on all tab pages. To start a Cycle press the "Cycle Start" button. The virtual LED on the bottom right corner of the button will light while the cycle is running. When the program execution ends the virtual LED goes off.
To run a single line of code only press the "Single Cycle" button, this will execute a single line of code only. The virtual LED in the bottom right corner of the button will light while the cycle is running and the LED goes off as soon as the execution of the line finishes.
When a Cycle or a Single line is being executed press the "Stop Cycle" button any time to stop the Cycle. When pressing the "Stop Cycle" button the Cycle is stopped which stops the motion with decceleration in case any axis was moving. The decceleration of the axis happens with using the acceleration values set for the axis.
To stop the Cycle and movement immediately without a decceleration press the "Reset button". This will stop the Cycle and the motion without decceleration, immediately. The "Reset button" is a toggle type of button, it blinks if pressed in and this blinking means that the machine is in reset and it cannot be moved, no motion can be executed. Pressing the "Reset button" again brings the machine out of reset.

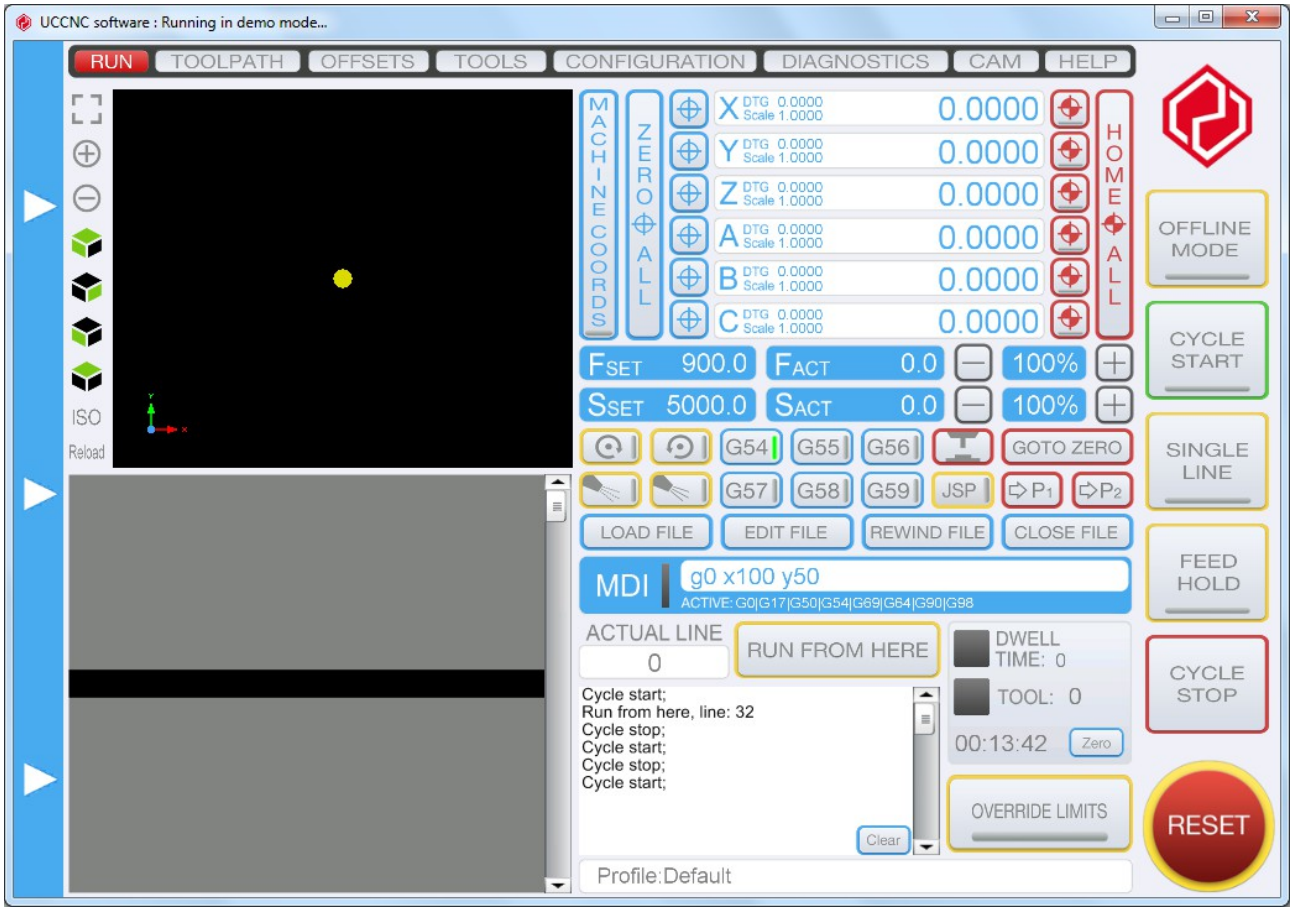
The software can be put to Offline mode with pressing the "Offline" button. In offline mode all outputs of the UC100 device goes to the default output state and no output signals are generated. The software in the offline state operates as if it was just simulating the motions and the functions. The offline mode can be used for example to run a G-code software without the machine moving and check what the machine will do prior the real machining cycle.

### 4.3.3 .Executing code via the MDI

The MDI (Manual Data Input) control provides the ability to execute a G-code command without loading a G-code program. The MDI is an input field where a text can be entered and after typing the text command pressing the enter will make the software interpret the entered text and execute the command. The textfield of the MDI control will blink green color if the input text is understandable and executable by the interpreter and it will blink red color if the text is not interpretable. After a red clor blinking no action will be taken by the software. After a green blink the software will execute the command which was entered.

The right side of the MDI control shows the active motion mode of the MDI. The active motion modes can be G0, G1, G2 or G3. When any of these modes are active inputting only axis woords with coordinates is valid. For example if the G0 is the active motion mode then a 'X1 Y2' text typed into the MDI will move the axis with a rapid linear movement from the current coordinates to the X=1 and Y=2 coordinates.

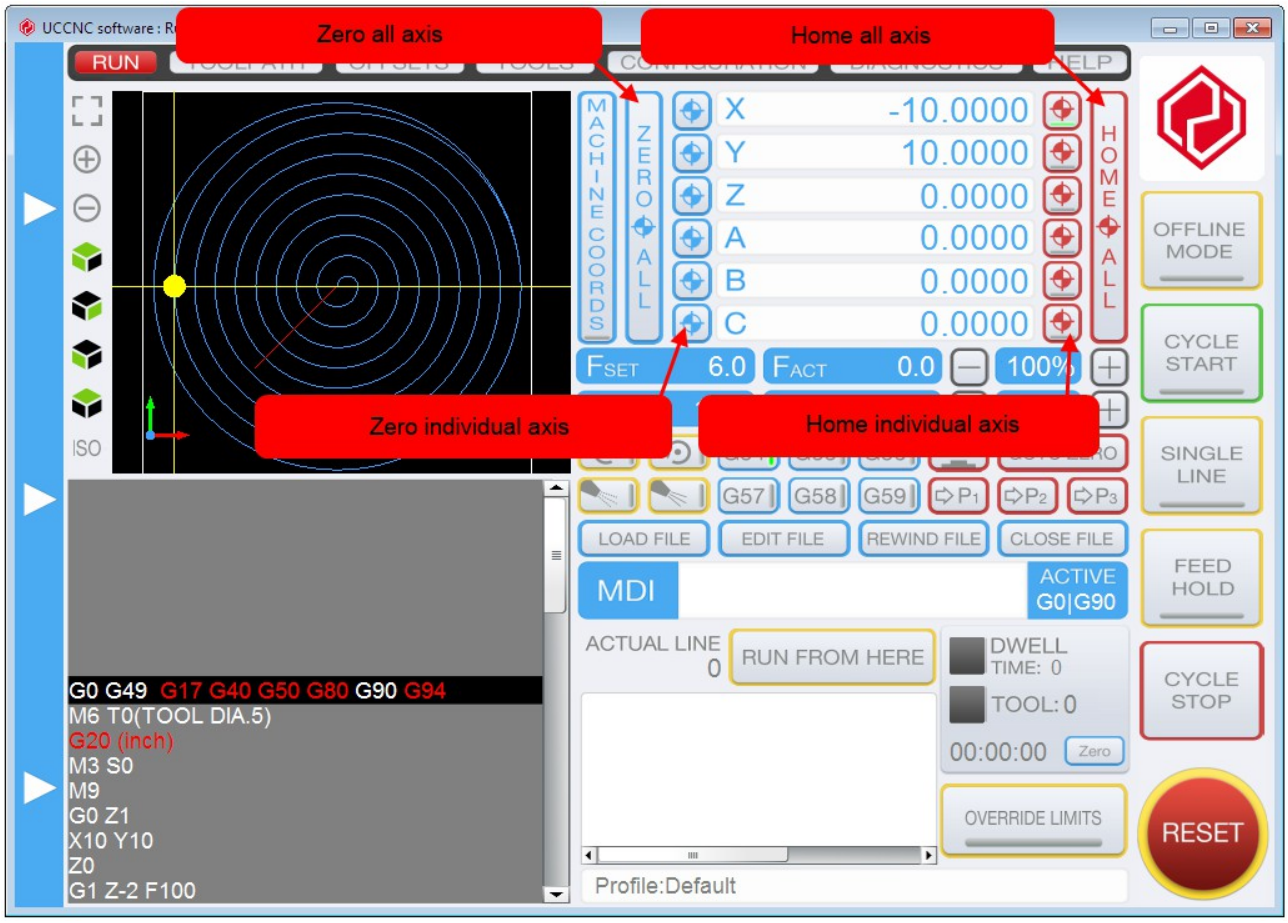
The following printscreen picture shows text input in the MDI control:




### 4.3.4 .Zeroing and referencing one or more axis

All of the 6-axis can be zeroed and referenced (homed) using the axis zero and home buttons.
Axis can be homed and referenced individually or all pressing a single button. When all axis are referenced, the homing is done in sequence.

The following picture shows the ref and home buttons:




### 4.3.5 .Using offsets

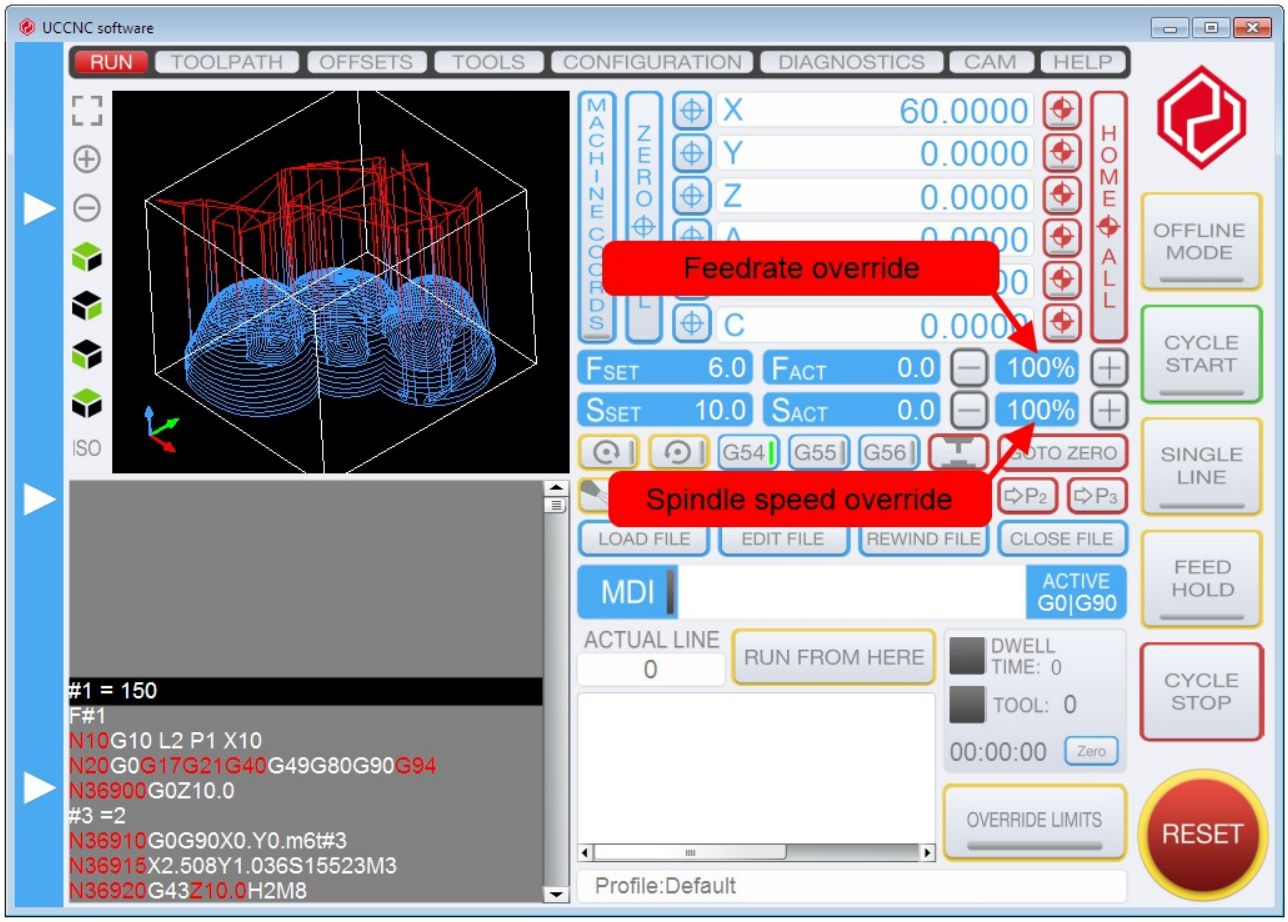
Offsets are used to define different origin coordinate system inside and derived from the machine coordinate system. The machine coordinate system origin point is defined by the reference (home) points. The offset coordinate system are offset from the machine coordinates by the user defined lengths. Different work offset values can be defined for each axis and for the Z-axis also a Tool length offset can be defined which offset is useful when compensating the drilling or milling tool's length in the machine's chuck. There are in total 6 pieces of offset coordinate systems, these are named G54 to G59. The different coordinate systems can be selected using the on screen offset buttons or programming the G54 to G59 codes in program or via the MDI.
When selecting a coordinate system the screen set offsets are applied to the system and the coordinate DROs' values are compensated with the offset values.
The Offset coordinates can be edited on the Offsets tab page.

The following picture shows the Offsets tab page:


### 4.3.6 .Overriding the feedrate and the spindle speed

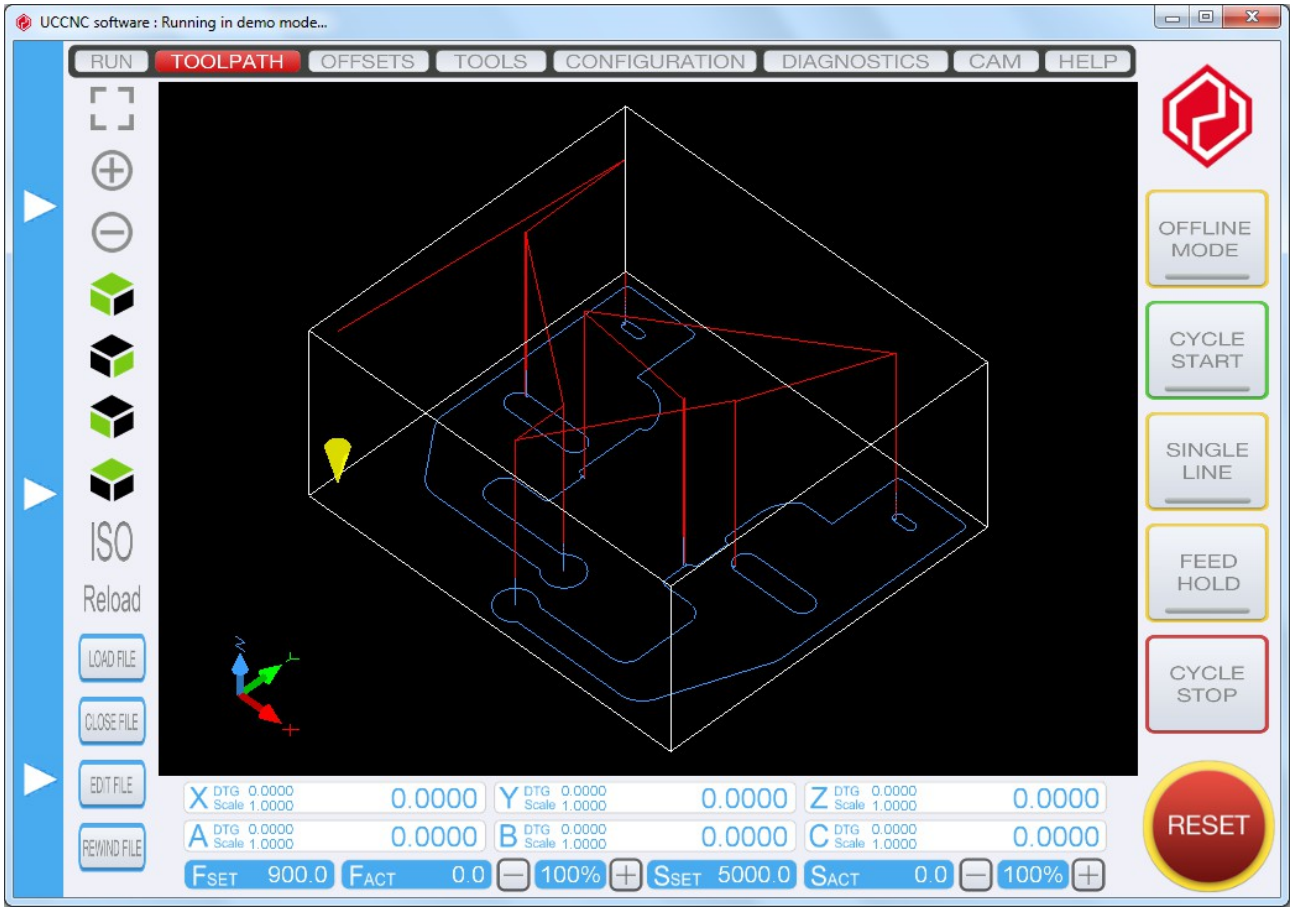
The programmed feedrate and spindle speed can be overridden any time, even when a code is executed and when a motion is in progress. Both properties can be overridden in the range of 0-300% of the programmed value. The override feature is useful for example if the programmed feedrate is too slow or too fast and need to be adjusted, but the operator does not want to bother with regenerating the code with new feedrate values. The same is true for the spindle speed override, if the spindle speed is controlled from software then the programmed speed can be overriden on the fly which makes the system more comfortable for the operator and more productive saving lots of additional work and time.
To override the feedrate use the + and – buttons next to the feedrate and the spindle speed DROs. Pressing the plus button increases the value with +10% if the value is equal or above 10% and it increases +1% if the actual value is below 10%. Pressing the minus button decreases the value with 10% in case the current value is above 10% and decreases the value with 1% if the actual value is equal or below 10%.
If the feedrate override is set to 0% then the motion stops and stalls till the DRO value is changed to any higher value.

The following picture shows the Feedrate override and the Spindle speed override interfaces:




### 4.3.7 .Using the toolpath and G-code viewers.

The toolpath viewer is a 3D openGL control which shows the actually loaded toolpath. The toolpath drawing in the toolpath window can be rotated, panned, zoomed and moved using the mouse pointer and mouse buttons.
There are also buttons next to the window to apply different viewpoints to the window.




### 4.3.8 .Jog control, jogging the machine

The jog controller is used to move the machine manually with pressing screen buttons. Also the manual pulse generator and the associated buttons can be found on this screen.

The jog panel is located on the very left side of the screen and on startup this screen is hidden, only it's right side border is shown. Touching the panel's border with the mouse pointer makes the jog panel to pop and appear on the screen. The panel is not available when a motion is in progress, when the machine controller is busy executing commands.

There are + and – jog buttons on the jog panel, pressing these buttons all axis can be jogged to the negative and to the positive directions. The Jog feed sets the feedrate of the jog movements, the value is defined in percentage of the set maximum (G0) feedrate of the axis.

The mode of the jogging can be selected with buttons on the screen, there are continous and stepping modes and also there is an MPG mode.

In continous jogging mode the machine axis jogs while the jog button of the axis is being pressed and the jog finishes when the button gets released.

In step jogging mode the axis moves the selected distance for every jog button presses. Currently there are 0.01, 0.1 and 1 unit lengths are selectable.

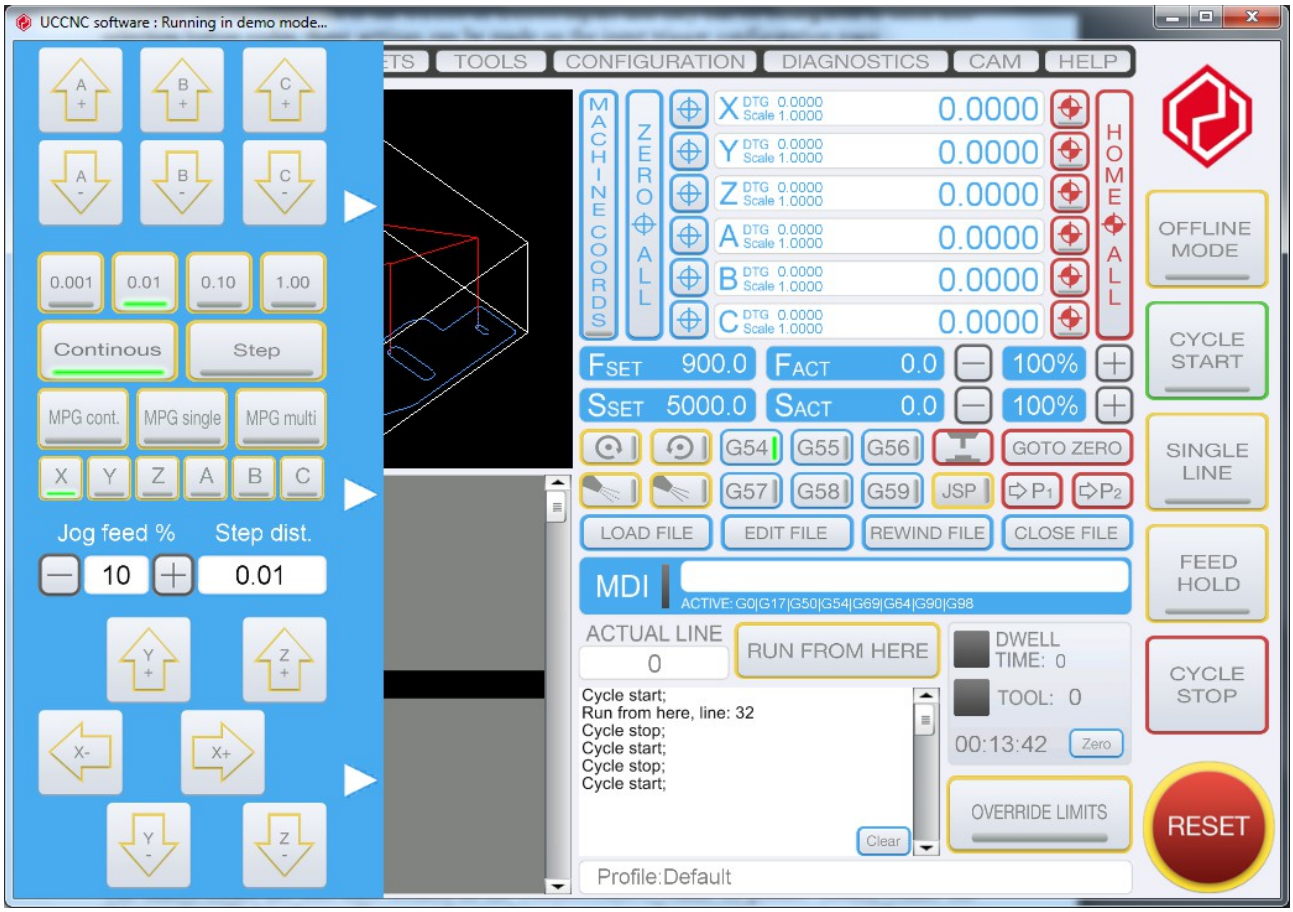
In MPG mode the external MPG device is selected for the jogging and selecting this mode disables the screen buttons jogging. The MPG mode has 3 running modes, the MPG cont., the MPG single and the MPG multi modes.

In the MPG cont. mode when rotating the MPG wheel the axis continously moving with the wheel rotation.

In MPG single mode when rotating the MPG wheel the axis is moving the jog step distance on the first encoder tick, the distance is selected with the distance buttons 0.001, 0.01, 0.1 or 1.00 Units. The next movement happens after the motion is finished and on the next encoder tick.

In MPG multi mode when rotating the MPG wheel the axis is moving the jog step distance on all encoder ticks, the distance is selected with the distance buttons 0.001, 0.01, 0.1 or 1.00 Units.

The axis to move with the MPG can be selected with the X,Y,Z,A,B,C buttons.
Also there are button codes for the MPG axis selection, so external buttons or switches on the MPG controller can be connected to the UC100 or UC300 inputs and they can be configured to these axis selection button codes, these settings can be made on the input trigger configuration page.
The associated buttons codes are documented in the buttons_by_number.htm document in the documentation section of the UCCNC installation.
The following screen shows the jog panel on the left side of the UCCNC software window:




## 5 .Macro-ing capability.

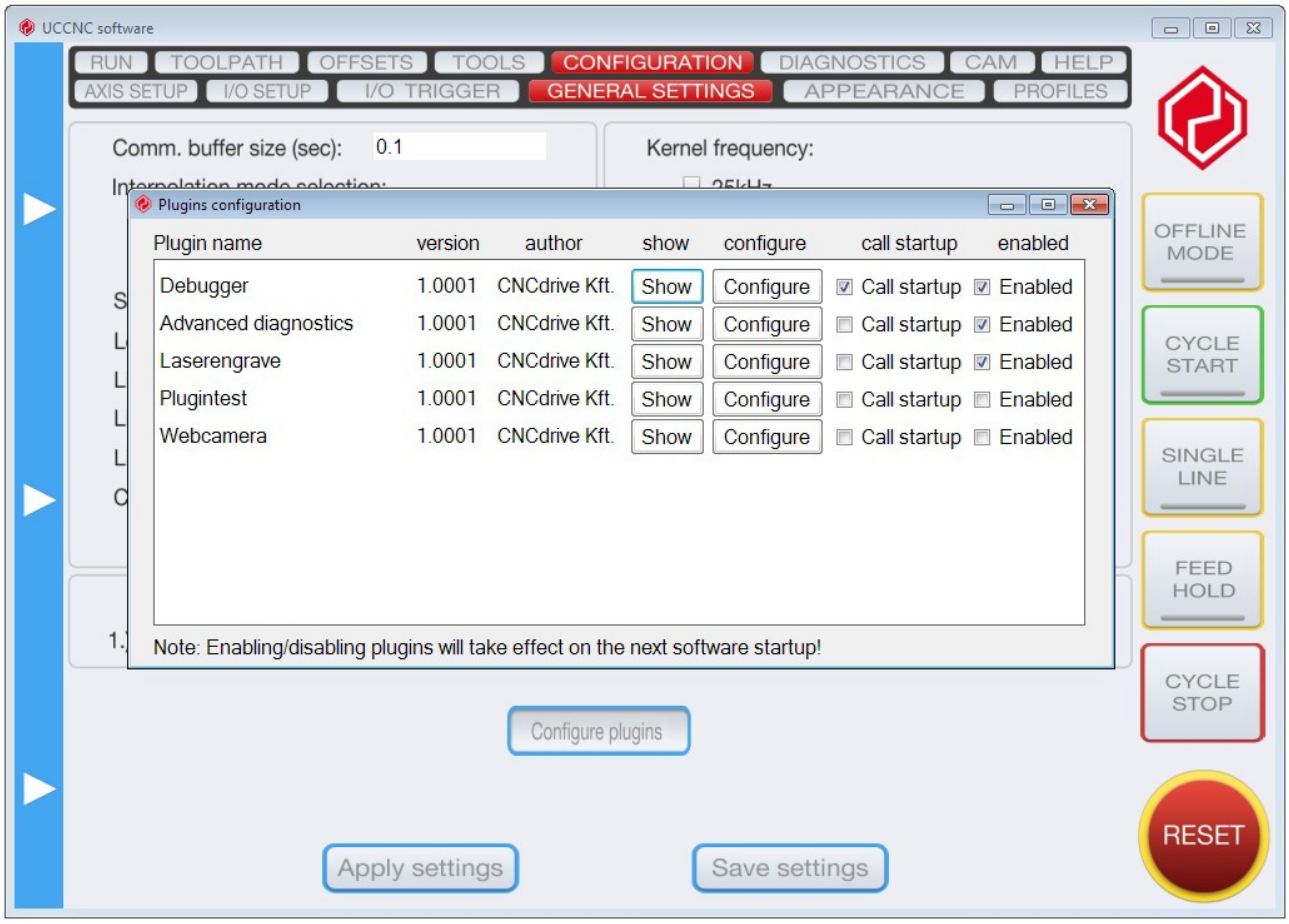
The macro codes are text files located in the Profiles\Macro_Name of profile\ folder, where the "Name of profile" is the profile name the machine is running. A macro can contain simple or even complex code to execute even a bunch of movements and I/O manipulations.
A macro can be called from program execution or via the MDI control with simply programming Mx, where the "x" means an integer number. When calling a macro the corresponding macro file is read, compiled and executed by the interpreter. If the macro file contains a syntax error then the macro is ignored and skipped and an error message is shown in the status list control.

There are default macros in the RS274 programming language, but also custom macros can be written by the user with creating the required macro file in the profile's macro folder and writing the script into the file using notepad.exe for example.

Macros can be edited using a text editor like notepad. The macro files can be edited on the fly, because the macro scripts are read and compiled on runtime, when the macro is called.
For informations about writing custom macro codes please read the Macroing_capability_detailed.pdf document in the UCCNC/Documentation folder.

# 6 .Plugin control modules

Plugin controls are software modules which builds into and works within and together with the UCCNC software.
The following picture shows the plugin configuration window which window can be opened with pressing the Configure plugins button on the Configuration/General settings tab page.




## 6.1 .Installing plugins

To install new plugins simply copy the plugin .dll file to the /Plugins subfolder in the UCCNC installation folder. The software will automatically detect the new plugin on the next software startup. By default the a newly installed plugins are disabled, in order to make them run they have to be enabled on the Plugin configuration window.

## 6.2 .Enabling, configuring and using plugins

Every plugin can be configured using the Plugin configuration window. The plugin properties are shown in this windows. Every row on this windows lists one plugin and the first text section of this row shows the name, the version number of the plugin and the name of the author. The plugins can be opened up with pressing the Show button in the row of the plugin. Pressing the Show button calls the Showup event of the plugin if the actual plugin has this interface implemented. In most cases this event opens and shows a Window which represents the plugin. If the plugin does not implement the Showup event then the UCCNC software will show an error message with a message

about this. Some plugins which just running in the background may not need a visual interface and may not implement the Showup event.

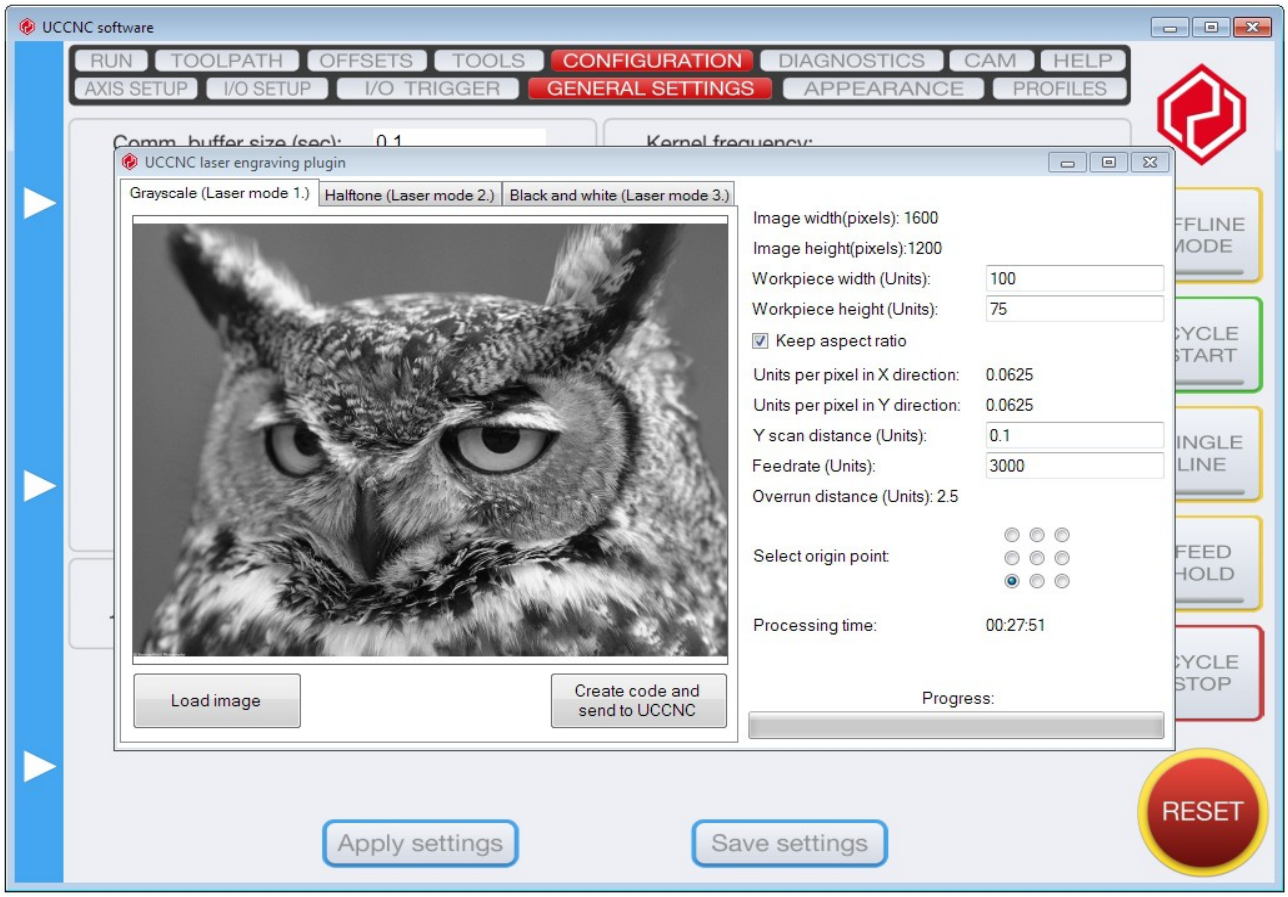
To configure a plugin press the Configure button in the row of the plugin, this will call the Configuration event of the plugin. Usually this event will popup a new window where the user can set parameters about the plugin. If the plugin does not implement the Configuration interface then the software will show an error message about this. Some plugins may not need a configuration UI, because there is nothing to configure in them or the plugin writter decided to place all configuration settings on the main window of the plugin which can be called with the Show button as described previously.

The plugins can be individually enabled with checking the enable checkbox of the plugin. If a plugin is enabled then it will start running when the UCCNC software starts up and when it has finished all of the initialisation process. If the plugin is disabled then the UCCNC will not call the create the plugin object and also no other messages will be sent to the disabled plugin, in other words the plugin will be not running.

If checking the call startup checkbox of the plugin then the plugin's startup event will be called on the UCCNC startup. It must be noted that the plugin mist be enabled for the startup function to be called. The startup event mostly contains code which opens the plugin Window, but how it is implemented depends on the plugin's developer. Plugins which has no visual appearance just running in the background may not show any windows on the software startup, but may still do things in the background when this event is called. How and for what the plugins can be used for depends on the purpose of the plugin. Plugins can implement different and even complex features which features may be useful for some users with different machine setups and different needs for different functions.

An example plugin is the laser engraving plugin. With the use of this plugin scan type laser engraving can be done with the UCCNC software. The plugin can read picture images and converts it to grayscale or black and white image and then to vectors and executable code for the UCCNC.

The following picture shows the laser engraving plugin:




### 6.3 .Writing/creating custom plugins

The plugin interface allows programmers to create their own plugin modules for the UCCNC software. A plugin modul is a .dll (dynamic link library) file which is placed into the UCCNC installation directory and which software module can then communicate and interact with the UCCNC software. To write a plugin the plugininterface.dll file (located in the UCCNC directory) should be included into the plugin project. The plugin interface is a .NET 4.0 based code and so the plugin has to be also .NET 4.0 based. We advice to use Visual Studio to create new plugins.

In the plugins folder there is an Example folder which contains the source code for an example plugin. This source code could be studied and may be modified for a new plugin creation. Since Visual Studio has intellisense the available function calls to the UCCNC can be easily explored and accessed via the plugin interface dynamic library.

## 7 .Screen editor

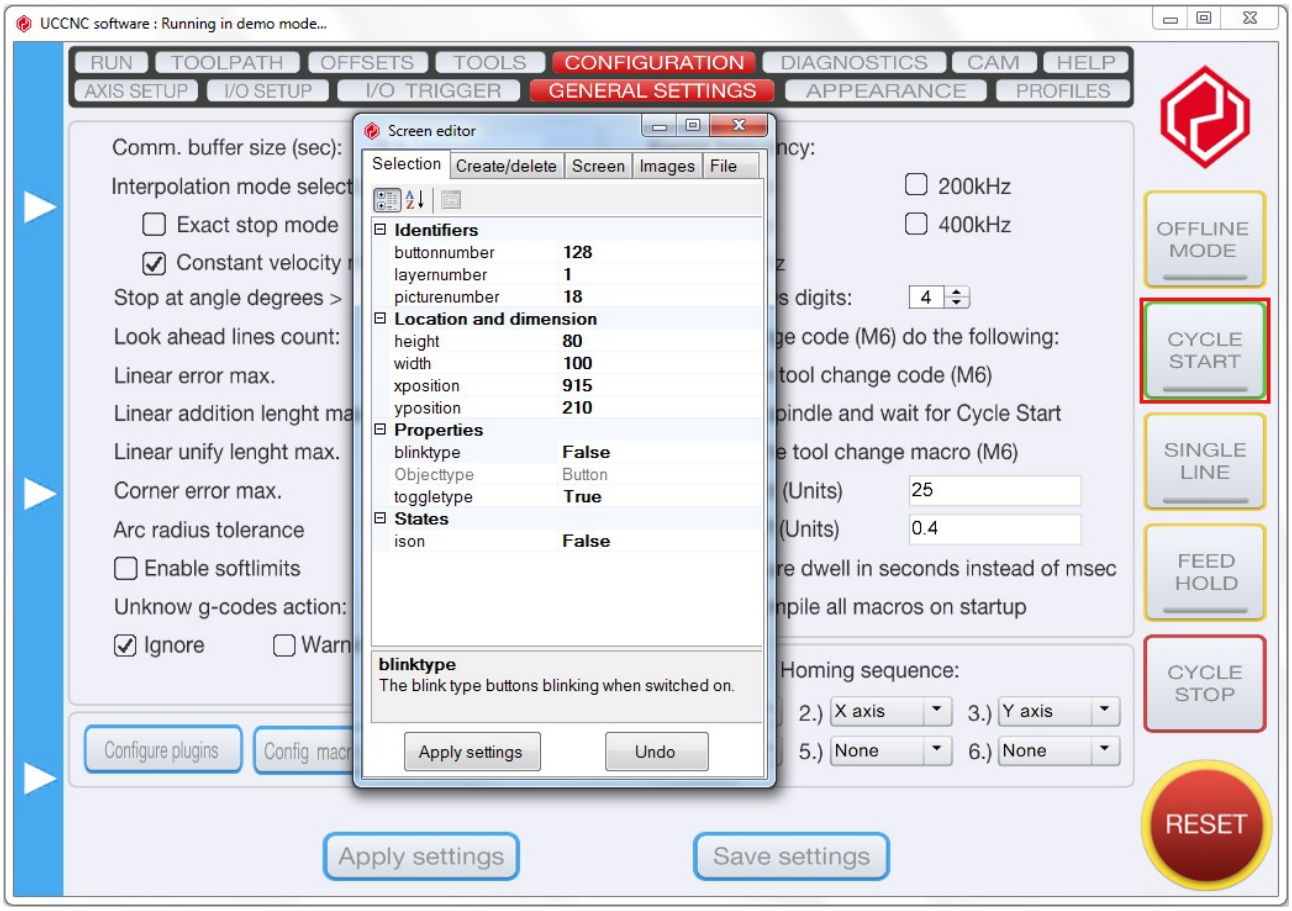
### 7.1 .Description of the screen editor

The UCCNC software has a built in screen editor module. To enter the screen editing mode press the Edit screen button on the Configuration/General settings tab page.

Entering the screen editor mode any of the screen elements like the buttons, textfields, labels, tab pages, backgrounds, LEDs, etc. can be resized, replaced, removed or any type and number of new elements can be added to the screen.

In screen editor mode all functions of the UCCNC are disabled/inactive, pressing buttons will take no effect. To use the software functions again the screen editor mode must be left, this can be done with closing the screen editor window.

The following printscreen shows the screen editor window with the screen editor mode active:




## 7.2 .Editing screen elements

When in screen editor mode pressing any screen items will select the item. The selection will be indicated with a blinking red border around the selected item.

After selecting an item the property grid control on the Selection tab page of the screen editor window will list the properties of the item. The properties can be changed with overwriting the values in the property grid control. To validate the new values press the Apply settings on the bottom of the tab page. After validating the new values the changes will immediately take effect and the item will change position or size etc. depending on which properties where changed.

The screen items can be also dragged and resized using the mouse. After the item was clicked and selected clicking the item again with holding the mouse button down and moving the mouse will drag the item. When releasing the mouse button the item's new position will be registered and shown in the property grid. To resize an item move the mouse on the blinking selection border and when the cursor changes to the resize arrows then press the mouse button, hold it down and move the cursor to pull and resize the item.

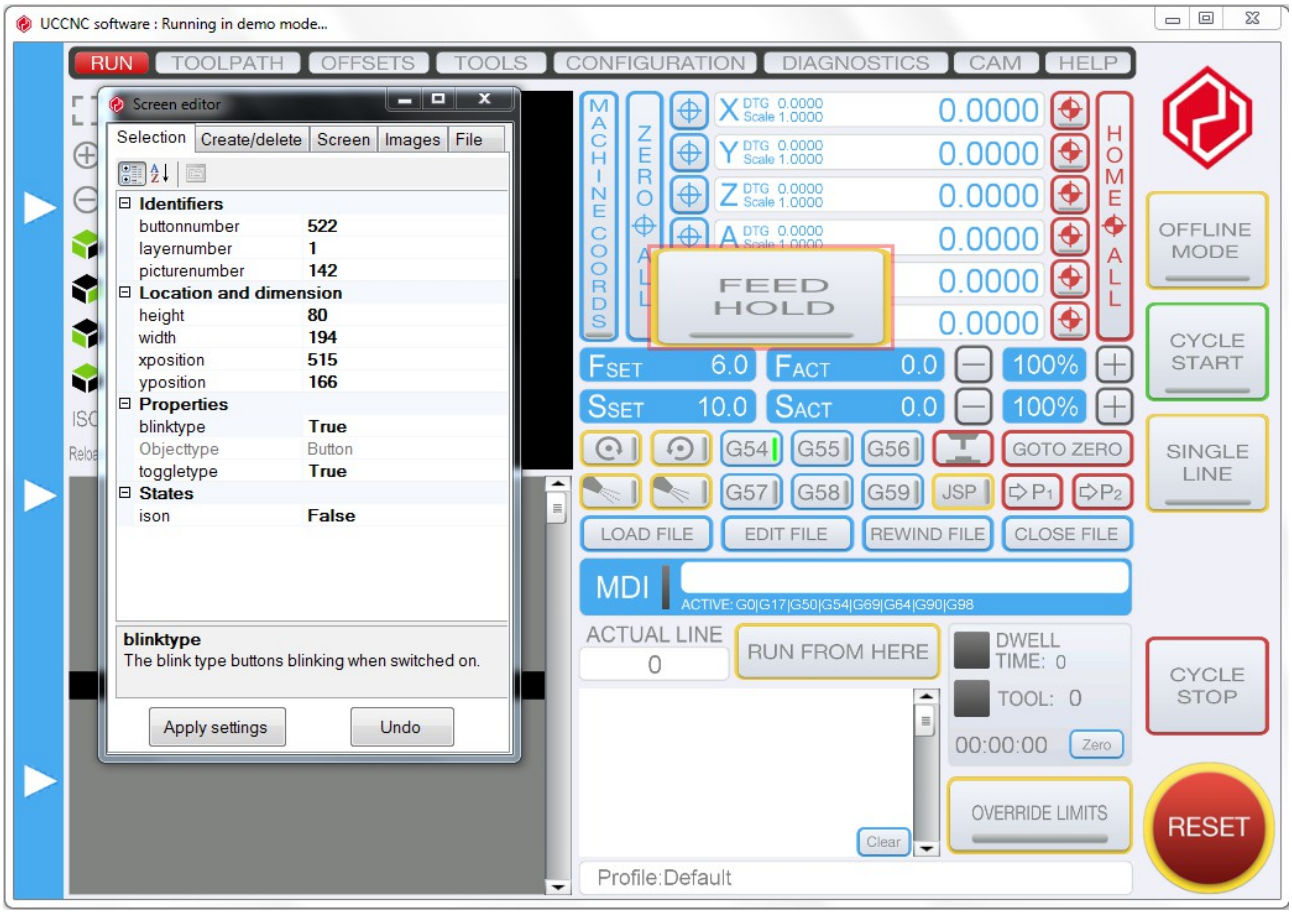
To undo the last change press the Undo button. The Undo button will only reverse the last change and only while the item is still selected.

To change between tab screens while in screen editor mode press and hold the Shift keyboard key and click the tab page's header with the mouse.

To edit the fills (hider panels) press the control keyboard key. Pressing the Control key will toggle the fill items' visibility when working in screen editing mode. When the fills are visible then they

can be selected and edited just like the other screen items.

The following printscreen shows an example where the Feedhold button' is selected and it's position and dimensions were changed.



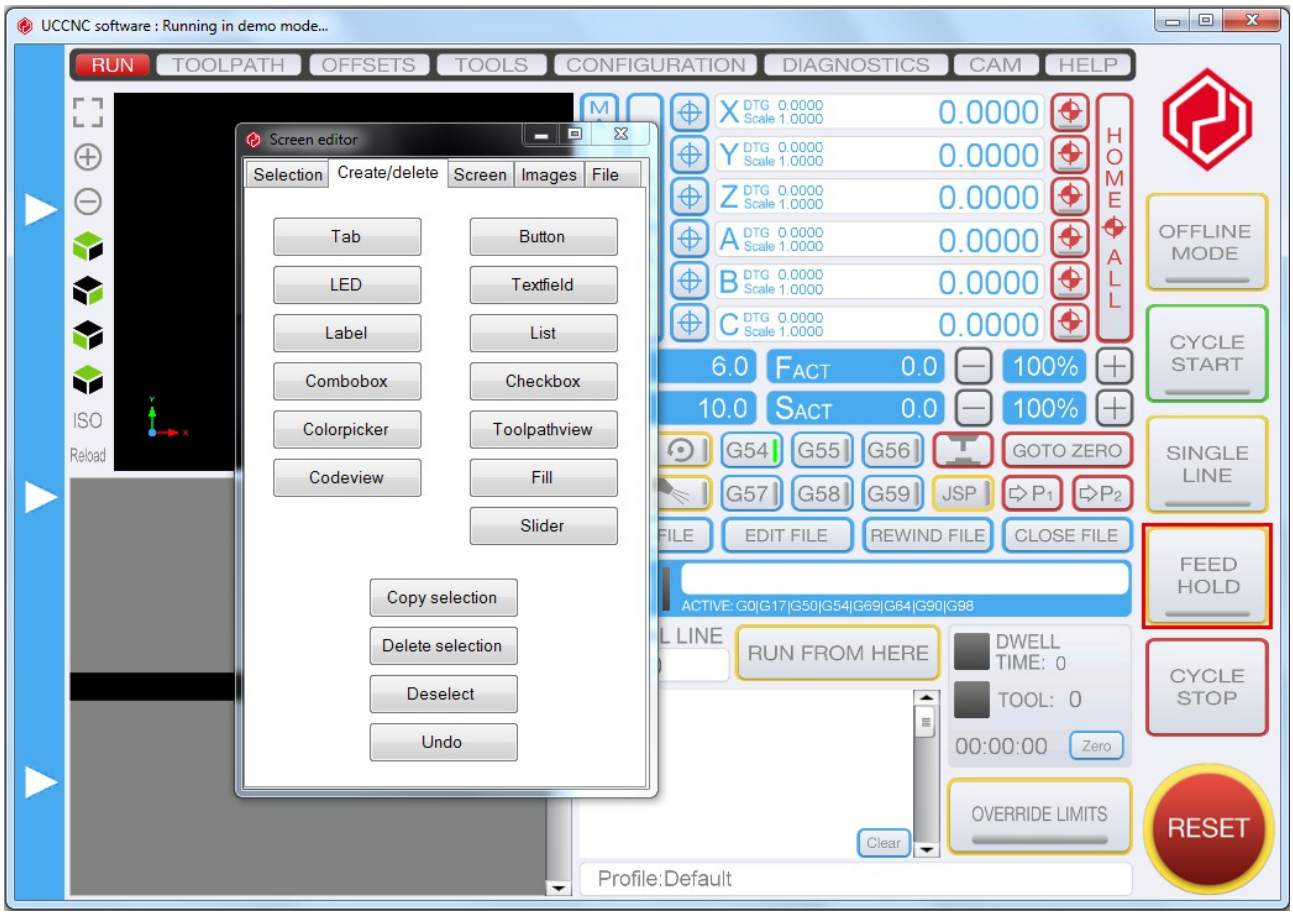

To add new items or to delete an existing one select the item with clicking on it and go to the Create/delete tab page. There are individual buttons to create different type of new objects. Pressing these buttons will create a blank new screen item of the selected type.

To delete an item after selecting it click the Delete selection button.

To copy an existing item select the item and press the Copy selection button.

The new item will be created and shown on the middle of the screen. The new item will be blank, the properties of the new item can be filled afterwards.

The following printscreen shows the Create/delete tab page of the screen editor window:




## 7.3 .Editing the canvas properties

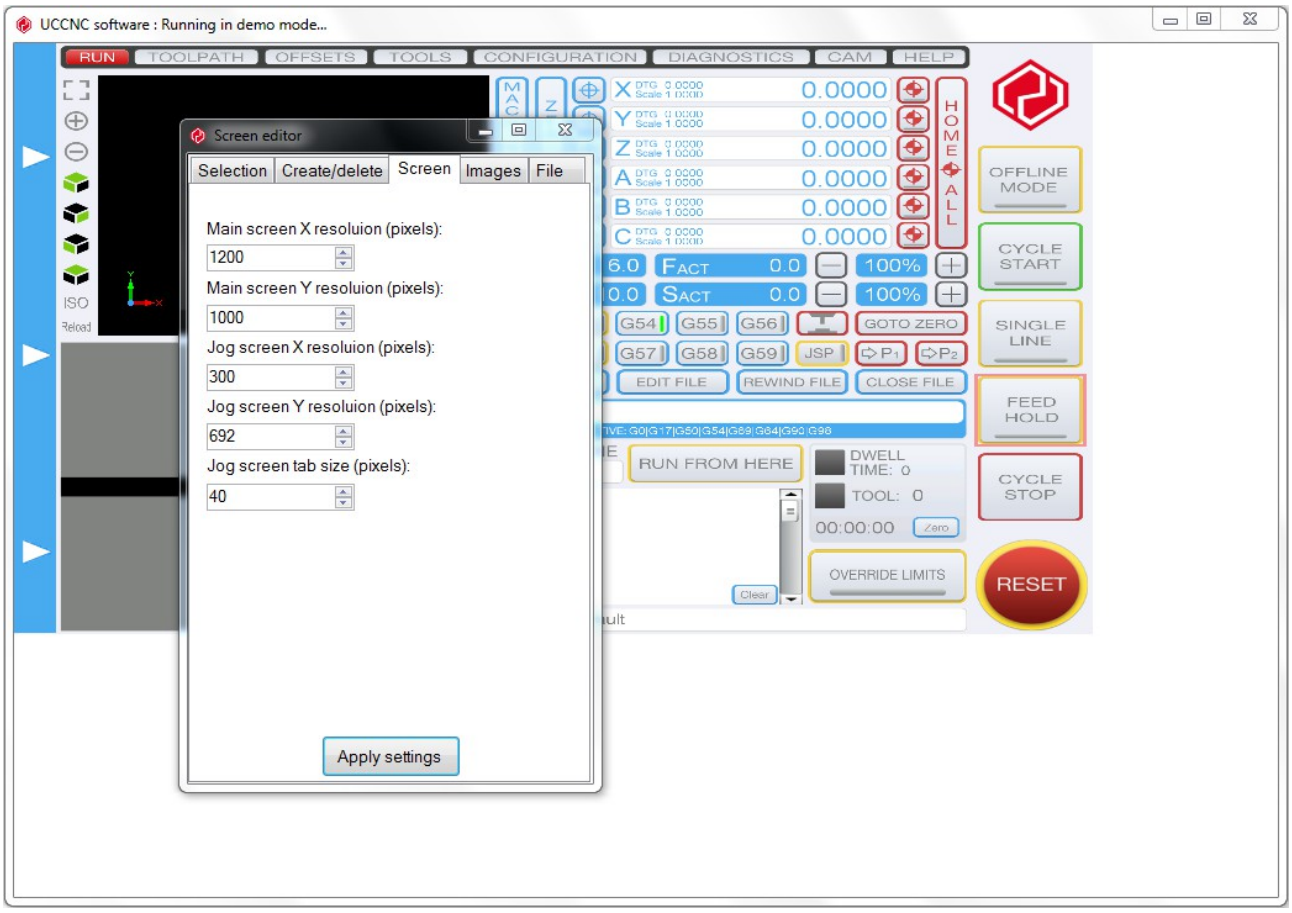
The resolution of the main screen canvas can be set on the screen editor window on the Screen tab page. The screen resolution is defined in pixels. Changing the screen resolution can be useful when different monitors, aspect ratios are used. Different screensets can be designed to fit specific monitor resolutions.

The coordinates of the screenset items will be relative to the set resolution. For example if the screen vertical resolution is 1000 pixels and a button's vertical position is at 500 pixels then the button' left side will be shown on the middle of the vertical resolution.

If the screen vertical resolution is changed to 2000 pixels and if the button will still have the 500 vertical coordinate then it will appear at the first quarter of the screen.

The vertical and horizontal resolutions of the jog screen can be set separately on the same tab page and in addition the tab size of the jog screen can be also defined. The tab size means the number of pixels which will remain visible from the jog screen at the left side of the window when the jog screen is in it's hidden state.

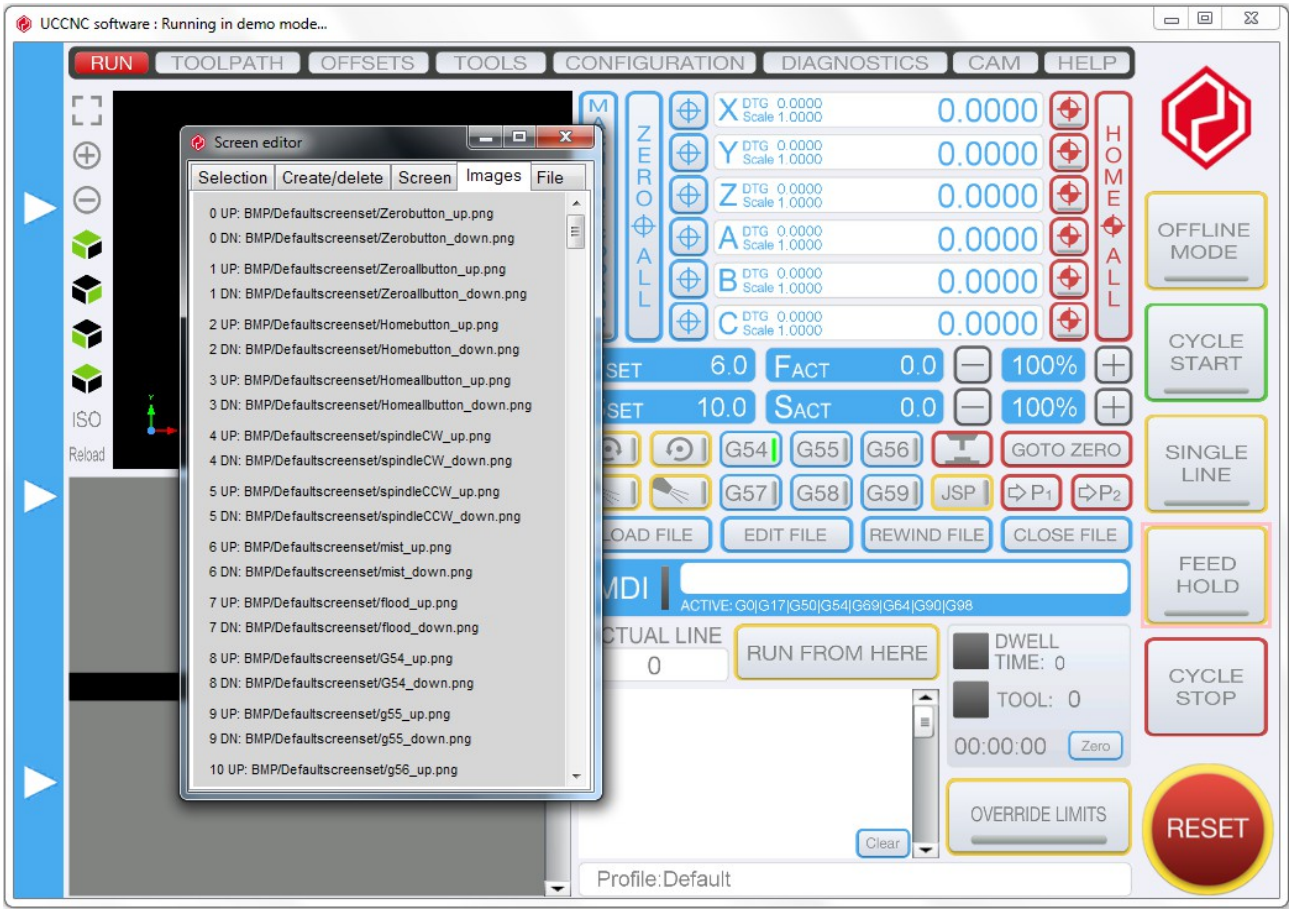
The following printscreen shows a case when the main screen's horizontal and vertical resolutions ware both changed:




## 7.4 .Editing picture images

Most of the screen items like the buttons, LEDs, tab screens, backgrounds rendering images. These images are defined in the screenset file. To edit the to be loaded images select the Images tab page on the screen editor window. This page lists all the to be loaded images names. All images in the list has 2 file references, because all buttons and LEDs etc. have an up and a down or on/off states. Separate images are used in these states of the screen items. To change the image reference click on the filename in the list, this will open an open file dialog and browse and select the new image file to be used. At the bottom of the page there are buttons to add new images or to delete an existing images from the list. Set the number of the new image in the numeric up/down counter control next to these buttons. Adding a new image will make a new entry in the image list.

The following printscreen shows the images list on the screenset editor window:

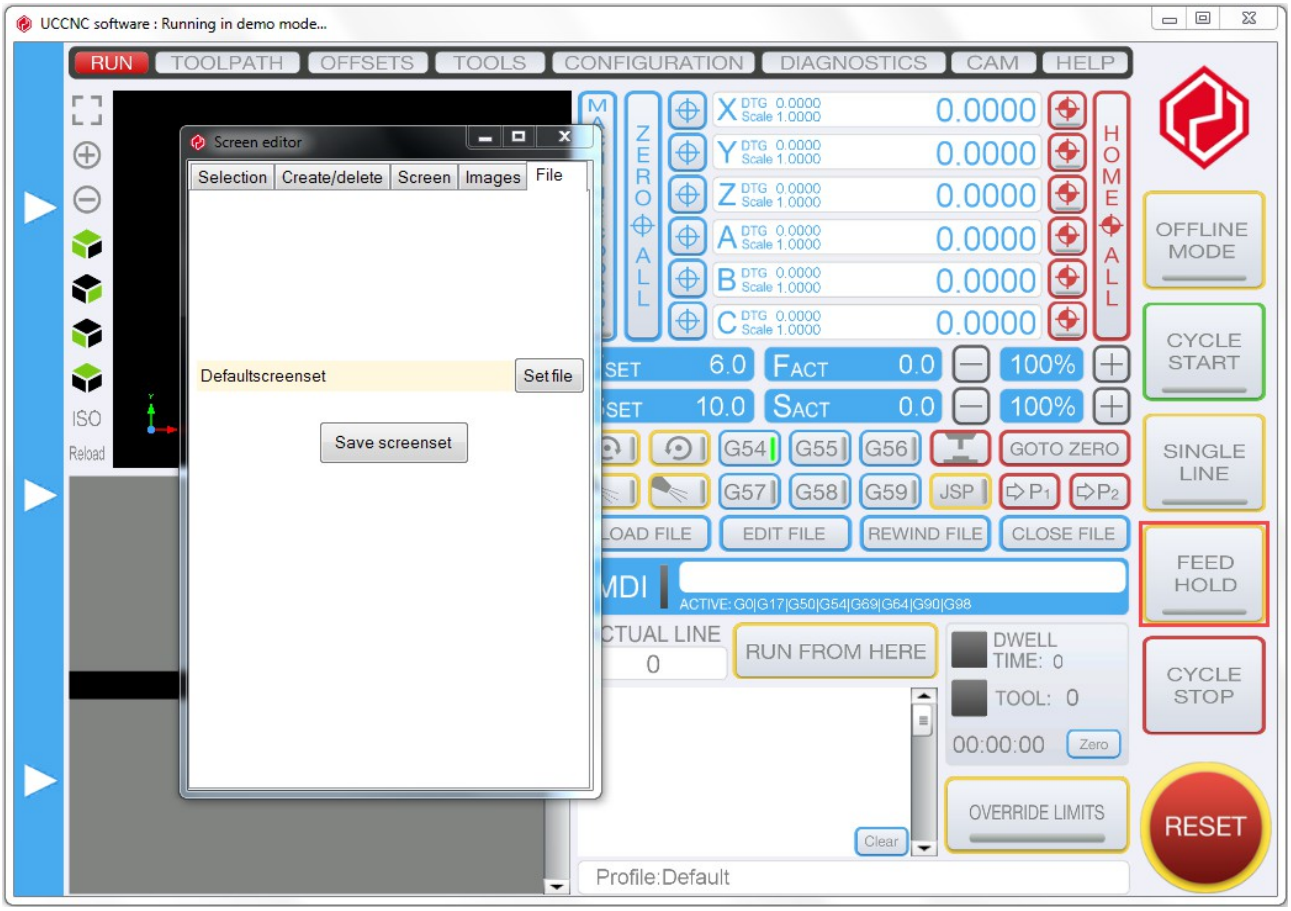



## 7.5 .Saving the screen

To save all changes to the screenset select the File tab page on the screen editor window and press the Save screenset button. To save the screenset to a different screenset file select the file with pressing the Set file button on the same page.

The screens for different motion controller devices are defined in sections inside the screenset file. For example if a UC100 motion controller or the UC100 demo mode is loaded then saving the screenset will save the screenset datas to the UC100 section in the screenset file. The section start and section ends are marked in the file with the //region devicetype and //endregion devicetype, where the devicetype means the device name. When starting up the UCCNC the software will read the screenset datas for the used motion controller device with reading the datas only from between the region definition markers. In case a device has no screenset region defined in a screenset file then the software will remove all the other regions and will execute the remaining datas which are outside all of the region declarations, in most cases this will mean that the screen will be empty white.

The following printscreen shows the File tab page of the screen editor:




# 8 .Macro loops

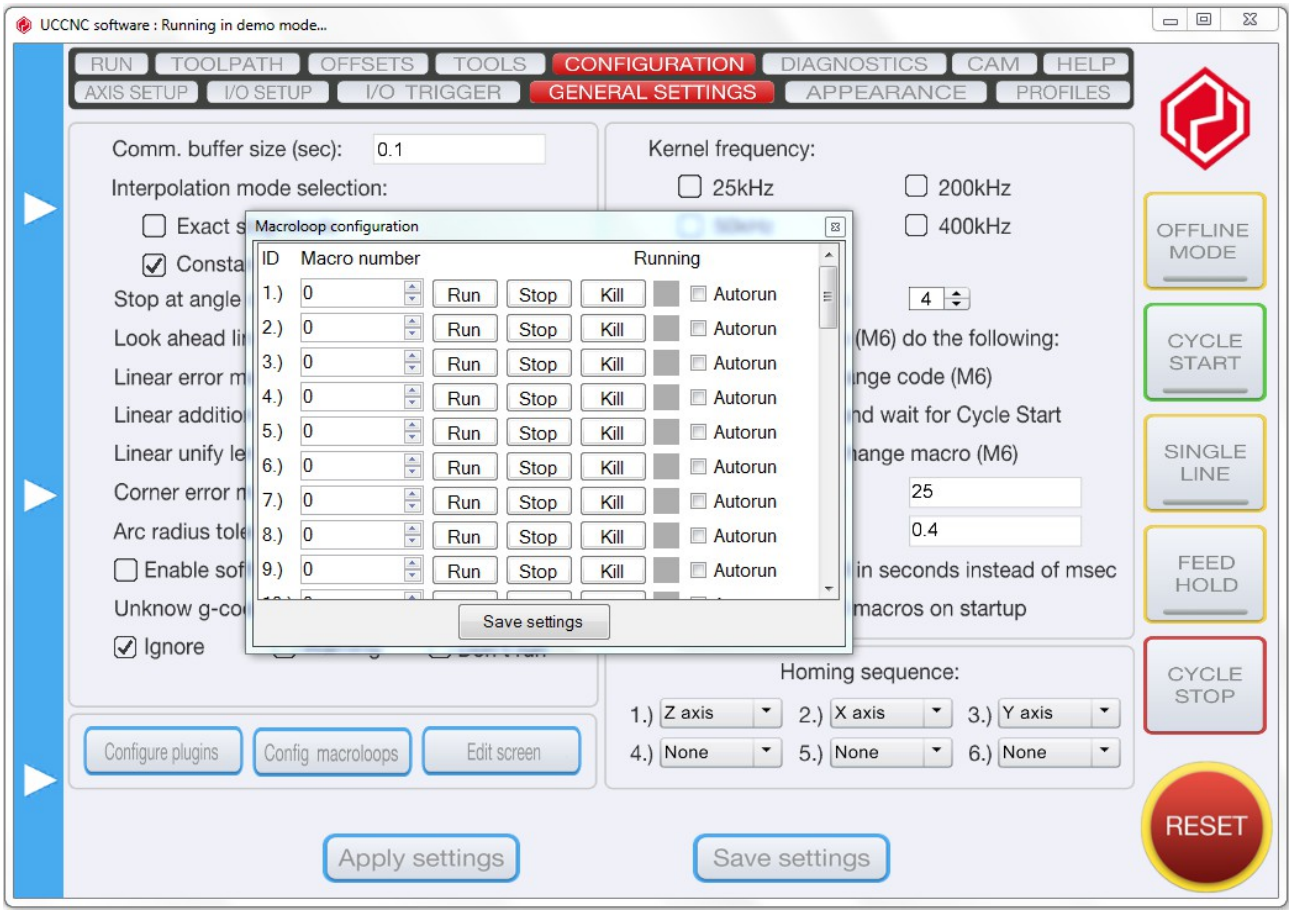
## 8.1 .Description of macro loops

Macro loops are program loops or in other words software threads. These threads can run one macro in an infinite loop. The loop runs in the background, asynchronously and independently from the other operations of the software. A macro loop can do it's own task while the software is operating and doing the user commanded operations. For example the macro in a loop can continuously monitor the value of a DRO and can trigger outputs of the motion controller based on the value read from the DRO or it can write new values to another DRO. One thing we recommend not to do is executing G-code and other motion commands from a macro loop, because they could conflict with other motions commanded by the user via the UCCNC GUI interface.

There are currently 48 available macro loop slots which means that up to 48 macros can be placed and run in separated threads.

The macro loops can be configured on the General settings page in the UCCNC pressing the Config macroloops button.

The following picture shows the macro loop configuration window:




## 8.2 .Writing macros for loops

The programming language, syntax and available functions are the same as for the standard macros, the only difference is the running in a separated thread, in a loop.
The code which surrounds the macro and written into the macroloop interface is the following:

```
while(loop)
{
//The macro code is executed here...
Thread.Sleep(50);
}
```

The 'loop' variable is internally defined and it's value is set based on the running state of the macro. If the loop is commanded to run then the loop variable is true, so the macro code is looped. When the loop is Stopped then the loop variable value becomes false, so the loop execution stops as soon as the macro code finishes.
There is an 50mseconds wait time programmed in the loop, so the highest frequency a macro could run is around 1/50msec = 20Hz. Of course the loop time can be longer than this, because the macro code itself takes some time to execute.

Because the macro runs in a loop the whole macro code is repeated from the beginning, this

arrises a problem that variables will always gets their original value on the next cycle repeat. For example:

```
int i= 0;
i++;
```

The above code defines a local variable and then it increments the variable with one.
The issue is that the loop repeats, so the variable will then again gets the value 0.
To overcome this problem one or more internal loops can be programmed inside the macro, like I nthe following code:

```
int i=0;
while(loop)
{
  i++;
  //Do the rest of the work here...
  Thread.Sleep(50);//It is important to do some wait, otherwise the CPU will be overloaded!
}
```

Please note that the loop variable was checked in the internal while loop and as previously described this variable will change value to false as soon as the macro is commanded to stop, so then the macro can safely terminate.

Important to note, that when the macro language is selected to be VisualBasic instead of C# in any macros with the #VB directive written in the first line of the macro then the loop variable is internally declared as mloop, because the loop keyword is a directive in VisualBasic and so it can't be used as a variable name.

### 8.3 .Running, stopping, killing macro loops

To place a macro into a loop open up the macro loop configuration Windows and select the number of the macro to run and then press the Run button. The green virtual LED in the same line will indicate if the macro starts running and the LED will be green until the loop stops running. To stop the loop press the Stop button, this will make the loop variable to become false and the thread can terminate.
To run the loop automatically every time the UCCNC is loaded and starts up select the Autorun option.
There can be cases when a macro can't be stopped, a simple macro code example for this is below:

```
while(true)
{
  Thread.Sleep(50);
}
```

The above code will not terminate on stop command, because the while loop is infinate and the execution will not jump out of the loop even in the macro stop button was pressed.
In this case the green Running LED will remain on indicating that the loop is still running.
The Kill button can be used to abort/kill the macro execution. However this will likely not

cause serious problems, but if a macro loop can't be stopped we recommend to investigate and correct the issue in the macro code.
To save the settings press the save button on the bottom of the window.