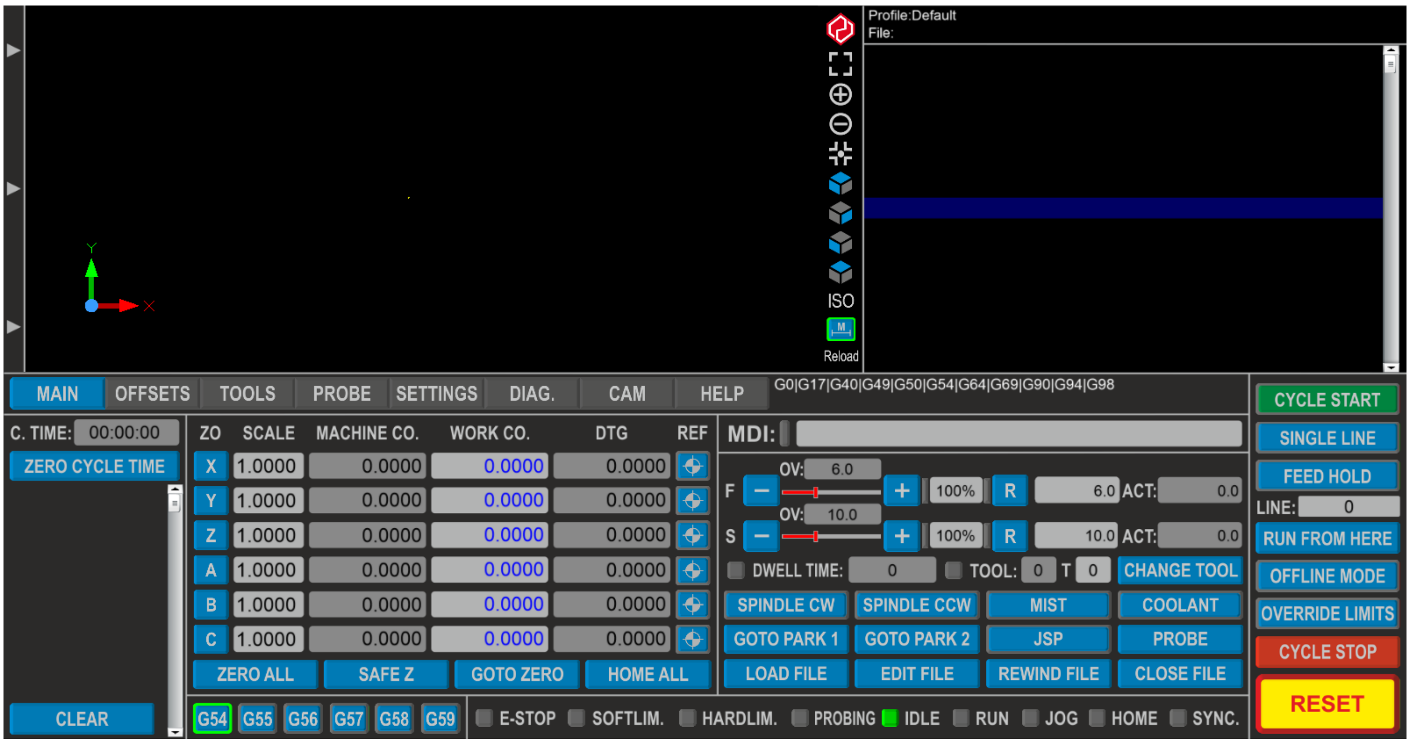

UCCNC MACROS, SCREENSET & MODBUS FUNCTIONS, BUTTON, FIELD, LED & CHECKBOX CODES

UCCNC VERSION 1.2113 REV A (28/12/20)

Robertspark

THINK BEFORE YOU PRINT

Contents

LIST OF G-CODES

G0 : Linear interpolation with maximum feedrate (Parameters X,Y,Z,A,B,C)
G1 : Linear interpolation with set feedrate (Parameters X,Y,Z,A,B,C)
G2 : CW arc interpolation with set feedrate (Parameters X,Y,Z,I,J,K,R)
G3 : CCW arc interpolation with set feedrate (Parameters X,Y,Z,I,J,K,R)
G4 : Dwell, wait for the programmed time interval (Parameters P in milliseconds)
G10 L1 : Load tool table offset (Parameters P ,Z) Notes: P can be a value 1-96 for tools number 1-96 in the tools table.
G10 L2 : Load offset origin point (Parameters P ,X,Y,Z,A,B,C) Notes: P can be a value 1-6 for G54-G59 offset tables.
G17/G18/G19 : XY/XZ/YZ plane selection (No parameters)
G28 : Run to home position (Parameters X,Y,Z,A,B,C) Note: The parameters are the intermediate coordinates.
G28.1 : Home axis (Parameters X,Y,Z,A,B,C) Note: The parameters are the intermediate coordinates.
G31 : Straight probe (Parameters X,Y,Z,A,B,C) Note: Only one axis is supported at a time.
G33 : Spindle synchronized motion (Parameters X,Z,K,Q) Note: K is the pitch per revolution. Q is the angle of penetration.
G33.1 : Rigid tapping, right-hand tap (Parameters Z,K,Q) Note: K is the pitch per revolution. Q is the peck depth.
G33.2 : Rigid tapping, left-hand tap (Parameters Z,K,Q) Note: K is the pitch per revolution. Q is the peck depth
G40 : Cancel cutter radius compensation (No parameters)
G41/G42 : Start cutter radius comensation left / right (Parameters D) Note D is the tool number in the tool table used for the cutter radius compensation.
G43 : Set tool length offset (Parameters H) Note: The H parameter number of tool length offset is loaded from tool table to tool offset.
G44 : Set tool length offset (Parameters H) Note: Identical to G43 code, but for negative tool length offsets.
G49 : Cancel tool length offset (No parameters)
G50: Reset all scale factors to 1 (No parameters)
G51: Set scale factors (Parameters X,Y,Z,A,B,C).
G52 : Temporary offset coordinate system (Parameters X,Y,Z,A,B,C).
G53 : Linear interpolation in the machine coordinate system (Parameters X,Y,Z,A,B,C) Note: Can be called with G0 or G1 modal active.
G54 - G59 : Work offset selection (No parameters)
G61/G61.1 : Set path control to exact stop mode (No parameters)
G64 : Set path control to constant velocity mode (Parameters D,E,H,L,P,Q)
G68: Rotate coordinate system (Parameters A, B, R)
G69: Reset coordinate system rotation (No parameters)
G73 : Peck drilling cycle with set backoff (Parameters X,Y,Z,Q,R) Note's is the final depth, Q is the depth increment, R is the retract plane.
G76 : Threading cycle (Parameters P,Z,I,J,K,E,L,Q,H) Note: See the documentation for detailed information.
G80 : Cancel canned cycle (No parameters)
G81 : Drilling cycle (Parameters X,Y,Z,R) Note: Z is the depth, R is the retract plane.
G82 : Drilling cycle with dwell (Parameters X,Y,Z,R,P) Note: Z is the depth, R is the retract plane, P is the dwell time in msec.
G83 : Peck drilling cycle with full back off (Parameters X,Y,Z,Q,R) Note: Z is the final depth, Q is the depth increment, R is the retract plane.

G84 : Rigid peck tapping cycle (Parameters X,Y,Z,K,Q,R,H) Note:Z is the final depth, K is the pitch, Q is the depth increment, R is the retract plane, H is the direction.
G85 : Boring cycle with feed out (Parameters X,Y,Z,R) Note:Z is the depth, R is the retract plane.
G86 : Boring cycle with dwell and spindle stop/start (Parameters X,Y,Z,R) Note:Z is the depth, R is the retract plane.
G89 : Boring cycle with dwell and feed out (Parameters X,Y,Z,R) Note:Z is the depth, R is the retract plane.
G90 : Select absolute distance mode (No parameters)
G91 : Select relative distance mode (No parameters)
G92 : Temporary offset to programmed coordinates (Parameters X,Y,Z,A,B,C).
G92.1 : Reset temporary offset coordinates (No parameters)
G93: Inverse time feedrate mode (No parameters)
G94: Units per miniute feedrate mode (No parameters)
G98 : Canned cycle return level to initial plane (No parameters)
G99 : Canned cycle return level to R plane (No parameters)

LIST OF M-CODES

M0, M1, M60 : Program stop
M2 : Program end
M3 : CW spindle relay on
M4 : CCW spindle relay on
M5 : CW and CCW relays off
M6 : Tool change
M7 : Mist coolant on
M8 : Flood coolant on
M9 : Mist and flood coolants off
M10 : Fast synchronous output (laser) on (Parameters Q) Note: Q parameter range is 0-255, it controls the laser power, intensity.
M11 : Fast synchronous output(laser) off
M10.1 to M10.10 : Fast synchronous digital outputs (10 number) on
M11.1 to M11.10: Fast synchronous digital outputs (10 number) off
M30 : Program end and program rewind
M31 : Z-axis straight probe macro
M40 : Start digitizing
M41 : Stop digitising
M47 : Program rewind and continue running
M48 : Enable the FRO and SRO controls.
M49 : Disable the FRO and SRO controls.
M50 : Enable / Disable the FRO controls. (Parameter P) Note: P0 disables the control.
M51 : Enable / Disable the SRO controls. (Parameter P) Note: P0 disables the control.
M66 : Wait on input. (Parameters P, E, L, Q)
M98 : Subroutine call (Parameters P, L) Note: P is the subroutine number, L is the times of call.
M99 : Return from subroutine
M106 : Turns the spindle PWM on (Parameters P) The range of P is 0 to 255. This macro is used to control the FAN speed in 3D printing
M107 : Turns the spindle PWM off
M200 : Park Position #1 (onscreen button #193)
M201 : Park Position #2 (onscreen button #194)
M202 : Park Position #3 (onscreen button #195)
M203 : Z-touch with retract for plasma zero height measurement (setting)
M204 : Go to Zero with Safe Z (onscreen button #131)
M205 : Turns THC on in synchronous with the motion.
M206 : Turns THC off in synchronous with the motion.
M207 : Turns the THC delay on in synchronous with the motion.
M208 : Turns the THC delay off in synchronous with the motion.
M209 : Turns the THC anti dive on in synchronous with the motion.
M210 : Turns the THC anti dive off in synchronous with the motion.
M211 : Turns the THC anti down on in synchronous with the motion.
M212 : Turns the THC anti down off in synchronous with the motion.
M213 : Turns the safe probe mode on.
M214 : Turns the safe probe mode off.
M215 : Changes the spindle pulley. (Parameters P) Note: P parameter is the number of the spindle pulley in use.
M216 : Go to Safe Z (onscreen button #216)

M20000 to M21999 : User macros called with on-screen button codes
M99998 : Constructor macro, called once on software startup.
M99999 : Destructor macro, called once on software shutdown.

OTHER SUPPORTED CODES

F : Feedrate value (Parameter in Unit/min)
S : Spindle speed (Parameter in rotational speed of 1/min)
T : Load tool (Parameter is the tool number, can be 1-20, example: T2)
O : Subroutine label (Parameter is the number of the subroutine, example: O11)
# : Reference an internal variable instead of a constant as parameter.
(Example: G1 X#1 Y#2)
? : Show the value of an internal variable.
(Example: ?#1, Note: This command works in MDI input only.)

ALPHABETICAL MACRO & SCREENSET FUNCTION INDEX

MACRO FUNCTIONS

This section describes the **UCCNC software** macro function calls capability.

The UCCNC software can have different profiles. Each profile can have different machine setup/settings and therefore different macros.

The macro files are located in the installation folder of the UCCNC software /Profile/Macro_name of profile where the "name of profile" is the profile name the software is loaded with.

The macros are plain text files with a "***.txt***" extension and have file names that start with an "***M***" followed by the number of the macro. Eg. ***M3.txt, M4.txt, etc.***

The user can make and edit new macros simply by creating a new macro file and adding it to the profile macro folder. Macro files, as they are plain text files are editable with the built in notepad.exe in Windows.

Tip: Notepad++ which is a free software programme that can edit text type files has many advantages over the default Windows notepad.exe programme. A UCCNC style editor has been created and also has built in UCCNC + C# language file which provides function syntax tips and auto-complete assistance. Notepad++ can be downloaded from here:
The UCCNC+C# Notepad++ style editor, language files and installation help manual can be downloaded from here:

Macros are programmed C# (C-sharp) programming language. The language is not described in this documentation, but it is very similar to C language and for those who are yet not familiar with C# programming it is recommended that you study the following Wiki page:
http://en.wikipedia.org/wiki/C_Sharp_syntax

There is an option to change any macros or macroloops from C# language to VB (VisualBasic) language compilation with writting the #VB directive into the very first line of the macro.

If the first line of the macro code contains the #VB directive then the software uses the VisualBasic compiler to compile the macro instead of the C# compiler.

When the Visual Badic language is selected in a macro then the whole language have to be written in VB language following the VB syntax.

The UCCNC specific functions are the same for the C# and the VB compilation, but this manual describes all functions in C# syntax only.

The macros are compiled and executed in runtime, so they can be changed at any time using your chosen text editor. In case the macro contains a syntax error and cannot be compiled, UCCNC software will show a script error notice in the status box. Also it is possible to create a runtime error, for example with dividing by zero in the macro code, in this case, UCCNC software will also show an error notice in the status box.

Tip: Inside the profile folder {e.g. C:\UCCNC\Profiles\Macro_Default } is a file called "Errorlog.txt", this file will log all reasons why a macro did not execute. It is a helpful file to have open at the same time as editing / creating and testing macros and I usually keep it open in Notepad++ so following executing / testing a macro, as soon as I switch back to Notepad++, it will popup and inform me a file has changed, do I want to review it, which normally means my macro has an error. It is worth noting that the line numbers are offset within the Errorlog.txt file by about 10-12 lines when an error is reported within the macro, hence in notepad++ as the line numbers are given I normally look around 10~12 lines LESS than the line number reported by the Errorlog file to find the error and recorrect it.

It is recommended that when writing, editing or modifying macros that they are first tested with the machine in Offline mode, and once the code compiles and executes successfully that they are then tested with the machine "air cutting" (either without a tool) or with a suitable Z-axis offset for safety reasons (limit switches are also recommended as are readily accessible Emergency Stop latching mushroom buttons).

All macros are compiled into the Macro class. The Macro class has visibility to the following Namespaces and objects:

- "**exec**" is the executer, this is the object in the UCCNC software which makes all motion execution, I/O manipulations etc.
- "**AS3**" is the main screen, this is the object on the screen where all fields, buttons, LEDs are placed, the value of these can be read from the AS3 object.
- "**AS3jog**" is the jog panel object on the screen, all fields, buttons, LEDs values on the jogpanel can be read from the AS3jog object.
- **System**, **System.Windows.Forms**, **System.Drawing**, **System.Threading** namespaces.

The macro text typed into the macro file is inside a function of a class and therefore defining other functions and global variables directly inside the macro is not possible.

Defining global variables and functions is possible only at the end of the macro text file with writing the #Events text into the macro, this text will let UCCNC know that the remaining text of the macro has to be compiled outside of the function, but still inside the macro class.

The following example shows a simple macro which creates a Windows Form and adding a button to it and assigning an event handler to the button's click event. The example also declares a function which is then called from inside the macro.

```
Button MyButton = new Button();
MyButton.Size = new System.Drawing.Size(80, 40);
MyButton.Location = new System.Drawing.Point(110, 130);
MyButton.Text = "Press me";
MyButton.Click += new EventHandler(MyButton_Click);

MyForm = new Form();
MyForm.Size = new System.Drawing.Size(300, 300);
MyForm.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
MyForm.Controls.Add(MyButton);
MyForm.ShowDialog();

MyFunction();

#Events

Form MyForm; //This is a global variable, a Windows Form

void MyButton_Click(object sender, EventArgs e)
{
        MessageBox.Show("Mybutton was clicked!");
        MyForm.Close();
}

void MyFunction()
{
        exec.Code("G0 X10");
}
```

The following list contains the Executer (***exec.***___) object's functions which are callable from any macros.

IsMoving()

Function: bool IsMoving(void)

Description: This function returns *true* if the software is executing motion OR a function. It returns *false* if the machine AND software are idle.

Example: ***while(exec.IsMoving()){}*** *// do nothing while machine is moving*

IsMovingTHC()

Function: bool IsMovingTHC(void)

Description: The function is the same as the IsMoving except that this function checks the Idle bit only and does not check the motion buffer count. This function can be used for example to detect when the motion stopped when the THC control is enabled and because the controller is waiting for the ArcOK signal to go on. In this situation the software may still fill the motion buffer if there are more motion commands in the g-code buffer, but the idle bit will be inactive and because this function depends on the idle bit only, it will work in this scenario.

Example: ***while(exec.IsMovingTHC()){}***

Get[XYZABC]machpos()

Function: double Get[XYZABC]machpos(void)

Description: This function returns the actual machine position of AN axis, this is the absolute (machine) position excluding any offsets.

Example: ***double Xmachposvariable = exec.GetXmachpos();***

Notes: [XYZABC] denotes the axis being called. Only axis per call.

Get[XYZABC]pos()

Function: double Get[XYZABC]pos(void)

Description: This function returns the actual position of AN axis including the selected offset (G54 – G59) and the corresponding (G54-G59) tool offset.

Example: ***double Xposvariable = exec.GetXpos();***

Notes: [XYZABC] denotes the axis being called. Only one axis per call.

Get[XYZABC]scale()

Function: double Get[XYZABC]scale(void)

Description: This function returns the actual G51 scale value of an axis.

Example: ***double Xscalevariable = exec.GetXscale();***

Notes: [XYZABC] denotes the axis being called. Only one axis per call.

StopWithDeccel()
Function: void StopWithDeccel(void)
Description: This function stops ALL axes using the set deceleration profile.
Example: ***exec.StopWithDeccel();***

Stop()
Function: void Stop(void)
Description: This function causes an instant stop on ALL axes.
Example: ***exec.Stop();***

Wait()
Function: void Wait(int milliseconds)
Description: This function causes the loop to stop & wait for the set amount of time in milliseconds before continuing with the next line.
Example: ***exec.Wait(1000);*** *// wait 1000 mSec before continuing code*

Setofflinemode()
Function: void Setofflinemode(bool setoffline)
Description: This function sets the machine to offline mode or to online mode. In offline mode the UCxxx outputs are ignored / inhibited, so the machine can't move, however the UCCNC software (graphical front end) will continue to execute machine code and appear as an operating machine.
Example: ***exec.Setofflinemode(true);*** *// machine is set Offline*
exec.Setofflinemode(false); *// machine is set Online*

Getcurrenttool()
Function: int Getcurrenttool(void)
Description: This function returns the current actual selected tool number. It returns zero (0) if no tool is presently selected.
Example: ***int Currenttoolvariable = exec.Getcurrenttool();***

Setcurrenttool()
Function: void Setcurrenttool(int toolnumber)
Description: This function sets the selected tool number in the UCCNC software. Useful when using automatic tool changer. It is recommended that the new tool number is set at the END of any toolchange macro.
Example: ***exec.Setcurrenttool(2);***

Getnewtool()

Function: int Getnewtool(void)
Description: This function reads the tool number next to the M6 code, for example if a code M6 T2 was executed then this function returns the number 2.
Example: ***int newtoolnumber = exec.Getnewtool();***

Ismacrostopped()

Function: bool Ismacrostopped(void)
Description: This function checks if a stop was pressed by the user on the UCCNC software GUI.
Example: ***if(exec.Ismacrostopped){return;}***

Setoutpin()

Function: void Setoutpin(int portnumber, int pinnumber)
Description: This function sets the selected output pin to a HIGH level. Note: If a pin is called which has been pre-configured for a hardware function, such as a step or direction pin, the *Setoutpin* function will be ignored, and the pre-configured hardware function will take precedence.
Example: ***exec.Setoutpin(1, 2);***

Clroutpin()

Function: void Clroutpin(int portnumber, int pinnumber)
Description: This function sets the selected output pin to low level. Note: If a pin is called which has been pre-configured for a hardware function, such as a step or direction pin, the *Clroutin* function will be ignored, and the pre-configured hardware function will take precedence.
Example: ***exec.Clroutpin(1, 2);***

Code()

Function: void Code(string code)
Description: This function is the most complex of all, it makes it possible to execute G-code from inside a macro. The G-code is sent as a string in the parameter of the function and is interpreted at execution time.
Example: ***exec.Code("G0 X10 Y20 Z0");***
Note: It is also possible to join string values within the *exec.Code()* function by joining a previously defined string such as:
Example: ***string Offset = 1.4;*** *// define a string variable and assigns it a value of 1.4*
exec.Code("G0 Z" + Offset); *// displays G0 Z1.4*

Codelist()

Function: void Codelist(List<string> codelist)

Description: This function is similar to the Code function, but it executes not only a single, but multiply lines of g-codes from inside a macro. The g-code lines are sent as a string List in the parameter of the function and are interpreted in execution time. This function differs from calling multiply Code functions, because the Codelist function loads all the code lines the same time into the motion control API, so the lines are optimised by the Constant velocity interpolator while the multiply Code function calls are executed one-by-one separately.

Example:

List<string> codelist = new List<string>(); //Create a new List of strings.
codelist.Add("G0 Z-25"); //Add g-code lines to the List.
codelist.Add("M3");
codelist.Add("G1 X0 Y0 F500");
codelist.Add("#5 = 12.23");
codelist.Add("G1 X#5 Y2");
exec.Codelist(codelist); //Execute the List of g-codes.

Callbutton()

Function: void Callbutton(int buttonnumber)

Description: This function calls an internal function of the UCCNC software. The buttonnumber represents an internal function number of the UCCNC software. The buttonnumber list is provided elsewhere within this document.

Example: ***exec.Callbutton(100);***

Readkey()

Function: string Readkey(string section, string key, string defaultvalue)

Description: This function READS a key value from the profile (.pro) file. The section and the key parameters define which key to read from the profile file and the function returns with the default value parameter if the key does not exist in the profile file. The function returns the value as a string datatype.

Example: ***string mykeyvalue = exec.Readkey("axessettingscontrolX", "Axisenabled", "False");***

Writekey()

Function: void Writekey(string section, string key, string value)
Description: This function WRITES a key value into the profile (.pro) file. The section and the key parameters define which key to write to the profile file. The value parameter is the new value to write. If the key already exists in the profile file then the function overwrites the key with the new value. If the key does not exist then the function creates the key in the file with the set value.
Example: ***exec.Writekey("axessettingscontrolX", "Axisenabled", "False");***

Pluginshowup()

Function: void Pluginshowup(string Pluginfilename)
Description: This function calls a Plugin file's Showup_event(); function. This is useful when the plugin should do something on a button press event, for example when it should show its Graphical User Interface (GUI).
Example: ***exec.Pluginshowup("Plugintest.dll");***

Swapaxis()

Function: void Swapaxis(int axis1, int axis2)
Description: This function swaps the step and direction pin numbers and pin negate settings of one axis with another axis. Axis swapping may be done at anytime, the software saves the swapping sequences.
The axis number parameter can be in the range of 0 to 5 corresponding to the X to C axes in numerical order.
Be careful with saving the axis settings when the swapaxis function is in use. If the settings are saved without pressing the Apply settings button first then the swapped pins will be saved for the axis!
Example: ***exec.Swapaxis(0,1);*** *//Swaps the X-axis with Y-axis.*

Resetswapaxis()

Function: void Resetswapaxis(void)
Description: This function resets the swap axis sequence by rolling back all the previously called axis swapping sequences.
Example: ***exec.Resetswapaxis();***

Slaveaxis()

Function: void Slaveaxis(int masteraxis, int slaveaxis)
Description: This function slaves an axis to an axis. The master axis can be either X, Y or Z axes (parameter numbers 0, 1, 2 respectively) and the slave axis can be A, B and C axes (parameter numbers 3, 4, 5 respectively). To remove the slave from the master axis use value zero (0) on the slaveaxis parameter.
Be careful with saving the axis settings when the slaveaxis function is in use. If the settings are saved [Callbutton(167) / screen button press] without applying the Apply Settings Callbutton(168) function [or screen button press] first, then the slave parameters will be saved for the master axis!
Example: ***exec.Slaveaxis(0,3);*** *//Makes A-axis slave to the X-axis.*
Example: ***exec.Slaveaxis(0,0);*** *//Clears slave to the X-axis.*

Getanaloginput()

Function: int Getanaloginput(int channel)
Description: This function returns the actual value of an analog input channel signal. The parameter defines the channel to be read. If the device has no analog input channel with the channel number called, then the return value will be -1. The valid return range for the function is 0-65535.
Example: ***int analogin1 = exec.Getanaloginput(1);***

Getanalogoutput()

Function: int Getanalogoutput(int channel)
Description: This function returns the actual value of an analog output channel signal. The parameter defines the channel to be read. If the device has no analog output channel with the channel number called, then the return value will be -1. The valid return range for the function is 0-65535.
Example: ***int analogout1 = exec.Getanalogoutput(1);***

Getvar()

Function: double Getvar(int varnum)
Description: This function returns the value of an internal variable.
Example: ***double myvar = exec.Getvar(1);***

Setvar()

Function: void Setvar(double value, int varnum)
Description: This function sets the value of an internal variable.
Example: ***exec.Setvar(1.234, 1);***

TextQuestion()

Function: string TextQuestion(string Questiontext)
Description: This function shows a Question form waiting for a string as the answer. The text of the question shown on the Form is the parameter.
Example: ***string val = exec.TextQuestion("Stop code execution?");***

Question()

Function: double Question(string Questiontext)
Description: This function shows a Question form waiting for a double value as the answer. The text of the question shown on the Form is the parameter.
Example: ***double val = exec.Question("What X position to move?");***

Loadfile()

Function: void Loadfile(string filename)
Description: This function loads a g-code file. The parameter is a string which is the full path of the file to be loaded.
Example: ***exec.Loadfile("C:/UCCNC/Example_codes/holders.tap");***

IsLoading()

Function: bool IsLoading(void)
Description: The function returns true if the software is loading a g-code file and returns false if no file is being loaded.
Example: ***while(exec.IsLoading()){}***

Showplugin()

Function: int Showplugin(string pluginfilename)
Description: This function calls the Showup event of a UCCNC plugin installed in the /Plugins directory. The parameter is a string which is the name of the plugin file including the .dll extension.
The possible return values are the following:
0: The plugin started without problems.
1: The plugin is not enabled and can't run.
2: The plugin was not found, there is no plugin installed with this filename.
3.: The plugin does not have the Showup event implemented.
Example: ***int returnval = exec.Showplugin("Diagnostics.dll");***

Configplugin()

Function: int Configplugin(string pluginfilename)
Description: This function calls the Config event of a UCCNC plugin installed in the /Plugins directory. The parameter is a string which is the name of the plugin file including the .dll extension.
The possible return values are the following:
0: The plugin started without problems.
1: The plugin is not enabled and can't run.
2: The plugin was not found, there is no plugin installed with this filename.
3.: The plugin does not have the Config event implemented.
Example: ***int returnval = exec.Configplugin("Diagnostics.dll");***

Informplugin()

Function: object Informplugin(string Pluginfilename, object Message)
Description: This function sends data to one plugin. The Pluginfilename parameter defines the name of the plugin to send the message to and the parameter is an object and the return value is also an object.
Example:

```
string teststr = "Hello Plugintest.dll...";
object Returnvalue = exec.Informplugin("Plugintest.dll",
(object)teststr);
if (Returnvalue is string)
{
  string str = Returnvalue as string;
  MessageBox.Show(exec.mainform, "Return message was: " + str);
}
```

Informplugins()

Function: void Informplugins(object Message)
Description: This function sends data to all plugins. The parameter is an object and there is no return value.
Example:

```
string teststr = "Hello to all plugins...";
exec.Informplugins((object)teststr);
```

Ismacrorunning()

Function: int Ismacrorunning(int macronumber)
Description: The function returns the number of instances a text-macro is currently running. The function can be used to check if a macro is already running or not. If for example the macro is only allowed to run in one instance then the macro should contain a code to return without the execution of the code in another instance. The function is useful for example when calling text macros via button codes (M20000 to M21999) and when it is unwanted to run the macro in more instances if the user clicks the screen button more than one time while the macro is still running. The valid range for the macronumber parameter is 0 to 99999. The function returns -1 if the parameter is out of range. Note, that the macroloop runs are excluded from this function, those macro instances will not count in the returned value of the function.
Example:
if(exec.Ismacrorunning(20000)>1){ return; } *//The return value 1 indicates that only this one macro instance is running. If the number is greater means that this is a second or third etc. instance of the macro to be run.*
//Do macro functions here...

RemoveRunfromhere()

Function: void RemoveRunfromhere()
Description: This function removes the Initial movement Window of the Run from here function to be shown condition after the Run from here button was pressed or the button code was previously called, so the Initial movement Window will not be shown and inital movement will not be made. The function can be used when a plugin or macro changes the g-code line number to be executed and with calling the Run from here button code and if the plugin or macro don't need the initial movement preparation.
Example: ***exec.RemoveRunfromhere();***

GetRotate()

Function: void GetRotate(out double Rx, out double Ry, out double Angle)
Description: This function returns the current G68 rotation point and rotation angle.
Example: ***double Rx, Ry, Angle;***
exec.GetRotate(out Rx, out Ry, out Angle); *// returns values into the defined variables Rx, Ry and Angle*

Getgcodelinetext()

Function: string Getgcodelinetext(int rownumber)
Description: This function returns one row of text from the loaded g-code file pointed by the rownumber parameter.
Example: ***string linetext = exec.Getgcodelinetext(0);*** *//Reads the first row of the g-code file.*

Getcurrgcodelinetext()

Function: string Getcurrgcodelinetext(void)
Description: This function returns one row of text from the loaded g-code file pointed by the current line DRO pointer.
Example: ***string linetext = exec.Getcurrgcodelinetext();***

Getcurrentgcodelinenumber()

Function: int Getcurrentgcodelinenumber(void)
Description: This function returns the line number of the g-code file pointed by the current line DRO pointer.
Example: ***int linenumber = exec.Getcurrentgcodelinenumber();***

Ismacrorunning()

Function: int Ismacrorunning(int macronumber)
Description: The function returns the number of instances a text-macro is currently running. The function can be used to check if a macro is already running or not. If for example the macro is only allowed to run in one instance then the macro should contain a code to return without the execution of the code in another instance. The function is useful for example when calling text macros via button codes (M20000 to M21999) and when it is unwanted to run the macro in more instances if the user clicks the screen button more than one time while the macro is still running. The valid range for the macronumber parameter is 0 to 99999. The function returns -1 if the parameter is out of range. Note, that the macroloop runs are excluded from this function, those macro instances will not count in the returned value of the function.
Example: ***if(exec.Ismacrorunning(20000)>1){ return; }*** *//The return value 1 indicates that only this one macro instance is running. If the number is greater means that this is a second or third etc. instance of the macro to be run.*

```
        //Do macro functions here...
```

Plugininterface.Datatypes.Tooltablestruct[]

Function: Plugininterface.Datatypes.Tooltablestruct[] Gettooltabledata(void)
Description: The function returns an array of structures with all tools' tool table parameter values in the detailed tooltable.
Example: ***Plugininterface.Datatypes.Tooltablestruct[] Tdata = exec.Gettooltabledata();***
MessageBox.Show(Tdata[1].Description); *//Shows the Description parameter value of Tool#1*

List<Plugininterface.Datatypes.Layerdatastruct>

Function: List<Plugininterface.Datatypes.Layerdatastruct> Getlayerslist(bool isAS3)
Description: The function returns a List of structures with all screen layers' parameters. If the AS3 parameter is true then the main screen's layers parameters are return otherwise the jog screen layers parameters are returned.
Example: ***List<Plugininterface.Datatypes.Layerdatastruct> Ldata = exec.Getlayerslist(true);***
MessageBox.Show("" + Ldata[1].Isactive); *//Shows the Isactive property of the first layer in the List.*

DospinCW()

Function: void DospinCW()
Description: Function equivent to M-code M3 (clockwise spindle run) via code function [*exec.Code("M3");]*
Example: ***exec.DospinCW();***

DospinCCW()

Function: void DospinCCW()
Description: Function equivent to M-code M4 (counter clockwise spindle *run) via code function [exec.Code("M4");]*
Example: ***exec.DospinCCW();***

Stopspin()

Function: void Stopspin()
Description: Function equivent to M-code M5 (spindle stop) via code function [*exec.Code("M5");]*
Example: ***exec.Stopspin();***

Miston()
Function: void Miston()
Description: Function equivent to M-code M7 (Coolant Mist on) via code function [*exec.Code("M7");]*
Example: ***exec.Miston();***

Floodon()
Function: void Floodon()
Description: Function equivent to M-code M8 (Coolant Flood on) via code function [*exec.Code("M8");]*
Example: ***exec.Floodon();***

Stopcoolant()
Function: void Stopcoolant()
Description: Function equivent to M-code M9 (Stop Coolant (M7 & M8 off)) via code function [*exec.Code("M9");]*
Example: ***exec.Stopcoolant();***

ChangeaxisDROvalue()
Function: void ChangeaxisDROvalue(int axisnumber, string value)
Description: Changes the DRO value of an axis, X = 0, Y=2, etc, equivalent to G92, but only changes 1 axis per function call.
Example: ***exec.ChangeaxisDROvalue(0, "100");***

AddStatusmessage()
Function: void AddStatusmessage(string StatusMessage)
Description: Adds a message to the Status Listbox
Example: ***exec.AddStatusmessage("Check Ohmic Probe");***

Getgcodefilename()
Function: string Getgcodefilename(void)
Description: This function returns the current loaded g-code filename as a string.
Example: ***string val = exec.Getgcodefilename();***

Getprofilename()
Function: string Getgprofilename(void)
Description: This function returns the current loaded profile filename as a string.
Example: ***string val = exec.Getprofilename();***

Setaxishomestate()

Function: void Setaxishomestate(int axis, bool ishomed)
Description: This function sets / unsets a specific axis as "homed".
Parameters:

- ***axis*** is the number of the axis, 0=X, 1=Y, 2=Z....
- ***ishomed*** defines the checkbox state to be set (true / false)

Example: ***exec.Setaxishomestate(2, true);***

MessageBox.Show()

Function: MessageBox.Show (string PopupText)
Description: This is not a uccnc function, but a built in C# function which can be used in Macros [more information is available on the web regarding c# MessageBox.Show() options. It is added to provide an ease of reference to a useful function within UCCNC which creates a popup Message Box. There are two variants to the MessageBox that may be useful in UCCNC. The "simple" acknowledgement message box popup which you call to halt the macro, until the "OK" is pressed, and the second which has "Yes" and "No" option buttons, which will allow the macro to skip / bypass sections of code.
Example:
Simple popup:

MessageBox.Show("The machine was not yet homed, home the machine before run to parking position!");

Yes / No Option popup:

The following list contains the AS3 and AS3jog object's functions which are callable from any macros. Each of the functions must either be prefixed with either AS3 or AS3jog which will either permit access to the main screens or the jog screens respectively.

Getfield()

Function: string Getfield(int fieldnumber)
Description: This function reads the value of a field object and returns the value in as a string datatype.
Example: ***string fieldvalue = AS3.Getfield(100);***

Setfield()

Function: void Setfield(double value, int fieldnumber)
Description: This function SETS the DOUBLE value of a field object. If the field exists on the screen OR on any tab pages then it will update it's value. If the field object does not exist then this function will do nothing.
Note that there are fields like the position, federate and spindle speed DROs etc which are updated in UCCNC from internal variables in loops, these fields cannot be updated permanently using this method, because the internal functions will rewrite these fields with the internal variable values. Call the Validatefield function after the Setfield function to change the value of the field.
Example: ***AS3.Setfield(15.23, 100);***

Validatefield()

Function: void Validatefield(int fieldnumber)
Description: This function VALIDATES the value of a field object. If the field exists on the screen OR on any tab pages then the function executes. If it does not exist, then this function will do nothing.
When this function is called then the field value changed event is called inside the UCCNC core and the software will make the actions necessary to validate the actual value of the field.
Example: ***AS3.Setfield(12.34, 97);*** *//Sets the value of the X current coordinate field to 12.34*
AS3.Validatefield(97); *//Validates the 12.34 value for the X current coordinate field with writing the offset coordinates as necessary.*

Setfieldtext()

Function: void Setfieldtext(string value, int fieldnumber)
Description: This function SETS the STRING value of a field. If the field exists on the screen OR on any tab pages, then it will update its value. If it does not exist, then this function will do nothing.
Note that there are fields like the position, federate and spindle speed DROs etc which are updated in UCCNC from internal variables in loops, these fields cannot be updated permanently using this method, because the internal functions will rewrite these fields with the internal variable values. Call the *Validatefield* function after the *Setfield* function to change the value of the field.
Example: ***AS3.Setfieldtext("This is my field", 100);***
AS3.Validatefield(100); *//Validates fieldtext 100.*

GetLED()

Function: bool GetLED(int LEDnumber)
Description: This function returns the logic state of a LED screen object. If the LED is ON, then the function returns *true* or, if the LED is OFF, then this function returns *false*. If the LED does not exist on the screen, then the return value returned will be *false*.
Example: ***bool stateofmyLED = AS3.GetLED(18);***

SetLED()

Function: void SetLED(bool state, int LEDnumber)
Description: This function sets the logic state of a LED screen object. If the LED does not exist on the screen then this function will do nothing.
Note: There are LEDs whose values are refreshed from internal loops of UCCNC, these LEDs cannot be set with this function, because the LED's value will update from the UCCNC's internal loop. For example the idle and run LEDs cannot be set with this function. Basically this function is to set user LEDs which are not used by the UCCNC core.
Example: ***AS3.SetLED(true, 1000);***

Getbutton()

Function: bool Getbutton(int buttonnumber)
Description: This function returns *true* if the button is being pressed on the screen OR via an input trigger and returns *false* if the button is released.
Example: ***while(!AS3.Getbutton(128));*** *//Waits for the cycle start button press on the main screen.*

Getbuttonstate()

Function: bool Getbuttonstate(int buttonumber)
Description: This function works with toggle type buttons and returns true if the button is in it's on state and returns false when the button is in it's off state. The function always returns false for non-toggle type buttons.
Example: ***bool buttonstate = AS3.Getbuttonstate(114);*** *//Checks if the M3 spindle on/off button is active.*

Switchbutton()

Function: void Switchbutton(bool Ison, int Buttonnumber)
Description: This function works with toggle type buttons and switches the button between ON and OFF states. If the *Ison* parameter is *true*, the function switches the button to the ON state, and *false* switches the button to the OFF state. The function can be used for example in buttons' macros to change the state of the button by toggling its visual state.
Example: ***AS3.Switchbutton(true, 128)***

Getcomboboxselection()

Function: string Getcomboboxselection(int labelnumber)
Description: This function returns the selected text string of a combobox screen item.
Example: ***string selectedtext = AS3.Getcomboboxselection(1);***

Getlist()

Function: List< string > Getlist(int labelnumber)
Description: This function returns all items of a screen list. The return type is a list of strings.
Example: ***List<string> mylist = AS3.Getlist(2);***

GetMDIhistory()

Function: List< string > GetMDIhistory()
Description: This function returns all items previously typed into the MDI control.
Example: ***List<string> mylist = AS3.GetMDIhistory();***

ClearMDIhistory()

Function: void ClearMDIhistory()
Description: This function clears the MDI history list.
Example: ***AS3.ClearMDIhistory();***

Sendimageview()

Function: void Sendimageview(Image img, int Labelnumber)
Description: This functions sends a picture image to an imageviewer.
Example: //This example code sends and rotates a picture image around it's center and displays it in an imageview control

```
Image img1 = Image.FromFile(Application.StartupPath + "/a.png");
for (int i = 0; i < 360; i++)
{
 Image imgdraw = (Image)img1.Clone();
 using (Graphics g = Graphics.FromImage(imgdraw))
 {
  // Set the rotation point to the center in the matrix
  g.TranslateTransform(imgdraw.Width / 2, imgdraw.Height / 2);
  // Rotate
  g.RotateTransform(i);
  // Restore rotation point in the matrix
  g.TranslateTransform(-imgdraw.Width / 2, -imgdraw.Height / 2);
  // Draw the image on the bitmap
  g.DrawImage(imgdraw, new Point(0, 0));
}

AS3.Sendimageview(imgdraw, 1);
imgdraw.Dispose();
Thread.Sleep(30);
if(exec.Ismacrostopped())
{
 return;
}
}
```

Additemtolist()

Function: void Additemtolist(string value, int labelnumber)
Description: Adds a string value to a Listbox with the corresponding labelnumber.
Example: ***AS3.Additemtolist("Check Ohmic Probe" , 18);***

Setcheckboxstate()

Function: void Setcheckboxstate(bool state , int boxnumber)
Description: This function sets the state of a checkbox
Parameters:

- ***state*** defines the checkbox state to be set (true / false)

- ***boxnumber*** is the unique identifier of the checkbox control. The checkboxes numbers and their meaning are listed and described elsewhere in this document.

Example: ***AS3.Setcheckboxstate(true,10);***

Getcheckboxstate()

Function: bool Getcheckboxstate(int boxnumber)
Description: This function returns the state of a checkbox (true / false)
Parameters:

- ***boxnumber*** is the unique identifier of the checkbox control. The checkboxes numbers and their meaning are listed and described elsewhere in this document.

Example: ***AS3.Getcheckboxstate(10);***

Addcomboboxitem()

Function: void Addcomboboxitem(string value, int comboboxnumber)
Description: This function adds a value to a combo box.
Parameters:

- ***value*** defines the string to be added to the combobox.
- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Addcomboboxitem("Spiral drill", 11);***

Validatenewcomboboxitems()

Function: void Validatenewcomboboxitems(int comboboxnumber)
Description: This function validates the items added to a combo box, it is called after all *Addcomboboxitem* functions have been called to add items to the combobox with the same unique comboboxnumber.
Parameter:

- ***comboboxnumber*** is the unique identifier of the combobox object.

Example: ***AS3.Validatenewcomboboxitems(11);***

Getcomboboxselection()

Function: string Getcomboboxselection(int comboboxnumber)
Description: This function returns the value selected within a combobox.
Parameter:

- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Getcomboboxselection (11);***

Clearcomboboxitems()

Function: void Clearcomboboxitems(int comboboxnumber)
Description: This function clears all items from a combo box.
Parameter:

- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Clearcomboboxitems(11);***

Updatecomboboxselection()

Function: void Updatecomboboxselection(int index, int comboboxnumber)
Description: This function updates the combo box and sets focus on the index item (starting at index zero (0)).
Parameter:

- ***index*** is the item number in the list to be selected at screen load.
- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Updatecomboboxselection(0,11);***

Setslider()

Function: void Setslider(int value, int fieldnumber)
Description: This function sets the logic state of a slider screen object. If the slider does not exist on the screen then this function will do nothing.
Example: ***AS3.SetSlider(89, 150);***

Getcolorpickercolor()

Function: int Getcolorpickercolor(int LEDnumber)
Description: This function returns the colorpicker object colour as RGB color code in integer format.
Parameters:

- ***LEDnumber*** is the unique identifier of the object.

Example: ***AS3.Addcolorpick(1);***

Getselectedindexinlist()

Function: int Getselectedindexinlist(int listboxnumber)
Description: This function returns the selected items index position within the the list of items.
Parameters:

- ***listboxnumber*** is the unique identifier of the object.

Example: ***AS3.Getselectedindexinlist(2);***

Setcheckboxstate()

Function: void Setcheckboxstate(bool ison, int checknumber)
Description: This function sets the logic state of a checkbox screen object. If the checkbox does not exist on the screen then this function will do nothing.
Parameters:

- ***LEDnumber*** is the unique identifier of the object.

Example: ***AS3.Setcheckboxstate(true, 27);***

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

MODBUS MACRO FUNCTIONS

GetAllModbusArray()

Function: ushort[] GetAllModbusArray(void)
Description: This function returns the whole Modbus register table, and returns the values within an array which contains unsigned short numbers (integers between 0 and 65535). Arrays in C# are indexed starting with zero (0).
Example: ***ushort [] MbusArray = GetAllModbusArray();***
ushort ArrayPos0 = MbusArray[0]; *// first array position*
ushort ArrayPos1 = MbusArray[1]; *// second array position*

SetModbusregister()

Function: bool SetModbusregister(int Registernumber, ushort Value)
Description: This function writes a single register to the Modbus register table.
Parameters:

- ***Registernumber*** parameter is the position in the Modbus register table to be written to.
- ***Value*** defines the value you want to write (must be an unsigned short value (integer between 0 and 65535))
- ***Return*** The function returns *true* if it executed correctly or *false* if it did not / was corrupt.

Example: ***ushort Writevalue = (ushort)AS3.Getfielddouble(2000);***
if(!exec.SetModbusregister(0, Writevalue))
{
//The write returned with error, handle it...
}

SetModbusregisters()

Function: bool SetModbusregisters(int Startregister, ushort[] Values)
Writes multiply registers in the modbus register table.
Description: This function writes multiple registers to the Modbus register table.
Parameters:

- ***Startregister*** defines the starting modbus register to write to.
- ***Value*** defines the value you want to write (must be an unsigned short value (integer between 0 and 65535))
- ***Return*** The function returns *true* if it executed correctly or *false* if it did not / was corrupt.

GetModbusregister()

Function: bool GetModbusregister(int Registernumber, **out** ushort Value)
Description: This function a single register from the Modbus table.
Parameters:

- ***Registernumber*** parameter is the position in the Modbus register table to be read from
- ***Value*** defines the value you want to read from (returned as an unsigned short value (integer between 0 and 65535))
- ***Return*** The function returns *true* if it executed correctly or *false* if it did not / was corrupt.

Example:

```
ushort Readvalue;
if(exec.GetModbusregister(0, out Readvalue))
{
  AS3.Setfield(Readvalue, 2000); // do something with
  Readvalue
}
else
{
   //The read returned with error, handle it...
}
```

GetModbusregisters()

Function: bool GetModbusregisters(int Startregister, int Registercount, **out** ushort[] Values)
Reads multiply registers from the modbus register table.
Description: This function reads multiple registers from the Modbus register table.
Parameters:

- ***Startregister*** defines the starting modbus register to read from.
- ***Registercount*** defnes the incremental number of registers to read from following the Startregister position.
- ***Values*** are returned as unsigned short values (integers between 0 and 65535), to a unsigned short array "Values".
- ***Returns*** The function returns *true* if it executed correctly or *false* if it did not / was corrupt.

Example:

```
ushort [] Readvalues;
if(exec.GetModbusregister(5, 3, out Readvalues))
{ // start at Modbus register table position 5, & read 3 values
// do something with Readvalues
AS3.Setfield(Readvalues[0], 2000);
AS3.Setfield(Readvalues[1], 2001);
AS3.Setfield(Readvalues[2], 2002);
}
else
{
   //The read returned with error, handle it...
}
```

WriteModbusString()

Function: void WriteModbusString(string String, int Startregister, bool HightoLowByteorder)
Description: This function converts a string into a ushort array and writes it into the Modbus register table and returns nothing.
Parameters:

- ***String*** defines the text string of charactors to transmit.
- ***Startregister*** defines the starting modbus register
- ***HightoLowByreorder*** defines the Big & Little Edian Byte Order.

Example:

```
string SendString = "testing";
WriteModbusString(SendString, 3, true);
```

// starting at Modbus register 3, send the variable SendString to the Modbus register table, converting the sting to a ushort array, using HightoLowByteorder

Example MODBUS macro codes

The following example code reads the modbus register 0 from the UCCNC modbus table and writes the value into the Textfield (DRO) with ID=2000. To test run the macro create a Textfield with ID=2000 using the Screen editor and place the below text into a text macro file in the /Profiles/Macros folder.
Setup a Modbus Connection and Function in the Modbus plugin which function reads value into register 0.

Execute the macro via MDI.
To execute the macro in a loop place the macro file to the macro loops in the General settings/Configure macroloops and place your macro number into one loop slots.
Running the macro in a loop will continuously read the modbus register 0 and will write the read value into the DRO.

The code is the following:

```
ushort Readvalue;
if(exec.GetModbusregister(0, out Readvalue))
{
  AS3.Setfield(Readvalue, 2000);
}
else
{
  //The read returned with error, handle it...
}
```

The following example code reads a DRO and writes the value into the modbus register 0. To test run the macro create a Textfield with ID=2000 using the Screen editor and place the below text into a text macro file in the /Profiles/Macros folder. Setup a Modbus Connection and Function in the Modbus plugin which writes register 0 value onto the Modbus.

Execute the macro via MDI.
To execute the macro in a loop place the macro file to the macro loops in the General settings/Configure macroloops and place your macro number into one loop slots. Running the macro in a loop will continuously read the DRO value and will write it to the modbus.

The code is the following:

```
ushort Writevalue = (ushort)AS3.Getfielddouble(2000);
if(!exec.SetModbusregister(0, Writevalue))
{
 //The write returned with error, handle it...
}
```

SCREENSET FUNCTIONS

This section describes the UCCNC software **screenset** function calls capability.

All of the screen elements in the UCCNC software are defined in the external screenset script file which file is located in the **/Screens** folder of the UCCNC installation.

The screenset files have a ***.ssf** (Screen Set File) extension and are plain text files editable with notepad in Windows.

There may be more than one screenset file in the /Screens directory.
Which screenset file is loaded on the software startup is defined in the profile file,
the key which defines the name of the screenset file to be loaded is in the following section with the following name:

```
[Screensetsettings]
mainscreenfilename=Defaultscreenset
```

The UCCNC software first loads the profile file and reads the name of the screenset file and executes the script in this file.

The screenset file contains the definition of all screen elements. It defines the tab layers, backgrounds, buttons, fields, placements, dimensions, images etc.

Basically, the UCCNC GUI is blank and all screen elements are loaded to the screen from the screenset file.
This makes the screen customisable and freely editable to the user, because all the screen elements can be changed in the external screenset script file.

The screenset file's programming language is C# (C-sharp). The language is not described in this documentation, but it is very similar to C language and for those who are yet not familiar with C# programming it is recommended that you study the following Wiki page: http://en.wikipedia.org/wiki/C_Sharp_syntax

There are 2 different display objects or let's call them canvas' on the screen, the mainform.AS3 and the mainform.AS3jog. The **AS3** is the main screen canvas and the **AS3jog** is the jog panel on the left side of the screen. Screen elements can be uploaded onto these separately.

Tip: When editing screenset objects using the inbuilt screen editor within UCCNC, accessible via the Configuration >> General Settings Tab >> Edit Screen Button. When editing screensets using the inbuilt screen editor, users can change between tabs by holding down "shift" key and clicking on the desired tab with the left mouse button.

This documentation seeks to describe the screenset associated function calls, the prototypes of these functions with some descriptions and examples are shown below.

Addbackground()

Function: void Addbackground(double posX, double posY, double width, double height, int picturenumber, int backgroundnumber, int layernumber)
Description: This function renders an image onto a tab layer, in other words it places a background onto a tab layer.
Parameters:

- ***posX*** and ***posY*** defines the X and Y position, in pixels, of the top-left corner of the image to be placed.
- ***width*** and ***height*** defines how wide and high the image will be in pixels. Note, if the image has different dimensions than the width and height parameters, then the image will be scaled to the placement dimensions.
- ***picturenumber*** defines the numeric identifier of the image used to render the button, this identifier is set in the *Loadpicture* function and any picture which was loaded previously can be selected.
- ***backgroundnumber*** defines the identifier of this background.
- ***layernumber*** defines the number of the tab layer to place the image onto.

Example: ***AS3.Addbackground(0, 0, 1024, 692, 121, 11, 12);***

Addbutton()

Function: void Addbutton(double posX, double posY, double width, double height, bool toggletype, bool blinktype, int picturenumber, int buttonnumber, int layernumber)
Description: This function adds one button object to the screen.
Parameters:

- ***posX*** and ***posY*** defines the X and Y position, in pixels, of the top-left corner of the button control.
- ***width*** and ***height*** defines how wide and high the button will be in pixels.
- ***toggletype*** if set to *true*, defines that the button toggles between ON and OFF states, by rendering the left hand side half of the image onto the button in the OFF state and rendering the right hand side half of the image in the ON state of the button.
- ***blinktype*** if set to *true*, then in the ON state of the button the rendering of the left AND right hand side images are rendered periodically creating a blinking effect.
- ***picturenumber*** defines the numeric identifier of the image used to render the button, this identifier is set in the *Loadpicture* function and any picture which was loaded previously can be selected.
- ***buttonnumber*** defines the identifier for this button and this identifier also defines the function of this button. Eg. Button Number 128. is the cycle start button code. Note, the different button codes are listed and described elsewhere in this document.
- ***layernumber*** defines on which tab layer the button will appear on.

Example: ***AS3.Addbutton(50, 60, 100, 100, false, 18, 128, 2);***

Addcheckbox()

Function: void Addcheckbox(string labeltext, string labelfont, int fontsize, int fontcolor, int posX, int posY, int boxnumber, int layernumber)
Description: This function adds a checkbox control to the screen.
Parameters:

- ***labeltext*** defines the text string appearing before the checkbox control.
- ***labelfont*** defines the font type used for the text string checkbox label
- ***fontsize*** sets the size of the font in pixels.
- ***fontcolor*** is the RGB color code in integer format.
- ***posX*** and ***posY*** defines the X and Y position, in pixels, of the top-left corner of the checkbox control.
- ***boxnumber*** is the unique identifier of the checkbox control. The checkboxes numbers and their meaning are listed and described elsewhere in this document.
- ***layernumber*** defines on which tab layer the checkbox will appear on.

Example: ***AS3.Addcheckbox("", "Arial", 16, 0, 50, 100, 100, 2);***

Addcodeview()

Function: void Addcodeview (string labeltext, string labelfont, string textalign, int fontsize, int fontcolor, int posX, int posY, int Width, int Height, int layernumber)
Description: This function provides the machine code viewing window box.
Parameters:

- ***labeltext*** defines the text string title of the codeview control.
- ***labelfont*** defines the font type used to render the text in the control
- ***textalign*** defines the text inside the control's field.
- ***fontsize*** defines the size of the font in pixels.
- ***fontcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the control.
- ***Width*** and ***Height*** define the width and height of the control.
- ***layernumber*** defines which layer the control will appear on.

Example: ***AS3.Addcodeview("", "Arial", "left", 18, -1, 42, 350, 387, 339, 2);***

Addcolorpick()

Function: void Addcolorpick(double posX, double posY, double Width, double Height, int LEDnumber, int layernumber)
Description: This function adds a colour pick control to the screenset.
Parameters:

- ***posX*** and ***posY*** defines the top-left corner position of the control.
- ***Width*** and ***Height*** define the width and height of the control.
- ***LEDnumber*** is the unique identifier of the object.
- ***layernumber*** defines which layer the control will appear on.

Example: ***AS3.Addcolorpick(570, 168, 26, 26, 1, 10);***

Addcombobox()

Function: void Addcombobox(int posX, int posY, int Width,
int fontsize, int fontcolor, int Numberofaxis, int comboboxnumber ,
int layernumber)
Description: This function adds a combo box object.
Parameters:

- ***posX*** and ***posY*** define the top-left corner position of the combobox object, and are in pixels.
- ***Width*** defines the dimensions of the combobox object, and is in pixels.
- ***fontcolor*** is the color of the font in integer format of the RGB color code.
- ***fontsize*** defines the size of the font used to render the field text.
- ***Numberofaxis*** defines
- ***comboboxnumber*** is the unique identifier of the combobox object.
- ***layernumber*** is the number of the tab layer the combobox object will appear on.

Example: ***AS3.Addcombobox(532, 126, 130, 16, -16777216, 3, 11, 45);***

Addcomboboxitem()

Function: void Addcomboboxitem(string value, int comboboxnumber)
Description: This function adds a value to a combo box.
Parameters:

- ***value*** defines the string to be added to the combobox.
- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Addcomboboxitem("Spiral drill", 11);***

Addfield()

Function: void Addfield(string labeltext, string labelfont, string textalign, int fontsize, int fontcolor, double posX, double posY, int fieldwidth, string type, double min, double max, int labelnumber, int layernumber)
Description: This function adds one field object onto the screen. Each field object represents a text label on the UCCNC screen.
Parameters:

- ***labeltext*** defines a text string which will be written in front of the field on the screen.
- ***labelfont*** defines the font used for both the text string and the field text.
- ***textalign*** defines where the text is aligned inside the field, the possible values are "left", "right" and "center".
- ***fontsize*** defines the size of the font used to render the field text.
- ***fontcolor*** is the color of the font in integer format of the RGB color code.
- ***posX*** and ***posY*** define the top-left corner of the field
- ***fieldwidth*** define the length of the field in pixels.
- ***type*** is a special parameter, the possible values are the: "textfield", "textfieldnb", "showfield", "showfieldnb", "field", "fieldnb".
    - The "textfield", "textfieldnb", "field", "fieldnb" type fields can be read and written from macro code and are editable by the user on the UCCNC GUI.
    - The "textfield", "textfieldnb" can contain any kind of text including words.
    - The "field", "fieldnb" can contain numerical values only.
    - The "showfield", "showfieldnb" type labels can be read and written from macro code and are not editable by the user on the UCCNC GUI.
        - The "nb" at the end of the parameter means "no border", so these types of fields will have no visual border on the screen.
- The ***min*** and ***max*** define the minimum and maximum numerical range values for the "field", "fieldnb" type fields. Users cannot enter numbers outside of this range.
- The ***labelnumber*** is the identifier number of the field, the different label numbers and their meaning are listed elsewhere in this document.
- The ***layernumber*** is the number of the tab layer the field will appear on.

Example: ***AS3.Addfield(" ", "Arial", "right", 21, 7961465, 50, 100, 128, "showfieldnb", double.MinValue, double.MaxValue, 866, 2);***

Addfill()

Function: void Addfill(int fillcolor, int posX, int posY, int Width, int Height, double transparency, int Fillnumber, int layernumber)
Description: This function adds a colour fill to the screenset.
Parameters:

- ***fillcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the control.
- ***Width*** and ***Height*** define the width and height of the control.
- ***Transparency*** is the degree of transparency 0 to 1.
- ***Fillnumber*** is the unique identifier of the object.
- ***layernumber*** defines which layer the control will appear on.

Example: ***AS3.Addfill(-8355712, 30, 52, 878, 700, 0.35, 1, 4);***

Additemtolist()

Function: void Additemtolist(string value, int listboxnumber)
Description: This function adds a list box to the screenset.
Parameters:

- ***value*** defines the string to be added to the listbox.
- ***listboxnumber*** is the unique identifier of the object.

Example: ***AS3.Additemtolist("X-Axis", 2);***

Addlabel()

Function: void Addlabel(string labeltext, string labelfont, string textalign, int fontsize, int fontcolor, int posX, int posY, int layernumber)
Description: This function adds a label to the screenset.
Parameters:

- ***labeltext*** defines the text sting of the label to be displayed.
- ***labelfont*** defines the font type used to render the text in the control
- ***textalign*** defines the text inside the control's field.
- ***fontsize*** defines the size of the font in pixels.
- ***fontcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the control.
- ***layernumber*** defines which layer the control will appear on.

Example: ***AS3.Addlabel(" ", "Arial", "left", 16, -16777216, 214, 602, 13);***

Addled()

Function: void Addled(double posX, double posY, double width, double height, int picturenumber, int LEDnumber, int layernumber)
Description: This function adds an LED object to the screen.
Parameters:

- The ***posX*** and ***posY*** define the top-left corner position of the LED, and are in pixels.
- The ***width*** and ***height*** define the dimensions of the LED and are in pixels.
- The ***picturenumbe***r is the number of the image to render onto the LED.
- The ***LEDnumber*** is the unique identifier of the LED. LED numbers and their meanings are described elsewhere in this document.
- The ***layernumber*** is the number of the tab layer the LED will appear on.

Example: **AS3.Addled(530, 420, 6, 37, 68, 216, 2);**

Addlist()

Function: void Addlist(string labelfont, string textalign, int fontsize, int fontcolor, int posX, int posY, int Width, int Height, int listboxnumber, int layernumber)
Description: This function adds a list box to the screenset.
Parameters:

- ***labelfont*** defines the font type used to render the text in the control
- ***textalign*** defines the text inside the control's field.
- ***fontsize*** defines the size of the font in pixels.
- ***fontcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the control.
- ***Width*** and ***Height*** define the width and height of the control.
- ***listboxnumber*** is the unique identifier of the object.
- ***layernumber*** defines which layer the control will appear on.

Example: ***AS3.Addlist("Arial", "left", 14, -16777216, 457, 520, 292, 136, 2, 2);***

Addmdi()

Function: void Addmdi(string labeltext, string labelfont, string textalign, int fontsize, int fontcolor, int posX, int posY, int width, int MDInumber, int layernumber)

Description: This function adds a Manual Data Input (MDI) control to the screen.

Parameters:

- ***labeltext*** defines the text string title of the MDI control.
- ***labelfont*** defines the font type used to render the text in the control
- ***textalign*** defines the text inside the control's field.
- ***fontsize*** defines the size of the font in pixels.
- ***fontcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the MDI control.
- ***width*** defines the width of the MDI textfield in pixels.
- ***MDInumber*** is the identifier of the MDI object.
- ***layernumber*** defines which layer the MDI will appear on.

Example: ***AS3.Addmdi("This is an MDI:", "Arial", "left", 26, 4564978, 541, 421, 280, 2);***

Addslider()

Function: void Addslider(int posX, int posY, int length, int colorcode1, int colorcode2, int minvalue, int maxvalue, bool isvertical, int fieldnumber, int layernumber)

Description: This function adds a slider control to the screen.

Parameters:

- ***posX*** and ***posY*** defines the top-left corner position of the MDI control.
- ***length*** defines the width of the slider in pixels.
- ***colorcode1*** defines the RGB color code in integer format.
- ***colorcode2*** defines the RGB color code in integer format.
- ***minvalue*** defines the minimum slider value as an integer.
- ***maxvalue*** defines the maximum slider value as an integer.
- ***isvertical*** defines if the slider is vertical / horizontal.
- ***feildnumber*** is the identifier of the slider object.
- ***layernumber*** defines which layer the slider will appear on.

Example: ***AS3.Addslider("This is an MDI:", "Arial", "left", 26, 4564978, 541, 421, 280, 2);***

Addtab()

Function: void Addtab(string labeltext, string labelfont, string textalign, int fontsize, int fontcolor, int posX, int posY, int labelwidth, int labelheight, int picturenumber, int labelnumber, int parentnumber)

Description: This function adds one tab layer on the screen. Tab layers are the base screen elements in the UCCNC software and they are like layers in CAD software.

The tab layers can contain other screen elements like buttons, fields, background images etc.

There can be any number of tab layers on the screen, and by default tab layers are all transparent.

Every tab layer object has an identifier which is the ***labelnumber*** parameter and also a ***parentnumber*** .

Tab layers can be in parent-child relation with each other, the ***parentnumber*** defines the parent tab layer.

Tabs which have the same ***parentnumber*** are on the same level and just like in Windows tab controls, only one layer can be selected from the same level. Tabs and the screen items on the other same level tab layers are hidden.

Every tab layer has a **label** which is a graphical object and is used to allow the user to press and select the tab layer. The label position and width and height can be defined with the ***posX***, ***posY***, ***labelwidth***, ***labelheight*** parameters and the label can also have a picture rendered onto it, the ***picturenumber*** defines which previously loaded picture is rendered onto the label.

Parameters:

- ***labeltext*** defines the text string title of the tab.
- ***labelfont*** defines the font type of the text string title.
- ***textalign*** defines the justification of the text string title.
- ***fontsize*** defines the size of the font in pixels.
- ***fontcolor*** defines the RGB color code in integer format.
- ***posX*** and ***posY*** defines the top-left corner position of the tab control.
- ***labelwidth*** and ***labelheight*** define width and height of the tab control
- ***picturenumber*** define
- ***labelnumber*** define
- ***parentnumber*** define

Example: ***AS3.Addtab("", "Arial", "center", 14, 0, 69, 30, 100, 18, 94,20, 4);***

Addtoolpath()

Function: void Addtoolpath(int posX, int posY, int Width, int Height, int layernumber);
Description: This function adds a toolpath object.
Parameters:

- The ***posX*** and ***posY*** define the top-left corner position of the toolpath, and are in pixels.
- The ***Width*** and ***Height*** define the dimensions of the toolpath, and are in pixels.
- The ***layernumber*** is the number of the tab layer the toolpath will appear on.

Example: ***AS3.Addtoolpath(80, 39, 372, 306, 2);***

AddUCCAM()

Function: void AddUCCAM(int posX, int posY, int Width, int Height int labelnumber, int layernumber)
Description: This function adds a UCCAM object to the screenset. A UCCAM object is basically a window which allows you to display a DXF file within to allow you to view & generate toolpath operations (as shown on the CAM tab).
Parameters:

- ***posX*** and ***posY*** define the top-left corner position of the object, and are in pixels.
- ***Width*** and ***Height*** define the dimensions of the object, and are in pixels.
- ***labelnumber*** is the unique identifier of the UCCAM object.
- ***layernumber*** is the number of the tab layer the UCCAM object will appear on.

Example: ***AS3.AddUCCAM(48, 46, 468, 468, 1, 45);***

Filterfieldtext()

Function: void Filterfieldtext(string value, int fieldnumber)
Description: This function filters the characters which can be input into a field.
Parameters:

- ***value*** defines the filtered characters which can be input to the field, all other characters are prohibited. Note, that if a '-' character needs to be allowed in the filter then the \\ mark must be placed before the '-' character, because the '-' char means a range.
- ***fieldnumber*** defines the number of the field to apply the filter to.

Example: ***AS3.Filterfieldtext("0123456789.\\-%", 232);***
AS3.Filterfieldtext("0-9", 232); *//Allows characters 0123456789, because the - character means range.*

Loadpicture()

Function: void Loadpicture(string pictureupURL, string picturedownURL, int picturenumber, bool IsLastpicturetoload)
Description: This function loads 2 images into memory; the first 2 parameters are the filenames with the directory path for the image files. The base path is the UCCNC installation /Flashscreen/ path.
Parameters:

- ***pictureupURL*** defines the image which is shown when a button is not pressed (i.e. is released)
- ***picturedownURL*** defines the image which is shown when the button is being pressed. For screen elements like for backgrounds where only one image is used set both parameters the same and the software will automatically load one instance only into the memory.
- ***picturenumber*** defines the numeric identifier for the picture, this identifier can be later used to identify this image when the image has to be rendered onto a screen object like on a button or a background, tab layer etc.
- ***IsLastpicturetoload*** legacy function, no longer used, set to false.

Example: ***AS3.Loadpicture(bitmapfolder + "plus_up.png", bitmapfolder + "plus_down.png", 22, false);***

selectlayer()

Function: void selectlayer(int layernumber)
Description: This function selects a tab layer, so that any other tab layers on the same level will be hidden and this tab layer will be shown on the software startup.
Parameters:

- ***layernumber*** defines the initial / startup layer for the main screen

Example: ***AS3.selectlayer(2);***

Setfield()

Function: void Setfield(double value, int labelnumber)
Description: This function sets the initial value of a field object.
Parameters:

- ***value*** defines a numeric value for a field.
- ***labelnumber*** defines the number of the label to set the numeric value to.

Example: ***AS3.Setfield(0, 866);***

Setfieldtext()

Function: void Setfieldtext(string value, int labelnumber)
Description: This function sets the initial text of a field object.
Parameters:

- ***value*** defines the text string value for a field.
- ***labelnumber*** defines the number of the label to set the text string value to.

Example: ***AS3.Setfieldtext("initialtext", 866);***

Clearcomboboxitems()

Function: void Clearcomboboxitems(int comboboxnumber)
Description: This function clears all items from a combo box.
Parameter:

- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Clearcomboboxitems(11);***

Clearlist()

Function: void Clearlist(int listboxnumber)
Description: This function clears all items from a list box.
Parameter:

- The ***listboxnumber*** is the unique identifier of the object.

Example: ***AS3.Clearlist(11);***

Validatenewcomboboxitems()

Function: void Validatenewcomboboxitems(int comboboxnumber)
Description: This function validates the items added to a combo box, it is called after all *Addcomboboxitem* functions have been called to add items to the combobox with the same unique comboboxnumber.
Parameter:

- ***comboboxnumber*** is the unique identifier of the combobox object.

Example: ***AS3.Validatenewcomboboxitems(11);***

Updatecomboboxselection()

Function: void Updatecomboboxselection(int index, int comboboxnumber)
Description: This function updates the combo box and sets focus on the index item (starting at index zero (0)).
Parameter:

- ***index*** is the item number in the list to be selected at screen load.
- ***comboboxnumber*** is the unique identifier of the object.

Example: ***AS3.Updatecomboboxselection(0,11);***

Setcolorpickercolor()

Function: void Setcolorpickercolor(int theColor, int LEDnumber)
Description: This function returns the colorpicker object colour as RGB color code in integer format.
Parameters:

- ***theColor*** defines the color picker RGB color code in integer format.
- ***LEDnumber*** is the unique identifier of the object.

Example: ***AS3.*** Setcolorpickercolor ***(-8355712, 1);***

Setscreensize()

Function: void Setscreensize(int Width, int Height)
Description: This function sets the width & height of the screenset
Parameters:

- ***Width*** defines the width of the screenset / Jogscreen, in pixels.
- ***Height*** defines the height of the screenset / Jogscreen, in pixels.

Example: ***AS3.Setscreensize(1024, 692);***
Example: ***AS3jog.Setscreensize(300, 692);***

Setjogpaneltabsize()

Function: void Setjogpaneltabsize(int Width)
Description: This function sets the width of the retracted jog panel
Parameters:

- ***Width*** defines the width of the retracted jog panel, in pixels.

Example: ***AS3jog.Setjogpaneltabsize(40);***

UCCNC INTERNAL VARIABLES

This section describes the UCCNC software **internal variables** that are available.

All variables are available by other means such as via fields which are the recommended route to read all variables data.

It should be noed that UCCNC do not now recommend the use of variables to access these listed values and the recommended route to obtain these values are via the listed uccnc functions listed elsewhere.

The list below is provided for completeness of all information available and has been collated by the writer of this manual. The writer has attemped to clarify what data each of the variables exactly relates to. Errors and omissions are accepted. Please feel free to provide any corrections you may discover and I shall endeavour to update the manual for completeness of correct information.

int exec.actcannedreturnlevel

int exec.actplane

int exec.actualdistmode

double exec.actualfeedoverride

int exec.actualmodal

string exec.actualprofilename

double exec.actualSS

bool exec.Ahomed (ABCXYZ)

int exec.AnIn1 (1234)

int exec.AnOut1 (1234)

double exec.ArcstartposX

double exec.ArcstartposY

double exec.Astepsize (ABCXYZ)

bool[] exec.Buttonstates

bool exec.Call1000

bool exec.calledfromscreen

bool exec.constructmacro

long exec.currentOutput

int exec.Currenttool

bool exec.CycleStopped

bool exec.destructed

bool exec.destructmacro

bool exec.didinitialmovement

string exec.digitsformat

bool exec.dohoming

bool exec.doinitialmovement

double exec.dwelltime

bool exec.Estopin

double exec.eTOZval

double exec.eWzval

bool exec.exit

bool exec.feedholdbuttonstate

int exec.Fieldenteredscreennumber

bool exec.fileloading

bool exec.Filereloadcommand

bool exec.G68active

int exec.gcodelinenumber

bool exec.go

int exec.grabbedkey

int exec.homi

bool exec.hotkeysenabled

bool exec.Isabs

bool[] exec.ISenableLEDset

bool exec.Isfeedhold

bool exec.IsHoming

int[] exec.Ismacrorun

double[] exec.ivars

static int exec.ivarssize

bool exec.jogAmbuttonpressed (ABCXYZ)

bool exec.jogApbuttonpressed (ABCXYZ)

double exec.JogFeedrate

bool exec.joghidden

int exec.jogmode

double exec.jogsteprate

bool exec.Lastactionwasstepjog

double exec.lastfeed

string exec.laststatusmessage

static int exec.LEDssize

int[] exec.LEDstates

bool exec.Limitsoverridemode

bool exec.Loadimages

bool exec.Loadjogscreen

bool exec.Loadmainscreen

bool[] exec.looprun

bool exec.M3on (3478)

double? exec.MachposK

double? exec.MachposQ

bool exec.macrocall

bool exec.macroloopcall

bool exec.macrostop

bool exec.MDIgo

int exec.MDInumber

bool exec.MDIstopped

int exec.MPGaxis

int exec.MPGmode

string exec.newstatusmessage

bool exec.Offlinemode

string exec.OperatorPassword

double exec.Oz

double? exec.PeckZ

int exec.prevmodal

int exec.rememberjogmode

double exec.rotAngle

double exec.rotX

double exec.rotY

double exec.scaleA (ABCXYZ)

bool exec.setnextlinepressed

bool exec.showmachinecoords

int exec.startobjectnumber

bool exec.stepjoggoing

bool exec.stopped

static int exec.substackdepth

bool exec.tabbing

bool exec.THCvirtualARCON

bool exec.THCvirtualDOWN

bool exec.THCvirtualUP

bool exec.theAutozero

int exec.TotalM3s (3478)

double exec.Wa (abcxyz)

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

BUTTONS (SORT BY NUMBER)

This section describes the **UCCNC software** Button screen objects.

Each button represents an internal function of the UCNC software.
Buttons can be called by their number from macro code.
This documentation lists all the accessible buttons, sorted **numerically.**

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
100	ZeroX	Zeros the X axis position.
101	ZeroY	Zeros the Y axis position.
102	ZeroZ	Zeros the Z axis position.
103	ZeroA	Zeros the A axis position.
104	ZeroB	Zeros the B axis position.
105	ZeroC	Zeros the C axis position.
106	ZeroAll	Zeros All axis position.
107	HomeX	Runs the X axis to the home sensor.
108	HomeY	Runs the Y axis to the home sensor.
109	HomeZ	Runs the Z axis to the home sensor.
110	HomeA	Runs the A axis to the home sensor.
111	HomeB	Runs the B axis to the home sensor.
112	HomeC	Runs the C axis to the home sensor.
113	HomeAll	Runs all axis to the home sensor. The homing sequence is defined in the setup.
114	M3toggle	Toggles the M3 spindle CW button.
115	M4toggle	Toggles the M4 spindle CCW button.
116	M7toggle	Toggles the M7 mist button.
117	M8toggle	Toggles the M8 flood button.
118	G54select	Selects the G54 coordinate offset.
119	G55select	Selects the G55 coordinate offset.
120	G56select	Selects the G56 coordinate offset.
121	G57select	Selects the G57 coordinate offset.
122	G58select	Selects the G58 coordinate offset.
123	G59select	Selects the G59 coordinate offset.
124	OpenGcodefile	Starts an Open G-code file dialog.
125	CloseGcodefile	Closes the G-code file which is loaded.
126	EditGcodefile	Opens the notepad to edit the loaded G-code file.
127	RewindGcodefile	Rewinds the loaded G-code file with jumping to the first row in the file.
128	Cyclestart	Makes a cyclic run on the loaded G-code file.
129	Runsingleline	Executes one line of code (the actual code line) of the loaded G-code file.
130	Cyclestop	Stops the G-code execution.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
131	Gotozero	Moves all axis to the zero position with a rapid linear interpolation movement (uses M204 macro)
132	FROincrease	Increases the Feedrate override value.
133	FROdecrease	Decreases the Feedrate override value.
134	SROincrease	Increases the spindle speed override value.
135	SROdecrease	Decreases the spindle speed override value.
136	Toolpathzoomin	Zooms in the toolpath viewer.
137	Toolpathzoomout	Zooms out the toolpath viewer.
138	Toolpathzoomcontents	Zooms the contents in the toolpath viewer.
139	Toolpathview45	Sets the toolpath to a 45° look viewing mode.
140	Toolpathviewright	Sets the toolpath to a right side look viewing mode.
141	Toolpathviewleft	Sets the toolpath to a left side look viewing mode.
142	Toolpathviewtop	Sets the toolpath to a top side look viewing mode.
143	ToolpathviewISO	Sets the toolpath to Isometric viewing mode.
144	Resettoggle	Toggles the reset button.
145	Offlinetoggle	Toggles the offline button.
146	Limitsoverridetoggle	Toggles the limits override button.
147	JogX+	Jogs the X axis to positive direction.
148	JogX-	Jogs the X axis to negative direction.
149	JogY+	Jogs the Y axis to positive direction.
150	JogY-	Jogs the Y axis to negative direction.
151	JogZ+	Jogs the Z axis to positive direction.
152	JogZ-	Jogs the Z axis to negative direction.
153	JogA+	Jogs the A axis to positive direction.
154	JogA-	Jogs the A axis to negative direction.
155	JogB+	Jogs the B axis to positive direction.
156	JogB-	Jogs the B axis to negative direction.
157	JogC+	Jogs the C axis to positive direction.
158	JogC-	Jogs the C axis to negative direction.
159	Jograteincrease	Increases the jog rate.
160	Jogratedecrease	Decreases the jog rate.
161	Jogmodecont	Sets the jog mode to continuous.
162	Jogmodestep	Sets the jog mode to stepping.
163	R*	
164	Jogsteprate001	Sets the jog distance when stepping mode to 0.01 Units.
165	Jogsteprate010	Sets the jog distance when stepping mode to 0.10 Units.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
166	Jogsteprate100	Sets the jog distance when stepping mode to 1.00 Units.
167	Savesettings	Saves all settings to the profile file.
168	Applysettings	Applies the settings on the setup screens.
169	Setnextline	Sets the G-code execution pointer to the line defined on the screen in the 128. labelfield.
170	G54offsetcurrentpos	Offsets the actual position of the G54 coordinate system.
171	G55offsetcurrentpos	Offsets the actual position of the G55 coordinate system.
172	G56offsetcurrentpos	Offsets the actual position of the G56 coordinate system.
173	G57offsetcurrentpos	Offsets the actual position of the G57 coordinate system.
174	G58offsetcurrentpos	Offsets the actual position of the G58 coordinate system.
175	G59offsetcurrentpos	Offsets the actual position of the G59 coordinate system.
176	G54offsetclear	Clears the offset in the G54 coordinate system.
177	G55offsetclear	Clears the offset in the G55 coordinate system.
178	G56offsetclear	Clears the offset in the G56 coordinate system.
179	G57offsetclear	Clears the offset in the G57 coordinate system.
180	G58offsetclear	Clears the offset in the G58 coordinate system.
181	G59offsetclear	Clears the offset in the G59 coordinate system.
182	Tooloffsetclear	Clears the tool offset.
183	Tooloffsetclear	Clears the tool offset.
184	Tooloffsetclear	Clears the tool offset.
185	Tooloffsetclear	Clears the tool offset.
186	Tooloffsetclear	Clears the tool offset.
187	Tooloffsetclear	Clears the tool offset.
188	Listprofiles	Lists the available profile file names in the 1.Listcomponent.
189	Deleteprofile	Deletes the actually selected profile file.
190	Loadprofile	Loads the actually selected profile file.
191	Createnewprofile	Create a new profile, the name of the profile is held in the 216.inputfield.
192	DoXMLimport	Imports a Mach3 .xml setup file. The function start an open file dialog.
193	Gotoparkposition1	Commands the machine to park position 1. (Code executed in Macro M200)
194	Gotoparkposition2	Commands the machine to park position 2. (Code executed in Macro M201)
195	Gotoparkposition3	Commands the machine to park position 3. (Code executed in Macro M202)

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
196	Toollengthmeasurement	Commands a tool length measurement. (Code executed in Macro M31)
197	Machinecoordstoggle	Toggles the position DROs between the machine coordinates and the actual offset coordinates view.
198	CalibrateX	Calibrates the steps per units value with travel measurement for the X-axis.
199	CalibrateY	Calibrates the steps per units value with travel measurement for the Y-axis.
200	CalibrateZ	Calibrates the steps per units value with travel measurement for the Z-axis.
201	CalibrateA	Calibrates the steps per units value with travel measurement for the A-axis.
202	CalibrateB	Calibrates the steps per units value with travel measurement for the B-axis.
203	CalibrateC	Calibrates the steps per units value with travel measurement for the C-axis.
204	Softlimitstoggle	Toggles the software limits enable function.
205	THCtoggle	Toggles the THC control enable function.
206-215	R*	
216	?	Go to Safe Z (used with M216)
217	R*	
218	Zeroworktimer	Zeros the work timer.
219	ClearG92offset	Clears the G92 offset coordinates.
220	MPGXaxisselect	Selects the X axis for the MPG jog.
221	MPGYaxisselect	Selects the Y axis for the MPG jog.
222	MPGZaxisselect	Selects the Z axis for the MPG jog.
223	MPGAaxisselect	Selects the A axis for the MPG jog.
224	MPGBaxisselect	Selects the B axis for the MPG jog.
225	MPGCaxisselect	Selects the C axis for the MPG jog.
226	MPGcontmodeselect	Selects the continuous jog mode for the MPG.
227	MPGsinglemodeselect	Selects the single step jog mode for the MPG.
228	MPGmultimodeselect	Selects the multi step jog mode for the MPG.
229	JogXplusoff	Switches the X axis positive direction jogging off.
230	JogXminusoff	Switches the X axis negative direction jogging off.
231	JogYplusoff	Switches the Y axis positive direction jogging off.
232	JogYminusoff	Switches the Y axis negative direction jogging off.
233	JogZplusoff	Switches the Z axis positive direction jogging off.
234	JogZminusoff	Switches the Z axis negative direction jogging off.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
235	JogAplusoff	Switches the A axis positive direction jogging off.
236	JogAminusoff	Switches the A axis negative direction jogging off.
237	JogBplusoff	Switches the B axis positive direction jogging off.
238	JogBminusoff	Switches the B axis negative direction jogging off.
239	JogCplusoff	Switches the C axis positive direction jogging off.
240	JogCminusoff	Switches the C axis negative direction jogging off.
241	Jogsteprate0001	Sets the jog distance when stepping mode to 0.001 Units.
242 - 292	Sethotkeycodes1to48	Opens the Hotkeys keyboard key selection window.
293 - 343	Sethotkeyfunctions1to48	Opens a Function selection window for the hotkeys.
344 - 394	Setinputtriggerfunctions1to48	Opens a Function selection window for the input triggers.
400 - 449	SelectTABlayer	Selects the TAB layer on the screen. The layer number selected is the function code - 400, so TAB layer 1. selected with code 401. Note: To select a sub layer with a hotkey or with an input trigger configure the same key or pin to select all of it's parent layers and also the layer itself, so the parent layers and the sub layers will also be selected and the sub layer will always showup this way, no matter which parent layer was originally shown.
450 - 500	SetoutputtriggerLEDs1to48	Opens a LED selection window for the output triggers.
501	CAM_ImportDXF	Imports a dxf drawing file to UCCAM.
502	CAM_Generatetoolpath	Generates the toolpath in UCCAM.
503	CAM_Generategcode	Creates the gcode in UCCAM.
504	M3on	Switches the M3 spindle CW button on.
505	M3off	Switches the M3 spindle CW button off.
506	M4on	Switches the M4 spindle CCW button on.
507	M4off	Switches the M4 spindle CCW button off.
508	M7on	Switches the M7 mist button on.
509	M7off	Switches the M7 mist button off.
510	M8on	Switches the M8 flood button on.
511	M8off	Switches the M8 flood button off.
512	Reseton	Switches the Reset button on.
513	Resetoff	Switches the Reset button off.
514	Offlineon	Switches the Offline button on.
515	Offlineoff	Switches the Offline button off.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
516	Machinecoordson	Switches the machine coordinates button on.
517	Machinecoordsoff	Switches the machine coordinates button off.
518	Softlimitson	Switches the softlimits setting on.
519	Softlimitsoff	Switches the softlimits setting off.
520	THCon	Switches the THC control on.
521	THCoff	Switches the THC control off.
522	Feedholdtoggle	Toggles the feedhold button.
523	Feedholdon	Switches the feedhold button on.
524	Feedholdoff	Switches the feedhold button off.
525	Configplugins	Opens the plugin configuration window
526	Editscreen	Enters the screen editor mode
527	Savealloffsets	Saves all offset values to the profile file.
528	Clearstatusmessages	Clears the Status message box
529	THCAntiDiveon	Switches the Anti Dive function for the THC on.
530	THCAntiDiveoff	Switches the Anti Dive function for the THC off.
531	THCAntiDivetoggle	Toggles the Anti Dive function for the THC.
532	Configmacroloops	Opens the macroloop configuration window.
533	THCDelayon	Switches the THC Delay function for the THC on.
534	THCDelayoff	Switches the THC Delay function for the THC off.
535	THCDelaytoggle	Toggles the Delay function for the THC.
536	THCarconsignalon_emulation	Switches the THC arcon virtual signal on. Can be used to emulate the THC arcon signal from keyboard.
537	THCarconsignaloff_emulation	Switches the THC arcon virtual signal off. Can be used to emulate the THC arcon signal from keyboard.
538	THCupsignalon_emulation	Switches the THC up virtual signal on. Can be used to emulate the THC up signal from keyboard.
539	THCupsignaloff_emulation	Switches the THC up virtual signal off. Can be used to emulate the THC up signal from keyboard.
540	THCdownsignalon_emulation	Switches the THC down virtual signal on. Can be used to emulate the THC down signal from keyboard.
541	THCdownsignaloff_emulation	Switches the THC down virtual signal off. Can be used to emulate the THC down signal from keyboard.
542	THCAntiDownon	Switches the Anti Down function for the THC on.
543	THCAntiDownoff	Switches the Anti Down function for the THC off.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
544	THCAntiDowntoggle	Toggles the Anti Down function for the THC.
545	Safeprobemodeon	Switches the safe probe mode on.
546	Safeprobemodeoff	Switches the safe probe mode off.
547	Safeprobemodetoggle	Toggles the safe probe mode.
548	Operatorlock	Opens the operator lock/unlock window.
549	Showstatistics	Opens the machine statistics window.
550	ShowSpindlepulleys	Opens the Spindle pulleys selection window.
551	Digitize_setfilename	Sets the file name to save the Digitized points to.
552	JogSafeprobemodeon	Switches the jog safe probe mode on.
553	JogSafeprobemodeoff	Switches the jog safe probe mode off.
554	JogSafeprobemodetoggle	Toggles the jog safe probe mode.
555	ReloadGcodefile	Reloads the g-code file if a file is already loaded.
556	JograteResetto100percent	Resets the jog rate field to 100%.
557	FROResetto100percent	Resets the feed override to 100%.
558	SROResetto100percent	Resets the spindle speed override to 100%.
559	M1optionalstopon	Switches the M1 optional stop on.
560	M1optionalstopoff	Switches the M1 optional stop off.
561	M1optionalstoptoggle	Toggles the M1 optional stop state.
562 - 612	Sethotkeycodes49to96	Opens the Hotkeys keyboard key selection window.
613 - 663	Sethotkeyfunctions49to96	Opens a Function selection window for the hotkeys.
664 - 714	Setinputtriggerfunctions49to96	Opens a Function selection window for the input triggers.
715 - 762	SetoutputtriggerLEDs49to96	Opens a LED selection window for the output triggers.
763 - 769	R*	R* = Reserved address, do not use this address.
770	JogpanelShow	Shows the jog panel.
771	JogpanelHide	Hides the jog panel.
772	JogpanelToggle	Toggles the jog panel.
773	OpenSpindlePIDsetup	Opens the Spindle PID controller setup window.
774	ToolpathTCPfollowOn	Locks the toolpath center to the TCP point.
775	ToolpathTCPfollowOff	Unlocks the toolpath center to the TCP point.
776	ToolpathTCPfollowToggle	Toggles the toolpath center to the TCP point.
777	ShowToolpathZplatesetup	Shows the toolpathZplate setup window.
778	ShowCutterRcompwarnings	Shows the Cutter Radius Compensation warnings window.
779	OpenTooltabledetails	Opens the detailed tooltable window.
780	Savetooltabledatas	Saves all the data of the tool table to file

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
781	DereferenceAllHomes	Clears the homed state of all home buttons.
782	CloseResetandQuestionForms	Closes the Reset warning and the Question forms if any of them is active.
783	ChangetoolM6	Executes the tool changing process.
784	Toggletoolpathdimensionsvisibility	Toggles the dimensions drawings visibility in the toolpath view.
785	Toolpathdimensionsvisibilityon	Switches on the dimensions drawings visibility in the toolpath view.
786	Toolpathdimensionsvisibilityoff	Switches off the dimensions drawings visibility in the toolpath view.
787	SelectScreenset	Opens the screenset selector window.
788	Camerareloadcapturedevicelist	Reloads the available camera list into listbox#3.
789	Cameracapturetoggle	Toggles the camera video capture on/off.
790	Cameracaptureon	Switches the camera video capture on.
791	Cameracaptureoff	Switches the camera video capture off.
792	Movecameraoffset	Moves the XY axes with the defined camera offset distance.
793	Cameragrayscalefiltertoggle	Toggles the camera grayscale filter on/off.
794	Cameragrayscalefilteron	Switches the camera grayscale filter on.
795	Cameragrayscalefilteroff	Switches the camera grayscale filter off.
796	Cameraedgefiltertoggle	Toggles the camera edge filter on/off.
797	Cameraedgefilteron	Switches the camera edge filter on.
798	Cameraedgefilteroff	Switches the camera edge filter off.
799	R*	R* = Reserved address, do not use this address.
800	ProbeToolQuickJump	Jumps to the probe screen and selects tool probe
801	ToolProbeMode	Selects tool probe mode
802	SimpleProbeMode	Selects simple probe mode
803	AngleProbeMode	Selects angle probe mode
804	PocketProbeMode	Selects pocket probe mode
805	OuterProbeMode	Selects outer probe mode
806	InnerCornerProbeMode	Selects inner corner probe mode
807	OuterCornerProbeMode	Selects outer corner probe mode
808	RectangleInnerProbeMode	Selects rectangle inner probe mode
809	CircleInnerProbeMode	Selects circle inner probe mode
810	RectangleOuterProbeMode	Selects rectangle outer probe mode
811	CircleOuterProbeMode	Selects circle outer probe mode
812	R*	
813	InnerCornerBottomLeft	Selects bottom left corner for inner corner probe
814	InnerCornerBottomRight	Selects bottom right corner for inner corner probe
815	InnerCornerTopRight	Selects top right corner for inner corner probe

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
816	InnerCornerTopLeft	Selects top left corner for inner corner probe
817	OuterCornerBottomLeft	Selects bottom left corner for outer corner probe
818	OuterCornerBottomRight	Selects bottom right corner for outer corner probe
819	OuterCornerTopRight	Selects top right corner for outer corner probe
820	OuterCornerTopLeft	Selects top left corner for outer corner probe
821	StartProbe	Starts probing sequence or confirm probing question
822	ProbeAxis1SelectX	Selects X axis for Axis1 on probe screen
823	ProbeAxis1SelectY	Selects Y axis for Axis1 on probe screen
824	ProbeAxis1SelectZ	Selects Z axis for Axis1 on probe screen
825	ProbeAxis1SelectA	Selects A axis for Axis1 on probe screen
826	ProbeAxis1SelectB	Selects B axis for Axis1 on probe screen
827	ProbeAxis1SelectC	Selects C axis for Axis1 on probe screen
828	ProbeAxis2SelectX	Selects X axis for Axis2 on probe screen
829	ProbeAxis2SelectY	Selects Y axis for Axis2 on probe screen
830	ProbeAxis2SelectZ	Selects Z axis for Axis2 on probe screen
831	ProbeAxis2SelectA	Selects A axis for Axis2 on probe screen
832	ProbeAxis2SelectB	Selects B axis for Axis2 on probe screen
833	ProbeAxis2SelectC	Selects C axis for Axis2 on probe screen
834	ProbeAxisCSelectX	Selects X axis for clearance axis on probe screen
835	ProbeAxisCSelectY	Selects Y axis for clearance axis on probe screen
836	ProbeAxisCSelectZ	Selects Z axis for clearance axis on probe screen
837	ProbeAxisCSelectA	Selects A axis for clearance axis on probe screen
838	ProbeAxisCSelectB	Selects B axis for clearance axis on probe screen
839	ProbeAxisCSelectC	Selects C axis for clearance axis on probe screen
840	TouchToolProbeMode	Selects touch probe as tool probe mode
841	MobileToolProbeMode	Selects mobile probe as tool probe mode
842	FixedToolProbeMode	Selects fixed probe as tool probe mode
843	SetAsWpProbePos	Sets current position as workpiece probe position
844	ProbeInWpProbePos	Toggles probe in workpiece probe position
845	GotoWpProbePos	Moves to workpiece probe position
846	SetAsMobileProbePos	Sets current position as mobile probe position
847	SetAsFixedProbePos	Sets current position as fixed probe position
848	GotoProbePos	Moves to mobile or fixed probe position

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
849	RefCurrentAsWp	References current position as workpiece zero position
850	RefProbeAsWp	Probes and references the probed position as workpiece zero position
851	ZeroProbedAxis	Toggles zero probed axis
852	ZeroOnAllOffsets	Toggles zero on all offsets
853	SingleProbeMode	Toggles single probe mode
854	EnableRetract	Toggles enable retract
855	PauseBeforeProbe	Toggles pause before probe
856	CommonAxisSettings	Toggles common axis settings
857	ProbeInfo	Show probe screen help
858	GotoRef	Moves axes to reference coordinates
859	Axis1CountGage	Toggles axis 1 count gage
860	Axis1CountDia	Toggles axis 1 count dia
861	Axis2CountGage	Toggles axis 2 count gage
862	Axis2CountDia	Toggles axis 2 count dia
863	MobileProbePosAEna	Toggles mobile probe position A axis enable
864	MobileProbePosBEna	Toggles mobile probe position B axis enable
865	MobileProbePosCEna	Toggles mobile probe position C axis enable
866	FixedProbePosAEna	Toggles fixed probe position A axis enable
867	FixedProbePosBEna	Toggles fixed probe position B axis enable
868	FixedProbePosCEna	Toggles fixed probe position C axis enable
869	SafeXPositive	Toggles greater Z is safer in G19
870	SafeYPositive	Toggles greater Z is safer in G18
871	SafeXPositive	Toggles greater Z is safer in G17
872	SaveMobileProbePos	Toggles save workpiece references on exit
873	SaveWpReferences	Toggles save mobile probe position on exit
874	OverrideProbeDia	Toggles override probe diameter
875	LimitTraverseSpeed	Toggles limit traverse speed
876	SeparateSettingsPerMode	Toggles separate settings per mode
877	NoBlower	Disables blower
878	BlowWithM7	Sets blower to use M7
879	BlowWithM8	Sets blower to use M8
880	BlowUsingOutput	Sets blower to use output
881	BlowerPinActiveLow	Toggles blower output pin active low
882	BlowToolProbes	Sets blower to blow only on tool probes
883	BlowAllProbes	Sets blower to blow on all probes
884	AirBlower	Starts the blower and is on while blowing
885-999	R*	R* = Reserved address, do not use this address.

BUTTON NUMBER	FUNCTION NAME	DESCRIPTION
1000	Increasecameratargetcircles	Increases the number of camera target circles
1001	Decreasecameratargetcircles	Decreases the number of camera target circles
1002	Cameratargetlinetoggle	Toggles the camera target line visibility
1003	Cameratargetlineon	Makes the camera target line visible.
1004	Cameratargetlineoff	Makes the camera target line invisible.
1005	MovetocameraZheight	Makes the Z axis to move the the set camera height.
1006	Camerainvertimagetoggle	Makes the camera image colors invertion function to toggle.
1007	Camerainvertimageon	Makes the camera image colors invertion on.
1008	Camerainvertimageoff	Makes the camera image colors invertion off.
20000 - 21999	Macro_call_buttons	These buttons codes call the same number of macros.

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

USER BUTTON NUMBER	MACROCODE DESCRIPTION
Users can define M-codes M20000 to M21999 which correspond to the same Button Numbers, this allows UCCNC to call the macros directly from screen button presses. It is highly recommended that you DO NOT call Macros from Macros, given there is no way to pass back to the original macro given they run from start to finish (M98 excluded), hence will loop. This section is provided to allow users to record their M-codes / User Button Numbers along with any descriptions for future reference.	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	
2	

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

FIELDS (SORT BY NUMBER)

This section describes the **UCCNC software** field screen objects.

Each field object represents a text label on the UCCNC screen.
"textfield", "textfieldnb", "field", "fieldnb" type labels can be read and written from macro code and are **editable** by the user on the UCCNC GUI.
The "textfield", "textfieldnb" can contain any kind of text including words.
The "field", "fieldnb" can contain numerical values only.
"showfield", "showfieldnb" type labels can be read and written from macro code and are **not editable** by the user on the UCCNC GUI.
This documentation lists all the accessible field objects, sorted **numerically.**

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
1	Xaxissteppin	
2	Xaxisdirpin	
3	Xaxislimitminuspin	
4	Xaxislimitpluspin	
5	Xaxishomepin	
6	Xaxishomingspeed	
7	Xaxishomeoffset	
8	Xaxisstepsper	
9	Xaxisvelocity	
10	Xaxisacceleration	
11	Xaxissoftlimitminus	
12	Xaxissoftlimitplus	
13	Xaxiscompaccel	
14	Xaxisbacklash	
15	Xaxishomingspeeddown	
16	Yaxissteppin	
17	Yaxisdirpin	
18	Yaxislimitminuspin	
19	Yaxislimitpluspin	
20	Yaxishomepin	
21	Yaxishomingspeed	
22	Yaxishomeoffset	
23	Yaxisstepsper	
24	Yaxisvelocity	
25	Yaxisacceleration	
26	Yaxissoftlimitminus	
27	Yaxissoftlimitplus	
28	Yaxiscompaccel	
29	Yaxisbacklash	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
30	Yaxishomingspeeddown	
31	Zaxissteppin	
32	Zaxisdirpin	
33	Zaxislimitminuspin	
34	Zaxislimitpluspin	
35	Zaxishomepin	
36	Zaxishomingspeed	
37	Zaxishomeoffset	
38	Zaxisstepsper	
39	Zaxisvelocity	
40	Zaxisacceleration	
41	Zaxissoftlimitminus	
42	Zaxissoftlimitplus	
43	Zaxiscompaccel	
44	Zaxisbacklash	
45	Zaxishomingspeeddown	
46	Aaxissteppin	
47	Aaxisdirpin	
48	Aaxislimitminuspin	
49	Aaxislimitpluspin	
50	Aaxishomepin	
51	Aaxishomingspeed	
52	Aaxishomeoffset	
53	Aaxisstepsper	
54	Aaxisvelocity	
55	Aaxisacceleration	
56	Aaxissoftlimitminus	
57	Aaxissoftlimitplus	
58	Aaxiscompaccel	
59	Aaxisbacklash	
60	Aaxishomingspeeddown	
61	Baxissteppin	
62	Baxisdirpin	
63	Baxislimitminuspin	
64	Baxislimitpluspin	
65	Baxishomepin	
66	Baxishomingspeed	
67	Baxishomeoffset	
68	Baxisstepsper	
69	Baxisvelocity	
70	Baxisacceleration	
71	Baxissoftlimitminus	
72	Baxissoftlimitplus	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
73	Baxiscompaccel	
74	Baxisbacklash	
75	Baxishomingspeeddown	
76	Caxissteppin	
77	Caxisdirpin	
78	Caxislimitminuspin	
79	Caxislimitpluspin	
80	Caxishomepin	
81	Caxishomingspeed	
82	Caxishomeoffset	
83	Caxisstepsper	
84	Caxisvelocity	
85	Caxisacceleration	
86	Caxissoftlimitminus	
87	Caxissoftlimitplus	
88	Caxiscompaccel	
89	Caxisbacklash	
90	Caxishomingspeeddown	
91	Estoppin	
92	Probepin1	
93	Indexpin	
94	Indexprescaler	
95	Chargepumppin	
96	-	
97	G54_CurrentcoordX	
98	G54_CurrentcoordY	
99	G54_CurrentcoordZ	
100	G54_CurrentcoordA	
101	G54_CurrentcoordB	
102	G54_CurrentcoordC	
103	G55_CurrentcoordX	
104	G55_CurrentcoordY	
105	G55_CurrentcoordZ	
106	G55_CurrentcoordA	
107	G55_CurrentcoordB	
108	G55_CurrentcoordC	
109	G56_CurrentcoordX	
110	G56_CurrentcoordY	
111	G56_CurrentcoordZ	
112	G56_CurrentcoordA	
113	G56_CurrentcoordB	
114	G56_CurrentcoordC	
115	G57_CurrentcoordX	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
116	G57_CurrentcoordY	
117	G57_CurrentcoordZ	
118	G57_CurrentcoordA	
119	G57_CurrentcoordB	
120	G57_CurrentcoordC	
121	G58_CurrentcoordX	
122	G58_CurrentcoordY	
123	G58_CurrentcoordZ	
124	G58_CurrentcoordA	
125	G58_CurrentcoordB	
126	G58_CurrentcoordC	
127	G59_CurrentcoordX	
128	G59_CurrentcoordY	
129	G59_CurrentcoordZ	
130	G59_CurrentcoordA	
131	G59_CurrentcoordB	
132	G59_CurrentcoordC	
133	G54_WorkoffsetX	
134	G54_WorkoffsetY	
135	G54_WorkoffsetZ	
136	G54_WorkoffsetA	
137	G54_WorkoffsetB	
138	G54_WorkoffsetC	
139	G55_WorkoffsetX	
140	G55_WorkoffsetY	
141	G55_WorkoffsetZ	
142	G55_WorkoffsetA	
143	G55_WorkoffsetB	
144	G55_WorkoffsetC	
145	G56_WorkoffsetX	
146	G56_WorkoffsetY	
147	G56_WorkoffsetZ	
148	G56_WorkoffsetA	
149	G56_WorkoffsetB	
150	G56_WorkoffsetC	
151	G57_WorkoffsetX	
152	G57_WorkoffsetY	
153	G57_WorkoffsetZ	
154	G57_WorkoffsetA	
155	G57_WorkoffsetB	
156	G57_WorkoffsetC	
157	G58_WorkoffsetX	
158	G58_WorkoffsetY	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
159	G58_WorkoffsetZ	
160	G58_WorkoffsetA	
161	G58_WorkoffsetB	
162	G58_WorkoffsetC	
163	G59_WorkoffsetX	
164	G59_WorkoffsetY	
165	G59_WorkoffsetZ	
166	G59_WorkoffsetA	
167	G59_WorkoffsetB	
168	G59_WorkoffsetC	
169	TooloffsetZ	
170	PWMspindle_PWMpin	
171	PWMspindle_dirpin	
172	PWMspindle_PWMfrequency	
173	Stepdirspindle_Steppin	
174	Stepdirspindle_Dirpin	
175	Stepdirspindle_Stepsperrotation	
176	Stepdirspindle_Acceleration	
177	Spindle_Minvelocity	
178	Spindle_Maxvelocity	
179	Spindle_M3relaypin	
180	Spindle_M4relaypin	
181	Spindle_M3delayon	
182	Spindle_M3delayoff	
183	Spindle_M4delayon	
184	Spindle_M4delayoff	
185	Spindle_M7relaypin	
186	Spindle_M8relaypin	
187	Spindle_M7delayon	
188	Spindle_M8delayon	
189	Spindle_M9delay	
190	Comm_buffer_size	
191	CV_stopangledegrees	
192	CV_Lookaheadlines	
193	CV_Linearerrormax	
194	CV_Cornererrormax	
195	PositionDROsdigits	
196	ToolZoffset1	
197	ToolZoffset2	
198	ToolZoffset3	
199	ToolZoffset4	
200	ToolZoffset5	
201	ToolZoffset6	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
202	ToolZoffset7	
203	ToolZoffset8	
204	ToolZoffset9	
205	ToolZoffset10	
206	ToolZoffset11	
207	ToolZoffset12	
208	ToolZoffset13	
209	ToolZoffset14	
210	ToolZoffset15	
211	ToolZoffset16	
212	ToolZoffset17	
213	ToolZoffset18	
214	ToolZoffset19	
215	ToolZoffset20	
216	Newprofilename	
217	CV_Linearadditionlenght	
218	CV_Linearunifylength	
219	THC_onpin	
220	THC_uppin	
221	THC_downpin	
222	THC_min_height	
223	THC_max_height	
224	THC_feedrate	
225	SafeZheight	
226	XposDRO	
227	YposDRO	
228	ZposDRO	
229	AposDRO	
230	BposDRO	
231	CposDRO	
232	FRODRO	
233	SRODRO	
234	THC_ondelay	
235	G73backoff	
236	R*	
237	R*	
238	R*	
239	R*	
240	R*	
241	Xaxisstepport	
242	Xaxisdirport	
243	Xaxislimitminusport	
244	Xaxislimitplusport	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
245	Xaxishomeport	
246	Yaxisstepport	
247	Yaxisdirport	
248	Yaxislimitminusport	
249	Yaxislimitplusport	
250	Yaxishomeport	
251	Zaxisstepport	
252	Zaxisdirport	
253	Zaxislimitminusport	
254	Zaxislimitplusport	
255	Zaxishomeport	
256	Aaxisstepport	
257	Aaxisdirport	
258	Aaxislimitminusport	
259	Aaxislimitplusport	
260	Aaxishomeport	
261	Baxisstepport	
262	Baxisdirport	
263	Baxislimitminusport	
264	Baxislimitplusport	
265	Baxishomeport	
266	Caxisstepport	
267	Caxisdirport	
268	Caxislimitminusport	
269	Caxislimitplusport	
270	Caxishomeport	
271	Estopport	
272	Probeport1	
273	Indexport	
274	Chargepumpport	
275	-	
276	THC_onport	
277	THC_upport	
278	THC_downport	
279	PWMspindle_PWMport	
280	PWMspindle_dirport	
281	Stepdirspindle_Stepport	
282	Stepdirspindle_Dirport	
283	Spindle_M3relayport	
284	Spindle_M4relayport	
285	Spindle_M7relayport	
286	Spindle_M8relayport	
287	MPGpinA	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
288	MPGportA	
289	MPGpinB	
290	MPGportB	
291	Inputtrigger1pin	
292	Inputtrigger1port	
293	Inputtrigger1function	
294	Inputtrigger2pin	
295	Inputtrigger2port	
296	Inputtrigger2function	
297	Inputtrigger3pin	
298	Inputtrigger3port	
299	Inputtrigger3function	
300	Inputtrigger4pin	
301	Inputtrigger4port	
302	Inputtrigger4function	
303	Inputtrigger5pin	
304	Inputtrigger5port	
305	Inputtrigger5function	
306	Inputtrigger6pin	
307	Inputtrigger6port	
308	Inputtrigger6function	
309	Inputtrigger7pin	
310	Inputtrigger7port	
311	Inputtrigger7function	
312	Inputtrigger8pin	
313	Inputtrigger8port	
314	Inputtrigger8function	
315	Inputtrigger9pin	
316	Inputtrigger9port	
317	Inputtrigger9function	
318	Inputtrigger10pin	
319	Inputtrigger10port	
320	Inputtrigger10function	
321	Inputtrigger11pin	
322	Inputtrigger11port	
323	Inputtrigger11function	
324	Inputtrigger12pin	
325	Inputtrigger12port	
326	Inputtrigger12function	
327	Inputtrigger13pin	
328	Inputtrigger13port	
329	Inputtrigger13function	
330	Inputtrigger14pin	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
331	Inputtrigger14port	
332	Inputtrigger14function	
333	Inputtrigger15pin	
334	Inputtrigger15port	
335	Inputtrigger15function	
336	Inputtrigger16pin	
337	Inputtrigger16port	
338	Inputtrigger16function	
339	Inputtrigger17pin	
340	Inputtrigger17port	
341	Inputtrigger17function	
342	Inputtrigger18pin	
343	Inputtrigger18port	
344	Inputtrigger18function	
345	Inputtrigger19pin	
346	Inputtrigger19port	
347	Inputtrigger19function	
348	Inputtrigger20pin	
349	Inputtrigger20port	
350	Inputtrigger20function	
351	Inputtrigger21pin	
352	Inputtrigger21port	
353	Inputtrigger21function	
354	Inputtrigger22pin	
355	Inputtrigger22port	
356	Inputtrigger22function	
357	Inputtrigger23pin	
358	Inputtrigger23port	
359	Inputtrigger23function	
360	Inputtrigger24pin	
361	Inputtrigger24port	
362	Inputtrigger24function	
363	Inputtrigger25pin	
364	Inputtrigger25port	
365	Inputtrigger25function	
366	Inputtrigger26pin	
367	Inputtrigger26port	
368	Inputtrigger26function	
369	Inputtrigger27pin	
370	Inputtrigger27port	
371	Inputtrigger27function	
372	Inputtrigger28pin	
373	Inputtrigger28port	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
374	Inputtrigger28function	
375	Inputtrigger29pin	
376	Inputtrigger29port	
377	Inputtrigger29function	
378	Inputtrigger30pin	
379	Inputtrigger30port	
380	Inputtrigger30function	
381	Inputtrigger31pin	
382	Inputtrigger31port	
383	Inputtrigger31function	
384	Inputtrigger32pin	
385	Inputtrigger32port	
386	Inputtrigger32function	
387	Inputtrigger33pin	
388	Inputtrigger33port	
389	Inputtrigger33function	
390	Inputtrigger34pin	
391	Inputtrigger34port	
392	Inputtrigger34function	
393	Inputtrigger35pin	
394	Inputtrigger35port	
395	Inputtrigger35function	
396	Inputtrigger36pin	
397	Inputtrigger36port	
398	Inputtrigger36function	
399	Inputtrigger37pin	
400	Inputtrigger37port	
401	Inputtrigger37function	
402	Inputtrigger38pin	
403	Inputtrigger38port	
404	Inputtrigger38function	
405	Inputtrigger39pin	
406	Inputtrigger39port	
407	Inputtrigger39function	
408	Inputtrigger40pin	
409	Inputtrigger40port	
410	Inputtrigger40function	
411	Inputtrigger41pin	
412	Inputtrigger41port	
413	Inputtrigger41function	
414	Inputtrigger42pin	
415	Inputtrigger42port	
416	Inputtrigger42function	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
417	Inputtrigger43pin	
418	Inputtrigger43port	
419	Inputtrigger43function	
420	Inputtrigger44pin	
421	Inputtrigger44port	
422	Inputtrigger44function	
423	Inputtrigger45pin	
424	Inputtrigger45port	
425	Inputtrigger45function	
426	Inputtrigger46pin	
427	Inputtrigger46port	
428	Inputtrigger46function	
429	Inputtrigger47pin	
430	Inputtrigger47port	
431	Inputtrigger47function	
432	Inputtrigger48pin	
433	Inputtrigger48port	
434	Inputtrigger48function	
435 - 442	R*	
443	MPGprescaler	
444	Analoginput1var	
445	Analoginput2var	
446	R*	
447	FROanalogchannelnumber	
448	FROanalogminpercent	
449	FROanalogmaxpercent	
450	SROanalogchannelnumber	
451	SROanalogminpercent	
452	SROanalogmaxpercent	
453	JROanalogchannelnumber	
454	JROanalogminpercent	
455	JROanalogmaxpercent	
456	MPGfilterconstant	
457	MPGspeedmultiplier	
458	PWMspindle_PWMmindutycycle	
459	PWMspindle_PWMmaxdutycycle	
460	Analogoutput1var	
461	Analogoutput2var	
462	SpindlePWM_analogoutputchanne l	
463	Xaxisenablepin	
464	Xaxisenableport	
465	Yaxisenablepin	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
466	Yaxisenableport	
467	Zaxisenablepin	
468	Zaxisenableport	
469	Aaxisenablepin	
470	Aaxisenableport	
471	Baxisenablepin	
472	Baxisenableport	
473	Caxisenablepin	
474	Caxisenableport	
475 - 499	R*	
500	G92_offsetX	
501	G92_offsetY	
502	G92_offsetZ	
503	G92_offsetA	
504	G92_offsetB	
505	G92_offsetC	
506	Hotkey_keycode1	
507	Hotkey_function1	
508	Hotkey_keycode2	
509	Hotkey_function2	
510	Hotkey_keycode3	
511	Hotkey_function3	
512	Hotkey_keycode4	
513	Hotkey_function4	
514	Hotkey_keycode5	
515	Hotkey_function5	
516	Hotkey_keycode6	
517	Hotkey_function6	
518	Hotkey_keycode7	
519	Hotkey_function7	
520	Hotkey_keycode8	
521	Hotkey_function8	
522	Hotkey_keycode9	
523	Hotkey_function9	
524	Hotkey_keycode10	
525	Hotkey_function10	
526	Hotkey_keycode11	
527	Hotkey_function11	
528	Hotkey_keycode12	
529	Hotkey_function12	
530	Hotkey_keycode13	
531	Hotkey_function13	
532	Hotkey_keycode14	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
533	Hotkey_function14	
534	Hotkey_keycode15	
535	Hotkey_function15	
536	Hotkey_keycode16	
537	Hotkey_function16	
538	Hotkey_keycode17	
539	Hotkey_function17	
540	Hotkey_keycode18	
541	Hotkey_function18	
542	Hotkey_keycode19	
543	Hotkey_function19	
544	Hotkey_keycode20	
545	Hotkey_function20	
546	Hotkey_keycode21	
547	Hotkey_function21	
548	Hotkey_keycode22	
549	Hotkey_function22	
550	Hotkey_keycode23	
551	Hotkey_function23	
552	Hotkey_keycode24	
553	Hotkey_function24	
554	Hotkey_keycode25	
555	Hotkey_function25	
556	Hotkey_keycode26	
557	Hotkey_function26	
558	Hotkey_keycode27	
559	Hotkey_function27	
560	Hotkey_keycode28	
561	Hotkey_function28	
562	Hotkey_keycode29	
563	Hotkey_function29	
564	Hotkey_keycode30	
565	Hotkey_function30	
566	Hotkey_keycode31	
567	Hotkey_function31	
568	Hotkey_keycode32	
569	Hotkey_function32	
570	Hotkey_keycode33	
571	Hotkey_function33	
572	Hotkey_keycode34	
573	Hotkey_function34	
574	Hotkey_keycode35	
575	Hotkey_function35	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
576	Hotkey_keycode36	
577	Hotkey_function36	
578	Hotkey_keycode37	
579	Hotkey_function37	
580	Hotkey_keycode38	
581	Hotkey_function38	
582	Hotkey_keycode39	
583	Hotkey_function39	
584	Hotkey_keycode40	
585	Hotkey_function40	
586	Hotkey_keycode41	
587	Hotkey_function41	
588	Hotkey_keycode42	
589	Hotkey_function42	
590	Hotkey_keycode43	
591	Hotkey_function43	
592	Hotkey_keycode44	
593	Hotkey_function44	
594	Hotkey_keycode45	
595	Hotkey_function45	
596	Hotkey_keycode46	
597	Hotkey_function46	
598	Hotkey_keycode47	
599	Hotkey_function47	
600	Hotkey_keycode48	
601	Hotkey_function48	
602	Hotkey_keycode49	
603	Hotkey_function49	
604	Hotkey_keycode50	
605	Hotkey_function50	
606	Hotkey_keycode51	
607	Hotkey_function51	
608	Hotkey_keycode52	
609	Hotkey_function52	
610	Hotkey_keycode53	
611	Hotkey_function53	
612	Hotkey_keycode54	
613	Hotkey_function54	
614	Hotkey_keycode55	
615	Hotkey_function55	
616	Hotkey_keycode56	
617	Hotkey_function56	
618	Hotkey_keycode57	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
619	Hotkey_function57	
620	Hotkey_keycode58	
621	Hotkey_function58	
622	Hotkey_keycode59	
623	Hotkey_function59	
624	Hotkey_keycode60	
625	Hotkey_function60	
626	Hotkey_keycode61	
627	Hotkey_function61	
628	Hotkey_keycode62	
629	Hotkey_function62	
630	Hotkey_keycode63	
631	Hotkey_function63	
632	Hotkey_keycode64	
633	Hotkey_function64	
634	Hotkey_keycode65	
635	Hotkey_function65	
636	Hotkey_keycode66	
637	Hotkey_function66	
638	Hotkey_keycode67	
639	Hotkey_function67	
640	Hotkey_keycode68	
641	Hotkey_function68	
642	Hotkey_keycode69	
643	Hotkey_function69	
644	Hotkey_keycode70	
645	Hotkey_function70	
646	Hotkey_keycode71	
647	Hotkey_function71	
648	Hotkey_keycode72	
649	Hotkey_function72	
650	Hotkey_keycode73	
651	Hotkey_function73	
652	Hotkey_keycode74	
653	Hotkey_function74	
654	Hotkey_keycode75	
655	Hotkey_function75	
656	Hotkey_keycode76	
657	Hotkey_function76	
658	Hotkey_keycode77	
659	Hotkey_function77	
660	Hotkey_keycode78	
661	Hotkey_function78	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
662	Hotkey_keycode79	
663	Hotkey_function79	
664	Hotkey_keycode80	
665	Hotkey_function80	
666	Hotkey_keycode81	
667	Hotkey_function81	
668	Hotkey_keycode82	
669	Hotkey_function82	
670	Hotkey_keycode83	
671	Hotkey_function83	
672	Hotkey_keycode84	
673	Hotkey_function84	
674	Hotkey_keycode85	
675	Hotkey_function85	
676	Hotkey_keycode86	
677	Hotkey_function86	
678	Hotkey_keycode87	
679	Hotkey_function87	
680	Hotkey_keycode88	
681	Hotkey_function88	
682	Hotkey_keycode89	
683	Hotkey_function89	
684	Hotkey_keycode90	
685	Hotkey_function90	
686	Hotkey_keycode91	
687	Hotkey_function91	
688	Hotkey_keycode92	
689	Hotkey_function92	
690	Hotkey_keycode93	
691	Hotkey_function93	
692	Hotkey_keycode94	
693	Hotkey_function94	
694	Hotkey_keycode95	
695	Hotkey_function95	
696	Hotkey_keycode96	
697	Hotkey_function96	
698 - 699	R*	
700	Outputtrigger1pin	
701	Outputtrigger1port	
702	Outputtrigger1LED	
703	Outputtrigger2pin	
704	Outputtrigger2port	
705	Outputtrigger2LED	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
706	Outputtrigger3pin	
707	Outputtrigger3port	
708	Outputtrigger3LED	
709	Outputtrigger4pin	
710	Outputtrigger4port	
711	Outputtrigger4LED	
712	Outputtrigger5pin	
713	Outputtrigger5port	
714	Outputtrigger5LED	
715	Outputtrigger6pin	
716	Outputtrigger6port	
717	Outputtrigger6LED	
718	Outputtrigger7pin	
719	Outputtrigger7port	
720	Outputtrigger7LED	
721	Outputtrigger8pin	
722	Outputtrigger8port	
723	Outputtrigger8LED	
724	Outputtrigger9pin	
725	Outputtrigger9port	
726	Outputtrigger9LED	
727	Outputtrigger10pin	
728	Outputtrigger10port	
729	Outputtrigger10LED	
730	Outputtrigger11pin	
731	Outputtrigger11port	
732	Outputtrigger11LED	
733	Outputtrigger12pin	
734	Outputtrigger12port	
735	Outputtrigger12LED	
736	Outputtrigger13pin	
737	Outputtrigger13port	
738	Outputtrigger13LED	
739	Outputtrigger14pin	
740	Outputtrigger14port	
741	Outputtrigger14LED	
742	Outputtrigger15pin	
743	Outputtrigger15port	
744	Outputtrigger15LED	
745	Outputtrigger16pin	
746	Outputtrigger16port	
747	Outputtrigger16LED	
748	Outputtrigger17pin	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
749	Outputtrigger17port	
750	Outputtrigger17LED	
751	Outputtrigger18pin	
752	Outputtrigger18port	
753	Outputtrigger18LED	
754	Outputtrigger19pin	
755	Outputtrigger19port	
756	Outputtrigger19LED	
757	Outputtrigger20pin	
758	Outputtrigger20port	
759	Outputtrigger20LED	
760	Outputtrigger21pin	
761	Outputtrigger21port	
762	Outputtrigger21LED	
763	Outputtrigger22pin	
764	Outputtrigger22port	
765	Outputtrigger22LED	
766	Outputtrigger23pin	
767	Outputtrigger23port	
768	Outputtrigger23LED	
769	Outputtrigger24pin	
770	Outputtrigger24port	
771	Outputtrigger24LED	
772	Outputtrigger25pin	
773	Outputtrigger25port	
774	Outputtrigger25LED	
775	Outputtrigger26pin	
776	Outputtrigger26port	
777	Outputtrigger26LED	
778	Outputtrigger27pin	
779	Outputtrigger27port	
780	Outputtrigger27LED	
781	Outputtrigger28pin	
782	Outputtrigger28port	
783	Outputtrigger28LED	
784	Outputtrigger29pin	
785	Outputtrigger29port	
786	Outputtrigger29LED	
787	Outputtrigger30pin	
788	Outputtrigger30port	
789	Outputtrigger30LED	
790	Outputtrigger31pin	
791	Outputtrigger31port	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
792	Outputtrigger31LED	
793	Outputtrigger32pin	
794	Outputtrigger32port	
795	Outputtrigger32LED	
796	Outputtrigger33pin	
797	Outputtrigger33port	
798	Outputtrigger33LED	
799	Outputtrigger34pin	
800	Outputtrigger34port	
801	Outputtrigger34LED	
802	Outputtrigger35pin	
803	Outputtrigger35port	
804	Outputtrigger35LED	
805	Outputtrigger36pin	
806	Outputtrigger36port	
807	Outputtrigger36LED	
808	Outputtrigger37pin	
809	Outputtrigger37port	
810	Outputtrigger37LED	
811	Outputtrigger38pin	
812	Outputtrigger38port	
813	Outputtrigger38LED	
814	Outputtrigger39pin	
815	Outputtrigger39port	
816	Outputtrigger39LED	
817	Outputtrigger40pin	
818	Outputtrigger40port	
819	Outputtrigger40LED	
820	Outputtrigger41pin	
821	Outputtrigger41port	
822	Outputtrigger41LED	
823	Outputtrigger42pin	
824	Outputtrigger42port	
825	Outputtrigger42LED	
826	Outputtrigger43pin	
827	Outputtrigger43port	
828	Outputtrigger43LED	
829	Outputtrigger44pin	
830	Outputtrigger44port	
831	Outputtrigger44LED	
832	Outputtrigger45pin	
833	Outputtrigger45port	
834	Outputtrigger45LED	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
835	Outputtrigger46pin	
836	Outputtrigger46port	
837	Outputtrigger46LED	
838	Outputtrigger47pin	
839	Outputtrigger47port	
840	Outputtrigger47LED	
841	Outputtrigger48pin	
842	Outputtrigger48port	
843	Outputtrigger48LED	
844 - 851	R*	
852	EncoderApin	
853	EncoderAport	
854	EncoderBpin	
855	EncoderBport	
856	CAM_Tooldia	
857	CAM_Startdepth	
858	CAM_Cutdepth	
859	CAM_Cutperpass	
860	CAM_SafeZ	
861	CAM_Feedrate	
862	CAM_Tooloverlap	
863	CAM_Plungerate	
864	EncoderPPR	
865	CAM_Spindlespeed	
866	Setnextlinefield	
867	Setfeedrate	
868	Actfeedrate	
869	Setspindlespeed	
870	Actspindlespeed	
871	MachinecoordX	
872	MachinecoordY	
873	MachinecoordZ	
874	MachinecoordA	
875	MachinecoordB	
876	MachinecoordC	
877	Activemodal	
878	IOmonitor_XPOS	
879	IOmonitor_YPOS	
880	IOmonitor_ZPOS	
881	IOmonitor_APOS	
882	IOmonitor_BPOS	
883	IOmonitor_CPOS	
884	Motionbuffer	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
885	Diagnostics_minX	
886	Diagnostics_minY	
887	Diagnostics_minZ	
888	Diagnostics_maxX	
889	Diagnostics_maxY	
890	Diagnostics_maxZ	
891	Diagnostics_sizeX	
892	Diagnostics_sizeY	
893	Diagnostics_sizeZ	
894	Diagnostics_totalnumberofobjects	
895	Diagnostics_Filename	
896	Dwell_time	
897	Active_toolnumber	
898	Worktimer	
899	Active_fixture	
900	Profile_name	
901	Licensed_to	
902	Softwareversion	
903	Firmwareversion	
904	Hardwareversion	
905	APIversion	
906	Serialnumber	
907	Devicetype	
908	CAM_Statuslabel	
909	Analog_inputvalue1	
910	Analog_inputvalue2	
911	Analog_outputvalue1	
912	Analog_outputvalue2	
913	Jogfeedrate	
914	Laseroutputpin	
915	Laseroutputport	
916 - 920	R*	
921	ToolZoffset21	
922	ToolZoffset22	
923	ToolZoffset23	
924	ToolZoffset24	
925	ToolZoffset25	
926	ToolZoffset26	
927	ToolZoffset27	
928	ToolZoffset28	
929	ToolZoffset29	
930	ToolZoffset30	
931	ToolZoffset31	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
932	ToolZoffset32	
933	ToolZoffset33	
934	ToolZoffset34	
935	ToolZoffset35	
936	ToolZoffset36	
937	ToolZoffset37	
938	ToolZoffset38	
939	ToolZoffset39	
940	ToolZoffset40	
941	ToolZoffset41	
942	ToolZoffset42	
943	ToolZoffset43	
944	ToolZoffset44	
945	ToolZoffset45	
946	ToolZoffset46	
947	ToolZoffset47	
948	ToolZoffset48	
949	ToolZoffset49	
950	ToolZoffset50	
951	ToolZoffset51	
952	ToolZoffset52	
953	ToolZoffset53	
954	ToolZoffset54	
955	ToolZoffset55	
956	ToolZoffset56	
957	ToolZoffset57	
958	ToolZoffset58	
959	ToolZoffset59	
960	ToolZoffset60	
961	ToolZoffset61	
962	ToolZoffset62	
963	ToolZoffset63	
964	ToolZoffset64	
965	ToolZoffset65	
966	ToolZoffset66	
967	ToolZoffset67	
968	ToolZoffset68	
969	ToolZoffset69	
970	ToolZoffset70	
971	ToolZoffset71	
972	ToolZoffset72	
973	ToolZoffset73	
974	ToolZoffset74	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
975	ToolZoffset75	
976	ToolZoffset76	
977	ToolZoffset77	
978	ToolZoffset78	
979	ToolZoffset79	
980	ToolZoffset80	
981	ToolZoffset81	
982	ToolZoffset82	
983	ToolZoffset83	
984	ToolZoffset84	
985	ToolZoffset85	
986	ToolZoffset86	
987	ToolZoffset87	
988	ToolZoffset88	
989	ToolZoffset89	
990	ToolZoffset90	
991	ToolZoffset91	
992	ToolZoffset92	
993	ToolZoffset93	
994	ToolZoffset94	
995	ToolZoffset95	
996	ToolZoffset96	
997 - 999	R*	
1000	MDIinput	
997 - 2000	R*	
2001	THCAntidivevelocitypercentage	
2002	Analog_inputvalue3	
2003	Analog_inputvalue4	
2004	Analoginput3var	
2005	Analoginput4var	
2006	Analogoutput3var	
2007	Analogoutput4var	
2008	Analog_outputvalue3	
2009	Analog_outputvalue4	
2010	Plasmapieceheight	
2011	Probepin2	
2012	Probeport2	
2013	Pulleynumber	
2014	Pulleyratio	
2015	Digitize_numberofdigits	
2016	Digitize_proberadius	
2017	Digitize_filename	
2018	Digitize_numberofstoredpoints	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2019	Chargepump2pin	
2020	Chargepump2port	
2021	THCenableoutputpin	
2022	THCenableoutputport	
2023	Antidiveoutputpin	
2024	Antidiveoutputport	
2025	Antidownoutputpin	
2026	Antidownoutputport	
2027	JogStepDistance	
2028-2099	R*	
2100	Inputtrigger49pin	
2101	Inputtrigger49port	
2102	Inputtrigger49function	
2103	Inputtrigger50pin	
2104	Inputtrigger50port	
2105	Inputtrigger50function	
2106	Inputtrigger51pin	
2107	Inputtrigger51port	
2108	Inputtrigger51function	
2109	Inputtrigger52pin	
2110	Inputtrigger52port	
2111	Inputtrigger52function	
2112	Inputtrigger53pin	
2113	Inputtrigger53port	
2114	Inputtrigger53function	
2115	Inputtrigger54pin	
2116	Inputtrigger54port	
2117	Inputtrigger54function	
2118	Inputtrigger55pin	
2119	Inputtrigger55port	
2120	Inputtrigger55function	
2121	Inputtrigger56pin	
2122	Inputtrigger56port	
2123	Inputtrigger56function	
2124	Inputtrigger57pin	
2125	Inputtrigger57port	
2126	Inputtrigger57function	
2127	Inputtrigger58pin	
2128	Inputtrigger58port	
2129	Inputtrigger58function	
2130	Inputtrigger59pin	
2131	Inputtrigger59port	
2132	Inputtrigger59function	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2133	Inputtrigger60pin	
2134	Inputtrigger60port	
2135	Inputtrigger60function	
2136	Inputtrigger61pin	
2137	Inputtrigger61port	
2138	Inputtrigger61function	
2139	Inputtrigger62pin	
2140	Inputtrigger62port	
2141	Inputtrigger62function	
2142	Inputtrigger63pin	
2143	Inputtrigger63port	
2144	Inputtrigger63function	
2145	Inputtrigger64pin	
2146	Inputtrigger64port	
2147	Inputtrigger64function	
2148	Inputtrigger65pin	
2149	Inputtrigger65port	
2150	Inputtrigger65function	
2151	Inputtrigger66pin	
2152	Inputtrigger66port	
2153	Inputtrigger66function	
2154	Inputtrigger67pin	
2155	Inputtrigger67port	
2156	Inputtrigger67function	
2157	Inputtrigger68pin	
2158	Inputtrigger68port	
2159	Inputtrigger68function	
2160	Inputtrigger69pin	
2161	Inputtrigger69port	
2162	Inputtrigger69function	
2163	Inputtrigger70pin	
2164	Inputtrigger70port	
2165	Inputtrigger70function	
2166	Inputtrigger71pin	
2167	Inputtrigger71port	
2168	Inputtrigger71function	
2169	Inputtrigger72pin	
2170	Inputtrigger72port	
2171	Inputtrigger72function	
2172	Inputtrigger73pin	
2173	Inputtrigger73port	
2174	Inputtrigger73function	
2175	Inputtrigger74pin	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2176	Inputtrigger74port	
2177	Inputtrigger74function	
2178	Inputtrigger75pin	
2179	Inputtrigger75port	
2180	Inputtrigger75function	
2181	Inputtrigger76pin	
2182	Inputtrigger76port	
2183	Inputtrigger76function	
2184	Inputtrigger77pin	
2185	Inputtrigger77port	
2186	Inputtrigger77function	
2187	Inputtrigger78pin	
2188	Inputtrigger78port	
2189	Inputtrigger78function	
2190	Inputtrigger79pin	
2191	Inputtrigger79port	
2192	Inputtrigger79function	
2193	Inputtrigger80pin	
2194	Inputtrigger80port	
2195	Inputtrigger80function	
2196	Inputtrigger81pin	
2197	Inputtrigger81port	
2198	Inputtrigger81function	
2199	Inputtrigger82pin	
2200	Inputtrigger82port	
2201	Inputtrigger82function	
2202	Inputtrigger83pin	
2203	Inputtrigger83port	
2204	Inputtrigger83function	
2205	Inputtrigger84pin	
2206	Inputtrigger84port	
2207	Inputtrigger84function	
2208	Inputtrigger85pin	
2209	Inputtrigger85port	
2210	Inputtrigger85function	
2211	Inputtrigger86pin	
2212	Inputtrigger86port	
2213	Inputtrigger86function	
2214	Inputtrigger87pin	
2215	Inputtrigger87port	
2216	Inputtrigger87function	
2217	Inputtrigger88pin	
2218	Inputtrigger88port	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2219	Inputtrigger88function	
2220	Inputtrigger89pin	
2221	Inputtrigger89port	
2222	Inputtrigger89function	
2223	Inputtrigger90pin	
2224	Inputtrigger90port	
2225	Inputtrigger90function	
2226	Inputtrigger91pin	
2227	Inputtrigger91port	
2228	Inputtrigger91function	
2229	Inputtrigger92pin	
2230	Inputtrigger92port	
2231	Inputtrigger92function	
2232	Inputtrigger93pin	
2233	Inputtrigger93port	
2234	Inputtrigger93function	
2235	Inputtrigger94pin	
2236	Inputtrigger94port	
2237	Inputtrigger94function	
2238	Inputtrigger95pin	
2239	Inputtrigger95port	
2240	Inputtrigger95function	
2241	Inputtrigger96pin	
2242	Inputtrigger96port	
2243	Inputtrigger96function	
2244-2249	R*	
2250	Outputtrigger49pin	
2251	Outputtrigger49port	
2252	Outputtrigger49LED	
2253	Outputtrigger50pin	
2254	Outputtrigger50port	
2255	Outputtrigger50LED	
2256	Outputtrigger51pin	
2257	Outputtrigger51port	
2258	Outputtrigger51LED	
2259	Outputtrigger52pin	
2260	Outputtrigger52port	
2261	Outputtrigger52LED	
2262	Outputtrigger53pin	
2263	Outputtrigger53port	
2264	Outputtrigger53LED	
2265	Outputtrigger54pin	
2266	Outputtrigger54port	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2267	Outputtrigger54LED	
2268	Outputtrigger55pin	
2269	Outputtrigger55port	
2270	Outputtrigger55LED	
2271	Outputtrigger56pin	
2272	Outputtrigger56port	
2273	Outputtrigger56LED	
2274	Outputtrigger57pin	
2275	Outputtrigger57port	
2276	Outputtrigger57LED	
2277	Outputtrigger58pin	
2278	Outputtrigger58port	
2279	Outputtrigger58LED	
2280	Outputtrigger59pin	
2281	Outputtrigger59port	
2282	Outputtrigger59LED	
2283	Outputtrigger60pin	
2284	Outputtrigger60port	
2285	Outputtrigger60LED	
2286	Outputtrigger61pin	
2287	Outputtrigger61port	
2288	Outputtrigger61LED	
2289	Outputtrigger62pin	
2290	Outputtrigger62port	
2291	Outputtrigger62LED	
2292	Outputtrigger63pin	
2293	Outputtrigger63port	
2294	Outputtrigger63LED	
2295	Outputtrigger64pin	
2296	Outputtrigger64port	
2297	Outputtrigger64LED	
2298	Outputtrigger65pin	
2299	Outputtrigger65port	
2300	Outputtrigger65LED	
2301	Outputtrigger66pin	
2302	Outputtrigger66port	
2303	Outputtrigger66LED	
2304	Outputtrigger67pin	
2305	Outputtrigger67port	
2306	Outputtrigger67LED	
2307	Outputtrigger68pin	
2308	Outputtrigger68port	
2309	Outputtrigger68LED	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2310	Outputtrigger69pin	
2311	Outputtrigger69port	
2312	Outputtrigger69LED	
2313	Outputtrigger70pin	
2314	Outputtrigger70port	
2315	Outputtrigger70LED	
2316	Outputtrigger71pin	
2317	Outputtrigger71port	
2318	Outputtrigger71LED	
2319	Outputtrigger72pin	
2320	Outputtrigger72port	
2321	Outputtrigger72LED	
2322	Outputtrigger73pin	
2323	Outputtrigger73port	
2324	Outputtrigger73LED	
2325	Outputtrigger74pin	
2326	Outputtrigger74port	
2327	Outputtrigger74LED	
2328	Outputtrigger75pin	
2329	Outputtrigger75port	
2330	Outputtrigger75LED	
2331	Outputtrigger76pin	
2332	Outputtrigger76port	
2333	Outputtrigger76LED	
2334	Outputtrigger77pin	
2335	Outputtrigger77port	
2336	Outputtrigger77LED	
2337	Outputtrigger78pin	
2338	Outputtrigger78port	
2339	Outputtrigger78LED	
2340	Outputtrigger79pin	
2341	Outputtrigger79port	
2342	Outputtrigger79LED	
2343	Outputtrigger80pin	
2344	Outputtrigger80port	
2345	Outputtrigger80LED	
2346	Outputtrigger81pin	
2347	Outputtrigger81port	
2348	Outputtrigger81LED	
2349	Outputtrigger82pin	
2350	Outputtrigger82port	
2351	Outputtrigger82LED	
2352	Outputtrigger83pin	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2353	Outputtrigger83port	
2354	Outputtrigger83LED	
2355	Outputtrigger84pin	
2356	Outputtrigger84port	
2357	Outputtrigger84LED	
2358	Outputtrigger85pin	
2359	Outputtrigger85port	
2360	Outputtrigger85LED	
2361	Outputtrigger86pin	
2362	Outputtrigger86port	
2363	Outputtrigger86LED	
2364	Outputtrigger87pin	
2365	Outputtrigger87port	
2366	Outputtrigger87LED	
2367	Outputtrigger88pin	
2368	Outputtrigger88port	
2369	Outputtrigger88LED	
2370	Outputtrigger89pin	
2371	Outputtrigger89port	
2372	Outputtrigger89LED	
2373	Outputtrigger90pin	
2374	Outputtrigger90port	
2375	Outputtrigger90LED	
2376	Outputtrigger91pin	
2377	Outputtrigger91port	
2378	Outputtrigger91LED	
2379	Outputtrigger92pin	
2380	Outputtrigger92port	
2381	Outputtrigger92LED	
2382	Outputtrigger93pin	
2383	Outputtrigger93port	
2384	Outputtrigger93LED	
2385	Outputtrigger94pin	
2386	Outputtrigger94port	
2387	Outputtrigger94LED	
2388	Outputtrigger95pin	
2389	Outputtrigger95port	
2390	Outputtrigger95LED	
2391	Outputtrigger96pin	
2392	Outputtrigger96port	
2393	Outputtrigger96LED	
2394-2399	R*	
2400	AUXencoder1pinA	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2401	AUXencoder1portA	
2402	AUXencoder1pinB	
2403	AUXencoder1portB	
2404	AUXencoder2pinA	
2405	AUXencoder2portA	
2406	AUXencoder2pinB	
2407	AUXencoder2portB	
2408	AUXencoder3pinA	
2409	AUXencoder3portA	
2410	AUXencoder3pinB	
2411	AUXencoder3portB	
2412	AUXencoder4pinA	
2413	AUXencoder4portA	
2414	AUXencoder4pinB	
2415	AUXencoder4portB	
2416	AUXencoder5pinA	
2417	AUXencoder5portA	
2418	AUXencoder5pinB	
2419	AUXencoder5portB	
2420	AUXencoder6pinA	
2421	AUXencoder6portA	
2422	AUXencoder6pinB	
2423	AUXencoder6portB	
2424	AUXencoder1countsper	
2425	AUXencoder2countsper	
2426	AUXencoder3countsper	
2427	AUXencoder4countsper	
2428	AUXencoder5countsper	
2429	AUXencoder6countsper	
2430	AUXencoder1position	
2431	AUXencoder2position	
2432	AUXencoder3position	
2433	AUXencoder4position	
2434	AUXencoder5position	
2435	AUXencoder6position	
2436	SpindlePIDoutputpercentvalue	
2437	Arcradiustolerance	
2438	ScaleX	
2439	ScaleY	
2440	ScaleZ	
2441	ScaleA	
2442	ScaleB	
2443	ScaleC	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2444	DistanceToGoX	
2445	DistanceToGoY	
2446	DistanceToGoZ	
2447	DistanceToGoA	
2448	DistanceToGoB	
2449	DistanceToGoC	
2450	Feedrateoverridden	
2451	Spindlespeedoverridden	
2452-2500	R*	
2501	ToolDiameter1	
2502	ToolDiameter2	
2503	ToolDiameter3	
2504	ToolDiameter4	
2505	ToolDiameter5	
2506	ToolDiameter6	
2507	ToolDiameter7	
2508	ToolDiameter8	
2509	ToolDiameter9	
2510	ToolDiameter10	
2511	ToolDiameter11	
2512	ToolDiameter12	
2513	ToolDiameter13	
2514	ToolDiameter14	
2515	ToolDiameter15	
2516	ToolDiameter16	
2517	ToolDiameter17	
2518	ToolDiameter18	
2519	ToolDiameter19	
2520	ToolDiameter20	
2521	ToolDiameter21	
2522	ToolDiameter22	
2523	ToolDiameter23	
2524	ToolDiameter24	
2525	ToolDiameter25	
2526	ToolDiameter26	
2527	ToolDiameter27	
2528	ToolDiameter28	
2529	ToolDiameter29	
2530	ToolDiameter30	
2531	ToolDiameter31	
2532	ToolDiameter32	
2533	ToolDiameter33	
2534	ToolDiameter34	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2535	ToolDiameter35	
2536	ToolDiameter36	
2537	ToolDiameter37	
2538	ToolDiameter38	
2539	ToolDiameter39	
2540	ToolDiameter40	
2541	ToolDiameter41	
2542	ToolDiameter42	
2543	ToolDiameter43	
2544	ToolDiameter44	
2545	ToolDiameter45	
2546	ToolDiameter46	
2547	ToolDiameter47	
2548	ToolDiameter48	
2549	ToolDiameter49	
2550	ToolDiameter50	
2551	ToolDiameter51	
2552	ToolDiameter52	
2553	ToolDiameter53	
2554	ToolDiameter54	
2555	ToolDiameter55	
2556	ToolDiameter56	
2557	ToolDiameter57	
2558	ToolDiameter58	
2559	ToolDiameter59	
2560	ToolDiameter60	
2561	ToolDiameter61	
2562	ToolDiameter62	
2563	ToolDiameter63	
2564	ToolDiameter64	
2565	ToolDiameter65	
2566	ToolDiameter66	
2567	ToolDiameter67	
2568	ToolDiameter68	
2569	ToolDiameter69	
2570	ToolDiameter70	
2571	ToolDiameter71	
2572	ToolDiameter72	
2573	ToolDiameter73	
2574	ToolDiameter74	
2575	ToolDiameter75	
2576	ToolDiameter76	
2577	ToolDiameter77	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2578	ToolDiameter78	
2579	ToolDiameter79	
2580	ToolDiameter80	
2581	ToolDiameter81	
2582	ToolDiameter82	
2583	ToolDiameter83	
2584	ToolDiameter84	
2585	ToolDiameter85	
2586	ToolDiameter86	
2587	ToolDiameter87	
2588	ToolDiameter88	
2589	ToolDiameter89	
2590	ToolDiameter90	
2591	ToolDiameter91	
2592	ToolDiameter92	
2593	ToolDiameter93	
2594	ToolDiameter94	
2595	ToolDiameter95	
2596	ToolDiameter96	
2597-2599	R*	
2600	Xenabledelay	
2601	Yenabledelay	
2602	Zenabledelay	
2603	Aenabledelay	
2604	Benabledelay	
2605	Cenabledelay	
2606	Xcurrenthilowpin	
2607	Ycurrenthilowpin	
2608	Zcurrenthilowpin	
2609	Acurrenthilowpin	
2610	Bcurrenthilowpin	
2611	Ccurrenthilowpin	
2612	Xcurrenthilowport	
2613	Ycurrenthilowport	
2614	Zcurrenthilowport	
2615	Acurrenthilowport	
2616	Bcurrenthilowport	
2617	Ccurrenthilowport	
2618	Debounce	
2619	DebounceTHC	
2620	DebounceHome	
2621	Xhomebackoff	
2622	Yhomebackoff	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2623	Zhomebackoff	
2624	Ahomebackoff	
2625	Bhomebackoff	
2626	Chomebackoff	
2627	Digitalsyncoutputpin1	
2628	Digitalsyncoutputport1	
2629	Digitalsyncoutputpin2	
2630	Digitalsyncoutputport2	
2631	Digitalsyncoutputpin3	
2632	Digitalsyncoutputport3	
2633	Digitalsyncoutputpin4	
2634	Digitalsyncoutputport4	
2635	Digitalsyncoutputpin5	
2636	Digitalsyncoutputport5	
2637	Digitalsyncoutputpin6	
2638	Digitalsyncoutputport6	
2639	Digitalsyncoutputpin7	
2640	Digitalsyncoutputport7	
2641	Digitalsyncoutputpin8	
2642	Digitalsyncoutputport8	
2643	Digitalsyncoutputpin9	
2644	Digitalsyncoutputport9	
2645	Digitalsyncoutputpin10	
2646	Digitalsyncoutputport10	
2647	Dragknifestopatangle	
2648	Dragkniferetractatangle	
2649	Dragkniferetractheight	
2650	Dragknifemovebackfeedrate	
2651	Dragknifepulloutfeedrate	
2652	Changetool	
2653	Laststatusmessage	
2654	Debounceprobe1	
2655	Debounceprobe2	
2656	CameraoffsetX	
2657	CameraoffsetY	
2658	CameraZheight	
2659-2699	R*	
2700	ProbeStatus	
2701	Axis1P	
2702	Axis2P	
2703	Axis3P	
2704	GuageHeight	
2705	ProbeDiameter	

FIELD NUMBER	FUNCTION NAME	DESCRIPTION
2706	FineDistance	
2707	Retract	
2708	SafeXYZ	
2709	FastFeed	
2710	FineFeed	
2711	ResultsPos1P	
2712	ResultsPos1N	
2713	ResultsSize1	
2714	ResultsCenter1	
2715	ResultsAngle	
2716	ResultsPos2P	
2717	ResultsPos2N	
2718	ResultsSize2	
2719	ResultsCentre2	
2720	MobileProbePosX	
2721	MobileProbePosY	
2722	MobileProbePosZ	
2723	MobileProbePosA	
2724	MobileProbePosB	
2725	MobileProbePosC	
2726	FixedProbePosX	
2727	FixedProbePosY	
2728	FixedProbePosZ	
2729	FixedProbePosA	
2730	FixedProbePosB	
2731	FixedProbePosC	
2732	SafeX	
2733	SafeY	
2734	SafeZ	
2735	SafeXYZLabel	
2736	OverrideProbeDia	
2737	TraverseSpeedLimit	
2738	BlowerPort	
2739	BlowerPin	
2734	BlowTime	

R* = Reserved address, do not use this address.

LED's (SORT BY NUMBER)

This section describes the **UCCNC software** LED screen objects.

Each LED represents an internal boolean variable of the UCNC software.
The LEDs' logic state can be read in macro code.
This documentation lists all the accessible LED variables, sorted **numerically.**

LED NUMBER	LED NAME	DESCRIPTION
1	OutputPT1PN1	Indicates the actual logic state of port#1 pin#1.
2	OutputPT1PN2	Indicates the actual logic state of port#1 pin#2.
3	OutputPT1PN3	Indicates the actual logic state of port#1 pin#3.
4	OutputPT1PN4	Indicates the actual logic state of port#1 pin#4.
5	OutputPT1PN5	Indicates the actual logic state of port#1 pin#5.
6	OutputPT1PN6	Indicates the actual logic state of port#1 pin#6.
7	OutputPT1PN7	Indicates the actual logic state of port#1 pin#7.
8	OutputPT1PN8	Indicates the actual logic state of port#1 pin#8.
9	OutputPT1PN9	Indicates the actual logic state of port#1 pin#9.
10	OutputPT1PN10	Indicates the actual logic state of port#1 pin#10.
11	OutputPT1PN11	Indicates the actual logic state of port#1 pin#11.
12	OutputPT1PN12	Indicates the actual logic state of port#1 pin#12.
13	OutputPT1PN13	Indicates the actual logic state of port#1 pin#13.
14	OutputPT1PN14	Indicates the actual logic state of port#1 pin#14.
15	OutputPT1PN15	Indicates the actual logic state of port#1 pin#15.
16	OutputPT1PN16	Indicates the actual logic state of port#1 pin#16.
17	OutputPT1PN17	Indicates the actual logic state of port#1 pin#17.
18	Idle	Indicates the Idle state of the device.
19	Run	Indicates a run state of the device. This is the opposite of the Idle state.
20	Jog	Active when a jog command is being executed.
21	Dwell	Indicates a dwell in progress.
22	Backlash	Active when the backlash compensation is being executed.

LED NUMBER	LED NAME	DESCRIPTION
23	Home	Active when the homing command is being executed.
24	Probing	Active when the probing command is being executed.
25	Reset	Indicates an active reset signal.
26	Hardlimit	Active when any of the configured limit inputs are triggered.
27	Limitsoverride	Active if the limits are overriden by the user.
28	Toolchangeinprogress	Active when a tool change macro is being executed.
29	Xlimitpositive	Active when the X-axis positive side limit switch is triggered.
30	Ylimitpositive	Active when the Y-axis positive side limit switch is triggered.
31	Zlimitpositive	Active when the Z-axis positive side limit switch is triggered.
32	Alimitpositive	Active when the A-axis positive side limit switch is triggered.
33	Blimitpositive	Active when the B-axis positive side limit switch is triggered.
34	Climitpositive	Active when the C-axis positive side limit switch is triggered.
35	Index	Active when the index input is triggered.
36	Estop	Active when the estop input is triggered.
37	Probe	Active when the probe input is triggered.
38	Xlimitnegative	Active when the X-axis negative side limit switch is triggered.
39	Ylimitnegative	Active when the Y-axis negative side limit switch is triggered.
40	Zlimitnegative	Active when the Z-axis negative side limit switch is triggered.
41	Alimitnegative	Active when the A-axis negative side limit switch is triggered.
42	Blimitnegative	Active when the B-axis negative side limit switch is triggered.
43	Climitnegative	Active when the C-axis negative side limit switch is triggered.
44	Xhome	Active when then X-axis home input is triggered.
45	Yhome	Active when then Y-axis home input is triggered.
46	Zhome	Active when then Z-axis home input is triggered.
47	Ahome	Active when then A-axis home input is triggered.
48	Bhome	Active when then B-axis home input is triggered.
49	Chome	Active when then C-axis home input is triggered.
50	SpindleCW	Active when the spindle is rotating Clockwise.
51	SpindleCCW	Active when the spindle is rotating Counter-clockwise

LED NUMBER	LED NAME	DESCRIPTION
52	Miston	Active when the mist coolant is on.
53	Floodon	Active when the flood coolant is on.
54	Cyclestart	Active when a G-code execution cycle is in progress.
55	Runsingleline	Active when a Single line G-code execution cycle is in progress.
56	Xhomed	Active when the X-axis was already homed.
57	Yhomed	Active when the Y-axis was already homed.
58	Zhomed	Active when the Z-axis was already homed.
59	Ahomed	Active when the A-axis was already homed.
60	Bhomed	Active when the B-axis was already homed.
61	Chomed	Active when the C-axis was already homed.
62	Machinecoords	Active when the machine coordinate system is selected to view in the position DROs.
63	THCon	Active when the THC on physical input is active.
64	THCup	Active when the THC up physical input is active.
65	THCdown	Active when the THC down physical input is active.
66	THCenabled	Active when the THC control is enabled.
67	Softlimitsenabled	Active when the Software limit function is enabled.
68	THCdelay	Active when the THC delay is ongoing.
69	OutputPT2PN1	Indicates the actual logic state of port#2 pin#1.
70	OutputPT2PN2	Indicates the actual logic state of port#2 pin#2.
71	OutputPT2PN3	Indicates the actual logic state of port#2 pin#3.
72	OutputPT2PN4	Indicates the actual logic state of port#2 pin#4.
73	OutputPT2PN5	Indicates the actual logic state of port#2 pin#5.
74	OutputPT2PN6	Indicates the actual logic state of port#2 pin#6.
75	OutputPT2PN7	Indicates the actual logic state of port#2 pin#7.
76	OutputPT2PN8	Indicates the actual logic state of port#2 pin#8.
77	OutputPT2PN9	Indicates the actual logic state of port#2 pin#9.
78	OutputPT2PN10	Indicates the actual logic state of port#2 pin#10.
79	OutputPT2PN11	Indicates the actual logic state of port#2 pin#11.
80	OutputPT2PN12	Indicates the actual logic state of port#2 pin#12.
81	OutputPT2PN13	Indicates the actual logic state of port#2 pin#13.

LED NUMBER	LED NAME	DESCRIPTION
82	OutputPT2PN14	Indicates the actual logic state of port#2 pin#14.
83	OutputPT2PN15	Indicates the actual logic state of port#2 pin#15.
84	OutputPT2PN16	Indicates the actual logic state of port#2 pin#16.
85	OutputPT2PN17	Indicates the actual logic state of port#2 pin#17.
86	OutputPT3PN1	Indicates the actual logic state of port#3 pin#1.
87	OutputPT3PN2	Indicates the actual logic state of port#3 pin#2.
88	OutputPT3PN3	Indicates the actual logic state of port#3 pin#3.
89	OutputPT3PN4	Indicates the actual logic state of port#3 pin#4.
90	OutputPT3PN5	Indicates the actual logic state of port#3 pin#5.
91	OutputPT3PN6	Indicates the actual logic state of port#3 pin#6.
92	OutputPT3PN7	Indicates the actual logic state of port#3 pin#7.
93	OutputPT3PN8	Indicates the actual logic state of port#3 pin#8.
94	OutputPT3PN9	Indicates the actual logic state of port#3 pin#9.
95	OutputPT3PN10	Indicates the actual logic state of port#3 pin#10.
96	OutputPT3PN11	Indicates the actual logic state of port#3 pin#11.
97	OutputPT3PN12	Indicates the actual logic state of port#3 pin#12.
98	OutputPT3PN13	Indicates the actual logic state of port#3 pin#13.
99	OutputPT3PN14	Indicates the actual logic state of port#3 pin#14.
100	OutputPT3PN15	Indicates the actual logic state of port#3 pin#15.
101	OutputPT3PN16	Indicates the actual logic state of port#3 pin#16.
102	OutputPT3PN17	Indicates the actual logic state of port#3 pin#17.
103	OutputPT4PN1	Indicates the actual logic state of port#4 pin#1.
104	OutputPT4PN2	Indicates the actual logic state of port#4 pin#2.
105	OutputPT4PN3	Indicates the actual logic state of port#4 pin#3.
106	OutputPT4PN4	Indicates the actual logic state of port#4 pin#4.
107	OutputPT4PN5	Indicates the actual logic state of port#4 pin#5.

LED NUMBER	LED NAME	DESCRIPTION
108	OutputPT4PN6	Indicates the actual logic state of port#4 pin#6.
109	OutputPT4PN7	Indicates the actual logic state of port#4 pin#7.
110	OutputPT4PN8	Indicates the actual logic state of port#4 pin#8.
111	OutputPT4PN9	Indicates the actual logic state of port#4 pin#9.
112	OutputPT4PN10	Indicates the actual logic state of port#4 pin#10.
113	OutputPT4PN11	Indicates the actual logic state of port#4 pin#11.
114	OutputPT4PN12	Indicates the actual logic state of port#4 pin#12.
115	OutputPT4PN13	Indicates the actual logic state of port#4 pin#13.
116	OutputPT4PN14	Indicates the actual logic state of port#4 pin#14.
117	OutputPT4PN15	Indicates the actual logic state of port#4 pin#15.
118	OutputPT4PN16	Indicates the actual logic state of port#4 pin#16.
119	OutputPT4PN17	Indicates the actual logic state of port#4 pin#17.
120	OutputPT5PN1	Indicates the actual logic state of port#5 pin#1.
121	OutputPT5PN2	Indicates the actual logic state of port#5 pin#2.
122	OutputPT5PN3	Indicates the actual logic state of port#5 pin#3.
123	OutputPT5PN4	Indicates the actual logic state of port#5 pin#4.
124	OutputPT5PN5	Indicates the actual logic state of port#5 pin#5.
125	OutputPT5PN6	Indicates the actual logic state of port#5 pin#6.
126	OutputPT5PN7	Indicates the actual logic state of port#5 pin#7.
127	OutputPT5PN8	Indicates the actual logic state of port#5 pin#8.
128	OutputPT5PN9	Indicates the actual logic state of port#5 pin#9.
129	OutputPT5PN10	Indicates the actual logic state of port#5 pin#10.
130	OutputPT5PN11	Indicates the actual logic state of port#5 pin#11.
131	OutputPT5PN12	Indicates the actual logic state of port#5 pin#12.
132	OutputPT5PN13	Indicates the actual logic state of port#5 pin#13.
133	OutputPT5PN14	Indicates the actual logic state of port#5 pin#14.

LED NUMBER	LED NAME	DESCRIPTION
134	OutputPT5PN15	Indicates the actual logic state of port#5 pin#15.
135	OutputPT5PN16	Indicates the actual logic state of port#5 pin#16.
136	OutputPT5PN17	Indicates the actual logic state of port#5 pin#17.
137	MPGAPIN	Indicates the logic state of the MPG A pin.
138	MPGBPIN	Indicates the logic state of the MPG B pin.
139	EnableXaxis	On when the enable output of the X axis is active
140	EnableYaxis	On when the enable output of the Y axis is active
141	EnableZaxis	On when the enable output of the Z axis is active
142	EnableAaxis	On when the enable output of the A axis is active
143	EnableBaxis	On when the enable output of the B axis is active
144	EnableCaxis	On when the enable output of the C axis is active
145	Jogmodecontinous	On when the continuous jog mode is selected.
146	Jogmodestep	On when the step jog mode is selected.
147	R*	R* = Reserved address, do not use this address.
148	Jograte0001	On when the step jog rate is set to 0.001 units.
149	Jograte0010	On when the step jog rate is set to 0.01 units.
150	Jograte0100	On when the step jog rate is set to 0.1 units.
151	Jograte1000	On when the step jog rate is set to 1 units.
152	MPGmodecont	On when the MPG continuous mode is selected.
153	MPGmodesingle	On when the MPG single mode is selected.
154	MPGmodemulti	On when the MPG multi mode is selected.
155	MPGXaxisselect	On when the X axis is selected for the MPG jog.
156	MPGYaxisselect	On when the Y axis is selected for the MPG jog.
157	MPGZaxisselect	On when the Z axis is selected for the MPG jog.
158	MPGAaxisselect	On when the A axis is selected for the MPG jog.
159	MPGBaxisselect	On when the B axis is selected for the MPG jog.
160	MPGCaxisselect	On when the C axis is selected for the MPG jog.

LED NUMBER	LED NAME	DESCRIPTION
161 to 211	TABlayervisible	On when the tab layer is visible. The layer number selected is the LED code - 161, so the 161. LED code is for the tab layer 0. and the 211. LED code is for the 50. tab layer. Note: When more than one tab layer is visible, for example if a tab layer has a sub-tab layer which is also visible then both LED codes are active.
212	Offlinemode	On when the offline mode is active.
213	Sync_thread	Active when a syncronous thread cutting is in execution.
214	EncoderApin	Indicates the logic state of the Encoder A pin.
215	EncoderBpin	Indicates the logic state of the Encoder B pin.
216	MDIrunning	On when an MDI command is in progress.
217	Feedhold	On when the feedhold button is avtice.
218	Isdemomode	On if the software is running in demo mode.
219	Laserdataloaded	On if a laser data object is loaded to the memory for laser engraving.
220	Laserrunning	On when a laser engraving is in progress using the laser data object.
221	OutputPT5PN26	Indicates the actual logic state of port#5 pin#26. (M44 motherboard only.)
222	OutputPT5PN27	Indicates the actual logic state of port#5 pin#27. (M44 motherboard only.)
223	OutputPT5PN28	Indicates the actual logic state of port#5 pin#28. (M44 motherboard only.)
224	OutputPT5PN29	Indicates the actual logic state of port#5 pin#29. (M44 motherboard only.)
225	OutputPT5PN30	Indicates the actual logic state of port#5 pin#30. (M44 motherboard only.)
226	OutputPT5PN31	Indicates the actual logic state of port#5 pin#31. (M44 motherboard only.)
227	OutputPT5PN32	Indicates the actual logic state of port#5 pin#32. (M44 motherboard only.)
228	OutputPT5PN33	Indicates the actual logic state of port#5 pin#33. (M44 motherboard only.)
229	M0stopactive	Indicates that the M0 stop is active.
230	M1stopactive	Indicates that the M1 stop is active.
231	M60stopactive	Indicates that the M60 stop is active.
232	Pause	On when any of the M0 or M1 or M60 stop mode is active.
233	THCAntidiveactive	On when in THC control and the Anti diving happens.
234	OutputPT1PN94	Indicates the actual logic state of port#1 pin#94. (5441 motherboard only.)
235	OutputPT1PN95	Indicates the actual logic state of port#1 pin#95. (5441 motherboard only.)
236	MPGJogOn	Indicates that the MPG jogging is active.
237	THCarcon_emulation	Indicates that the THCarcon emulation signal is active.

LED NUMBER	LED NAME	DESCRIPTION
238	THCup_emulation	Indicates that the THCup emulation signal is active.
239	THCdown_emulation	Indicates that the THCdown emulation signal is active.
240	THCantidiveenabled	On when the THC anti dive function is enabled.
241	THCdelayenabled	On when the THC delay function is enabled.
242	THCantidownenabled	On when the THC anti down function is enabled.
243	SafeProbeModeactive	On when the SafeProbeMode is active. The safe probe mode actives the Reset if the probe goes on when not probing.
244	ProbedOK	On if the last probing (G31) was finished with a probe touch, off if the last probing did not end with a touch.
245	Digitizing	On when the M40 digitizing command is active.
246	JogSafeProbeModeactive	On when the JogSafeProbeMode is active. The jog safe probe mode stops the jog motion if the probe goes on while jogging.
247	M1optionalstopenabled	On when the M1 optional stop is enabled
248	ScaleXactive	On when the X axis scaling differs from value 1.
249	ScaleYactive	On when the Y axis scaling differs from value 1.
250	ScaleZactive	On when the Z axis scaling differs from value 1.
251	ScaleAactive	On when the A axis scaling differs from value 1.
252	ScaleBactive	On when the B axis scaling differs from value 1.
253	ScaleCactive	On when the C axis scaling differs from value 1.
254	G68Rotationactive	On when the G68 rotation is active.
255	TCPfollowmodeactive	On when the toolpath TCP follow mode is active.
256	FROdisabled	On if the feedrate override control (FRO) is disabled.
257	SROdisabled	On if the spindle speed override control (SRO) is disabled.
258	FROover100	On if the FRO is over 100%
259	FROunder100	On if the FRO is under 100%
260	SROover100	On if the SRO is over 100%
261	SROunder100	On if the SRO is under 100%
262	Proberadiusdisabled	On if the probe radius compensation in the digitizing settings is disabled.
263	Rapidmovementselected	Warning LED that rapid G0 code is selected.
264	Cameracaptureon	On when the camera is activated.
265	Cameragrayscalefilteron	On when the camera grayscale filter is activated.
266	Cameraedgefilteron	On when the camera edge filter is activated.

LED NUMBER	LED NAME	DESCRIPTION
267	Cameratargetlineon	On when the camera target line is shown.
268	Softlimitactive	On if any of the machine axis reached the softlimit.
296	Cameraimageinverton	On when the camera image invertion filter is activated.
270-299	R*	
300	ProbeError	On if probe was not successful on probe screen
301	ProbePaused	On if probe is paused or question needs to be confirmed
302	Axis1Disable	On if Axis1 cannot be selected on probe screen
303	Axis2Disable	On if Axis2 cannot be selected on probe screen
304	Axis3Disable	On if Axis3 cannot be selected on probe screen
305	TouchProbeWpCoordsSaved	On if workpiece probe position is saved
306	MobileProbeCoordsSaved	On if mobile probe position is saved
307	FixedProbeCoordsSaved	On if fixed probe position is saved
308	MobileWpReferenced	On if workpiece is referenced to mobile probe
309	FixedWpReferenced	On if workpiece is referenced to fixed probe

R* = Reserved address, do not use this address.

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

CHECKBOX OBJECTS (SORT BY NUMBER)

This section describes the **UCCNC software** Checkbox screen objects.

Each checkbox permits the user to make a selection on the UCCNC software GUI.
Checkboxes are read only and their logic state can be read in macro code.
This documentation lists all the accessible Checkbox variables, sorted **numerically.**

CHECKBOX NUMBER	CHECKBOX NAME	DESCRIPTION
1	Xaxisenable	
2	Xaxissteppin_activelow	
3	Xaxisdirpin_activelow	
4	Xaxislimitminuspin_activelow	
5	Xaxislimitpluspin_activelow	
6	Xaxishomepin_activelow	
7	Xaxishomedirectionpositive	
8	Xaxishomepositionautoset	
9	Xaxisenablebacklash	
10	Yaxisenable	
11	Yaxissteppin_activelow	
12	Yaxisdirpin_activelow	
13	Yaxislimitminuspin_activelow	
14	Yaxislimitpluspin_activelow	
15	Yaxishomepin_activelow	
16	Yaxishomedirectionpositive	
17	Yaxishomepositionautoset	
18	Yaxisenablebacklash	
19	Zaxisenable	
20	Zaxissteppin_activelow	
21	Zaxisdirpin_activelow	
22	Zaxislimitminuspin_activelow	
23	Zaxislimitpluspin_activelow	
24	Zaxishomepin_activelow	
25	Zaxishomedirectionpositive	
26	Zaxishomepositionautoset	
27	Zaxisenablebacklash	
28	Aaxisenable	
29	Aaxissteppin_activelow	
30	Aaxisdirpin_activelow	
31	Aaxislimitminuspin_activelow	
32	Aaxislimitpluspin_activelow	
33	Aaxishomepin_activelow	

CHECKBOX NUMBER	CHECKBOX NAME	DESCRIPTION
34	Aaxishomedirectionpositive	
35	Aaxishomepositionautoset	
36	Aaxisenablebacklash	
37	Baxisenable	
38	Baxissteppin_activelow	
39	Baxisdirpin_activelow	
40	Baxislimitminuspin_activelow	
41	Baxislimitpluspin_activelow	
42	Baxishomepin_activelow	
43	Baxishomedirectionpositive	
44	Baxishomepositionautoset	
45	Baxisenablebacklash	
46	Caxisenable	
47	Caxissteppin_activelow	
48	Caxisdirpin_activelow	
49	Caxislimitminuspin_activelow	
50	Caxislimitpluspin_activelow	
51	Caxishomepin_activelow	
52	Caxishomedirectionpositive	
53	Caxishomepositionautoset	
54	Caxisenablebacklash	
55	Estoppin_activelow	
56	Probepin1_activelow	
57	Chargepumppin_activelow	
58	R*	
59	Chargepump_alwayson	
60	-	
61 - 72	R*	
73	Generalsettings_exactstoppathmode	
74	Generalsettings_constantvelocitypathmode	
75	Generalsettings_enablesoftlimits	
76	Toolchange_igonoretoolchangemacro	
77	Toolchange_stopspindleandwaitforcyclestart	
78	Toolchange_runM6macro	
79	Profiles_createshortcutondesktop	
80	Appearance_3DTCPmarker	
81	IOsetup_EnableTHCcontrol	
82	IOsetup_THConpinactivelow	
83	IOsetup_THCuppinactivelow	
84	IOsetup_THCdownpinactivelow	
85	IOsetup_ControlTHCevenifTHConisnotactive	
86	Generalsettings_Kernelfrequency25kHz	
87	Generalsettings_Kernelfrequency50kHz	

CHECKBOX NUMBER	**CHECKBOX NAME**	**DESCRIPTION**
88	Generalsettings_Kernelfrequency100kHz	
89	Appearance_ShowconeicononTCP	
90	Appearance_ShowcrosshaironTCP	
91 - 138	Inputtrigger1_activelow to Inputtrigger48_activelow	
139	R*	
140	R*	
141	Xaxis_enable	
142	Yaxis_enable	
143	Zaxis_enable	
144	Aaxis_enable	
145	Baxis_enable	
146	Caxis_enable	
147	Appearance_Maximizescreenonstartup	
148	IOsetup_attachJROtoMPG	
149	R*	
150 - 197	Outputtrigger1_activelow to Outputtrigger48_activelow	
198	R*	
199	R*	
201	Reverseencodercountdirection	
202 - 210	CAM_select_origin	
211	Laseroutput_activelow	
212	Dwelltimeinseconds	
213	Enableusertabpage	
214	Generalsettings_Kernelfrequency200kHz	
215	Generalsettings_Kernelfrequency400kHz	
216	Unknowgcode_ignore	
217	Unknowngcode_warning	
218	Unknowngcode_donotrun	
219	RotateTCPwithplaneselection	
220	Showzeromark	
221	THCAntidiveenable	
222	Showmessageonsoftlimits	
223	THCDelayenable	
224	THCAntidownenable	
225	Probepin2_activelow	
226	UseSpindlepulleys	
227	ValidateTextfieldswithEnterkeyonly	
228	Digitize_addaxisnames	
229	Digitize_commaseparatedCSV	
230	Digitize_includeaxisX	
231	Digitize_includeaxisY	

CHECKBOX NUMBER	CHECKBOX NAME	DESCRIPTION
232	Digitize_includeaxisZ	
233	Digitize_includeaxisA	
234	Digitize_includeaxisB	
235	Digitize_includeaxisC	
236	Digitize_Clearfilenamewhenfinished	
237	Chargepump2pin_activelow	
238	THCenableoutput_activelow	
239	Antidiveoutput_activelow	
240	Antidownoutput_activelow	
241	Precompileallmacrosonstartup	
242	Disablejogpanelautopopup	
243 - 290	Inputtrigger49_activelow to Inputtrigger96_activelow	
291 - 338	Outputtrigger49_activelow to Outputtrigger96_activelow	
339	R*	
340	Showrotationpointmark	
341	Showtoolpathboundariesmark	
342	Softlimitsgcodefileprecheck	
343	ResetG68onfileload	
344	ResetG51onfileload	
345	ShowRcompAnalysisonFileload	
346	DisableVirtualMouse	
347	Xcurrenthilow_activelow	
348	Ycurrenthilow_activelow	
349	Zcurrenthilow_activelow	
350	Acurrenthilow_activelow	
351	Bcurrenthilow_activelow	
352	Ccurrenthilow_activelow	
353	G41_G42roundjoints	
354	Savetooltableonclosing	
355	Dontresettoolpathviewonreload	
356	R*	
357	Digitalsyncoutput1_activelow	
358	Digitalsyncoutput2_activelow	
359	Digitalsyncoutput3_activelow	
360	Digitalsyncoutput4_activelow	
361	Digitalsyncoutput5_activelow	
362	Digitalsyncoutput6_activelow	
363	Digitalsyncoutput7_activelow	
364	Digitalsyncoutput8_activelow	
365	Digitalsyncoutput9_activelow	
366	Digitalsyncoutput10_activelow	

CHECKBOX NUMBER	CHECKBOX NAME	DESCRIPTION
367	CheckHomeonCycleStart	
368	ResetDereferenceAllHomes	
369	Aaxisrotary	
370	Baxisrotary	
371	Caxisrotary	
372	ResetG94onfileload	
373	Aaxisrotaryrollover	
374	Baxisrotaryrollover	
375	Caxisrotaryrollover	
376	Disabletoolpath	
377	Donotcompressimages	
378	Toolpathdimsaxiscolored	
379	Showtoolpathdimensions	

R* = Reserved address, do not use this address.

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

VARIABLES

UCCNC allows for the use of parametric programming (as shown within the main UCCNC user manual), the following is an extract from the UCCNC manual for ease of reference. The purpose of this section is to provide a list of the internal variables as tabled below and also a location for the user to note their own variables should they be using them and to avoid (or allow for) sharing them between macros.

Parametric programming

Instead of constants, variables could be also used to define coordinates or feedrate, spindle speed etc. There are 1000 spaces for internal variables currently, named as #0 to #999.
When a number is programmed with the '**#**' suffix means that this parameter 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 values to variables.
For example to give a value of **1.23** to variable **#2**, *code:* **'#2 = 1.23'**.
Equations should always be on 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. Only one variable can be programmed as a parameter on a single line.

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)

Typing **?#4** into the MDI will return the value stored within variable #4 to the status message box on screen.

To program complex equations it is often required to use brackets. Because in RS274NGC programming, the round brackets '**(**' and '**)**' are used to defined comments for the equations, rectangular brackets '**[**' and '**]**' should be used instead. Also, to define a comma for functions which have more than one parameter use the semicolon '**;**' character instead of the '**,**' comma.

For example: **#1= [1 + #2]*3**
#4=max[#1;# 2]*7

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

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 supported functions are listed below:

OPERATOR	NAME OF FUNCTION	SHORT DESCRIPTION
+	Summation (addition)	Summarise (add) numbers.
-	Deduction (subtraction)	Deduct (subtract) 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. For example: ?#1 prints the value of #1 variable. ***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	hyperbolic cosine	Returns the hyperbolic cosine of the specified angle.
e	exponentiation	Returns e (the base of natural logarithms) raised to the specified power.
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. Example: min[1; 2] gives an output of 1.
max[;]	maximum	Returns the larger of two numbers. Example: max[1;2] gives an output of 2.
pi	Pi (π)	~ 22/7 or 3.142…

OPERATOR	NAME OF FUNCTION	SHORT DESCRIPTION
pow[;]	power	Returns a specified number raised to the specified power. Example: 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. Example: 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

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

The following is the list of variables that are known and referenced within the UCCNC manual, as others are made available the list will be updated, space is provided to add the users own variables to avoid (or allow for) sharing them between macros:

DEFINED VARIABLES	**FUNCTION / DESCRIPTION**
#5060	Probe Success = 0, Probe not tripped = 1, Probe active = 2
#5061	G31 Probe, X Axis Touch / Stop Coordinate
#5062	G31 Probe, Y Axis Touch / Stop Coordinate
#5063	G31 Probe, Z Axis Touch / Stop Coordinate
#5064	G31 Probe, A Axis Touch / Stop Coordinate
#5065	G31 Probe, B Axis Touch / Stop Coordinate
#5066	G31 Probe, C Axis Touch / Stop Coordinate
USER VARIABLES	**FUNCTION / DESCRIPTION**
Permitted range: #0 to #999	

Intentionally blank page to facilitate updated sections to be printed and inserted to retain pagination for other sections as new Macros are added.

TIPS:

The following is the list of tips that you may or may not find useful at putting together your macros (and to some extent plugins) using standard C# code.

Variable Conversion:

A lot of time you will either extract a variable in one variable type and need to convert it to another type.

E.g: AS3.Getfield(100) will return the field as a string variable.
However, to AS3.Setfield(15.23, 100), "15.23" must be a double {or you can use AS3.Setfieldtext("15.23", 100)}

The common ones I use are as follows:

Convert.ToBoolean
Conver.ToByte
Convert.ToChar
Convert.ToDecimal
Convert.ToDouble
Convert.ToInt32
Convert.ToString

More info available here:
https://docs.microsoft.com/en-us/dotnet/api/system.convert?view=netframework-3.0

MessageBox:

Messageboxes are usefull if you need a message box to popup with a warning or some sort of input required etc.

The most simplistic MessageBox just pops up with a string message and an OK button.

Code:

```
+ Environment.NewLine +
```

e.g.

The fun stuff starts when you want something more than just a message, and maybe want a couple of buttons to allow for options, maybe a heading on the MessageBox and maybe an Icon then you have a lot more options to customise the message box with simple standard settings.

If you want to break a string message within a MessageBox, break it be using the following code:

Code:

```
+ Environment.NewLine +
```

e.g.