Mach3 Version 3.x
Macro Programmers Reference Manual

Draft Revision 0.23
August 22nd, 2010

For Mach3 v3 versions thru 3.43.19.

Table of Contents

Introduction

This Programmers Reference Manual documents the commonly used Cypress Basic (CB) function calls available to macro programmers using Mach3 version 3. This information is being provided primarily to help Mach3 users understand existing macro code. While this interface will continue to be supported by future Mach3 versions for some period of time, Mach3 version 4 will provide a completely new, much more regular interface, ***much*** higher functionality interface for CB macro programming. It is ***strongly*** recommended that all new CB code use the new interface, as support for this old one ***will*** be discontinued at some point in the, possibly not too distant, future. In addition, it is unlikely there will be any further updates or bug fixes to this now obsolete interface after the release of Mach3 version 4, so any existing bugs and anomalies (and there are quite a few) will remain.

No attempt has been made to make this an exhaustive document covering all of the CB functionality. There are many functions which were never previously documented, or which were documented incompletely or incorrectly. In many cases, these functions are not included in this document. There are a number of functions which were partially documented, but found to either not function as documented, to have significant restriction in their operation, or, in some cases, were felt to be either of no real value, or even risky to use. These functions are generally not included in this document. The functionality described herein has been tested against Mach3 version 3.042.020. Some functions may behave differently in other versions. Some functions will be missing entirely in some earlier versions.

Legacy Scripting Functions Grouped Alphabetically

ActivateSignal

Sub ActivateSignal (SigNum As Integer)

This function causes the specified Mach output signal to be driven to its active state. If the signal is defined in Config->Ports&Pins as ActiveHigh, it will be driven to a logic High level, otherwise it will be driven to a logic Low level.

Arguments:

SignalID must be one of the pre-defined Mach3 CB output signal constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:

None

Example:

```
ActivateSignal(OUTPUT2)  ‘ Turn on Flux Capacitor
Sleep(1000)              ‘ Give it time to charge
fully
DeactivateSignal(OUTPUT2) ‘ Turn it off
```

See also:

DeactivateSignal(), CB Constants

AppendTeachFile

Function AppendTeachFile(Filename As String) As Integer

This function re-opens an existing Teach file at Gcode\Filename in the Mach3 directory, and appends any commands subsequently executed via MDI or Code() to that file, until CloseTeachFile() is executed. The specified file must already exist.

Arguments:

Filename is the name of the Teach file to be re-opened. The file must reside in the Gcode subdirectory of the Mach3 install directory.

Return Value:

A non-zero value is returned if the operation was successful.

Example:

```
' Create a new Teach File in Mach3\Gcode
MyTeachFile = "TeachMe.nc"
Err = OpenTeachFile(MyTeachFile)
If Err <> 0 Then
     ' Teach file created successfully
     ' Write some G Code to it
     Code "G00 X0 Y0"
     Code "G02 X0 Y0 I-1 J0 F40"
     ' Close the Teach file
     CloseTeachFile()
     ' Now load the teach file for execution
     LoadTeachFile()
Else
     ' OpenTeachFile failed
     Message "Unable to open Teach File"
End If
```

See also:

OpenTeachFile(), CloseTeachFile()

AskTextQuestion

Function AskTextQuestion(Prompt As String) As String

This function displays a dialog box containing the specified prompt string, and waits for the user to enter a text string in the dialogs text box. The user entered string is returned to the caller.

Arguments:

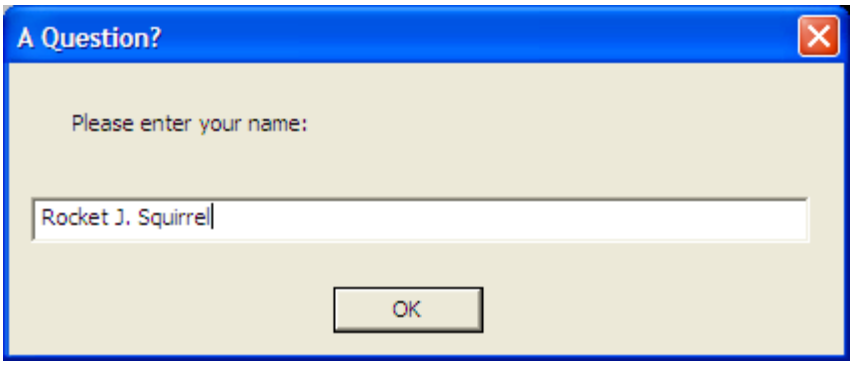
Prompt string is the string that will be displayed above the text box when the dialog is displayed.

Return Value:

The text string entered by the user.

Example:

```
Dim UsersName As String
UsersName = AskTextQuestion("Please enter your name:")
Message "Hello, " & UsersName & "!"
```



See also:
Message(), Question(), AskTextQuestion(), MachMsg(), GetCoord()

CloseDigFile

Sub CloseDigFile()

This function closes an open digitizing file. If there is no open digitizing file, it does nothing.

Arguments:
None

Return Value:
None

Example:

```
OpenDigFile()       ' Open the digitizing file
ProbeOutline()      ' Call my custom probing function
CloseDigFile()      ' Close the digitizing file
```

See also:
OpenDigFile(), SetProbeActive(), IsProbing()

CloseTeachFile

Sub CloseTeachFile()

This function closes an open Teach file. The file must have previously been opened by either OpenTeachFile, or AppendTeachFile(). If not Teach file is currently open, it does nothing.

Arguments:
None

Return Value:

None

Example:

```
TeachFile = "MyTeachFile.nc"
Dim Err As Integer
Err = OpenTeachFile(TeachFile)
If Err = 0 Then
     MsgBox("Unable To Open: " & Teachfile)
Else
     ' MDI commands entered here are written to
TeachFile
     Code("G55")
     Code("G00 X0.000 Y0.000 Z0.500")
End If
' Now close TeachFile
CloseTeachFile()
' Do something else here
' Now re-open TeachFile for append
Err = AppendTeachFile(TeachFile)
If Err = 0 Then
     MsgBox("Unable To Open: " & TeachFile)
Else
     ' MDI commands entered here are written to
TeachFile
     Code("G56")
     Code("G00 X0.000 Y0.000 Z0.500")
End If
' Now close TeachFile
CloseTeachFile()
```

See also:

OpenTeachFile(), AppendTeachFile()

Code

Sub Code(Gcode As String)

This function executes the single line of G-code passed as the argument, exactly as if it had been entered on the MDI line, or encountered in a G-Code program. Note that the G-code line is placed in the execution queue, but, in general, the Code() function will return before the line has actually been executed. If it is important that your program know that the line has completed execution, a While loop must be used with the IsMoving() function, as shown in the example below.

Note also that if a Teach file is currently open, the G-code line will not actually be executed, but will simply be written out to the Teach file.

The following optional modes of the Code() function are documented here only for completeness. Their use is discouraged, and support for them may be discontinued at any time, without warning.

Sub Code("LOAD:" & FilePath)

This mode loads a G-Code file from the specified FilePath. The LoadFile() function should be used instead.

Sub Code("SAVE_XML")

This method immediately saves the XML configuration file.

Arguments:
A single G-code line to be executed as a String

Return Value:
None

Example:

```
' Select our fixture
Code("G55")
 ' Move away from the vise
Code("G00 X-4.000 Y1.000 Z1.000")
' Wait for movement to complete
While (IsMoving())
     ' Sleep, so other threads can run while we're
waiting
     Sleep(100)
Wend
' Done
```

See also:

CoupleSlave

Sub Function CoupleSlave(State As Integer)

This function causes any slaved axis to be coupled to, or de-coupled from, its master axis during homing.

Arguments:

State is an Integer value of 0 or 1 that defines whether the slave axis should be coupled to its master axis during homing. If State is 0, the axes will be un-coupled. If State is 1, the axes will be coupled.

Return Value:

None

Example:

```
' Define the axes
Dim Xaxis As Integer
Xaxis = 0
Dim Yaxis As Integer
Yaxis = 0

' Couple the slaved A axis to its master X axis
CoupleSlave(1)
' Home the master and slave
SingleVerify(Xaxis)
' Un-couple the X and A axes
CoupleSlave(0)
```

See also:

None

DeactivateSignal

Sub DeactivateSignal (SigNum As Integer)

This function causes the specified Mach output signal to be driven to its inactive state. If the signal is defined in Config->Ports&Pins as ActiveHigh, it will be driven to a logic Low level, otherwise it will be driven to a logic High level.

Arguments:

SignalID must be one of the pre-defined Mach3 CB output signal constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:

None

Example:

```
ActivateSignal (OUTPUT2)      ' Turn on Flux Capacitor
Sleep(1000)                   ' Give it time to charge
fully
DeactivateSignal (OUTPUT2)    ' Turn it off
```

See also:

ActivateSignal (), CB Constants

DoMenu

Sub DoMenu (MenuIndex As Integer, MenuItem As Integer)

This function allows a macro script to invoke any function available through any of the Mach3 menus, exactly as if the user had clicked on the menu with the mouse. The specific menu item to be invoked is specified by the two arguments. The first indicates which menu the item to be invoked resides in. The second indicates which item within that menu is to be invoked.

Arguments:

MenuIndex is the 0-based index of the menu to activate. The File menu is index 0, the Config menu is index 1, etc.

MenuItem is the 0-based index of the menu item to activate. The first item on a given menu is index 0, the second is index 1, etc.

Return Value:

None

Example:

```
' Define menu indices, left-to-right, starting with 0
Dim FileMenu As Integer
FileMenu = 0
Dim ConfigMenu As Integer
ConfigMenu = 1
…
' Define File menu items, top-to-bottom, starting with
0
Dim FileMenuLoadGCodeMenuItem
FileMenuLoadGCodeMenuItem = 0
Dim FileMenuLazyCAMMenuItem As Integer
FileMenuLazyCAMMenuItem = 1
Dim FileMenuCloseFilesMenuItem As Integer
FileMenuCloseFilesMenuItem = 2
Dim FileMenuExitMenuItem As Integer
FileMenuExitMenuItem = 3
' Define Config menu items, top-to-bottom, starting
with 0
Dim ConfigMenuDefineNativeUnitsMenuItem As Integer
ConfigMenuDefineNativeUnitsMenuItem = 0
Dim ConfigMenuPortsAndPinsMenuItem As Integer
ConfigMenuPortsAndPinsMenuItem = 1
…
' Pop-up the Config->Ports & Pins dialog
```

```
DoMenu(ConfigMenu, ConfigMenuPortsAndPinsMenuItem)
```

See also:

DoButton(), DoOEMButton()

DoButton

Sub DoButton(ButtonNum As Integer)

This legacy function allows a macro to execute an on-screen button function which has an assigned Button code. The specified function is invoked exactly as if the user had clicked the corresponding on-screen button with the mouse. Note that no actual on-screen button need exist for this function to work. This is simply an easy means to execute any of the "Button" functions through CB.

The use of DoButton is no longer recommended practice and this function exists only to support preexisting legacy scripts. This function is deprecated, and its use is ***strongly*** discouraged.

> Legacy script note: Over time, there have been two different Button numbering schemes used with Mach; the "Button number" series and the "OEMButton number" series. This function uses the "Button number" series.
>
> Within the "Button number" range, valid ButtonNums were from 0 to 31, which, at one time, corresponded to OEM LED numbers 1000 10 1031.
>
> The numerical correspondence between the numbering series is **not guaranteed** for future releases of Mach.

Use the DoOEMButton function instead of this function.

Arguments:

ButtonNum must be one of the pre-defined Mach3 OEM Button Number constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:

None

Example:

```
' Define OEM codes for Mist On and Mist Off
Const RewindButton = 2

' rewind the gcode to start
DoButton(RewindButton)
```

See also:
DoButton(), DoOEMButton()

DoOEMButton

Sub DoOEMButton(OEMButtonCode As Integer)

This function allows a macro to execute any on-screen button function which has an assigned OEM Button code. The specified function is invoked exactly as if the user had clicked the corresponding on-screen button with the mouse. Note that no actual on-screen button need exist for this function to work. This is simply an easy means to execute any of the "OEM Button" functions through CB.

Arguments:
OEMButtonCode must be one of the pre-defined Mach3 OEM Button Code constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:
None

Example:

```
' Define OEM codes for Mist On and Mist Off
Const OEMButtonMistOn = 226
Const OEMButtonMistOff = 227

' Turn Mist coolant on for 3 seconds
DoOEMButton(OEMButtonMistOn)
Sleep(3000)
DoOEMButton(OEMButtonMistOff)
```

See also:
DoButton(), DoOEMButton(), DoMenu()

DoSpinCCW

Sub DoSpinCCW()

This function turns the spindle on, rotating counter-clockwise.

Arguments:
None

Return Value:
None

Example:
```
' Turn on the spindle, turning CCW
DoSpinCCW()
' Let it run 5 seconds
Sleep(5000)
' Now turn it off
DoSpinStop()
```

See also:
DoSpinCW(), DoSpinStop()

DoSpinCW

Sub DoSpinCW()

This function turns the spindle on, rotating clockwise.

Arguments:
None

Return Value:
None

Example:
```
' Turn on the spindle, turning CW
DoSpinCW()
' Let it run 5 seconds
Sleep(5000)
' Now turn it off
DoSpinStop()
```

See also:
DoSpinCCW(), DoSpinStop()

DoSpinStop

Sub DoSpinStop()

This function turns off the spindle.

Arguments:
None

Return Value:
None

Example:

```
' Turn on the spindle, turning CW
DoSpinCW()
' Let it run 5 seconds
Sleep(5000)
' Now turn it off
DoSpinStop()
```

See also:
DoSpinCW(), DoSpindCCW()

EndTHC

Sub EndTHC()

This function turns off torch height control. It is functionally identical to THCOff().

Arguments:
None

Return Value:
None

Example:

```
StartTHC()' Turn on torch height control
…          ' Do some cutting here
EndTHC()   ' Turn off torch height control
```

See also:
StartTHC(), THCOn(), EndTHC(), THCOff(), ZeroTHC(), ResetTHC()

FeedRate

Sub FeedRate() As Double

This function gets the current feed rate. Note that Feed rate is specified in units per minute.

Arguments:
Feed rate specified in units/minute, as a Double

Return Value:
None

Example:

```
' Set the feed rate to 123.456 inches/minute
SetFeedRate(123.456 / 60)
' Get the current feed rate, in inches/minute, and
display it
CurrentFeedrate = FeedRate()
' Display it on the status line
Message "Current feed rate = " & CurrentFeedrate
```

See also:

SetFeedrate()

FileName

Function FileName() As String

This function returns the filename and path of the currently loaded G-Code file, if any. If no file is currently loaded, the string "No File Loaded." is returned instead.

Arguments:

None

Return Value:

Current G-Code file name and path, or "No File Loaded."

Example:

```
' Show user current G-Code file name and path
Message "Current file is: " & FileName()
```

See also:

LoadFile(), LoadRun()

GetABSPosition

Function GetABSPosition(Axis As Integer) As Double

This function returns the machine position of the specified axis.

Arguments:

Axis is the axis whose machine position is being requested as follows:

0 = X Axis
1 = Y Axis
2 = Z Axis
3 = A Axis
4 = B Axis
5 = C Axis

Return Value:

Machine position as a Double

Example:

```
' Define the axes
Dim Xaxis As Integer
Xaxis = 0
Dim Yaxis As Integer
Yaxis = 1
Dim Zaxis As Integer
Zaxis = 2

' Get Y Axis Machine Position
Dim AxisPos As Double
AxisPos = GetABSPosition(Yaxis)
' Put it on the status line
Message "Y Axis Machine Pos = " & AxisPos
```

See also:

SetMachZero(), MinX(), MaxX(), MinY(), MaxY()

GetACoor

Function GetACoor() As Double

This function is used in conjunction with the GetCoord() function to get X, Y, Z and A axis coordinate values from the user. The GetACoor() function will return the A value entered by the user in the last GetCoord() function call.

Arguments:

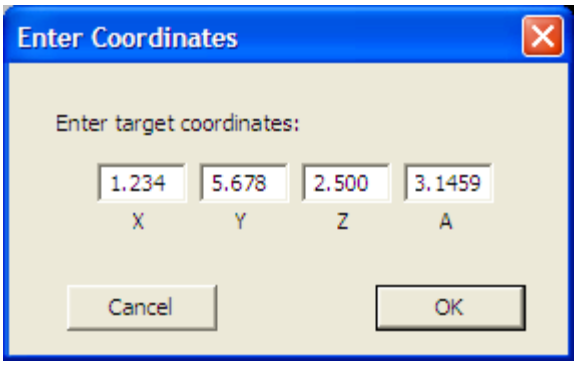
None

Return Value:

A Axis coordinate value from last GetCoord() call

Example:

```
GetCoord("Enter target coordinates:")
Message "Coordinates are: " & GetXCoor() & " " &
GetYCoor() & " " & GetZCoor() & " " & GetACoor()
```



See also:
GetCoord(), GetXCoor(), GetYCoor(), GetZCoor()

GetActiveProfileDir

Function GetActiveProfileDir() As string

This function is used to retrieve the string with the full path of the active profile.

Arguments:
None

Return Value:
The path to the active profile.

Example:

```
MsgBox “The Running Profile path is “ &
GetActiveProfileDir()
```

See also:
GetActiveProfileDir

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

GetActiveProfileName

Function GetActiveProfileName() As string

This function is used to retrieve the name of the currently running profile.

Arguments:
None

Return Value:

The name of the currently active profile.

Example:

```
MsgBox “The Running Profile is: “ &
GetActiveProfileName()
```

See also:

GetActiveProfileDir

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

GetActiveScreenSetName

Function GetActiveScreenSetName() As string

This function is used to retrieve the name of the currently active Screen set.

Arguments:

None

Return Value:

The name of the currently active screen set.

Example:

```
MsgBox “The Running screen set is: “ &
GetActiveScreenSetName()
```

See also:

n/a

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

GetCoord

Sub GetCoord(Prompt As String)

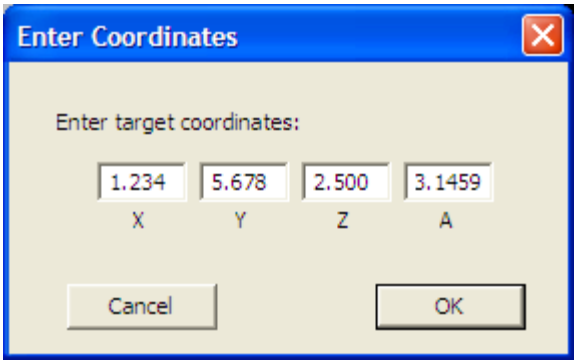
This function displays a dialog box containing the Prompt string, along with four textboxes, labeled X, Y, Z and A, into which the user can enter four coordinate values. The values are stored in variables within Mach3 which can be retrieved using the GetXCoor(), GetYCoor(), GetZCoor() and GetACoor() functions.

Arguments:

Prompt is a String that is displayed in the dialog box.

Return Value:
None

Example:
```
GetCoord("Enter target coordinates:")
Message "Coordinates are: " & GetXCoor() & " " &
GetYCoor() & " " & GetZCoor() & " " & GetACoor()
```



See also:
GetXCoor(), GetYCoor(), GetZCoor(), GetACoor()

GetCurrentTool

Function GetCurrentTool() As Integer

This function returns the currently active tool number.

Arguments:
None

Return Value:
Current tool number, 1-253.

Example:
```
' Tell the user which tool is active
Message "Current Tool is " & GetCurrentTool() & " = >
" & GetToolDesc()
```

See also:
SetCurrentTool(), GetSelectedTool(), ToolLengthOffset(), GetToolParam(), SetToolParam(), GetToolChangeStart(), GetToolDesc(), SetToolX(), SetToolZ()

GetDRO

Function GetDRO(DRONum As Integer) As Double

This legacy function takes the DRO number passed as its argument, and returns the value of the Mach DRO of that number.

The use of GetDRO is no longer recommended practice and this function exists only to support preexisting legacy scripts. This function is deprecated, and its use is ***strongly*** discouraged.

> Legacy script note: Over time, there have been two different DRO numbering schemes used with Mach; the "DRO number" series and the "OEMDRO number" series. This function uses the "DRO number" series.
>
> The "DRO number" series was further subdivided into "User" and "OEM" ranges. Within the "OEM" range, valid DRONums were from 0 to 200, which, at one time, corresponded to OEM DRO numbers 800 to 1000.
>
> The numerical correspondence between the numbering series is **not guaranteed** for future releases of Mach.

Use the GetOEMDRO and GetUserDRO functions instead of this function.

Arguments:

DRONum is the DRO number to read. The value has to be within the "DRO number series".

Return Value:

Contents of DRO DRONum

Example:

```
' Define the axes
Const XaxisMultiFunctionDRONum = 0
Const YaxisMultiFunctionDRONum = 1
Const ZaxisMultiFunctionDRONum = 2

' Read the Z axis DRO
MsgBox "Using GetDRO() Z Axis DRO reads: " &
GetDRO(ZaxisMultiFunctionDRONum)
```

See also:

SetOEMDRO(), GetOEMDRO(), SetUserDRO(), GetUserDRO()

GetDROString

Function GetDROString(DRONum As Integer) As String

This legacy function takes the DRO number passed as its argument and returns the value of the Mach DRO of that number, rounded to four decimal places, and formatted as a String.

The use of GetDROString is no longer recommended practice and this function exists only to support preexisting legacy scripts. This function is deprecated, and its use is ***strongly*** discouraged.

> Legacy script note: Over time, there have been two different DRO numbering schemes used with Mach; the "DRO number" series and the "OEMDRO number" series. This function uses the "DRO number" series.
>
> The "DRO number" series was further subdivided into "User" and "OEM" ranges. Within the "OEM" range, valid DRONums were from 0 to 200, which, at one time, corresponded to OEM DRO numbers 800 to 1000.
>
> The numerical correspondence between the numbering series is **not guaranteed** for future releases of Mach.

Use of the GetOEMDRO and GetUserDRO functions instead; then use cStr to converts the numerical value to a string.

Arguments:
DRONum is the DRO number to read.

Return Value:
Contents of DRO DRONum

Example:

```
' Define the axes
Const XaxisMultiFunctionDRONum = 0
Const YaxisMultiFunctionDRONum = 1
Const ZaxisMultiFunctionDRONum = 2

Dim ZPositionString as String

' Read the Z axis DRO
ZPosition = cStr(GetDRO(ZaxisMultiFunctionDRONum))
```

```
MsgBox “Using GetDRO() Z Axis DRO reads: “ &
ZPositionString
```

See also:
SetOEMDRO(), GetOEMDRO(), SetUserDRO(), GetUserDRO()

GetIJMode

Function GetIJMode() As Integer

This function returns the current IJ mode (absolute/incremental), as set in Config->GeneralConfig.

Arguments:
None

Return Value:
0 indicates absolute IJ mode is enabled
1 indicates incremental IJ mode is enabled

Example:

```
‘ Show user the current IJ mode
If GetIJMode() Then
     Message "IJ Mode is incremental"
Else
     Message "IJ Mode is absolute"
End If
```

See also:
SetIJMode()

GetLED

Function GetLED(LEDNum As Integer) As Integer

This legacy function takes the LED number passed as its argument, and returns the value of the Mach LED of that number.

The use of GetLED is no longer recommended practice and this function exists only to support preexisting legacy scripts. This function is deprecated, and its use is ***strongly*** discouraged.

Legacy script note: Over time, there have been two different LED numbering schemes used with Mach; the “LED number” series and the “OEMLED number” series. This function uses the “LED number” series.

The “LED number” series was further subdivided into “User” and “OEM” ranges. Within the “OEM” range, valid LEDNums were from 0 to 55, which, at one time, corresponded to OEM LED numbers 800 to 855.

The numerical correspondence between the numbering series is **not guaranteed** for future releases of Mach.

Use the GetOEMLED and GetUserLED functions instead of this function.

Arguments:
LEDNum is the OEM LED number to read.

Return Value:
0 indicates LEDNum is currently turned off
1 indicates LEDNum is currently turned on

Example:

```
‘ Define the LEDs
Const InchModeLED = 1
Const mmModeLED = 2

‘ Are we in inch or metric mode?
Code “G20”
‘Code “G21”

If GetLED(InchModeLED) Then
     ‘ We are in Inch Mode
     Message “Inch Mode”
End If

If GetLED(mmModeLED) Then
     ‘ We are in mm Mode
     Message “mm Mode”
End If
```

See also:
GetOEMLED(), SetUserLED(), GetUserLED()

GetLoadedGCodeDir

Function GetLoadedGCodeDir() As string

This function is used to retrieve the string with the full path to the loaded g-code file name.

Arguments:
None

Return Value:
The path to the loaded G-Code file.
If no Gcode is loaded the function returns a null string (“”).

Example:

```
MsgBox “The G-Code file path is “ &
GetloadedGCodeDir()
```

See also:
GetloadedGCodeFileName

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

GetLoadedGCodeFileName

Function GetLoadedGCodeFileName() As string

This function is used to retrieve the name of the loaded g-code file.

Arguments:
None

Return Value:
The name of the loaded G-Code file. The extension is included in the string.
If no Gcode is loaded the function returns a null string (“”).

Example:

```
MsgBox “The G-Code file name is “ &
GetloadedGCodeFileName()
```

See also:
GetLoadedGCodeDir

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

GetMachVersion

Function GetMachVersion(ByRef Major as Integer, ByRef Minor as Integer, ByRef Build as Integer) As Boolean

This function returns the full path to the Mach3 installation folder.

Arguments:

The functions arguments are passed by reference and used to return the Mach version numbers.
Major: Major Version number
Minor: Minor version number
Build: build Version number

You must declare (DIM) the variables you pass to the function as integers. Non declared variables are created as type Var which can't be passed by reference.

Return Value:

True: the returned version values are valid
False: an error occurred processing the GetMachVersion call; the version values may not be valid.

Example:

```
If (GetMachVersion(Major, Minor, Build) <> true) Then
     MsgBox("Unable to get Version info")
Else
     MsgBox("Mach3 version = " & Major & "." & Minor &
"." & Build )
End If
```

See Also:

n/a

First Mach3 version with API:

This API was first implemented in Mach3 version 3.42.30.

GetMainFolder

Function GetMainFolder() As String

This function returns the full path to the Mach3 installation folder.

Arguments:
None

Return Value:
String full file system path to Mach3 installation folder

Example:
```
' Show the user where Mach3 is installed
Message "Mach3 is installed at: " & GetMainFolder()
```

See also:

GetMyWindowsHandle

Function GetMyWindowsHandle() as Long

This function is used to retrieve the windows handle for the Mach3 window. This is useful for passing to OS calls which require the callers windows handle as a parameter.

Arguments:
None

Return Value:
The Mach 3 windows handle

Example:
```
MsgBox "My Windows handle as number is " &
GetMyWindowsHandle()
```

See also:
n/a

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

GetOEMDRO

Function GetOEMDRO(DRONum As Integer) As Double

This function returns the value of OEM DRO DRONum.

Arguments:
DRONum must be a valid OEM DRO number.

Return Value:

Returns a Double value of the specified DRO

Example:

```
' Define the axes
Const XaxisMultiFunctionOEMDRONum = 800
Const YaxisMultiFunctionOEMDRONum = 801
Const ZaxisMultiFunctionOEMDRONum = 802

' Write 1.2345 to Z axis DRO using SetOEMDRO
SetOEMDRO(ZaxisMultiFunctionDRONum, 1.2345)

' Show the user the Z Axis DRO value, using
GetOEMDRO()
MsgBox "After using SetOEMDRO() the Z Axis DRO reads:
" & GetOEMDRO(ZaxisMultiFunctionDRONum)
```

See also:

SetOEMDRO(), SetUserDRO(), GetUserDRO()

GetOEMLED

Function GetOEMLED(LEDNum As Integer) As Integer

This function returns the value of OEM LED LEDNum.

Arguments:

LEDNum must be a valid OEM LED number.

Return Value:

Returns an Integer value representing the current state of the specified LED. 0 indicates the LED of off (unlit), 1 indicated the LED is on (lit).

Example:

See also:

SetOEMLED(), SetUserLED(), GetUserLED()

GetPage

Function GetPage() As Integer

This function returns the number of the currently active screen set page.

Arguments:
None

Return Value:
Current screen set page number, as Integer

Example:

```
' Make sure user is on Diagnostics page
If Not GetPage() = 5 Then
     MsgBox "Please switch to Diagnostics page…"
End If
```

See also:
SetPage()

GetParam

Function GetParam(ParamName As String) As Double

This function allows a number of Mach3 internal parameters (not to be confused with G-code parameters) to be read. Each Mach3 parameter is identified by name. The current value of the parameter whose name is given by ParamName is returned as a Double. Valid parameters are:

Parameter	***Description***
XMachine	X axis machine position
YMachine	Y axis machine position
ZMachine	Z axis machine position
Encoder1	Encoder1 Count
Encoder2	Encoder2 Count
Encoder3	Encoder3 Count
Encoder4	Encoder4 Count
MPG1	MPG1 Count
MPG2	MPG2 Count
MPG3	MPG3 Count
XScale	X axis scale factor
YScale	Y axis scale factor
ZScale	Z axis scale factor
AScale	A axis scale factor
BScale	B axis scale factor
CScale	C axis scale factor
FeedRate	Feed rate
Units	Current units (inch/mm). 0 = mm, 1 = inch
StepsPerAxisX	X axis steps per unit

StepsPerAxisY	Y axis steps per unit
StepsPerAxisZ	Z axis steps per unit
StepsPerAxisA	A axis steps per unit
StepsPerAxisB	B axis steps per unit
StepsPerAxisC	C axis steps per unit
VelocitiesX	X axis maximum velocity, from motor tuning, in units/second
VelocitiesY	Y axis maximum velocity, from motor tuning, in units/second
VelocitiesZ	Z axis maximum velocity, from motor tuning, in units/second
VelocitiesA	A axis maximum velocity, from motor tuning, in units/second
VelocitiesB	B axis maximum velocity, from motor tuning, in units/second
VelocitiesC	C axis maximum velocity, from motor tuning, in units/second
AccelerationX	X axis maximum acceleration, from motor tuning
AccelerationY	Y axis maximum acceleration, from motor tuning
AccelerationZ	Z axis maximum acceleration, from motor tuning
AccelerationA	A axis maximum acceleration, from motor tuning
AccelerationB	B axis maximum acceleration, from motor tuning
AccelerationC	C axis maximum acceleration, from motor tuning
SpindleSpeed	Should modify Spindle Speed, but does not work in all versions. User SetSpinSpeed() instead.
ZInhibitOn	Z Inhibit Enable. 0=Z inhibit disabled, 1=Z inhibit enabled. When Z inhibit is enabled, the Z axis will not be allowed to move below the depth specified by the ZinhibitDepth parameter.
ZInhibitDepth	Z Inhibit Depth. When Z inhibit is enabled, the Z axis will not be allowed to move below the depth specified by the ZinhibitDepth parameter.
SafeZ	SafeZ height
XDRO	X axis DRO
YDRO	Y axis DRO
ZDRO	Z axis DRO
ADRO	A axis DRO
BDRO	B axis DRO
CDRO	C axis DRO
Boundry	Toolpath Boundaries display enable. 0=>disable boundaries display, 1=>enable boundaries display
XRefPer	X axis homing speed, as % of rapid speed
YRefPer	Y axis homing speed, as % of rapid speed
ZRefPer	Z axis homing speed, as % of rapid speed
ARefPer	A axis homing speed, as % of rapid speed
BRefPer	B axis homing speed, as % of rapid speed
CRefPer	C axis homing speed, as % of rapid speed
TotalHours	Running count of total Mach3 up-time

Arguments:

ParamName is the String name of the parameter to be returned. This must be one of the above names.

Return Value:

Current value of the requested parameter as a Double

Example:

```
' Get the new scale factor from the user
ScaleFactor = Question "Enter new scale factor:"
' Set the new scale factor for X/Y/Z
SetParam("Xscale", ScaleFactor)
SetParam("Yscale", ScaleFactor)
SetParam("Zscale", ScaleFactor)
```

See also:

SetParam()

GetPortByte

Function GetPortByte(PortAddr As Integer) As Integer

This function reads the 8-bit PC I/O port whose address is given by PortAddr, and returns the 8-bit data value read from the port as an unsigned integer value. This function can be used for reading hardware devices not directly supported by Mach3.

Note that this function is available only when the parallel port driver is loaded.

Arguments:

PortAddr is the Integer address of the port to be read

Return Value:

Unsigned 8-bit integer value read from the port.

Example:

```
' Our port address
PortAddr = 1016 ' 0x3f8
' Read data register of parallel port at 0x3f8
PortData = GetPortByte(PortAddr)
```

See also:

PutPortByte()

GetToolParam

Function GetToolParam(ToolNum As Integer, ParamNum As Integer)

This function allows any tool parameter, except the description text, for any tool to be read. ToolNum is the number of the tool whose parameters are being set, and can be from 1 to 255. ParamNum is a parameter number, defined as follows:

For Mach3Mill:
- 1 = Diameter
- 2 = Z Offset
- 3 = X Wear
- 4 = Z Wear

For Mach3Turn:
- 1 = Tip Type
- 2 = Tool Radius
- 3 = X Offset
- 4 = Z Offset
- 5 = X Wear
- 6 = Z Wear
- 7 = Turret Angle

Arguments:
ToolNum is an Integer tool number, and must be between 1 and 255.

Return Value:
Requested parameter value, as a Double

Example:
```
‘ Define some constants
DiameterParam = 1
ZoffsetParam = 2
XwearParam = 3
ZwearParam = 4
‘ Display tool #23 parameters
Diam = GetToolParam(23, DiameterParam)
Length = GetToolParam(23, ZoffsetParam)
Xwear = GetToolParam(23, XwearParam)
Zwear = GetToolParam(23, ZwearParam)
Desc = GetToolDesc(23)
Message “Tool 23: Diam=“ & Diam & “ Length=” & Length
_
& “ Xwear=” & Xwear & “ Zwear=” & Zwear & “ Desc=” &
Desc
```

See also:
SetToolParam(), GetToolDesc()

GetRPM

Function GetRPM() As Double

This function returns the currently commanded spindle speed (S-word) as a Double. Note that this returns the most recent S-word value, and not the actual spindle RPM.

Arguments:
None

Return Value:
Currently commanded spindle speed as a Double.

Example:

```
' Show current S-word
Message "S-Word = " & GetRPM()
```

See also:
SetSpinSpeed(), DoSpinCW(), DoSpinCCW(), DoSpinStop()

GetSafeZ

Function GetSafeZ() As Double

This function returns the current SafeZ height.

Arguments:
None

Return Value:
Current SafeZ height as a Double.

Example:

```
' Get current SafeZ height
OldSafeZ = GetSafeZ()
' Set new SafeZ height
NewSafeZ = 1.5750
SetSafeZ(NewSafeZ)
…
' Restore old SafeZ height
SetSafeZ(OldSafeZ)
```

See also:
SetSafeZ()

GetScale

Function GetScale(Axis As Integer) As Double

This function returns the current scale factor for axis *Axis*.

Arguments:
Axis is the Integer Axis. 0=X, 1=Y, 2=Z, 3=A, etc.

Return Value:
Current scale factor for specified axis, as a Double

Example:
```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Get the current axis scale factors
OldXScaleFactor = GetScale(Xaxis)
OldYScaleFactor = GetScale(Yaxis)
OldZScaleFactor = GetScale(Zaxis)
 ' Get the new scale factor from the user
ScaleFactor = Question("Enter new scale factor:")
' Set new scale factors for X/Y/Z
SetScale(Xaxis, ScaleFactor)
SetScale(Yaxis, ScaleFactor)
SetScale(Zaxis, ScaleFactor)
```

See also:
SetScale()

GetSelectedTool

Function GetSelectedTool() As Integer

This function returns the tool specified by the most recent tool change (M6) command. This function is typically used in the M6Start macro to make the selected tool the current tool.

Arguments:
None

Return Value:
Selected tool as an Integer

Example:
```
' Sample M6Start macro
' Get selected tool
NewTool = GetSelectedTool()
```

```
' Make it the current tool
SetCurrentTool(NewTool)
```

See also:
GetCurrentTool(), SetCurrentTool()

GetSetupUnits

Function GetSetupUnits() As Integer

This function returns the native setup units of the machine. The native setup units are a characteristic of machine and the return value of this API does not change with the use of G20/G21.

Arguments:
None

Return Value:
0 = mm units
1 = inch units

Example:

```
' display the machine's setup units

Case Select GetSetupUnits()
Case 0:
     MsgBox "Setup units are millimeters"
Case 1:
     MsgBox "Setup units are inches"
Else:
     MsgBox "Unknown setup units value"
End case
```

See also:
n/a

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

GetTimer

Function GetTimer(TimerNum As Integer) As Double

This function returns the current count for the specified timer. Mach3 provides 25 timers, numbered 0 to 24, which can be used for timing in CB scripts. To time an event, first

clear the timer using SetTimer(), then use GetTimer() to read the timer. Note that this function works only with the parallel port driver, and support for this function may be removed without notice in a future release.

Arguments:
TimerNum is an Integer timer number, which must be between 0 and 24.

Return Value:
Double value of timer TimerNum.

Example:

```
' Clear timer 15
SetTimer(15)
' Wait for OEM Trigger 10 to go active
While IsActive(OEMTRIG10) = False Then
     Sleep 10
Wend
' See how long it took
Message "OEMTRIG10 active after " & GetTimer(15) & "
seconds"
```

See also:
SetTimer()

GetToolChangeStart

Function GetToolChangeStart(Axis As Integer) As Double

This function returns the position of the specified axis at the time a tool change started. This is typically used in an M6End macro to restore the axis positions to the positions they were in before the tool change.

Arguments:
Axis is the Integer Axis. 0=X, 1=Y, 2=Z, 3=A, etc.

Return Value:
Double position of the specified axis at the start of the last tool change.

Example:

```
' Example M6End macro
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Move all axes back to where they were before the
tool change
Xpos = GetToolChangeStart(Xaxis)
Ypos = GetToolChangeStart(Yaxis)
```

```
Zpos = GetToolChangeStart(Zaxis)
Code “G00 X” & Xpos & “ Y” & Ypos & “ Z” & Zpos
‘ Wait for move to complete
While IsMoving()
     Sleep 100
Wend
```

See also:
GetSelectedTool(), GetCurrentTool(), SetCurrentTool()

GetToolDesc

Function GetToolDesc(ToolNum As Integer) As String

This function returns the tool descriptor text for the tool specified by ToolNum.

Arguments:
ToolNum is an Integer tool number, from 1 to 255

Return Value:
Tool table description text for specified tool, as a String

Example:

```
‘ Show user the current tool description
Message “Tool “ & GetCurrentTool() & “: “ & _
GetToolDesc(GetCurrentTool())
```

See also:
GetToolParam(), SetToolParam(), SetToolDesc()

GetTurretAng

Function GetTurretAng() As Double

This function returns the current lathe tool turret angle.

Arguments:
None

Return Value:
Current lathe tool turret angle, as a Double

Example:

```
‘ Display the current tool turret angle
Message “Tool turret position = “ & GetTurretAng()
```

See also:

GetUserDRO

Function GetUserDRO(DRONum As Integer) As Double

This function returns the value of User DRO DRONum as a Double.

Arguments:

DRONum is the Integer User DRO number to be set. Valid User DRO numbers range from 1000-2254.

Return Value:

Current value of UserDRO DRONum as a Double

Example:

```
‘ Define some constants
MyWidgetDRO = 1125
‘ Set MyWidgetDRO to 1.234
SetUserDRO(MyWidgetDRO, 1.234)
…
‘ Get current value of MyWidgetDRO
MyDROVal = GetUserDRO(MyWidgetDRO)
```

See also:

GetUserDRO(), SetOEMDRO(), GetOEMDRO()

GetUserLabel

Function GetUserLabel(LabelNum As Integer) As String

This function allows the user to retrieve the value of an on-screen User label. “User” labels are those that are created in the screen designer with the default text containing the String “UserLabel” followed by one or more digits.

Arguments:

LabelNum is the numeric portion of the user label default text. LabelNum must be between 0 and 255.

Return Value:

None

Example:

```
‘ Change the text in UserLabel25
SetUserLabel(25, “This is Label 25”)
```

```
…
‘ Retrieve the text from UserLabel25
LabelText = GetUserLabel(25)
```

See also:

SetUserLabel()

GetUserLED

Function GetUserLED(LEDNum As Integer) As Integer

This function allows the current state of a User LED to be retrieved.

Arguments:

LEDNum is the User LED whose state is to be retrieved, which must be in the range of 1000 to 2254

Return Value:

Current state of the specified User LED. 0 indicates the LED is off (unlit), 1 indicates the LED is on (lit).

Example:

```
‘ Define some constants
FluxCapacitorControl = OUTPUT1  ‘ Output that controls
the flux capacitor
FluxCapacitorLED = 1234  ‘ LED that indicates flux
capacitor is active
‘ Turn on the Flux capacitor
ActivateSignal(FluxCapacitorControl)
‘ Turn on the Flux Capacitor LED for the operator
SetUserLED(FluxCapacitorLED, 1)
…
‘ Is the Flux Capacitor on?
FluxCapacitorOn = GetUserLED(FluxCapacitorLED)
```

See also:

GetUserLED(), SetOEMLED(), GetOEMLED()

GetVar

Function GetVar(VarNum As Integer) As Double

This function returns the current value of the Mach variable specified by VarNum as a Double. Mach variables are accessible both to CB scripts, using the SetVar() and GetVar() functions, as well as G-code programs, using the #nnnn syntax.

Arguments:
VarNum is Integer the number of the Mach variable to be retrieved.

Return Value:
Current value of the specified variable, as a Double.

Example:

```
' Set a variable 1234 to our target position of 2.3456
SetVar(1234, 2.3456)
' Now move X to our target position
Code "G0 X #1234"
' Or, another way…
Code "G0 X " & GetVar(1234)
```

See also:
SetVar()

GetXCoor

Function GetXCoor() As Double

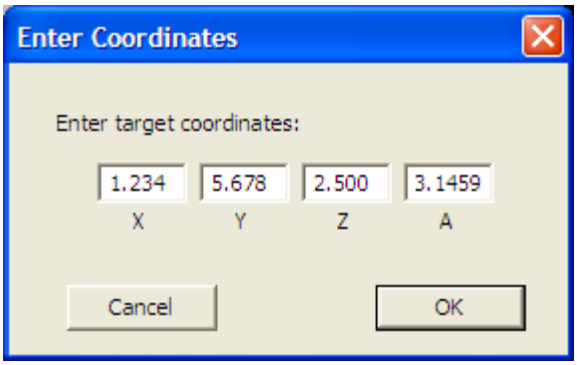
This function is used in conjunction with the GetCoord() function to get X, Y, Z and A axis coordinate values from the user. The GetXCoor() function will return the X value entered by the user in the last GetCoord() function call.

Arguments:
None

Return Value:
X Axis coordinate value from last GetCoord() call

Example:

```
GetCoord("Enter target coordinates:")
Message "Coordinates are: " & GetXCoor() & " " &
GetYCoor() _
& " " & GetZCoor() & " " & GetACoor()
```



See also:

GetCoord(), GetYCoor(), GetZCoor(), GetACoor()

GetYCoor

Function GetYCoor() As Double

This function is used in conjunction with the GetCoord() function to get X, Y, Z and A axis coordinate values from the user. The GetYCoor() function will return the Y value entered by the user in the last GetCoord() function call.

Arguments:

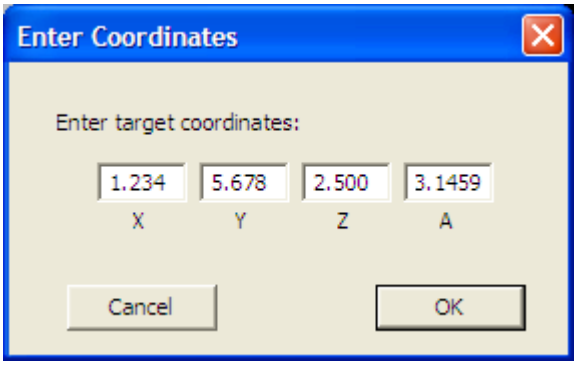
None

Return Value:

Y Axis coordinate value from last GetCoord() call

Example:

```
GetCoord("Enter target coordinates:")
Message "Coordinates are: " & GetXCoor() & " " &
GetYCoor() _
& " " & GetZCoor() & " " & GetACoor()
```



See also:

GetCoord(), GetXCoor(), GetZCoor(), GetACoor()

GetZCoor

Function GetZCoor() As Double

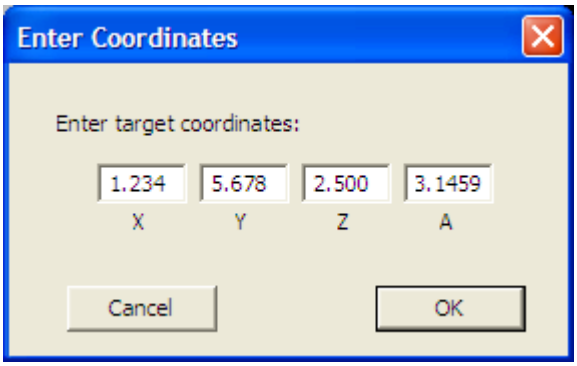
This function is used in conjunction with the GetCoord() function to get X, Y, Z and A axis coordinate values from the user. The GetZCoor() function will return the Z value entered by the user in the last GetCoord() function call.

Arguments:
None

Return Value:
Z Axis coordinate value from last GetCoord() call

Example:

```
GetCoord("Enter target coordinates:")
Message "Coordinates are: " & GetXCoor() & " " &
GetYCoor() _
& " " & GetZCoor() & " " & GetACoor()
```



See also:
GetCoord(), GetXCoor(), GetYCoor(), GetACoor()

GotoSafeZ

Sub GotoSafeZ()

This function will move the Z axis to the Safe_Z position, if Safe_Z is enabled in Config->Safe_Z Setup. If Safe_Z is not enabled, an error message will be displayed on the status line, and no move takes place.

Arguments:
None

Return Value:
None

Example:
```
' Move Z axis to Safe_Z position
GotoSafeZ()
```

See also:
GetSafeZ(), SetSafeZ()

HelpAbout

Sub HelpAbout()

This function displays a dialog box showing the current version of the scripting engine.

Arguments:
None

Return Value:
None

Example:
```
' Show the user the scripting engine version
HelpAbout()
```

See also:

IncludeTLOinZFromG31

Function IncludeTLOinZFromG31() As Boolean

This function is used to retrieve the state of the Mach menu config/general->config dialog option which determines if Mach includes the calculation of the current tool's TLO value in the Z coordinate value returned by a G31 probing operation.

Arguments:
None

Return Value:
True: The option is checked
False: the option is not checked

Example:

```
MsgBox "TLO in G31 option is " &
IncludeTLOinZFromG31()
```

See also:

n/a

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

IsActive

Function IsActive(Signal As Integer) As Boolean

This function returns a Boolean True if the current state of the specified input signal is its active state. Note that in terms of actual signal level, the term "active" depends on how the signal is defined. If the signal is defined in Config->Ports&Pins->InputSignals as ActiveLow, then IsActive() will return True when that signal is being driven to a logic low. If the signal is defined in Config->Ports&Pins->InputSignals as ActiveHigh, then IsActive() will return True when that signal is being driven to a logic high.

Arguments:

Signal is an integer value corresponding to one of pre-defined input signal constants.

Return Value:

False indicates the signal is currently in its inactive state
True indicates the signal is currently in its active state

Example:

```
' Show the user the state of the INPUT #1 input
If IsActive(INPUT1) Then
     Message "INPUT #1 input is active"
Else
     Message "INPUT #1 input is inactive"
End If
```

See also:

IsDiameter

Function IsDiameter() As Integer

This lathe-only function returns 1 if Mach3 is currently operating in Diameter mode.

Arguments:

None

Return Value:

0 indicates Mach3 is currently operating in Radius mode
1 indicates Mach3 is currently operating in Diameter mode

Example:

```
' Tell the user what mode we're in
If IsDiameter() Then
     Message "Currently in Diameter mode"
Else
     Message "Currently in Radius mode"
End If
```

See also:

IsEStop

Function IsEStop() As Integer

This function returns 1 if Mach3 is currently in E-Stop mode.

Arguments:

None

Return Value:

0 indicates Mach3 is currently not in E-Stop mode
1 indicates Mach3 is currently in E-Stop mode

Example:

```
Dim FluxCapacitorControl As Integer
FluxCapacitorControl = OUTPUT3

' Are we in E-Stop?
If IsEStop() Then
     ' Yes, so turn off the flux capacitor
     DeactivateOutput(FluxCapacitorControl)
Else
     ' No, so turn on the flux capacitor
     ActivateOutput(FluxCapacitorControl)
End If
```

See also:

IsLoading

Function IsLoading() As Integer

This function returns 1 if a G-code file is currently being loaded.

Arguments:
None

Return Value:
0 indicates a G-code file is not currently being loaded
1 indicates a G-code file is currently being loaded

Example:

```
Dim FluxCapacitorControl As Integer
FluxCapacitorControl = OUTPUT3

' Are we in loading a new G-code file?
If IsLoading() Then
     ' Yes, so turn off the flux capacitor
     DeactivateOutput(FluxCapacitorControl)
Else
     ' No, so turn on the flux capacitor
     ActivateOutput(FluxCapacitorControl)
End If
```

See also:

IsMoving

Function IsMoving() As Integer

This function returns 1 if any axis is currently moving. This is most often used when commanding motion within a macro, to pause macro execution until the motion is complete. . This is a complement to IsStopped().

Arguments:
None

Return Value:
0 if all axes currently stopped
1 if any axis is currently moving

Example:

```
' Move Z axis to Safe_Z position
GotoSafeZ()
' Wait for SafeZ move to complete
```

```
Sleep(100)
While IsMoving()
     Sleep(100)
Wend
```

See also:

IsStopped()

IsOutputActive

Function IsOutputActive(Signal As Integer) As Boolean

This function returns Boolean True if the current state of the specified output signal is its active state. Note that in terms of actual signal level, the term "active" depends on how the signal is defined. If the signal is defined in Config->Ports&Pins->OutputSignals as ActiveLow, then IsOutputActive() will return a True value when that signal is being driven to a logic low. If the signal is defined in Config->Ports&Pins->InputSignals as ActiveHigh, then IsOutputActive() will return a True value when that signal is being driven to a logic high.

Arguments:

Signal is an integer value corresponding to one of pre-defined output signal constants.

Return Value:

A Boolean value indicating the state of the specified output signal.

Example:

```
' Set OUTPUT1 to its active state
ActivateSignal(OUTPUT1)
' Show the user the state of the OUTPUT #1 input
If IsActive(OUTPUT1) Then
     MsgBox "OUTPUT #1 output is active"
Else
     MsgBox "OUTPUT #1 output is inactive"
End If
' Set OUTPUT1 to its inactive state
DeactivateSignal(OUTPUT1)
' Show the user the state of the OUTPUT #1 input
If IsActive(OUTPUT1) Then
     MsgBox "OUTPUT #1 output is active"
Else
     MsgBox "OUTPUT #1 output is inactive"
End If
```

See also:

IsPeriodicScriptRunning

Function IsPeriodicScriptRunning(ByVal ScriptQFN as String) as Boolean

This function is used to determine if a periodic script has been started.

Arguments:

ScriptQFN: the string of the Qualified File Name (QFN) for the script to check.
The QFN is relative to the Mach install directory.
The QFN passed must be identical to the QFN used to start the periodic script.

Return Value:

True = the Script is running.
False = the script is not running.

Example:

```
' check if the oiler has been started

If IsPeriodicScriptRunning("OilerScript") then
     Msgbox "Oiler script is running."
Else
     Msgbox "Oiler script is not running."
End If
```

See also:

StartPeriodicScript, StopPeriodicScript,

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

IsSafeZ

Function IsSafeZ() As Integer

This function returns 1 if Safe_Z is enabled in Config->Safe_Z Setup.

Arguments:

None

Return Value:

0 indicates Safe_Z is not enabled in Config->Safe_Z Setup
1 indicates Safe_Z is enabled in Config->Safe_Z Setup

Example:

```
' Show user IsSafeZ()
If IsSafeZ() Then
     Message "SafeZ  is enabled"
Else
     Message "SafeZ is disabled"
End If
```

See also:

IsStopped

Function IsStopped() As Integer

This function returns 1 if all axes are currently stopped. This is most often used when commanding motion within a macro, to pause macro execution until the motion is complete. This is a complement to IsMoving().

Arguments:
None

Return Value:
0 if any axis is currently moving
1 if all axes are currently stopped

Example:

```
' Move Z axis to Safe_Z position
GotoSafeZ()
' Wait for SafeZ move to complete
Sleep(100)
While Not IsStopped()
     Sleep(100)
Wend
```

See also:
IsMoving()

IsSuchSignal

Function IsSuchSignal(SignalID As Integer) As Integer

This function returns an Integer value indicating whether the specified signal is defined in Config->Ports&Pins. A 0 return value indicates the signal is not defined, while a non-zero return value indicates the signal is defined. This can be used, for example, to ensure a PROBE input is properly defined before trying to do probing.

Arguments:

SignalID must be one of the pre-defined Mach3 CB output signal constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:

Signal definition state, as an Integer. 0 => signal is not defined, 1 => signal is defined

Example:

```
' Is a YHOME input signal properly defined?
If IsSuchSignal(YHOME) = 0 Then
     Message "Error! No YHOME input is defined"
End If
```

See also:

JogOff

Function JogOff(Axis As Integer)

This function is used to stop jogging of the specified axis. It is typically used in conjunction with JogOn() to jog an axis under control of a script.

Arguments:

Axis is an Integer, specifying the axis for which to disable jogging. 0=X, 1=Y, 2=Z, 3=A, etc.

Return Value:

None

Example:

```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
Plus = 0
Minus = 1
' Jog the Y axis in the minus direction for one second
JogOn(Yaxis, Minus)
Sleep(1000)
' Now stop it
```

```
JogOff(Yaxis)
```

See also:

JogOn()

JogOn

Function JogOn(Axis As Integer, Dir as Integer)

This function is used to start an axis jogging in a specified direction at the current default jog speed. Once the axis starts jogging, it will continue until stopped by means if the JogOff() function, execution of a Stop button command, E-stop, or hitting a limit.

Arguments:

Axis is an Integer, specifying the axis for which to disable jogging. 0=X, 1=Y, 2=Z, 3=A, etc.

Dir is the direction in which to jog. 0=+ direction, 1=- direction

Return Value:

None

Example:

```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
Plus = 0
Minus = 1
' Jog the Y axis in the minus direction for one second
JogOn(Yaxis, Minus)
Sleep(1000)
' Now stop it
JogOff(Yaxis)
```

See also:

JogOff()

LoadFile

Sub LoadFile(FilePath As String)

This function loads the G-code file specified by FilePath.

Arguments:

FilePath is the full file system path to the G-code file to be loaded

Return Value:
None

Example:
```
' Load the roadrunner demo file
LoadFile("C:\Mach3\Gcode\roadrunner.tap")
' Now run it
RunFile()
```

See also:
RunFile()

LoadRun

Sub LoadRun(Filepath As String)

This function loads the G-code file specified by Filepath, then immediately begins execution. This is exactly equivalent to a LoadFile() followed by a RunFile().

Arguments:
Filepath is the String path to the G-code file to be run.

Return Value:
None

Example:
```
' Load and run the roadrunner demo file
LoadRun("C:\Mach3\Gcode\roadrunner.tap")
```

See also:
LoadFile(), RunFile(), Filename(), IsLoading()

LoadStandardLayout

Sub LoadStandardLayout()

This function re-loads the current default screen set. In most cases, this will be the currently loaded screen set. This is used primarily to re-load the default screen set when exiting a Wizard.

Arguments:
None

Return Value:
None

Example:

```
' Load default screen set
LoadStandardLayout()
```

See also:

ToggleScreens()

LoadTeachFile

Sub LoadTeachFile()

This function loads the most recent Teach file into Mach3 for execution. The file must first have been opened or created using OpenTeachFile() or AppendTeachFile() during the current Mach3 session.

Arguments:

None

Return Value:

None

Example:

```
' Create a new Teach File in Mach3\Gcode
MyTeachFile = "TeachMe.nc"
Err = OpenTeachFile(MyTeachFile)
If Err <> 0 Then
     ' Teach file created successfully
     ' Write some G Code to it
     Code "G00 X0 Y0"
     Code "G02 X0 Y0 I-1 J0 F40"
     ' Close the Teach file
     CloseTeachFile()
     ' Now load the teach file for execution
     LoadTeachFile()
Else
     ' OpenTeachFile failed
     Message "Unable to open Teach File"
End If
```

See also:

OpenTeachFile(), AppendTeachFile(), CloseTeachFile()

LoadWizard

Sub LoadWizard(WizardName As String)

This function loads and runs the specified wizard. Wizards reside in the Mach3\Addons directory, and each wizard consists of a number of files with a subdirectory of the Addons directory. The WizardName String passed as the argument to LoadWizard must be just the name of the top-level subdirectory for the wizard.

Arguments:

WizardName is the name of the Wizard to be run, as a String

Return Value:

None

Example:

```
' Load and run the "Circular Pocket" Wizard
LoadWizard("Circular Pocket")
```

See also:

MachMsg

Function MachMsg(Prompt As String, Title As String, DialogType As Integer) As Integer

This function display a dialog box with one of several different combinations of buttons, and waits for the user to click one of the buttons. The Title argument String is displayed in the title bar of the dialog. The Message argument String is displayed in the client portion of the dialog, above the buttons. Clicking on any button closes the dialog, and the return value of the function indicates which button the user clicked.

Arguments:

Message is the String to be displayed in the client area of the dialog, above the button(s)
Title is the String to be displayed in the title bar of the dialog
DialogType is an Integer value which defines which buttons will be displayed on the dialog as follows:

0 = OK button
1 = OK, Cancel buttons
2 = Abort, Retry, Ignore buttons
3 = Yes, No, Cancel buttons
4 = Yes, No buttons
5 = Retry, Cancel buttons

6 = Cancel, Try Again, Continue buttons

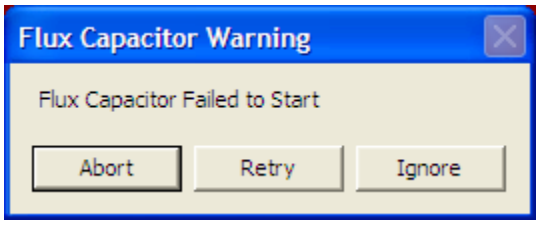
Return Value:

Integer value indicating which button the user clicked, as follows:

OK	= 1
Cancel	= 2
Abort	= 3
Retry	= 4
Ignore	= 5
Yes	= 6
No	= 7
Try Again	= 10
Continue	= 11

Example:

```
‘ Define some constants for MachMsg dialog types
MachMsgTypeOK = 0
MachMsgTypeOKCancel = 1
MachMsgTypeAbortRetryIgnore = 2
MachMsgTypeYesNoCancel = 3
MachMsgTypeYesNo = 4
MachMsgTypeRetryCancel = 5
MachMsgTypeCancelTryAgainContinue = 6
‘ Define some constants for MachMsg return codes
MachMsgReturnOK = 1
MachMsgReturnCancel = 2
MachMsgReturnAbort = 3
MachMsgReturnRetry = 4
MachMsgReturnIgnore = 5
MachMsgReturnYes = 6
MachMsgReturnNo = 7
MachMsgReturnTryAgain = 10
MachMsgReturnContinue = 11
‘ Display an Abort/Retry/Ignore dialog
Ret = MachMsg(“Flux Capacitor Failed to Start”, _
     “Flux Capacitor Warning”,
MachMsgTypeAbortRetryIgnore)
If Ret = MachMsgReturnAbort Then
     ‘ Handle Abort here
ElseIf Ret = MachMsgReturnRetry Then
     ‘ Handle Retry here
ElseIf Ret = MachMsgReturnIgnore Then
     ‘ Handle Ignore here
End If
```



See also:
Message(), Question(), AskTextQuestion(), GetCoord(),

MaxX

Function MaxX() As Double

This function returns the maximum X extent of the currently open G-code file as a Double

Arguments:
None

Return Value:
Maximum X extent of currently loaded G-code file as a Double

Example:

```
‘ Show user the current program extents
Message "MinX=" & MinX() & " MaxX=" & MaxX() & _
" MinY=" & MinY() & " MaxY=" & MaxY()
```

See also:
MinX(), MinY, MaxY()

MaxY

Function MaxY() As Double

This function returns the maximum Y extent of the currently open G-code file as a Double

Arguments:
None

Return Value:
Maximum Y extent of currently loaded G-code file as a Double

Example:

```
‘ Show user the current program extents
Message "MinX=" & MinX() & " MaxX=" & MaxX() & _
```

```
" MinY=" & MinY() & " MaxY=" & MaxY()
```

See also:
MaxX(), MinY, MaxY()

Message

Sub Message(MessageText As String)

This function displays MessageText on the status line.

Arguments:
MessageText is the String text to display on the status line.

Return Value:
None

Example:

```
‘ Display the current tool number on the status line
Message “Current tool is “ & GetCurrentTool()
```

See also:
Message(), Question(), Ask TextQuestion(), MachMsg(), GetCoord()

MinX

Function MinX() As Double

This function returns the minimum X extent of the currently open G-code file as a Double

Arguments:
None

Return Value:
Minimum X extent of currently loaded G-code file as a Double

Example:

```
‘ Show user the current program extents
Message "MinX=" & MinX() & " MaxX=" & MaxX() & _
" MinY=" & MinY() & " MaxY=" & MaxY()
```

See also:
MinX(), MinY, MaxY()

MinY

Function MinY() As Double

This function returns the minimum Y extent of the currently open G-code file as a Double

Arguments:
None

Return Value:
Minimum Y extent of currently loaded G-code file as a Double

Example:

```
' Show user the current program extents
Message "MinX=" & MinX() & " MaxX=" & MaxX() & _
" MinY=" & MinY() & " MaxY=" & MaxY()
```

See also:
MinX(), MinY, MaxY()

nFmt

Function nFmt(Val As Double, Digits As Integer) As Double

This function rounds a Double value to the specified number of decimal places

Arguments:
Val is the Double value to be rounded
Digits is the number of digits right of the decimal place to round to

Return Value:
Rounded vale, as a Double

Example:

```
' Round 1.23456789 to 4 decimal places
' Display it – will display as 1.2345
Message nFmt(1.23456789, 4)
```

See also:

NotifyPlugins

Sub NotifyPlugins(Event As Integer)

This function invokes the MyNotify method of all currently loaded Plug-ins, passing Event as the argument.

Arguments:
Event is an Integer value that will be passed as the argument to the MyNotify methods of all loaded plug-ins.

Return Value:
None

Example:

See also:

NumberPad

Function NumberPad (ByVal PadTitle As string) As Double

This function displays a number pad for data tnetry. The pad is large to allow easy use with touch screens. The number pad size can also be changed via dragging a border.

Arguments:
PadTitle is the string to display as the title of the number pad.

Return Value:
The value entered into the numberpad.

Example:

```
Option explicit

Dim d As Double
d = NumberPad("test num pad")
MsgBox "value entered = " & d
```

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

OpenDigFile

Sub OpenDigFile()

This function opens a digitization log file. When OpenDigFile is executed a FileOpen dialog will be displayed, and the user can select an existing file, or enter a new file name,

to which digitization (probing) data points will be written. Once opened, G31 commands will cause the probe trigger position to be written to the digitization file. When digitization is complete, CloseDigFile() can be used to close the file.

Arguments:
None

Return Value:
None

Example:

See also:

OpenTeachFile

Function OpenTeachFile (Filename As String) As Integer

This function re-opens an existing Teach file at Gcode\Filename in the Mach3 directory, and appends any commands subsequently executed via MDI or Code() to that file, until CloseTeachFile() is executed.

Arguments:
F

Return Value:
A

Example:

```
' Create a new Teach File in Mach3\Gcode
MyTeachFile = "TeachMe.nc"
Err = OpenTeachFile(MyTeachFile)
If Err <> 0 Then
     ' Teach file created successfully
     ' Write some G Code to it
     Code "G00 X0 Y0"
     Code "G02 X0 Y0 I-1 J0 F40"
     ' Close the Teach file
     CloseTeachFile()
     ' Now load the teach file for execution
     LoadTeachFile()
Else
     ' OpenTeachFile failed
     Message "Unable to open Teach File"
End If
```

See also:
AppendTeachFile(), CloseTeachFile()

Param1

Function Param1() As Double

This function returns the value of the P parameter passed to an M-macro.

Arguments:
None

Return Value:
Double value passed a P parameter to M-macro

Example:

```
' This macro expects three arguments: P, Q, & R
' If put into an M-macro, and invoked via MDI, it will
' display the argument values on the status line
' For example, if using M1200:
'    M1200 P1.234 Q2.345 R3.456
' Executing the above line to MDI will display:
'    P=1.234 Q=2.345 R=3.456
Parg = Param1()
Qarg = Param2()
Rarg = Param3()
Message "P=" & Parg & "P=" & Qarg & "R=" & Rarg
```

See also:
Param2(), Param3()

Param2

Function Param2() As Double

This function returns the value of the Q parameter passed to an M-macro.

Arguments:
None

Return Value:
Double value passed a Q parameter to M-macro

Example:

```
' This macro expects three arguments: P, Q, & R
```

```
' If put into an M-macro, and invoked via MDI, it will
' display the argument values on the status line
' For example, if using M1200:
'    M1200 P1.234 Q2.345 R3.456
' Executing the above line to MDI will display:
'    P=1.234 Q=2.345 R=3.456
Parg = Param1()
Qarg = Param2()
Rarg = Param3()
Message "P=" & Parg & "P=" & Qarg & "R=" & Rarg
```

See also:
Param1(), Param3()

Param3

Function Param3() As Double

This function returns the value of the R parameter passed to an M-macro.

Arguments:
None

Return Value:
Double value passed a R parameter to M-macro

Example:

```
' This macro expects three arguments: P, Q, & R
' If put into an M-macro, and invoked via MDI, it will
' display the argument values on the status line
' For example, if using M1200:
'    M1200 P1.234 Q2.345 R3.456
' Executing the above line to MDI will display:
'    P=1.234 Q=2.345 R=3.456
Parg = Param1()
Qarg = Param2()
Rarg = Param3()
Message "P=" & Parg & "P=" & Qarg & "R=" & Rarg
```

See also:
Param1(), Param2()

PlayWave

Sub PlayWave(Filename As String)

This function plays a .WAV file through the PCs audio system. Filename gives the name of the WAV file to be played.

Arguments:
Filepath is the String file path to the WAV file to be played

Return Value:
None

Example:

```
PlayWave("C:\WINDOWS\Media\Windows XP Startup.wav")
```

See also:

ProgramSafetyLockout

Function ProgramSafetyLockout() As Boolean

This function is used to retrieve the state of the Mach menu config/general-config dialog option which determines if input 1 is being used to control the Mach G-Code inhibit feature.

Arguments:
None

Return Value:
True: The option is checked
False: the option is not checked

Example:

```
MsgBox "Program Safety Lockout " &
ProgramSafetyLockout()
```

See also:
n/a

First Mach3 version with API:
This API was first implemented in Mach3 version 3.43.06.

PutPortByte

Sub PutPortByte(PortAddr As Integer, Value As Integer)

This function writes the 8-bit data given by Value to the 8-bit PC I/O port whose address is given by PortAddr. This function can be used for writing hardware devices not directly supported by Mach3.

Arguments:

PortAddr is the Integer address of the port to be written
Value is the Integer value to be written. Only the 8 least significant bits (LSBs) are written.

Return Value:

None

Example:

```
' Our port address
PortAddr = 1016 ' 0x3f8
PortData = 154  ' 0xa5
' Write 0xa5 to data register of parallel port at
0x3f8
PortData = PutPortByte(PortAddr, PortData)
```

See also:

GetPortByte()

Question

Function Question(Prompt As String) As Double

This function displays a dialog box with an OK button, and a text box into which the user can enter a numeric value. The Prompt string is displayed above the test box. This value is returned as a Double. This can be used for getting a single numeric value from the user.

Arguments:

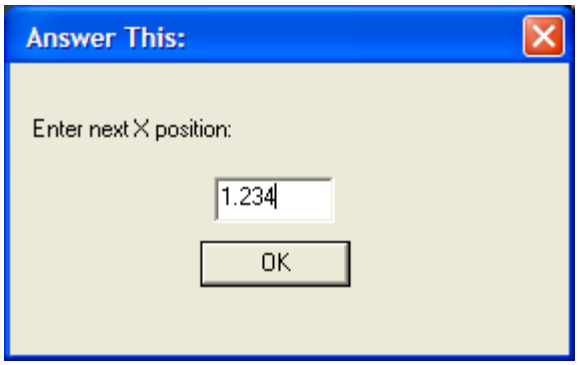
Prompt is the prompt string displayed in the Question dialog.

Return Value:

Double value entered into text box by user

Example:

```
' Get next X position from user
NextPos = Question("Enter next X position:")
' Go there
Code "G00 X" & NextPos
```



See also:
Message(), Question(), AskTextQuestion(), MachMsg(), GetCoord()

QueueDepth

Function QueueDepth() As Integer

This function returns the current number of entries in the trajectory planning queue.

Arguments:
None

Return Value:
Number of trajectory planner queue entries as an Integer

Example:

```
' Show current queue depth
Message "Queue depth = " & QueueDepth()
```

See also:

Random

Function Random() As Double

This function returns a pseudo-random number between 0 and 1. This can be used in place of the CB rnd() function.

Arguments:
None

Return Value:
Next pseudo-random number in sequence, as a Double

Example:

```
' Get next random number
NextRand = Random()
```

See also:

RefCombination

Sub RefCombination(Axes As Integer)

This function allows any combination of axes to be simultaneously referenced (homed). Which axes will be referenced is determined by the Axes argument, which is a bit-mapped variable, with the bits mapped as defined below.

Arguments:

Axes is a bit-mapped value the defines which axes are to be referenced. The value of Axes can be calculated by adding the values corresponding to the individual axes to be referenced. The axis values are:

X = 1
Y = 2
Z = 4
A = 8
B = 16
C = 32

So, for example, to reference the X, Z and B axes, Axes = 1 + 4 + 16 = 21.

Return Value:

None

Example:

```
' Define some constants
RefX = 1
RefY = 2
RefZ = 4
RefZ = 8
RefB = 16
RefC = 32
' Reference Y, Z and C axes
RefCombination(RefY + RefZ + RefC)
```

See also:

VerifyAxis(), SingleVerify(), SingleVerifyReport()

ResetAxisSwap

Sub ResetAxisSwap()

This function un-does the effect of a preceding SwapAxis(). Note that this is the only way to undo a Swap, and it is illegal to perform two SwapAxis() calls without an intervening ResetAxisSwap().

Arguments:
None

Return Value:
None

Example:

```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Swap X and Y axes
SwapAxis(Xaxis, Yaxis)
' Do something with swapped axes here…
' Undo the SwapAxis
ResetAxisSwap()
' Now Swap Y and Z axes
SwapAxis(Yaxis, Zaxis)
```

See also:
SwapAxis()

ResetTHC

Sub ResetTHC()

This function resets the Torch Height. This function is identical to ZeroTHC(), except if a program is running, it does nothing.

Arguments:
None

Return Value:
None

Example:

```
' Reset THC
ResetTHC()
```

See also:

THCOn(), THCOff()

RetractMode

Function RetractMode() As Integer

This function returns the current Peck Drilling retract mode. If a G98 was last run, it will return a 0. If a G99 was most recently run, it will return a 1.

Arguments:

None

Return Value:

Integer value indicating which peck cycle was last run. 0=G98, 1=G99.

Example:

```
' Run a G98
Code "G98 Z-0.5 R0.1 F10"
' Wait for it to complete
While IsMoving()
     Sleep 100
Wend
' Now show retract mode – should show 0
MsgBox("RetractMode = " & RetractMode())
Code "Hit CycleStart to continue…"
' Run a G99
Code "G99 Z-0.5 R0.1 F10"
' Wait for it to complete
While IsMoving()
     Sleep 100
Wend
' Now show retract mode – should show 1
MsgBox("RetractMode = " & RetractMode())
Code "Hit CycleStart to continue…"
```

See also:

roun

Function roun(Val As Double) As Double

This function rounds the Double value specified by Val to four decimal placed.

Arguments:
Val is the Double value to be rounded

Return Value:
Double value of Val rounded to four decimal places

Example:
```
' Round 1.23456789
Message "1.23456789 rounds to " & roun(1.23456789)
```

See also:
Round()

RunFile

Sub RunFile()

This function begins execution of the currently loaded G-code program, if any. This is functionally identical to DoOEMButton(CYCLESTART)

Arguments:
None

Return Value:
None

Example:
```
' Load our G-code file
LoadFile("MyGCode.tap")
' Run it
RunFile()
```

See also:
FileName(), LoadFile(), LoadRun(), IsLoading(), DoOEMButton()

RunScript

Function RunScript(ByVal QFN as string) as Integer

This function can be used to run the script in the specified filename.
The qualified file name (QFN) parameter is relative to the Mach install directory.
The function will load the specified script file, compile it, execute it and return when the script has completed.

Prior to the addition of this call, it was common practice to put script code into a Mxxx.m1s macro, and use the Code call to execute the Mxxx macro. The execution of an Mxxx macro involves the use of the Mach GCode interpreter (as what you are really

doing is executing a GCode M word block) and can result in the programmer having to invent and handle semaphores to coordinate the asynchronous execution of the Mxxx macro.

It is recommended that the use of the Code "as a subroutine call method" be avoided and that, when possible, the RunScript call should be used instead.

Arguments:

QFN: The Qualified File Name (without extension)of the script file to run. The extension is not included as Mach3 will look for either an .m1s or .mcc extension for the named macro.

Return Value:

0 = Script was found and invoked.
< 0 = error condition, script was not run.

Error Return values:

<TBD>

Example:

```
' Run script example
If RunScript("MsgBoxScript") < 0 then
        Msgbox "RunScript returned an error"
Else
        Msgbox "Script ran"
End If
```

Where the file MsgBoxScript.m1s contains

```
MsgBox "Message from script file"
```

Will result show a dialog box with the content "Message from script file". Followed by a dialog that says "Script Ran"

See also:

n/a

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

On 3.43.6 the API was defined as a Boolean function

Return Value:

True = Script was found and invoked.
False = Script was not found.

This API was revised in Mach3 version 3.43.19 to be an integer function.

SaveWizard

Sub SaveWizard()

This function can be used within a wizard to save the user-entered parameter value settings for the wizard, so the same values will be loaded the next time the wizard is invoked.

Arguments:
None

Return Value:
None

Example:

```
' Save current Wizard parameter values
SaveWizard()
```

See also:

SetButtonText

Sub SetButtonText(Text As String)

This function allows the text of an on-screen button to be changed by that buttons associated button script. This can be used to change the text on a button based on the state of a variable or mode.

Arguments:
Text is a String that specifies the new text to be displayed on the button.

Return Value:
None

Example:

```
' Example Spindle On/Off Toggle Button
SpindleCWLED = 11
If GetOEMLED(SpindleCWLED) Then
     ' Spindle is on, so turn it off
     DoSpinStop()
     SetButtonText("Spindle On")
Else
     ' Spindle is off, so turn it on
     DoSpinCW()
```

```
        SetButtonText("Spindle Off")
    End If
```

See also:

SetCurrentTool

Sub SetCurrentTool(ToolNum As Integer)

This function sets the current tool number to ToolNum. This is typically used in the M6Start script to make the selected tool the current tool.

Arguments:

ToolNum is an Integer tool number, from 1 to 255.

Return Value:

None

Example:

```
' Typical M6Start script
' Get selected tool
NewTool = GetSelectedTool()
' Make it the current tool
SetCurrentTool(NewTool)
```

See also:

GetCurrentTool(), GetSelectedTool()

SetDRO

Function SetDRO(DRONum As Integer, DROVal As Double)

This legacy function sets the DRO number specified by DRONum to the value specified by DROVal.

The use of SetDRO is no longer recommended practice and this function exists only to support preexisting legacy scripts. This function is deprecated, and its use is ***strongly*** discouraged.

> Legacy script note: Over time, there have been two different DRO numbering schemes used with Mach; the "DRO number" series and the "OEMDRO number" series. This function uses the "DRO number" series.

The “DRO number” series was further subdivided into “User” and “OEM” ranges. Within the “OEM” range, valid DRONums were from 0 to 200, which, at one time, corresponded to OEM DRO numbers 800 to 1000.

The numerical correspondence between the numbering series is **not guaranteed** for future releases of Mach.

Use the SetOEMDRO and SetUserDRO functions instead of this function.

Arguments:

DRONum is the Integer DRO number to read. The value has to be within the “DRO number series”.

DROVal is the Double value to which the specified DRO will be set.

Return Value:

None

Example:

```
‘ Define the axes
Const XaxisMultiFunctionDRONum = 0
Const YaxisMultiFunctionDRONum = 1
Const ZaxisMultiFunctionDRONum = 2

‘ Write 1.2345 to Z axis DRO using SetDRO
SetDRO(ZaxisMultiFunctionDRONum, 1.2345)

‘ Show the user the Z Axis DRO value, using GetDRO()
MsgBox “After using SetDRO() the Z Axis DRO reads: “ &
GetDRO(ZaxisMultiFunctionDRONum)
```

See also:

SetOEMDRO(), GetOEMDRO(), SetUserDRO(), GetUserDRO(), GetDRO()

SetFeedRate

Sub SetFeedRate(FeedRate As Double)

This function sets the feed rate. Note that FeedRate is specified in units per second, rather than units per minute.

Arguments:

FeedRate specified in units/second, as a Double

Return Value:

None

Example:

```
‘ Set the feed rate to 123.456 inches/minute
SetFeedRate(123.456 / 60)
‘ Get the current feed rate, in inches/minute, and display it
CurrentFeedrate = FeedRate()
‘ Display it on the status line
Message “Current feed rate = “ & CurrentFeedrate
```

See also:

FeedRate()

SetFormula

Sub SetFormula(Formula As String, Axis As Integer)

This function sets one of the axis formulas, accessible by FunctionCfg’s->Formulas. These allow motion for axes to be defined by formulas involving the positions of other axes, using algebraic and trigonometric functions. Note that for Formulas to take effect, you must check the “Formulas enabled” checkbox in FunctionCfg’s->Formulas.

Arguments:

Formula is a String that defines the algebraic/trigonometric function to be used to calculate axis position.

Axis is an Integer value that defines which axis the Formula is to be applied to

Return Value:

None

Example:

```
‘ Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
‘ Define the axis formulas such that a programmed
linear move
‘ Y causes a causes a sinusoidal move in X
‘ Z axis should move normally
SetFormula(“Z”, Zaxis)
SetFormula(“Y”, Yaxis)
SetFormula(“Sin(y)”, Xaxis)
```

See also:

SetIJMode

Sub SetIJMode(IJMode As Integer)

This function sets the IJ mode. IJMode = 0 sets absolute mode, while IJMode = 1 sets incremental mode.

Arguments:

IJMode is the Integer mode to select. 0=Absolute, 1=Incremental

Return Value:

None

Example:

```
' Define some constants
IJAbsolute = 0
IJIncremental = 1
' Set IJMode to incremental
SetIJMode(IJIncremental)
```

See also:

GetIJMode()

SetMachZero

Sub SetMachZero(Axis As Integer)

This function zeroes the machine position of the specified axis to the current position.

Arguments:

Axis is an Integer value identifying the axis to be zeroed. 0=X, 1=Y, 2=Z, 3=Z, etc.

Return Value:

None

Example:

```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
 ' Set machine zero for Y axis to current position
SetMachZero(Yaxis)
```

See also:

SetOEMDRO

Sub SetOEMDRO(DRONum As Integer, DROVal as Double)

This function sets the OEM DRO specified by DRONum to the value specified by DROVal.

Arguments:

DRONum is an Integer value or expression that evaluates to one of the OEM DRO numbers.

Return Value:

None

Example:

```
‘ Define the axes
Const XaxisMultiFunctionOEMDRONum = 800
Const YaxisMultiFunctionOEMDRONum = 801
Const ZaxisMultiFunctionOEMDRONum = 802

‘ Write 1.2345 to Z axis DRO using SetOEMDRO
SetOEMDRO(ZaxisMultiFunctionDRONum, 1.2345)

‘ Show the user the Z Axis DRO value, using
GetOEMDRO()
MsgBox “After using SetOEMDRO() the Z Axis DRO reads:
“ & GetOEMDRO(ZaxisMultiFunctionDRONum)
```

See also:

GetOEMDRO(), SetUserDRO(), GetUserDRO()

SetPage

Sub SetPage(PageNum As Integer)

This function switches the current display page to the one specified by PageNum.

Arguments:

PageNum is the Integer number of the display page to switch to.

Return Value:

None

Example:

```
' For 1024.set, change to Diagnostics page
SetPage(5)
```

See also:

GetPage()

SetParam

Sub SetParam(ParamName As String, ParamVal As Double)

This function allows a number of Mach3 internal parameters (not to be confused with G-code parameters) to be set. Each Mach3 parameter is identified by name. The parameter whose name is given by ParamName is set to the value given by ParamVal. Valid parameters are:

Parameter Name	***Description***
RPMOverRide	When this parameter is set to 1, the True Spindle Speed DRO (OEM DRO 39) is made writeable, over-riding Mach3s normal calculation of True Spindle Speed.
Xscale	X axis scale factor
YScale	Y axis scale factor
ZScale	Z axis scale factor
AScale	A axis scale factor
BScale	B axis scale factor
CScale	C axis scale factor
FeedRate	Feed rate
Units	Current units (inch/mm). 0 = mm, 1 = inch
StepsPerAxisX	X axis steps per unit
StepsPerAxisY	Y axis steps per unit
StepsPerAxisZ	Z axis steps per unit
StepsPerAxisA	A axis steps per unit
StepsPerAxisB	B axis steps per unit
StepsPerAxisC	C axis steps per unit
VelocitiesX	X axis maximum velocity, from motor tuning, in units/second
VelocitiesY	Y axis maximum velocity, from motor tuning, in units/second
VelocitiesZ	Z axis maximum velocity, from motor tuning, in units/second
VelocitiesA	A axis maximum velocity, from motor tuning, in units/second
VelocitiesB	B axis maximum velocity, from motor tuning, in units/second
VelocitiesC	C axis maximum velocity, from motor tuning, in units/second
AccelerationX	X axis maximum acceleration, from motor tuning
AccelerationY	Y axis maximum acceleration, from motor tuning
AccelerationZ	Z axis maximum acceleration, from motor tuning

AccelerationA	A axis maximum acceleration, from motor tuning
AccelerationB	B axis maximum acceleration, from motor tuning
AccelerationC	C axis maximum acceleration, from motor tuning
SpindleSpeed	Should modify Spindle Speed, but does not work on all versions. Use SetSpinSpeed() instead.
ZInhibitOn	Z Inhibit Enable. 0=Z inhibit disabled, 1=Z inhibit enabled. When Z inhibit is enabled, the Z axis will not be allowed to move below the depth specified by the ZinhibitDepth parameter.
ZInhibitDepth	Z Inhibit Depth. When Z inhibit is enabled, the Z axis will not be allowed to move below the depth specified by the ZinhibitDepth parameter.
SafeZ	SafeZ height
Boundry	Toolpath Boundaries display enable. 0=>disable boundaries display, 1=>enable boundaries display
XRefPer	X axis homing speed, as % of rapid speed
YRefPer	Y axis homing speed, as % of rapid speed
ZRefPer	Z axis homing speed, as % of rapid speed
ARefPer	A axis homing speed, as % of rapid speed
BRefPer	B axis homing speed, as % of rapid speed
CRefPer	C axis homing speed, as % of rapid speed
ToolPathLock	Toolpath Lock enable. 0=>Toolpath unlocked, 1=>Toolpath locked. When the tool path is locked, it cannot be moved, scaled or rotated.
PrepMove	Preparatory Move Dialog Inhibit. 0=>Allow showing of Preparatory Move dialogs, 1=>Inhibit showing of Preparatory Move dialogs
AutoToolChange	??? – Indicates auto tool changer in use?
ADirActive	??? – Reverse direction of A axis?

Arguments:

ParamName is the String name of the parameter to be set. This must be one of the above names.

ParamVal is the Double value to which the specified parameter will be set.

Return Value:

None

Example:

```
' Get the new scale factor from the user
ScaleFactor = Question "Enter new scale factor:"
' Set the new scale factor for X/Y/Z
SetParam("Xscale", ScaleFactor)
SetParam("Yscale", ScaleFactor)
SetParam("Zscale", ScaleFactor)
```

See also:

GetParam()

SetPulley

Sub SetPulley(Pulley As Integer)

This function sets the current spindle pulley number. This allows Mach3 to properly scale the spindle speed output based on the current pulley and commanded spindle speed. Pulley ratios and allowable speed ranges are configured in Config->SpindlePulleys

Arguments:
Pulley is an Integer pulley number, from 1 to 15

Return Value:
None

Example:

```
' Prompt the user for new pulley setting
NewPulley = Question "Enter new pulley number:"
' Tell Mach3
SetPulley(NewPulley)
```

See also:

SetSafeZ

Sub SetSafeZ(SafeZ As Double)

This function sets a new value for SafeZ. Note that SafeZ must be enabled in Config->SafeZConfig

Arguments:
SafeZ is the Double value to set SafeZ to

Return Value:
None

Example:

```
' Change SafeZ to +2.000
SetSafeZ(2.000)
```

See also:

SetScale

Sub SetScale(Axis As Integer, Scale As Double)

This function sets the scale factor for axis Axis to the value given by Scale.

Arguments:

Axis is the Integer Axis. 0=X, 1=Y, 2=Z, 3=A, etc.
Scale is the Double Scale factor

Return Value:

None

Example:

```
' Define some constants
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Get the current axis scale factors
OldXScaleFactor = GetScale(Xaxis)
OldYScaleFactor = GetScale(Yaxis)
OldZScaleFactor = GetScale(Zaxis)
 ' Get the new scale factor from the user
ScaleFactor = Question("Enter new scale factor:")
' Set new scale factors for X/Y/Z
SetScale(Xaxis, ScaleFactor)
SetScale(Yaxis, ScaleFactor)
SetScale(Zaxis, ScaleFactor)
```

See also:

GetScale()

SetSpinSpeed

Sub SetSpinSpeed(RPM As integer)

This function sets the spindle speed, in RPM, exactly as the "S" word in G-code does.

Arguments:

RPM is the Integer spindle speed desired.

Return Value:

None

Example:

```
' Turn spindle on at 2500 RPM
SetSpinSpeed(2500)
```

```
DoSpinCW()
```

See also:
DoSpinCW(), DoSpinCCW(), DoSpinStop()

SetTicker

Sub SetTicker(TickerNum As Integer, TickerText As String)

This function loads the text given by TickerText into the ticker specified by TickerNum. A Ticker is a scrolling text label, with its default text set to "Tickernn", where "nn" is a number between 0 and 255. A long message can be put in a Ticker, and it will scroll continuously so the entire message is visible, even if the label is shorter than the text.

Arguments:
TickerNum is the Integer number of the ticker to be written
TickerText is the String text to be written to ticker TickerNum

Return Value:
None

Example:

```
' Write the current file path to Ticker 25
SetTicker(25, FileName())
```

See also:
SetUserLabel(), Message()

SetTimer

Function SetTimer(TimerNum As Integer)

This function clears the specified timer. Mach3 provides 25 timers, numbered 0 to 24, which can be used for timing in CB scripts. To time an event, first clear the timer using SetTimer(), then use GetTimer() to read the timer. Note that this function works only with the parallel port driver, and support for this function may be removed without notice in a future release.

Arguments:
TimerNum is an Integer timer number, which must be between 0 and 24.

Return Value:
None

Example:

```
' Clear timer 15
SetTimer(15)
' Wait for OEM Trigger 10 to go active
While IsActive(OEMTRIG10) = False Then
     Sleep 10
Wend
' See how long it took
Message "OEMTRIG10 active after " & GetTimer(15) & "
seconds"
```

See also:

GetTimer()

SetToolDesc

Function SetToolDesc(ToolNum As Integer, TDesc As String)

This function set the text description of a tool in internal Mach the tool table.

Arguments:

ToolNum is an Integer tool number, from 0 to 254.
TDesc is a string with the description of the tool.

Return Value:

True: description was set for the tool
False: An error occurred attempting to set the tool description.

Example:

```
' Show user the current tool description
Dim TNum as Integer
Dim TDesc as string

TNum = 1  ' want to set description for tool # 1
TDesc = "1/4 135 degree split point drill"

If SetToolDescription(TNum, TDesc) then
     Message "Tool description was set"
Else
     Message "Error setting Tool description"
End if
```

See also:

GetToolDesc(), GetToolParam(), SetToolParam(),

First Mach3 version with API:

This API was first implemented in Mach3 version 3.42.30.

SetToolParam

Sub SetToolParam(ToolNum As Integer, ParamNum As Integer, ParamVal As Double)

This function allows any tool parameter for any tool to be set. ToolNum is the number of the tool whose parameters are being set, and can be from 1 to 255. ParamNum is a parameter number, defined as follows:

For Mach3Mill:
- 1 = Diameter
- 2 = Z Offset
- 3 = X Wear
- 4 = Z Wear

For Mach3Turn:
- 1 = Tip Type
- 2 = Tool Radius
- 3 = X Offset
- 4 = Z Offset
- 5 = X Wear
- 6 = Z Wear
- 7 = Turret Angle

Arguments:

ToolNum is an Integer tool number, and must be between 1 and 255.
ParamNum is an Integer parameter number, defined as described above.

Return Value:

None

Example:

```
‘ Define some constants
DiameterParam = 1
ZoffsetParam = 2
XwearParam = 3
ZwearParam = 4
‘ Define tool #23
SetToolParam(23, DiameterParam,  0.2500)     ‘ Set
tool diameter = 1/4”
SetToolParam(23, ZoffsetParam, 1.2500)  ‘ Set length
offset = 1.25”
SetToolParam(23, XwearParam, 0.0005)    ‘ Set X wear =
0.0005”
```

```
SetToolParam(23, ZwearParam, 0.0013)    ' Set Z wear =
0.0013"
```

See also:
GetToolParam(), GetToolDesc()

SetToolX

Function SetToolX(Pos As Double)

This function sets the X axis DRO to the value given by Pos, then displays the message "Work Offset Shifted" on the status line. Note that with the exception of this message, SetToolX() is functionally identical to using SetOEMDRO to modify the X axis DRO value directly.

Arguments:
Pos is the Double position to which the X axis DRO will be set.

Return Value:
None

Example:

```
' Prompt the user to zero the X axis
Message "Move X axis to zero position.  Press OK when
ready…"
' Zero the X axis DRO
SetToolX(0.0000)
```

See also:
SetToolZ()

SetToolZ

Function SetToolZ(Pos As Double)

This function sets the Z axis DRO to the value given by Pos, then displays the message "Work Offset Shifted" on the status line. Note that with the exception of this message, SetToolZ() is functionally identical to using SetOEMDRO to modify the Z axis DRO value directly.

Arguments:
Pos is the Double position to which the Z axis DRO will be set.

Return Value:
None

Example:

```
' Prompt the user to zero the Z axis
Message "Move Z axis to zero position.  Press OK when
ready…"
' Zero the Z axis DRO
SetToolZ(0.0000)
```

See also:

SetToolX()

SetTriggerMacro

Sub SetTriggerMacro(MacroNum As Integer)

This function allows an M-macro to be associated with OEM code 301. This can be used to cause one of the OEM Trigger signals to automatically run a macro when asserted. To do this, the OEM Trigger signal must first be configured in Config->Ports&Pins->InputSignals. Then the OEM Trigger must be associated with OEM code 301 in Config->SystemHotKeys. Finally, SetTriggerMacro must be used to define which M-macro will be executed when OEM button code 301 is executed.

Arguments:

MacroNum is the number of the M-macro to be run when OEM button code 301 is executed.

Return Value:

None

Example:

```
' Assume OEMTRIGGER5 is assigned to OEM code 301
' Assign the M1025 macro to OEM Code 301
SetTriggerMacro(1025)
' Now, when OEMTRIGGER5 is driven to its active level,
M1025.m1s will be executed
```

See also:

SetUserDRO

Sub SetUserDRO(DRONum As Integer, DROVal As Double)

This function sets the value of User DRO DRONum to DROVal.

Arguments:

DRONum is the Integer User DRO number to be set. Valid User DRO numbers range from 1000-2254.
DROVal is the Double value to which the User DRO will be set

Return Value:
None

Example:
```
' Define some constants
MyWidgetDRO = 1125
' Set MyWidgetDRO to 1.234
SetUserDRO(MyWidgetDRO, 1.234)
…
' Get current value of MyWidgetDRO
MyDROVal = GetUserDRO(MyWidgetDRO)
```

See also:
GetUserDRO(), SetOEMDRO(), GetOEMDRO()

SetUserLabel

Sub SetUserLabel(LabelNum As Integer, LabelText As String)

This function allows the user to change an on-screen "User" labels from CB, rather than by using a screen set editor. "User" labels are those that are created in the screen designer with the default text containing the String "UserLabel" followed by one or more digits.

Arguments:
LabelNum is the numeric portion of the user label default text. LabelNum must be between 0 and 255.
LabelText is the text to be placed into the label.

Return Value:
None

Example:
```
' Change the text in UserLabel25
SetUserLabel(25, "This is Label 25")
…
' Retrieve the text from UserLabel25
LabelText = GetUserLabel(25)
```

See also:

SetUserLED

Sub SetUserLED(LEDNum As Integer, State As Integer)

This function allows the state of a User LED to be set or cleared. User LEDs are numbered from

Arguments:

LEDNum is the User LED to be set, and must be in the range of 1000 to 2254
State is the new state of the User LED. 0 indicates the LED is off (unlit), 1 indicates the LED is on (lit).

Return Value:

None

Example:

```
‘ Define some constants
FluxCapacitorControl = OUTPUT1  ‘ Output that controls
the flux capacitor
FluxCapacitorLED = 1234  ‘ LED that indicates flux
capacitor is active
‘ Turn on the Flux capacitor
ActivateSignal(FluxCapacitorControl)
‘ Turn on the Flux Capacitor LED for the operator
SetUserLED(FluxCapacitorLED, 1)
…
‘ Is the Flux Capacitor on?
FluxCapacitorOn = GetUserLED(FluxCapacitorLED)
```

See also:

GetUserLED(), SetOEMLED(), GetOEMLED()

SetVar

Sub SetVar(VarNum As Integer, Val As Double)

This function sets the Mach variable specified by VarNum to the value given by Val. Mach variables are accessible both to CB scripts, using the SetVar() and GetVar() functions, as well as G-code programs, using the #nnnn syntax.

Arguments:

VarNum is Integer the number of the Mach variable to be set.
Val is the Double value to which the variable will be set

Return Value:

None

Example:

```
' Set a variable 1234 to our target position of 2.3456
SetVar(1234, 2.3456)
' Now move X to our target position
Code "G0 X #1234"
```

See also:

GetVar()

SingleVerify

Sub SingleVerify(Axis As Integer)

This function performs a position verify on a single axis by homing that axis, and, once homed, zeroing the work offset, then returning to the initial offset. This is commonly used at the beginning of a program to ensure the fixture offset is set properly, or after a crash to restore the correct work offset.

Arguments:

Axis in an Integer value that specifies the first axis to verify. 0=X, 1=Y, 2=Z, 3=A, etc.

Return Value:

None

Example:

```
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Verify Z axis work offset is properly set
SingleVerify(Zaxis)
```

See also:

SingleVerifyReport(), VerifyAxis(), RefCombination()

SingleVerifyReport

Function SingleVerifyReport(Axis As Integer)

This function performs a position verify on a single axis by homing that axis, and, once homed, zeroing the work offset, then returning to the initial offset. When complete, a message is displayed showing how far each axis was offset from its correct home position. This function is identical to the SingleVerify function except for this message

display. This is commonly used at the beginning of a program to ensure the fixture offset is set properly, or after a crash to restore the correct work offset.

Arguments:

Axis in an Integer value that specifies the first axis to verify. 0=X, 1=Y, 2=Z, 3=A, etc.

Return Value:

None

Example:

```
Xaxis = 0
Yaxis = 1
Zaxis = 2
' Verify Z axis work offset is properly set, and
report error, if any
SingleVerifyReport(Zaxis)
```

See also:

SingleVerifyReport(), VerifyAxis(), RefCombination()

Sleep

Sub Sleep(Time As Integer)

This function causes the CB program to pause for the specified period of time, in mSec. During this time, other processes can have access to the CPU. A Sleep() call should always be inserted in any loop which might remain active for any period of time, for example the While loop used to wait for a move command to complete.

Arguments:

Time, an Integer value expressing the sleep time in milliseconds.

Return Value:

None

Example:

```
' Move to zero position
Code "G0 X0.000 Y0.000"
' Wait for move to complete
While IsMoving()
     ' Let other processes have CPU while we wait
     Sleep 100
Wend
```

See also:

Speak

Sub Speak(TextToSpeak As String)

This function uses speech synthesis to "speak" the String argument. Note that your PC must have a working audio system, and speech must first be enabled by checking the "Allow Speech" checkbox in Config->GeneralConfig->GeneralConfiguration.

Arguments:

TextToSpeak is a String to be spoken

Return Value:

None

Example:

```
' Tell the user to load the next work piece
Speak("Please load next work piece.  Hit CycleStart
When Ready")
Code "M00"
```

See also:

StartPeriodicScript

Function StartPeriodicScript(ByVal ScriptQFN as String, ByVal UpdatePeriod as Double) as Integer

This function causes Mach to start a script that will be invoked with the specified periodicity.

Arguments:

ScriptQFN: the string of the Qualified File Name (QFN) for the script to be run. The qualified path name is relative to the Mach install directory. The QFN does not include the script extension. Mach will look first for ScriptQFN.mcc and if not found, then ScriptQFN.m1s.

If the QFN is the name of a script already started by a previous call to StartPeriodicScript(), an error condition is returned. Mach does not support multiple periodic instances of a single script.

UpdatePeriod: The length of the time period between runs of the script. The time units are seconds. The minimum value is 5ms, any value less than 5ms will be ignored and the minimum value of 5ms will be used for the UpdatePeriod.

It is recommended that periodic scripts be run with the longest UpdatePeriod appropriate for the script's task. This will help minimize the load on the machine from multiple periodic scripts.

Return Value:

0 = requested function was performed successfully (script is now running).
< 0 = error, requested function was not successful (script is not running).

Error Return Values:

<tbd>

Example:

```
' Initialize a script that runs the way oiler every 30
minutes

If StartPeriodicScript("OilerScript", 30*60*1000) then
     Msgbox "Oiler periodic script is running"
Else
     Msgbox "Oiler script was not started"
End If
```

See also:

IsPeriodicScriptRunning, StopPeriodicScript

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

It was defined as a Boolean function with

Return Value:

True = requested function was performed successfully (script is now running).
False = error, requested function was not successful (script is not running).

This API was revised in Mach3 version 3.43.19 to be an Integer function.

StartTHC

Sub StartTHC()

This function turns on torch height control. It is functionally identical to THCOn().

Arguments:

None

Return Value:

None

Example:

```
StartTHC()‘ Turn on torch height control
…          ‘ Do some cutting here
EndTHC()   ‘ Turn off torch height control
```

See also:

THCOn(), THCOff(), EndTHC(), ZeroTHC(), ResetTHC()

StopPeriodicScript

Function StopPeriodicScript(ByVal ScriptQFN as String) as Integer

This function is used to stop a previously started periodic script.

A periodic script only stops at the end of one of the script’s UpdatePeriod time quanta (as set by StartPeriodicScript) cycles. For example if a script is running with a 5 minute UpdatePeriod, and the stop request is issued 2 minutes into the cycle, the script will not stop until the current 5 minute cycle expires.

Arguments:

ScriptQFN: the string of the Qualified File Name (QFN) for the script to be stopped. The QFN is relative to the Mach install directory.
The QFN passed must be identical to the QFN used to start the periodic script. Attempts to stop a script that has not been started by StartPeriodicScript are ignored (this is not an error condition as the script is “stopped” upon return from the call).

Return Value:

0 = requested function was performed successfully (script is now running).
< 0 = error, requested function was not successful (script is not running).

Error Values:

<tbd>

Example:

```
‘ Stop a the running way oiler script

If StopPeriodicScript(“OilerScript”) then
     Msgbox “Oiler periodic script has been stopped”
Else
     Msgbox “Error, Oiler script was not stopped,
check the QFN passed.”
End If
```

See also:

IsPeriodicScriptRunning, StartPeriodicScript,

First Mach3 version with API:

This API was first implemented in Mach3 version 3.43.06.

It was defined as a Boolean function with

Return Value:

True = requested function was performed successfully (script is now running).

False = error, requested function was not successful (script is not running).

This API was revised in Mach3 version 3.43.19 to be an Integer function.

StraightFeed

Sub StraightFeed(X As Double, Y As Double, Z As Double, A As Double, B As Double, C As Double)

This function performs a feed rate move to the specified position. Note that all axis positions must be specified. This is exactly equivalent to Code "G1 Xn.nnn Yn.nnn Zn.nnn An.nnn Bn.nnn Cn.nnn".

Arguments:

X is a Double specifying the target position for the X axis
Y is a Double specifying the target position for the Y axis
Z is a Double specifying the target position for the Z axis
A is a Double specifying the target position for the A axis
B is a Double specifying the target position for the B axis
C is a Double specifying the target position for the C axis

Return Value:

None

Example:

```
' Send all axes to zero position at current feed rate
StraightTraverse(0.000, 0.000, 0.000, 0.000, 0.000,
0.000)
```

See also:

StraightTraverse(), Code()

StraightTraverse

Sub StraightTraverse(X As Double, Y As Double, Z As Double, A As Double, B As Double, C As Double)

This function performs a rapid move to the specified position. Note that all axis positions must be specified. This is exactly equivalent to Code "G0 Xn.nnn Yn.nnn Zn.nnn An.nnn Bn.nnn Cn.nnn".

Arguments:

X is a Double specifying the target position for the X axis
Y is a Double specifying the target position for the Y axis
Z is a Double specifying the target position for the Z axis
A is a Double specifying the target position for the A axis
B is a Double specifying the target position for the B axis
C is a Double specifying the target position for the C axis

Return Value:

None

Example:

```
' Send all axes to zero position
StraightTraverse(0.000, 0.000, 0.000, 0.000, 0.000,
0.000)
```

See also:

StraightFeed(), Code()

SwapAxis

Sub SwapAxis(FirstAxis As Integer, SecondAxis As Integer)

This function swaps the STEP and DIR pins for the two specified axes. This has precisely the same effect as changing the pin settings in Config->Ports&Pins. Note that no other axis parameters or settings are changed. If the two specified axes have different acceleration and velocity settings, unreliable operation will likely result. Note also that if you exit Mach3 while the swap is in effect, the swapped pins will be written to the XML configuration file, and the swap will still be in effect the next time you start Mach3. It is illegal to perform two consecutive swaps, without first executing a ResetAxisSwap(). Doing so will likely result in incorrect operation.

Arguments:

FirstAxis in an Integer value that specifies the first axis to swap. 0=X, 1=Y, 2=Z, etc.
SecondAxis in an Integer value that specifies the second axis to swap. 0=X, 1=Y, 2=Z, etc.

Return Value:

None

Example:

```
Xaxis = 0
Yaxis = 1
Code “G0 X0 Y0”
‘ Cut a 45 degree diagonal line from lower right to
upper left
Code “G1 X-1 Y1”
Code “G0 X0 Y0”
‘ Now swap X and Y axes
SwapAxis(Xaxis, Yaxis)
‘ Cut a 45 degree diagonal line from upper left to
lower right
Code “G1 X-1 Y1”
Code “G0 X0 Y0”
```

See also:

ResetAxisSwap()

SystemWaitFor

Sub SystemWaitFor(Signal As Integer)

This function it is used to tell Mach that a macro script is ending, and that any further GCode execution should wait for the Signal to go active. If used, this function should always be the last line in a macro and it should never be used within a loop.

Arguments:

SignalID must be one of the pre-defined Mach3 CB output signal constants (see CB Constants), or other value or expression that evaluates to one of those values.

Return Value:

None

Example:

See also:

THCOff

Sub THCOff()

This function turns off the torch height control.

Arguments:
None

Return Value:
None

Example:
```
' Turn off THC
THCOff()
```

See also:
THCOn(), ZeroTHC()

THCOn

Sub THCOn()

This function turns on the torch height control.

Arguments:
None

Return Value:
None

Example:
```
' Turn on THC
THCOn()
```

See also:
THCOff(), ZeroTHC()

ToggleScreens

Sub ToggleScreens()

This function toggles the active screen set between the “complex” and “simple” (.set and .sset) screen sets of the same base name. For example, if the current screen set is the default “1024.set”, this function will load the “1024.sset” screen set, and vice-versa. This function can be used to toggle between any two screen sets by simply giving both the same base filename, and giving one the .set extension, and the other the .sset extension.

Arguments:
None

Return Value:
None

Example:

```
' Toggle to simple screen set
ToggleScreens()
' Wait 5 seconds
Sleep 5000
' Toggle back to complex screen set
ToggleScreens()
```

See also:

ToolLengthOffset

Function ToolLengthOffset() As Double

This function returns the tool length offset for the current tool.

Arguments:
None

Return Value:
Tool length offset for current tool, as a Double

Example:

```
' Get the current tool length offset
LengthOffset = ToolLengthOffset
' Tell the user
Message "Tool length offset = " & LengthOffset
```

See also:
GetCurrentTool(), SetCurrentTool(), GetSelectedTool(), GetToolParam(), SetToolParam(), SetToolX(), SetToolZ(), GetToolDesc()

VerifyAxis

Sub VerifyAxis(Silent As Boolean)

This function verifies the positions of all axes by performing a home operation on all axes simultaneously, then moving all axes to position 0.0000 of the current fixture. If Safe_Z is enabled, a move to Safe_Z will be performed first. If the Silent argument is True, then upon completion, a message is displayed on the status line showing the offset errors of all axes from their home position. This function I used to ensure that no

position loss has occurred, and/or to correct any position loss that may have occurred since the previous home or verify operation.

Arguments:

Silent is a Boolean value that, if true, causes a message to be displayed on completion of the verify indicating the offsets, if any, of each axis from its correct position.

Return Value:

None. All axes machine zeroes are reset to their correct position, and all axes are moved to position 0.0000 of the current fixture.

Example:

```
' Make sure all axes are in correct position
VerifyAxis()
' Load the G-code file
LoadFile("MyGCodeFile.nc")
' Wait for Load to Complete
While IsLoading()
     Sleep 100
Wend
' Run the File
DoOEMButton(CycleStartButton)
' Wait for it to complete
While (GetOEMLED(StartLED))
     Sleep 100
Wend
```

See also:

SingleVerify(), SingleVerifyReport(), RefCombination()

ZeroTHC

Sub ZeroTHC()

This function zeroes the torch height correction factor.

Arguments:

None

Return Value:

None

Example:

```
' Zero THC correction
ZeroTHC()
```

See also:

THCOn(), THCOff()

Legacy Functions Grouped By Function

Digitizing

CloseDigFile – Close digitization points file

OpenDigFile – Open digitization points file

G-Code & G-code Files

Code – Execute a line of G-code

FileName – Get current G-code filename

GetVar – Get a G-code Var value

IsLoading – Get current G-code file loading status

LoadFile – Load a G-code file

LoadRun – Load and run a G-code file

MaxX – Get maximum X extent for current G-code file

MaxY – Get maximum Y extent for current G-code file

MinX – Get minimum X extent for current G-code file

MinY – Get minimum Y extent for current G-code file

Param1 – Get M-macro P parameter value

Param2 – Get M-macro Q parameter value

Param3 – Get M-macro Q parameter value

RunFile – Run currently loaded G-code file

SetVar – Set a G-code Var value

Lathe-only Functions

GetTurretAng – Get current tool turret angle

IsDiameter – Get current diameter/radius mode status

Mach3 Configuration & Status

GetIJMode – Get current IJ mode

GetMainFolder – Get Mach3 main folder path

GetParam – Get a Mach3 named parameter value

GetScale – Get current scale factor for an axis

GetTimer – Get a timer value

HelpAbout – Get current CB version

QueueDepth – Get current Mach3 trajectory queue depth

ResetAxisSwap – Un-swap swapped axes

RetractMode – Get current peck cycle retract mode

SetFormula – Set a new axis formula

SetIJMode – Set IJ mode

SetParam – Set a Mach3 names parameter

SetScale – Set a new axis scale factor

SetTimer – Reset a timer

SwapAxis – Swap axis outputs

Referencing, Verifying & Zeroing Axes

GetABSPosition – Get machine position for an axis

RefCombination – Reference any combination of axes

SetMachZero – Set machine zero to current position

SingleVerify – Verify position of a single axis

SingleVerifyReport – Verify position of a single axis, and report if error

VerifyAxis – Verify position of a single axis

SafeZ

GetSafeZ – Get current SafeZ height

GotoSafeZ – Go to current SafeZ height

IsSafeZ – Find out if machine is currently at or above SafeZ height

SetSafeZ – Set SafeZ height

Wizards& Plugins

LoadWizard – Load a wizard by name

NotifyPlugins – Send a notification to all plugins

SaveWizard – Save current wizard settings

Machine Status & Control

IsEstop – Find out if Mach3 is currently in E-Stop

CoupleSlave – Couple slave axis to its master for referencing

Motion Control

FeedRate – Get current feed rate

JogOff – Disable jogging for a single axis

JogOn – Jog a single axis in a specified direction

SetFeedRate – Set a new feed rate

StraightFeed – Perform a feed rate move

StraightTraverse – Perform a rapid move

IsMoving – Find out if machine is currently moving

IsStopped – Find out if machine is currently stopped

Spindle Control

DoSpinCCW – Turn spindle on CCW

DoSpinCW – Turn spindle on CW

DoSpinStop – Turn spindle off

GetRPM – Get current spindle RPM

SetPulley – Set a new spindle pulley

SetSpinSpeed – Set a new spindle speed

Tool Parameters and Tool Changes

GetCurrentTool – Get currently loaded tool

GetSelectedTool – Get newly selected tool

GetToolChangeStart – Get axis positions at start of tool change

GetToolDesc – Get the tool descriptor for the specified tool number

GetToolParam – Get value of a named tool parameter for the specified tool number

SetCurrentTool – Set the current tool to the specified value

SetToolParam – Set value of a named tool parameter for the specified tool number

SetToolX – Set the X axis offset

SetToolZ – Set the Z axis offset

ToolLengthOffset – Set the Z offset for the current tool

Torch Height Control

EndTHC – Turn off THC

ResetTHC – Reset THC

StartTHC – Start THC

THCOff – Turn off THC

THCOn – Turn on THC

ZeroTHC – Zero THC

Screen sets

DoMenu – Invoke a menu function

DoOEMButton – Invoke an OEM Button function

GetDRO – Get current value for specified DRO

GetDROString – Get current value for specified DRO as a String

GetLED – Get current state of specified OEM LED

GetOEMDRO – Get current value for specified OEM DRO

GetOEMLED – Get current state specified OEM LED

GetPage – Get current screen set page number

GetUserDRO – Get current value of specified User DRO

GetUserLabel – Get current value of specified User Label

GetUserLED – Get current state of specified User LED

LoadStandardLayout – Load default screen set

Message – Display a message on the Error SmartLabel

SetButtonText – Change the label text on the currently active button

SetDRO – Set the value of the specified OEM DRO

SetOEMDRO – Set the value of the specified OEM DRO

SetPage – Select a new screen set page by number

SetTicker – Set the value of a Ticker SmartLabel

SetUserDRO – Set the value of the specified User DRO

SetUserLabel – Set the value of the specified UserLabel SmartLabel

SetUserLED – Set the state of the specified User LED

ToggleScreens – Toggle between “complex”/.set and “simple”/.sset screen sets

User Dialogs

AskTextQuestion – Ask the user a question and get a String response

GetACoor – Get A coordinate given by last GetCoord() call

GetCoord – Get axis coordinates from user

GetXCoor – Get X coordinate given by last GetCoord() call

GetYCoor – Get Y coordinate given by last GetCoord() call

GetZCoor – Get Z coordinate given by last GetCoord() call

MachMsg – Display a dialog with one or more buttons on it

Question – As the user a question, and get a numeric response

Signals and Port I/O

ActivateSignal – Activate/Assert an output signal

DeactivateSignal – Deactivate/Deassert an output signal

GetPortByte – Read a byte from an I/O port

IsActive – Find out if the specified named input signal is in its active state

IsOutputActive – Find out if the specified named output signal is in its active state

IsSuchSignal – Find out if the specified named signal is defined

PutPortByte – Write a byte to an I/O port

SetTriggerMacro – Associate an M-macro with OEM code 301

SystemWaitFor – Pause script execution until specified named input signal is asserted

Teach Files

AppendTeachFile – Open a Teach file for append

CloseTeachFile – Close currently open Teach file

LoadTeachFile – Load current Teach file for execution

OpenTeachFile – Open a new Teach file for writing

Miscellaneous

nFmt – Round a Double value to a specified number of decimal places

PlayWave – Play a .WAV file

Random – Get a pseudo-random number

Roun – Round a Double value to four decimal places

Sleep – Pause script execution for specified length of time

Speak – Use voice synthesis to “speak” a String

Modbus Functions Grouped Alphabetically

GetInBit

Function

This function

Arguments:
F

Return Value:
A

Example:
A

See also:

GetInput

Function

This function

Arguments:
F

Return Value:
A

Example:
A

See also:

ResetOutBit

Function

This function

Arguments:
F

Return Value:
A

Example:

A

See also:

SetHomannString

Sub SetHomannString(Col As Integer, Row As Integer, Text As String)

This function writes the Text string to the LCD display on ModIO slave. Row and Col specify the row and column position of the first character of the string. This function is exactly equivalent to the SetModIOString function with a SlaveID of 1.

Arguments:

Slave is the ModBus Slave ID of the device whose LCD the string is written to
Col is the 0-based column to which the first character of the Text will be written. Col ***must*** be an even number (multiple of 2).
Row is the 0-based row this which the Text will be written
Text is the String to be written to the LCD

Return Value:

None

Example:

‘ Write “Hello, world!” to the second line of the 2x16 LCD on Modbus Slave 2
SetModIOString(2, 0, 1, “Hello, world!”)

See also:

FillFromCoil(), FillFromHolding(), FillFromInput(), FillFromStatus(), GetModWord(), ModGetInputWord(), SetModIOString(), SetModOutput(), SetModPlugString(), ResetOutBit(), GetInBit(), GetInput(), SetOutBit(), SetOutput(), WaitForPoll(),SetModIOString()

SetModIOString

Sub SetModIOString(SlaveID As Integer, Col As Integer, Row As Integer, Text As String)

This function writes the Text string to the ModIO devices LCD display. Row and Col specify the row and column position of the first character of the string.

Arguments:

Slave is the ModBus Slave ID of the device whose LCD the string is written to
Col is the 0-based column to which the first character of the Text will be written. Col ***must*** be an even number (multiple of 2).
Row is the 0-based row this which the Text will be written

Text is the String to be written to the LCD

Return Value:

None

Example:

‘ Write “Hello, world!” to the second line of the 2x16 LCD on Modbus Slave 2
SetModIOString(2, 0, 1, “Hello, world!”)

See also:

FillFromCoil(), FillFromHolding(), FillFromInput(), FillFromStatus(), GetModWord(), ModGetInputWord(), SetModIOString(), SetModOutput(), SetModPlugString(), ResetOutBit(), GetInBit(), GetInput(), SetOutBit(), SetOutput(), WaitForPoll(),SetHomannString()

SetModOutput

Function

This function

Arguments:

F

Return Value:

A

Example:

A

See also:

SetOutBit

Function

This function

Arguments:

F

Return Value:

A

Example:

A

See also:

WaitForPoll – Unreliable….

Function

This function

Arguments:

F

Return Value:

A

Example:

A

See also:

Serial Output Functions Grouped Alphabetically

There are a number of serial functions in the v3 code-base, some previously documented, some not. However, testing indicates the following is the only function that actually works. Serial input is not supported in Mach3 v3.

SendSerial

Sub SendSerial(Data As String)

This function send the String Data to the serial port specified in the Config->GeneralConfig serial port configuration. This provides transmit-only capability, at any supported BAUD rate.

Arguments:

String message to be sent to configured serial device

Return Value:

None

Example:

```
' Send "Hello, world!" to serial device
SendSerial("Hello, world!" & chr(10) & char(13))
```

See also:

Script Pre-processing Functionality

Starting with Mach3 v3 version 3.43.06, Mach provides some pre-processing capability.

#Expand

Text from external files can be included and expanded inline using the #Expand facility.

The syntax is:
<white space> --- <expand keyword> --- <white space> --
--------- <angle bracket path spec> ----------
| |
--- <Quoted path spec> ---------

Examples:

#expand <path-spec>
#expand "path-spec"

<white space> = any number of spaces or tabs
<expand keyword> = #expand - the word “Expand” can be any combination of upper or lower case letters.

<angle bracket path spec> = <*path-spec*>
<Quoted path spec> = “*path-spec*”

The *path-spec* is a filename optionally preceded by a directory specification. The filename must name an existing file. *path-spec* is a relative path and filename, hence it does not start with a leading “\”.

Both syntax forms cause replacement of the #expand directive by the entire contents of the specified include file.

The difference between the two forms is where the preprocessor searches for expand files and the inclusion (or not) of the .m1s extension in the path spec.

Syntax Form	Action
Angle-bracket form	This form instructs the preprocessor to search for the expand file in a location dependant on the current runtime environment. The rules are:

	1) if a wizard is NOT loaded (I.e. a regular screen set is loaded, the preprocessor will look for the path-spec in <Mach install dir>\ScreenSetMacros\<ActiveScreenSetName.set>\ 2) if a wizard is loaded, the preprocessor will look for the path-spec in <Mach install dir>\AddOns\<LoadedWizardName>\ For the Angle-bracket form, the extension is not included in the path-spec, and the pre-processor will only look for a corresponding .m1s file. This form is recommended for normal Mach screen set and wizard scripts.
Quoted form	This form instructs the preprocessor that the path-spec is a partially qualified File name and that the preprocessor should simply use the entire QFN as found in the #expand directive line. The QFN is assumed to be relative to the <Mach install dir>. As this is a QFN, it must also include the extension of the file to expand. (the extension is not restricted to .m1s, the preprocessor simply opens the specified FQFN). This for is recommended for user scripts which are not part of a screen set or wizard.

The primary intent of this facility is to enable script authors to keep script sources in one file and be able to reuse the source in multiple scripts.

For example many people prefer to define names for Mach's magic DRO and LED numbers.
This is a trivial example which uses this idea:

Macro script:

```
Option Explicit

#expand <MachConstants>

MsgBox "Mach's X DRO is # " & MachXDRONum
```

```
    Exit Sub    ' return to Mach
```

Where MachConstants.m1s contains:

```
    Const MachXDroNum = 800
```

The Mach #expand processing is recursive, so that you can include a file which in turns includes a file etc.

The mach3 script editor has been enhanced to deal with script expansions.

Screen Set Initialization and Clean up

As of Mach version 3.32.41 Mach now will run a macro on screen set load and on screen set unload.

The macros must be in
<mach3installdir>\ScreenSetMacros\<active screen set name with extension>\

For a screen set called MachStdMill, this would be:
C:\Mach3\ScreenSetMacros\MachStdMill.set\

The macro names Mach looks for within the dir are:
ScreenSetLoad
ScreenSetUnload

ScreenSetLoad is run by Mach before the screen set is loaded.
ScreenSetUnload is run by Mach when the screen set is unloaded.

Either .m1s (interpreted) or .mcc (compiled) macros may be used.

If the Screen Set directory or macros are not present, no error is thrown by Mach. This gives backwards compatibility with prior Mach versions and screen sets.

ScreenSetLoad has a couple of special characteristics:

1) If the LoadStandardLayout API is called by ScreenSetLoad, LoadStandardLayout will not invoke another instance of ScreenSetLoad (preventing a loop scenario).
2) ScreenSetLoad is run in a separate thread from the main Mach3 thread.

Brain Auto Initialization

While this is not strictly an Mach script language API, it is mentioned here as a compliment to the Screen Set initialization and clean up facilities.

Prior to Mach version 3.43.00, Mach looked in <Mach install dir>\Brains for installed brains. If a brain was enabled, it was loaded when Mach initialized. This still happens with Mach version 3.43.00+.

As of Mach version 3.43.0, Mach can auto load Brains each time Mach is started. Mach now also looks in <Mach install dir>\Brains\AutoLoad\ and any brains found there are automatically loaded, enabled, and run as part of Mach in initialization.

This provides a way for a screen set or wizard add-on to install a brain on disk and have it loaded when Mach initializes, without the need for the user to manually use the Mach menu to enable and load the brain.

OEM Series Button, DRO and LED numbers

As Mach3 developed, functions were added. Unfortunately documentation of the corresponding function numbers did not stay current.

The following tables were complied by several volunteers and represent the collectively known published OEM numbers as of the revision date of this manual. The list is not perfect and there may be errors contained within it.

OEM Button numbers

Type	New Function Description	Mach3 V3 #
OEM Button	Screen 1	1
OEM Button	Screen 2	2
OEM Button	Screen 3	3
OEM Button	Screen 4	4
OEM Button	Screen 5	5
OEM Button	Screen 6	6
OEM Button	Screen 7	7
OEM Button	Screen 8…98	8…98
OEM Button	Screen 99	99
OEM Button	Jog Increment Increment	100
OEM Button	Jog Increment Decrement	101
OEM Button	Reset Interp	102
OEM Button	Jog Enable Off/On Toggle	103
OEM Button	Safe Z Height Go To	104
OEM Button	Home/Reference All Z Then X,Y,A,B,C Set Coords	105
OEM Button	Units Inch/MM	106
OEM Button	Multi-function DROs Machine/Work Toggle 1	107
OEM Button	Feed Rate Override Increment (percentage)	108
OEM Button	Feed Rate Override Decrement (percentage)	109
OEM Button	Spindle CW Off/On Toggle	110
OEM Button	Jog Slow Increment	111
OEM Button	Jog Slow Decrement	112
OEM Button	Coolant Flood Off/On Toggle	113
OEM Button	Coolant Mist Off/On Toggle	114
OEM Button	G-code Edit	115
OEM Button	Rotational axis Diameter A Zero A	116

OEM Button	Rotational axis Diameter B Zero B	117
OEM Button	Rotational axis Diameter C Zero C	118
OEM Button	Soft limits On/Off Toggle	119
OEM Button	Tool Z Zero	120
OEM Button	Tool Table Open	121
OEM Button	Work Table Open	122
OEM Button	Torch Height Correction Enable Off/On Toggle	123
OEM Button	Torch Calibration Zero	124
OEM Button	Encoder Load From X	125
OEM Button	Encoder Load To X	126
OEM Button	Encoder Load From Y	127
OEM Button	Encoder Load To Y	128
OEM Button	Encoder Load From Z	129
OEM Button	Encoder Load To Z	130
OEM Button	Mill/Turn	131
OEM Button	Tool Path Off/On Toggle	132
OEM Button	Encoder 1 Zero X	133
OEM Button	Encoder 2 Zero Y	134
OEM Button	Encoder 3 Zero Z	135
OEM Button	Tool	136
OEM Button	Fixture	137
OEM Button	Go Home (G28)	138
OEM Button	Part X Zero	139
OEM Button	Part Y Zero	140
OEM Button	Part Z Zero	141
OEM Button	Part A Zero	142
OEM Button	Part B Zero	143
OEM Button	Part C Zero	144
OEM Button	Tool X Touch	145
OEM Button	Tool Z Zero	146
OEM Button	Joystick Throttle	147
OEM Button	Tool Touch Correction Enable Off/On Toggle	148
OEM Button	Homing/Limits Auto Over Ride Limit/Home Switches	149
OEM Button	Homing/Limit Switch Over Ride (Manual)	150
OEM Button	SS on Act4 Off/On Toggle	151
OEM Button	Reserved	152
OEM Button	Reserved	153
OEM Button	Reserved	154
OEM Button	Units Per Minute/Rev. Toggle	155

OEM Button	Set Next Line	156
OEM Button	Tool Path Jog Follow	157
OEM Button	Joystick On	158
OEM Button	Joystick Off	159
OEM Button	Tool Path Regen Display	160
OEM Button	Work X-Z Zero (turn)	161
OEM Button	Coordinate Mode Abs./Incremental.	162
OEM Button	Spindle Speed Override Increment by (percentage)	163
OEM Button	Spindle Speed Override Decrement by (percentage)	164
OEM Button	Laser Trigger Enable Off/On Toggle	165
OEM Button	Laser Grid Zero	166
OEM Button	Z Limiting Off/On Toggle Z	167
OEM Button	Tool Change Ignore Off/On Toggle	168
OEM Button	G-Code Close current File	169
OEM Button	G-Code Re-load Last File	170
OEM Button	Jog increment Selection Cycle	171
OEM Button	Clear error Label	172
OEM Button	Spindle CCW Off/On Toggle	173
OEM Button	MPG Parallel Port Encoder 3 Jog Off/On Toggle	174
OEM Button	MPG 1 Cycle Axis Controlled	175
OEM Button	Block Delete	176
OEM Button	Optional Stop Off/On Toggle	177
OEM Button	Inhibit All Off/On Toggle	178
OEM Button	Multi-function DROs Machine Coordinates Displayed	179
OEM Button	Multi-function DROs Work + G 92	180
OEM Button	Multi-function DROs Work	181
OEM Button	Spindle ??? (Override cancel)Actual Off/On Toggle	182
OEM Button	Home/Reference X Then Z (turn)	184
OEM Button	MPG 1 Jogs X	185
OEM Button	MPG 1 Jogs Y	186
OEM Button	MPG 1 Jogs Z	187
OEM Button	MPG 1 Jogs A	188
OEM Button	MPG 1 Jogs B	189
OEM Button	MPG 1 Jogs C	190
OEM Button	Jog Increment Selection 1	191
OEM Button	Jog Increment Selection 2	192

OEM Button	Jog Increment Selection 3	193
OEM Button	Jog Increment Selection 4	194
OEM Button	Jog Increment Selection 5	195
OEM Button	Jog Increment Selection 6	196
OEM Button	Jog Increment Selection 7	197
OEM Button	Jog Increment Selection 8	198
OEM Button	Jog Increment Selection 9	199
OEM Button	Jog Increment Selection 10	200
OEM Button	Joystick Feed Rate Override Off	201
OEM Button	Joystick Feed Rate Override Jog	202
OEM Button	Joystick Feed Rate Override Feed	203
OEM Button	Jog Mode Continuous 1	204
OEM Button	Jog Mode Incremental 1	205
OEM Button	Joystick On	206
OEM Button	Joystick Off	207
OEM Button	Tool Z Clear (turn)	208
OEM Button	Tool X Clear (turn)	209
OEM Button	Tool Touch Stock Correction Set to Zero	210
OEM Button	Home/Reference X and Z (turn)	211
OEM Button	Home/Reference X (turn)	212
OEM Button	Home/Reference Z (turn)	213
OEM Button	G-Code Show Recent Files	214
OEM Button	Display History	215
OEM Button	G-code Load	216
OEM Button	Tool Post Front/Rear Toggle	217
OEM Button	Z Limiting ON Z	218
OEM Button	Z Limiting Off Z	219
OEM Button	Port Bit-Test	220
OEM Button	Torch Height Correction Off/On Toggle	221
OEM Button	Torch Height Correction Off	222
OEM Button	Torch Height Correction ON	223
OEM Button	Coolant Flood ON	224
OEM Button	Coolant Flood Off	225
OEM Button	Coolant Mist ON	226
OEM Button	Coolant Mist Off	227
OEM Button	Teach File Load	228
OEM Button	Tool Path in Machine/Work Toggle	229
OEM Button	Wizard Display Selection Window	230
OEM Button	Wizard is done Load normal screen	231

OEM Button	Screen Simple/Complex Toggle	232
OEM Button	Output 4 On	233
OEM Button	Output 4 Off	234
OEM Button	Output 5 On	235
OEM Button	Output 5 Off	236
OEM Button	Output 6 On	237
OEM Button	Output 6 Off	238
OEM Button	Set Help Context	239
OEM Button	Home/Reference All Forced De-Reference	240
OEM Button	Tangential Off/On Toggle	241
OEM Button	G 59 Work Save Current XYZ to G 59.254 XYZ	242
OEM Button	Coordinates Machine G 53 Do G0G53	243
OEM Button	G 59 Move to G59.254 with midpoint selection	244
OEM Button	Jog Mode Continuous/Incremental/MPG Toggle Through	245
OEM Button	Home/Reference All Forced	246
OEM Button	CV Off/On Toggle	247
OEM Button	CV OFF	248
OEM Button	CV On	249
OEM Button	Jog Inhibit X Off/On Toggle	250
OEM Button	Jog Inhibit Y Off/On Toggle	251
OEM Button	Jog Inhibit Z Off/On Toggle	252
OEM Button	Jog Inhibit A Off/On Toggle	253
OEM Button	Jog Inhibit B Off/On Toggle	254
OEM Button	Jog Inhibit C Off/On Toggle	255
OEM Button	Multi-function DROs Machine/Work Toggle 2	256
OEM Button	Inhibit All Off	257
OEM Button	Inhibit All On	258
OEM Button	Encoder Jog X	259
OEM Button	Encoder Jog Y	260
OEM Button	Encoder Jog Z	261
OEM Button	Encoder Jog A	262
OEM Button	Encoder Jog B	263
OEM Button	Encoder Jog C	264
OEM Button	Jog Increment Value 1	265
OEM Button	Jog Increment Value 2	266
OEM Button	Jog Increment Value 3	267
OEM Button	Jog Increment Value 4	268
OEM Button	Jog Increment Value 5	269
OEM Button	Jog Increment Value 6	270

OEM Button	Jog Increment Value 7	271
OEM Button	Jog Increment Value 8	272
OEM Button	Jog Increment Value 9	273
OEM Button	Jog Increment Value 10	274
OEM Button	Jog Mode Incremental 2	275
OEM Button	Jog Mode Continuous 2	276
OEM Button	Feed Rate Increment (units)	277
OEM Button	Feed Rate Decrement (units)	278
OEM Button	Reverse Run	279
OEM Button	Wizard Switch to Last Used	280
OEM Button	MPG 2 Cycle Axis Controlled	281
OEM Button	MPG 2 Taper Mode Off/On Toggle	282
OEM Button	MPG Dual Flag Off/On Toggle	283
OEM Button	MPG Shuttle mode Off/On Toggle	284
OEM Button	Pause Preparation Moves	285
OEM Button	Pause (remember state)	286
OEM Button	Feed Rate Rapid Override On/Off Toggle	287
OEM Button	Dwell Crop current now	288
OEM Button	Set Formulas	289
OEM Button	MPG Push To Stop Jog X	290
OEM Button	MPG Push To Stop Jog Y	291
OEM Button	MPG Push To Stop Jog Z	292
OEM Button	MPG Push To Stop Jog A	293
OEM Button	MPG Push To Stop Jog B	294
OEM Button	MPG Push To Stop Jog C	295
OEM Button	Motor Screw Mapping	296
OEM Button	Dwell Unconditional crop any now	297
OEM Button	Reserved	298
OEM Button	Feed Rate Alternative Bypass Off/On Toggle	299
OEM Button	Tool Post Front/Rear Toggle	300
OEM Button	Code for OEMTriggers runs the macro in SetTriggerMacro 301	301
OEM Button	MPG Steps Multiple/Single/Step Velocity/Velocity Only Toggle	302
OEM Button	MPG Velocity Only	303
OEM Button	MPG Step/Velocity	304
OEM Button	MPG Steps Single	305
OEM Button	MPG Steps Multiple	306
OEM Button	Jog Push To Jog X Positive	307
OEM Button	Jog Push To Jog X Negative	308

OEM Button	Jog Push To Jog Y Positive	309
OEM Button	Jog Push To Jog Y Negative	310
OEM Button	Jog Push To Jog Z Positive	311
OEM Button	Jog Push To Jog Z Negative	312
OEM Button	Jog Push To Jog A Positive	313
OEM Button	Jog Push To Jog A Negative	314
OEM Button	MPG Calibrate	315
OEM Button	Tool Table Save	316
OEM Button	Work Table Save	317
OEM Button	Tool Path Mouse Drag Zooms Off/On Toggle	318
OEM Button	Tool Path Mouse Drag Pans Off/On Toggle	319
OEM Button	Wizard Display Selection Dialog	320
OEM Button	Wizard Run Newfangled	321
OEM Button	Emergency bailout	322
OEM Button	Screen Visibility of Screen 50 Off/On Toggle	323
OEM Button	Tool X Touch (turn)	324
OEM Button	Torch Volts Controlled by Spindle Step Line Off/On Toggle	325
OEM Button	Tool Z Touch (turn)	326
OEM Button	MPG Jog Mode	327
OEM Button	unknown	328
OEM Button	Jog Push To Jog B Positive	329
OEM Button	Jog Push To Jog B Negative	330
OEM Button	Jog Push To Jog C Positive	331
OEM Button	Jog Push To Jog C Negative	332
OEM Button	Multi-function DROs Distance To Go	333
OEM Button	Jog Push To Jog X Stop	334
OEM Button	Jog Push To Jog Y Stop	335
OEM Button	Jog Push To Jog Z Stop	336
OEM Button	Jog Push To Jog A Stop	337
OEM Button	Jog Push To Jog B Stop	338
OEM Button	Jog Push To Jog C Stop	339
OEM Button	Soft limits Set Temporary Minimum	340
OEM Button	Soft limits Set Temporary Maximum	341
OEM Button	Reserved 343-346	347
OEM Button	Screen Menu Bar Turn Off	348
OEM Button	Screen Menu Bar Turn On	349
OEM Button	Spindle Speed Requested Increment by (value)	350
OEM Button	Spindle Speed Requested Decrement by (value)	351
OEM Button	Jog Together X & A ++	352

OEM Button	Jog Together X & A –	353
OEM Button	Jog Together Y & B ++	354
OEM Button	Jog Together Y & B –	355
OEM Button	Start	1000
OEM Button	Pause Feed Hold	1001
OEM Button	G-Code Rewind	1002
OEM Button	Stop	1003
OEM Button	Single Block	1004
OEM Button	Resume	1005
OEM Button	Teach File Edit	1006
OEM Button	Work Zero All	1007
OEM Button	Work X Zero	1008
OEM Button	Work Y Zero	1009
OEM Button	Work Z Zero	1010
OEM Button	Work A Zero	1011
OEM Button	Work B Zero	1012
OEM Button	Work C Zero	1013
OEM Button	Feed Rate Override Cancel	1014
OEM Button	Estimate Job	1015
OEM Button	Run From Here	1016
OEM Button	Work Go To Zeros	1017
OEM Button	Multi-function DROs System	1018
OEM Button	Verify	1020
OEM Button	Reset	1021
OEM Button	Home/Reference X	1022
OEM Button	Home/Reference Y	1023
OEM Button	Home/Reference Z	1024
OEM Button	Home/Reference A	1025
OEM Button	Home/Reference B	1026
OEM Button	Home/Reference C	1027
OEM Button	Joystick Off/On Toggle	1028
OEM Button	Soft limits On/Off Toggle	1029
OEM Button	Unknown	1030
OEM Button	Jog Push to Jog Until stopped	1031

OEM DRO numbers

Type	New Function Description	Mach3 V3 #
OEM DRO	Jog Increment Incremental Size	1

OEM DRO	Pulse Freq.	2
OEM DRO	Jog Slow Decrement/Increment by (percentage %)	3
OEM DRO	Tool Path In Work Coordinates Minimum X	4
OEM DRO	Tool Path In Work Coordinates Minimum Y	5
OEM DRO	Tool Path In Work Coordinates Minimum Z	6
OEM DRO	Tool Path In Work Coordinates Minimum A	7
OEM DRO	Tool Path In Work Coordinates Minimum B	8
OEM DRO	Tool Path In Work Coordinates Minimum C	9
OEM DRO	Tool Path In Work Coordinates Maximum X	10
OEM DRO	Tool Path In Work Coordinates Maximum Y	11
OEM DRO	Tool Path In Work Coordinates Maximum Z	12
OEM DRO	Tool Path In Work Coordinates Maximum A	13
OEM DRO	Tool Path In Work Coordinates Maximum B	14
OEM DRO	Tool Path In Work Coordinates Maximum C	15
OEM DRO	G 92 Reposition/Threading X	16
OEM DRO	G 92 Reposition/Threading Y	17
OEM DRO	G 92 Reposition/Threading Z	18
OEM DRO	G 92 Reposition/Threading A	19
OEM DRO	G 92 Reposition/Threading B	20
OEM DRO	G 92 Reposition/Threading C	21
OEM DRO	Queue Depth	22
OEM DRO	Scale Time	23
OEM DRO	Spindle Pulse Width Modulation Base	24
OEM DRO	Torch Height Correction Speed	25
OEM DRO	Torch Height Correction Speed (Current)	26
OEM DRO	Torch Height Correction Speed Maximum	27
OEM DRO	Processor CPU Load	28
OEM DRO	Encoder 1 Position X	29
OEM DRO	Encoder 2 Position Y	30
OEM DRO	Encoder 3 Position Z	31
OEM DRO	Tool Length Minus Wear	32
OEM DRO	Home Off Distance X	33
OEM DRO	Home Off Distance Y	34
OEM DRO	Home Off Distance Z	35
OEM DRO	Home Off Distance A	36
OEM DRO	Home Off Distance B	37
OEM DRO	Home Off Distance C	38
OEM DRO	Spindle Speed TRUE	39
OEM DRO	Worst Case Interrupt	40

OEM DRO	Tool X	41
OEM DRO	Tool Length	42
OEM DRO	Tool Diameter	43
OEM DRO	Tool Tip Radius	44
OEM DRO	Tool Touch Correction	45
OEM DRO	Fixture Number	46
OEM DRO	Part X	47
OEM DRO	Part Y	48
OEM DRO	Part Z	49
OEM DRO	Part A	50
OEM DRO	Part B	51
OEM DRO	Part C	52
OEM DRO	Processor CPU Speed	53
OEM DRO	Safe Z Height Z	54
OEM DRO	Feed Rate Overridden	55
OEM DRO	Spindle Pulley	56
OEM DRO	Spindle Pulley Maximum Speed	57
OEM DRO	Feed Velocity Per Revolution	58
OEM DRO	Scale X	59
OEM DRO	Scale Y	60
OEM DRO	Scale Z	61
OEM DRO	Scale A	62
OEM DRO	Scale B	63
OEM DRO	Scale C	64
OEM DRO	Torch Height Correction Speed Minimum	65
OEM DRO	Threading Entrance Angle	66
OEM DRO	Limits Entrance Points Maximum	67
OEM DRO	Rotational axis Time Error	68
OEM DRO	Threading Trigger Angle	69
OEM DRO	Time Correction Derivative ???Threading diagnostic???	70
OEM DRO	Spindle Pulse Interrupts Per Rev.	71
OEM DRO	Spindle Pulse Count	72
OEM DRO	Spindle Adder	73
OEM DRO	Spindle Speed Override (percentage of requested)	74
OEM DRO	Stock Size	75
OEM DRO	Laser Grid Spacing X	76
OEM DRO	Laser Grid Spacing Y	77
OEM DRO	Repetitions	78

OEM DRO	Safe Z Height Decrement Z	79
OEM DRO	Z Limiting Distance Z	80
OEM DRO	Port Bit-test	81
OEM DRO	Torch Height Correction Feed Rate Limit %	82
OEM DRO	Coordinates Machine ABS X	83
OEM DRO	Coordinates Machine ABS Y	84
OEM DRO	Coordinates Machine ABS Z	85
OEM DRO	Coordinates Machine ABS A	86
OEM DRO	Coordinates Machine ABS B	87
OEM DRO	Coordinates Machine ABS C	88
OEM DRO	CV Distance Tolerance	89
OEM DRO	Spindle Pulse Number Of Disc Slots	90
OEM DRO	G 73 Peck Drill Pull-off Distance	91
OEM DRO	Tangential Lift Angle	92
OEM DRO	Tangential Lift Level Z	93
OEM DRO	Reserved	94
OEM DRO	Reserved	95
OEM DRO	Reserved	96
OEM DRO	CV Rate (CV applied at this rate)	97
OEM DRO	Feed Rate Decrement/ Increment by (units)	98
OEM DRO	Spindle PWM Control Ratio	99
OEM DRO	Encoder 4 Position A	100
OEM DRO	MPG 1 Count	101
OEM DRO	MPG 2 Count	102
OEM DRO	MPG 3 Count	103
OEM DRO	Feed Rate Rapid	104
OEM DRO	Tool Diameter (current)	105
OEM DRO	Tool Tip Direction	106
OEM DRO	Tool Tip Nose Radius	107
OEM DRO	Tool Offset X (current)	108
OEM DRO	Tool Offset Z (current)	109
OEM DRO	Tool Wear X	110
OEM DRO	Tool Wear Z	111
OEM DRO	Tool Turret Angle	112
OEM DRO	MPG 1 Velocity	113
OEM DRO	MPG 2 Velocity	114
OEM DRO	MPG 3 Velocity	115
OEM DRO	MPG Taper Angle	116
OEM DRO	Spindle Speed as Constant Surface Speed	117

OEM DRO	Coordinate System Rotation Angle X/Y (G68)	118
OEM DRO	Laser Grid Spacing X/Y	119
OEM DRO	Jog Increments That Can be Buffered	120
OEM DRO	Spindle Pulley Minimum Speed	121
OEM DRO	Feed Rate Alternative	122
OEM DRO	Tool Post Between Front and Rear	123
OEM DRO	MPG 1 Velocity Current	124
OEM DRO	MPG 2 Velocity Current	125
OEM DRO	MPG Step Jog Feed Rate	126
OEM DRO	Encoder Error Encoder 1 X	127
OEM DRO	Encoder Error Encoder 2 Y	128
OEM DRO	Encoder Error Encoder 3 Z	129
OEM DRO	Processor Interrupt Handler	130
OEM DRO	Laser SLS Distance	131
OEM DRO	Look ahead	133
OEM DRO	Input Modbus 64	146
OEM DRO	Input Modbus 65	147
OEM DRO	Input Modbus 66	148
OEM DRO	Input Modbus 67	149
OEM DRO	Soft limit Maximum X	150
OEM DRO	Soft limit Maximum Y	151
OEM DRO	Soft limit Maximum Z	152
OEM DRO	Soft limit Maximum A	153
OEM DRO	Soft limit Maximum B	154
OEM DRO	Soft limit Maximum C	155
OEM DRO	Soft limit Minimum X	156
OEM DRO	Soft limit Minimum Y	157
OEM DRO	Soft limit Minimum Z	158
OEM DRO	Soft limit Minimum A	159
OEM DRO	Soft limit Minimum B	160
OEM DRO	Soft limit Minimum C	161
OEM DRO	Tool Post Distance From Front to Rear	162
OEM DRO	Encoder 1 Units Position	170
OEM DRO	Encoder 2 Units Position	171
OEM DRO	Encoder 3 Units Position	172
OEM DRO	Encoder 4 Units Position	173
OEM DRO	Part X Touch Tool Table X Radius/Diameter X	175
OEM DRO	Part Z Touch Tool Table Z	176
OEM DRO	Torch Pierce Delay	177

OEM DRO	Work X	178
OEM DRO	Work Y	179
OEM DRO	Work Z	180
OEM DRO	Work A	181
OEM DRO	Work B	182
OEM DRO	Work C	183
OEM DRO	G 92/52 Control Point X	184
OEM DRO	G 92/52 Control Point Y	185
OEM DRO	G 92/52 Control Point Z	186
OEM DRO	G 92/52 Control Point A	187
OEM DRO	G 92/52 Control Point B	188
OEM DRO	G 92/52 Control Point C	189
OEM DRO	G 28 Home Location X	190
OEM DRO	G 28 Home Location Y	191
OEM DRO	G 28 Home Location Z	192
OEM DRO	G 28 Home Location A	193
OEM DRO	G 28 Home Location B	194
OEM DRO	G 28 Home Location C	195
OEM DRO	Multi-function DROs Distance To Go X	196
OEM DRO	Multi-function DROs Distance To Go Y	197
OEM DRO	Multi-function DROs Distance To Go Z	198
OEM DRO	Multi-function DROs Distance To Go A	199
OEM DRO	Multi-function DROs Distance To Go B	200
OEM DRO	Multi-function DROs Distance To Go C	201
OEM DRO	Spindle Speed Overridden (DRO 817 * DRO 74)	202
OEM DRO	Spindle Pulley Reversed Direction	203
OEM DRO	Tool X (current) (turn)	204
OEM DRO	Encoder Reading Corrected by Offsets X	208
OEM DRO	Encoder Reading Corrected by Offsets Y	209
OEM DRO	Encoder Reading Corrected by Offsets Z	210
OEM DRO	Tool Path In Machine Coordinates Maximum X	211
OEM DRO	Tool Path In Machine Coordinates Maximum Y	212
OEM DRO	Tool Path In Machine Coordinates Maximum Z	213
OEM DRO	Tool Path In Machine Coordinates Minimum X	214
OEM DRO	Tool Path In Machine Coordinates Minimum Y	215
OEM DRO	Tool Path In Machine Coordinates Minimum Z	216
OEM DRO	Spindle Dwell Spin Up CW	217
OEM DRO	Processor Brains Execution Time	220
OEM DRO	Probe Tip Diameter	221

OEM DRO	Sub Program Depth	222
OEM DRO	Feed Rate Rapid Override	223
OEM DRO	Macros Running Number of	224
OEM DRO	Multi-function DROs Machine/Work/DTG X	800
OEM DRO	Multi-function DROs Machine/Work/DTG Y	801
OEM DRO	Multi-function DROs Machine/Work/DTG Z	802
OEM DRO	Multi-function DROs Machine/Work/DTG A	803
OEM DRO	Multi-function DROs Machine/Work/DTG B	804
OEM DRO	Multi-function DROs Machine/Work/DTG C	805
OEM DRO	Feed Rate Velocity X	806
OEM DRO	Feed Rate Velocity Y	807
OEM DRO	Feed Rate Velocity Z	808
OEM DRO	Feed Rate Velocity A	809
OEM DRO	Feed Rate Velocity B	810
OEM DRO	Feed Rate Velocity C	811
OEM DRO	CV Blended Velocity	813
OEM DRO	Elapsed Time	814
OEM DRO	Estimate Job	815
OEM DRO	G-code Line Number (current)	816
OEM DRO	Spindle Speed Requested	817
OEM DRO	Feed Rate	818
OEM DRO	Feed Rate Override Decrement/Increment by (percentage)	821
OEM DRO	Tool Number	824
OEM DRO	Rotational Axis Diameter A	825
OEM DRO	Rotational Axis Diameter B	826
OEM DRO	Rotational Axis Diameter C	827
OEM DRO	Jog Increment Incremental Size	828
OEM DRO	Fixture Offset X	830
OEM DRO	Fixture Offset Y	831
OEM DRO	Fixture Offset Z	832
OEM DRO	Fixture Offset A	833
OEM DRO	Fixture Offset B	834
OEM DRO	Fixture Offset C	835
OEM DRO	Tool Length (current)	836

OEM LED numbers

Type	New Function Description	Mach3 V3 #
OEM OEM LED	G 92 in Effect	10
OEM OEM LED	Spindle CCW/CW (if either is requested)	11
OEM LED	Coolant Mist	12
OEM LED	Coolant Flood	13
OEM LED	Jog Mode Continuous	14
OEM LED	Jog Mode Incremental	15
OEM LED	Multi-function DROs Machine Coordinates Displayed	16
OEM LED	Feed Rate Override	17
OEM LED	Estimating Job	18
OEM LED	Emergency	19
OEM LED	Rotational Axis Diameter A Correction A	20
OEM LED	Rotational Axis Diameter B Correction B	21
OEM LED	Rotational Axis Diameter C Correction C	22
OEM LED	Soft limits On	23
OEM LED	Torch Height Correction Enable	24
OEM LED	Spindle Speed TRUE Accel.	25
OEM LED	Spindle Speed TRUE Decel.	26
OEM LED	Tool Path is On	27
OEM LED	Tool	28
OEM LED	Part	29
OEM LED	Joystick Throttle Is Slow Jog	30
OEM LED	Joystick Throttle Is Feed Rate	31
OEM LED	Homing/Limits Auto Over Ride Limit/Home Switches	33
OEM LED	Homing/Limit Switch Over Ride (Manual)	34
OEM LED	SS on Act4	35
OEM LED	Torch Arc Good	36
OEM LED	Torch Up Active	37
OEM LED	Torch Down Active	38
OEM LED	Feed Rate Per Revolution (G95)	39
OEM LED	Feed Rate Per Minute (G94)	40
OEM LED	Scale X	41
OEM LED	Scale Y	42
OEM LED	Scale Z	43
OEM LED	Scale A	44
OEM LED	Scale B	45
OEM LED	Scale C	46

OEM LED	Coordinate Mode Abs.	48
OEM LED	Coordinate Mode Incremental.	49
OEM LED	Threading Sync mode	50
OEM LED	Laser Trigger Enabled	51
OEM LED	Z Limiting ON Z	52
OEM LED	Tool Change Ignore	53
OEM LED	CV On G64 Mode Active (Constant Velocity)	54
OEM LED	Repetitions Enabled (M30)	55
OEM LED	Exact Stop on (G61 Mode Active)	56
OEM LED	MPG Jog Mode	57
OEM LED	Jog Rapid Enabled (shift key active)	58
OEM LED	MPG 1 Jogs X	59
OEM LED	MPG 1 Jogs Y	60
OEM LED	MPG 1 Jogs Z	61
OEM LED	MPG 1 Jogs A	62
OEM LED	MPG 1 Jogs B	63
OEM LED	MPG 1 Jogs C	64
OEM LED	Optional Stop	65
OEM LED	Block Delete	66
OEM LED	Inhibit All	67
OEM LED	Threading Feed Related to True Spindle Speed	68
OEM LED	Threading Index signal	69
OEM LED	Torch Height Correction ON	70
OEM LED	Spindle Speed Stable	71
OEM LED	IJ Abs. Mode	72
OEM LED	IJ Inc. Mode	73
OEM LED	Teaching File is Open	74
OEM LED	Offset In Effect	75
OEM LED	Output 4 Active	77
OEM LED	Output 5 Active	78
OEM LED	Output 6 Active	79
OEM LED	Pause	80
OEM LED	Tangential Control Active	81
OEM LED	Single Block	82
OEM LED	Jog Enable On	83
OEM LED	CV Enabled	84
OEM LED	Enhanced Pulsing Not in Use	85
OEM LED	Jog Inhibit X	86
OEM LED	Jog Inhibit Y	87

OEM LED	Jog Inhibit Z	88
OEM LED	Jog Inhibit A	89
OEM LED	Jog Inhibit B	90
OEM LED	Jog Inhibit C	91
OEM LED	Diameter mode Active (turn)	92
OEM LED	Timing Signal Active	93
OEM LED	Hotkeys enabled	94
OEM LED	Units Per Minute Mode	95
OEM LED	Units Per Rev. Mode	96
OEM LED	Reverse Run	97
OEM LED	MPG 2 Jogs X	98
OEM LED	MPG 2 Jogs Y	99
OEM LED	MPG 2 Jogs Z	100
OEM LED	MPG 2 Jogs A	101
OEM LED	MPG 2 Jogs B	102
OEM LED	MPG 2 Jogs C	103
OEM LED	MPG 2 Taper Mode Active	104
OEM LED	MPG Dual MPGs In Use	105
OEM LED	MPG Shuttle mode In Use	106
OEM LED	Spindle Speed as Constant Surface Speed G96	107
OEM LED	Coordinate System Rotation Angle Active X/Y (G68)	108
OEM LED	Feed Rate Rapid Override	109
OEM LED	Rotational axis Formula Mapping	110
OEM LED	Pause Feed Hold	111
OEM LED	Reverse Run	112
OEM LED	Feed Rate Alternative In Use	113
OEM LED	Tool Post Front Selected	114
OEM LED	Tool Post Rear Selected	115
OEM LED	Spindle Running (CCW Only)	116
OEM LED	MPG Velocity Only	117
OEM LED	MPG Step/Velocity	118
OEM LED	MPG Steps Single	119
OEM LED	MPG Steps Multiple	120
OEM LED	Jog Switch Active X Positive	121
OEM LED	Jog Switch Active X Negative	122
OEM LED	Jog Switch Active Y Positive	123
OEM LED	Jog Switch Active Y Negative	124
OEM LED	Jog Switch Active Z Positive	125
OEM LED	Jog Switch Active Z Negative	126

OEM LED	Jog Switch Active A Positive	127
OEM LED	Jog Switch Active A Negative	128
OEM LED	Tool Path Mouse Drag Zooms	129
OEM LED	Tool Path Mouse Drag Pans	130
OEM LED	MPG 3 Jogs X	131
OEM LED	MPG 3 Jogs Y	132
OEM LED	MPG 3 Jogs Z	133
OEM LED	MPG 3 Jogs A	134
OEM LED	MPG 3 Jogs B	135
OEM LED	MPG 3 Jogs C	136
OEM LED	Macro Is running	162
OEM LED	Multi-function DROs Distance To Go	163
OEM LED	Spindle CW	164
OEM LED	Spindle CCW	165
OEM LED	Conditions abnormal	166
OEM LED	CV Distance Tolerance On	168
OEM LED	Regen in progress	179
OEM LED	Tool change Auto TC mode	184
OEM LED	Tool Change Stop&Wait Mode	185
OEM LED	Reset	800
OEM LED	Units Inch	801
OEM LED	Units MMs	802
OEM LED	Processor Idle	803
OEM LED	Start	804
OEM LED	Pause Feed Hold	805
OEM LED	Tool Change In Process	806
OEM LED	Home/Reference X Warning	807
OEM LED	Home/Reference Y Warning	808
OEM LED	Home/Reference Z Warning	809
OEM LED	Home/Reference A Warning	810
OEM LED	Home/Reference B Warning	811
OEM LED	Home/Reference C Warning	812
OEM LED	Dwell (G04 and Spindle)	813
OEM LED	Joystick Enable	814
OEM LED	Fixture	816
OEM LED	Input 1 Active	821
OEM LED	Input 2 Active	822
OEM LED	Input 3 Active	823
OEM LED	Input 4 Active	824

OEM LED	Digitize Input	825
OEM LED	Spindle Index	826
OEM LED	Homing/Limit Switch Triggered	827
OEM LED	Limits Active Positive X	828
OEM LED	Limits Active Negative X	829
OEM LED	Home Negative X	830
OEM LED	Limits Active Positive Y	831
OEM LED	Limits Active Negative Y	832
OEM LED	Home Negative Y	833
OEM LED	Limits Active Positive Z	834
OEM LED	Limits Active Negative Z	835
OEM LED	Home Negative Z	836
OEM LED	Limits Active Positive A	837
OEM LED	Limits Active Negative A	838
OEM LED	Home Negative A	839
OEM LED	Limits Active Positive B	840
OEM LED	Limits Active Negative B	841
OEM LED	Home Negative B	842
OEM LED	Limits Active Positive C	843
OEM LED	Limits Active Negative C	844
OEM LED	Home Negative C	845
OEM LED	Enable 1	846
OEM LED	Enable 2	847
OEM LED	Enable 3	848
OEM LED	Enable 4	849
OEM LED	Enable 5	850
OEM LED	Enable 6	851
OEM LED	Output 1/Extrn1 Active	852
OEM LED	Output 2/Extrn2 Active	853
OEM LED	Output 3/Extrn3 Active	854
OEM LED	Digitize Output	855
OEM LED	System movement In effect or imminent	999