

# DSC Module User's Guide vs 3.0:

(DSC = Data, Signal and Control Functions)

by Scott "Poppa Bear" Shafer of S S Systems, LLC

## Summary:

The module wraps almost all the useable/practical functions in M4.

There are functions that:

- Get/Set all OEM Input/Outputs
- Get/Set Registers
- Get/Set Hardware I/O
- Get/Set Control functions
- Get/Set ToolPath color options.
- The Set functions are sometimes called "Do" since they lend themselves to button presses.

The following "header" needs to be put at the Top of any Macro, Script, or Wizard that you want to use its wrapper on. If you want to use it in a screen set, then it should go into the "Screen Load" script, prior to using it.

```
--////////// This is the REQUIRED Header code, Begin ////////////
package.path = "./?.lua;c:/Mach4Hobby/Modules/?.mcc;" --where the module file is.
DSC = require "DSC_Module";--load the module file, call it DSC, (if using it as a Global for screens)
--local DSC = require "DSC_Module";--load the module file, call it DSC, (if using it in a macro or script)
--////////// This is the REQUIRED Header code, End ////////////
```

## Get OEM Inputs:

### **DSC.GetInput(number);**

Function Purpose: Gets a Mach4 OEM Input Signal Input0-63

Returns: a numeric Bool, 1 = ON, 0 = OFF

Example:

```
local StatusOfInput0 = DSC.GetInput(0);--gets INPUT0 state
```

### **DSC.IsHome(number);**

Function Purpose: Active when Motor is on Home switch, 0-32

Returns: a numeric Bool, 1 = ON, 0 = OFF

Example:

```
local IsMotor0onHomeSwitch = DSC.IsHome(0);--gets motor0 home switch state
```

### **DSC.IsPlusLim(PlusLimSig);**

Function Purpose: Gets the status of the PLUS limit of the motor it belongs to, 0-32

Returns: a numeric Bool, 1 = ON, 0 = OFF

Example:

```
local IsMotor0onPlusLimit = DSC.IsPlusLim(0);--gets motor0 plus limit switch state
```

**DSC.IsMinusLim(MinusLimSig);**

Function Purpose: Gets the status of the Minus limit of the motor it belongs to, 0-32

Returns: a numeric Bool, 1 = ON, 0 = OFF

Example:

```
local IsMotor0onMinusLimit = DSC.IsMinusLim(0);--gets motor0 minus limit switch state
```

**DSC.GetOtherInput(OTHERINPUTSIG);**

Function Purpose: This handles all the Non-Sequential “Named” Inputs, in where a shortened name is passed, to a reverse look up function in the module.

Function Parameter(s): OTHERINPUTSIG is a shortened string (and all CAPS) i.e. drops the ISIG\_ So "ISIG\_DIGITIZE” becomes 'DIGITIZE' (ALL CAPS).

Param List:

DIGITIZE, INDEX, LIMITOVER, EMERGENCY, THCON, THCUP, THCDOWN, TIMING, JOGXP, JOGXN, JOGYP, JOGYN, JOGZP, JOGZN, JOGAP, JOGAN, JOGBP, JOGBN, JOGCP, JOGCN, SPINDLE\_AT\_SPEED, SPINDLE\_AT\_ZERO

Returns: a numeric Bool, 1 = ON, 0 = OFF

Example:

```
local IsDigitize = DSC.GetOtherInput('DIGITIZE');--is digitize signal active
```

**Set OEM Outputs:****DSC.OutputAxisSwitchStatus(AXISSWITCHSTATE, state);**

Function Purpose: To output the status of motors: Home, PlusLim or MinusLim switch state

Function Parameter(s): AXISSWITCHSTATE is a shortened string (and all CAPS) i.e. drops the OSIG\_

So "OSIG\_XLIMITPLUS” becomes 'XLIMITPLUS ' (ALL CAPS).

Param List:

XLIMITPLUS, XLIMITMINUS, XHOME, YLIMITPLUS, YLIMITMINUS, YHOME, ZLIMITPLUS, ZLIMITMINUS, ZHOME, ALIMITPLUS, ALIMITMINUS, AHOME, BLIMITPLUS, BLIMITMINUS, BHOME, CLIMITPLUS, CLIMITMINUS, CHOME

Sets: Can set an outgoing signal that tells if a Home, LimitPlus or LimitMinus switch is active.

Example:

```
local IsMotor0onHomeSwitch = DSC.IsHome(0);--gets motor0 home switch state
DSC.OutputAxisSwitchStatus('XHOME', IsMotor0onHomeSwitch);
```

**DSC.EnableSig(EnableSig, state);**

Function Purpose: To enable or disable a motors enable signal output

EnableSig = an integer from 0-31, state bool 1 = ON, 0 = OFF

Sets: Turns on or off a motors enable signal from 0-31 motors

Example:

```
DSC.EnableSig(0, 1);--enable motor0's enable
```

**DSC.OutputSig(OutputSig, state);**

Function Purpose: To turn ON or OFF a sequential OUTPUT signal from OUTPUT0-63

Sets: bool state = 1 on, 0 off,

Example:

```
local Output0 = DSC.OutputSig(0, 1) --turn on output0
```

**DSC.OutputMiscSig(OUTPUTMISCSIG, state);**

Function Purpose: To output the status of a miscellaneous control function.

Function Parameter(s): **OUTPUTMISCSIG** is a shortened string, (and all CAPS) i.e. drop the OSIG\_

so "OSIG\_RUNNING\_GCODE " becomes 'RUNNING\_GCODE ' (ALL CAPS).

Param List:

RUNNING\_GCODE, FEEDHOLD, BLOCK\_DELETE, SINGLE\_BLOCK, REVERSE\_RUN, OPT\_STOP, MACHINE\_ENABLED, TOOL\_CHANGE, DIST\_TOGO, MACHINE\_CORD, SOFTLIMITS\_ON, JOG\_INCJOG\_CONT, JOG\_ENABLED, JOG\_MPG, HOMED\_X, HOMED\_Y, HOMED\_Z, HOMED\_A, HOMED\_B, HOMED\_C, DWELL, TP\_MOUSE\_DN, LIMITOVER, CHARGE, CHARGE2, CURRENTHILOW, SPINDLEON, SPINDLEFWD, SPINDLEREV, COOLANTON, MISTON, DIGTRIGGER

Sets: Sets a misc. output for one of the above conditions being true or not, 1 or 0

Example:

```
DSC.OutputMiscSig(RUNNING_GCODE, 1); --G-Code is Running
```

**DEVICE REGISTERS:****DSC.ReadReg('RegisterPathAsString');**

Function Purpose: To read a "Register" can be a string value or number value

NOTE: Assumes you have set up your Modbus or whatever that fills the register your trying to get.

Returns: a string value or number value

Example:

```
local InputHoldingRegister1 = DSC.ReadReg('modbus0/InputHoldingReg1');
```

**DSC.WriteReg('RegisterPathAsString', val);**

Function Purpose: To write a value numeric or string to a register of a device

val = numeric or string value.

Sets: Sets a numeric or string value out to a device register

Example:

```
DSC.WriteReg('modbus0/OutputHoldingReg1', 4095);
```

## DEVICE DISCRETE IO:

### **DSC.GetHWIO('DeviceNameAsString', 'IoNameAsString');**

Function Purpose: To read the status of an "Input Point" on a device

Returns: Device Input status as a Bool number value

Example:

```
local InputX0 = DSC.GetHWIO('modbus0', 'InputX0');
```

### **DSC.SetHWIO('DeviceNameAsString', 'IoNameAsString', val);**

Function Purpose: To set the status of an "Output Point" on a device

Sets: A Bool number value to a Device Output point

Example:

```
DSC.SetHWIO('modbus0', 'OutputY0', 1);
```

## **DSC.DoFunc(...);**

The function can take up to 4 arguments depending on what func your calling. These are listed in the order they appear in the Mach4 API.h. Param1 is ALWAYS required it is the Function Name. There may be a Param 2, 3 and 4 depending on what the function is/does.

Here is the list:

1. DSC.DoFunc('DEREFALL'), Dereferences all axis
2. DSC.DoFunc('AXISENABLE', AxisID), Enables the defined axis number 0-5
3. DSC.DoFunc('AXISDISABLE', AxisID), Disables the defined axis number 0-5
4. DSC.DoFunc('HOMEALL'), Homes all axis
5. DSC.DoFunc('SETMCPOS', AxisID, val), Sets a MACHINE position for the defined axis, to a dbl value
6. DSC.DoFunc('SETAXPOS', AxisID, val), Sets an Axis DRO position (not machine coord.)
7. DSC.DoFunc('ENABLAXSL', AxisID), Enables the Soft Limits for the defined axis
8. DSC.DoFunc('DSABLAXSL', AxisID), Disables the Soft Limits for the defined axis
9. DSC.DoFunc('ZEROX'), Zero's the X axis
10. DSC.DoFunc('ZEROY'), Zero's the Y axis
11. DSC.DoFunc('ZEROZ'), Zero's the Z axis
12. DSC.DoFunc('ZEROA'), Zero's the A axis
13. DSC.DoFunc('ZEROB'), Zero's the B axis
14. DSC.DoFunc('ZEROC'), Zero's the C axis
15. DSC.DoFunc('ZEROALL'), Zero's all the axis
16. DSC.DoFunc('CLOSEGCODE'), close the G Code file
17. DSC.DoFunc('CYCLESTART'), cycle start
18. DSC.DoFunc('CYCLESTOP'), cycle stop
19. DSC.DoFunc('DRYRUNTO', LineNumber), dry runs to the line number
20. DSC.DoFunc('CNTLENBL'), Control Enable
21. DSC.DoFunc('CNTLDSBL'), Control Disable
22. DSC.DoFunc('ESTOP'), E-Stop

23. DSC.DoFunc('FEEDHOLD'), Feed Hold
24. DSC.DoFunc('MANTOOLCHG'), Do manual tool change (not auto)
25. DSC.DoFunc('EXEGCODE', StringBuffer), Will execute the G-code as string, StringBuffer = 'G0 X2'.
26. DSC.DoFunc('GOTOZ'), Go To Zero's
27. DSC.DoFunc('LOADGCODE', StringPath), Loads the requested G code file, i.e. StringPath = 'C:/mach4hobby/gcodefiles/RoadRunner.tap';
28. DSC.DoFunc('MDIEXE', StringBuffer), Execute MDI commands, as string, i.e. StringBuffer = 'G0 X2 Y2'.
29. DSC.DoFunc('STATECLEAR'), Machine State Clear
30. DSC.DoFunc('STATEPOP'), Machine State Pop
31. DSC.DoFunc('STATEPUSH'), Machine State Push
32. DSC.DoFunc('PROBEFLCL'), Probe File Close
33. DSC.DoFunc('PROBEFLOP', StringFileName, StringFileBuffer, OverWrite), File name as a string, string buffer for the data, bool for OverWrite, 1=yes, 0=no.
34. DSC.DoFunc('RESET'), Reset
35. DSC.DoFunc('REWIND'), Rewind G code
36. DSC.DoFunc('DIAMODE'), Lathe, Use Diameter Mode
37. DSC.DoFunc('RADMODE'), Lathe, Use Radius Mode
38. DSC.DoFunc('M1ON'), M1 Optional stop ON
39. DSC.DoFunc('M1OFF'), M1 Optional stop OFF
40. DSC.DoFunc('SET#VAR', VarNum, dval), Set a Pound VarNum, (number); to a dval, (dbl)
41. DSC.DoFunc('SNGLBLKON'), Turn Single Block ON
42. DSC.DoFunc('DEREFALL'), Turn Single Block OFF

## **DSC.GetFunc(...);**

Param1 is ALWAYS required it is the Function Name. There may be a Param 2, 3 and 4 depending on what the function is/does. NOTE: The name param has a prefix, if it is GET it returns a number or string. If prefix is IS it returns a bool.

If params are wrong, errors will be displayed in mach err, if call has an error, rc will be returned.

Here is the list:

1. DSC.GetFunc('GET#VAR', VarNum);--Gets the requested Pound VarNum, (number); returns a (dbl)
2. DSC.GetFunc('GETAXPOS', AxisID);--Gets position of the axis requested, (number); returns a (dbl)
3. DSC.GetFunc('ISAXSLEN', AxisID);--Returns a bool for if that axis has softlimits enabled
4. DSC.GetFunc('GETAXSLMAX', AxisID);--Gets the max soft limit val for that axis, as dbl
5. DSC.GetFunc('GETAXSLMIN', AxisID);--Gets the min soft limit val for that axis, as dbl
6. DSC.GetFunc('ISAXSPIN', AxisID);--Returns a bool if that axis is a spindle, 1=yes, 0=no
7. DSC.GetFunc('GETAXVEL', AxisID);--Gets the velocity that axis is set to, as dbl
8. DSC.GetFunc('ISAXENBL', AxisID);--Returns a bool if that axis is enabled, 1=yes, 0=no
9. DSC.GetFunc('ISAXHOMED', AxisID);--Returns a bool if that axis is homed, 1=yes, 0=no
10. DSC.GetFunc('ISAXSTILL', AxisID);--Returns a bool if that axis is not moving, 1=moving, 0=not
11. DSC.GetFunc('ISFEEDHLD', AxisID);--Returns a bool if that axis is FeedHeld, 1=yes, 0=no

12. DSC.GetFunc('GETGCODEPOS', AxisID) ;--Gets Axis pos at that the current line of Gcode, as dbl
13. DSC.GetFunc('ISBLKDEL', LineNum) ;-- Returns a bool if gcode line has a block delete, 1=yes, 0=no
14. DSC.GetFunc('GETDISTTOGO', AxisID) ;--Gets the distance left to go for that axis, as dbl
15. DSC.GetFunc('GETGCODELINE', LineNum) ;--Gets the line of Gcode at that number, as string
16. DSC.GetFunc('GETSTATENAME', StateNumber) ;--Gets the Machine State condition as string
17. DSC.GetFunc('GETTLOFFSET', AxisID) ;--Gets the current tool offset for that axis, as dbl
18. DSC.GetFunc('GETMODALGRP', GroupNum) ;--Gets the active modals for that group, as num
19. DSC.GetFunc('GETCOMPID') ;--Gets the computer ID as string
20. DSC.GetFunc('GETCOOLDLY') ;--Gets the Coolant Delay in seconds
21. DSC.GetFunc('ISDIAMODE') ;--Returns a bool if the lathe is in diameter mode, 1=yes, 0=no
22. DSC.GetFunc('GETFROPER') ;--Gets the Feed Rate Over-ride as percent
23. DSC.GetFunc('GETGCFILNM') ;--Gets the G-code files name, as string
24. DSC.GetFunc('GETLINECNT') ;--Gets the G-code line count, as num
25. DSC.GetFunc('GETLINENBR') ;--Gets the line number the G code is currently on, as num
26. DSC.GetFunc('GETLASTERR') ;--Gets the last Mach4 Err as string
27. DSC.GetFunc('GETLASTLOG') ;--Gets the last logged message as string
28. DSC.GetFunc('GETLICMODLS') ;--Gets the License Modules as string
29. DSC.GetFunc('ISLOGGING') ;--Returns if Logging is enabled or not, 1=yes, 0=no
30. DSC.GetFunc('GETM4DIR') ;--Gets the Mach4 Directory, as string
31. DSC.GetFunc('GETMISTDLY') ;--Gets the Mist coolant delay in seconds
32. DSC.GetFunc('GETMODE') ;--Gets the Machine type mode, 0=mill, 1=lathe, 2=plasma(not yet imp.)
33. DSC.GetFunc('ISOPTSTOP') ;--Returns a bool, if optional stop (M1) is enabled, 1=yes, 0=no
34. DSC.GetFunc('GETRRO') ;--Gets the Rapid Rate Override as percent
35. DSC.GetFunc('ISSGLBLK') ;--Returns if Single Block is Active, 1=yes, 0=no
36. DSC.GetFunc('GETSTATE') ;--Gets the Machines state, as number (feed this to GETSTATENAME)
37. DSC.GetFunc('ISINCYCLE') ;--Returns a bool if the machine is running gcode, 1=yes, 0=no
38. DSC.GetFunc('ISSTILL') ;--Returns a bool, if Anything is moving, 1=yes, 0=no
39. DSC.GetFunc('GETRUNTIME') ;--Get the amount of time, the current Gcode has been running

## Misc. Funcitons:

**DSC.DSCVersion();**-- Returns the version number of the DSC module as dbl and in Mach Err

**DSC.RGBtoLongInt(RedVal, GreenVal, BlueVal);** --RGB vals as nums 0-255, returns a Long Int representation of that color.

**DSC.HextoLongInt(HexVal);**--HexVal format 0x##### as Hex Number, returns a Long Int representation of that color.

## **Tool Path Set Functions:** (use the above RGB or Hex converters to long ints)

1. DSC.SetTPfunc('TPGEN');--Generate the Tool Path
2. DSC.SetTPfunc('TPABT');--Abort the Generation of the Tool Path
3. DSC.SetTPfunc('TPDWLM', BoolNum);--Tool Path, draw limits, 1=show, 0=no-show
4. DSC.SetTPfunc('TPFLWMD', BoolNum);--Enable follow mode, 1=yes, 0=no
5. DSC.SetTPfunc('TPAXCOL', AxisColor, LimitColor);--Sets the Axis color and Limit color (long ints)
6. DSC.SetTPfunc('TPBKCOL', TopColor, BotColor);--Sets the Axis Top and Bottom color (long ints)
7. DSC.SetTPfunc('TPPTHCOL', RapidColor, LineColor, ArcColor, HighlightColor);-- Set the Path colors, RapidColor, LineColor, ArcColor, and HighlightColor (long ints)

## **Tool Path Get Functions:**

1. DSC.GetTPfunc('TPGENPER');--Get how much in percent that the tool path is done generating
2. DSC.GetTPfunc('TPDWLM');--Returns if Tool Path draw limits is on
3. DSC.GetTPfunc('TPFLWMD');--Return if Tool Path Follow mode is on
4. DSC.GetTPfunc('TPGEN');--Return if the Tool Path is currently generating
5. DSC.GetTPfunc('TPAXCOL');--Gets the Axis color and Limit color (long ints)
6. DSC.GetTPfunc('TPBKCOL');-- Gets the Axis Top and Bottom color (long ints)
7. DSC.GetTPfunc('TPPTHCOL');-- Gets the Path colors, RapidColor, LineColor, ArcColor, and HighlightColor (long ints)

Also, as usual, if you find mistakes, bugs or have ideas for additions or improvements, let me know.

Scott "Poppa Bear" Shafer  
S S Systems, LLC