

## **COMMAND REFERENCE**

11/2/2012 10:58:53 AM

Galil Motion Control, Inc  
270 Technology Way  
Rocklin, California, 95765  
Phone: (916) 626-0101  
Fax: (916) 626-0102  
Technical: [support@galilmc.com](mailto:support@galilmc.com)  
Docs: [documentation@galilmc.com](mailto:documentation@galilmc.com)  
Web: [www.galil.com](http://www.galil.com)

---

## Table of Content

Table of Content	2
Overview	8
What is DMC code?	9
' Comment	14
- Subtraction Operator	15
# Label Designator	16
#AMPERR Amplifier error automatic subroutine	17
#AUTO Subroutine to run automatically upon power up	18
#AUTOERR Bootup Error Automatic Subroutine	19
#CMDERR Command error automatic subroutine	20
#COMINT Communication interrupt automatic subroutine	21
#ININT Input interrupt automatic subroutine	22
#LIMSWI Limit switch automatic subroutine	23
#MCTIME MC command timeout automatic subroutine	24
#POSERR Position error automatic subroutine	25
#SERERR Serial Encoder Error Automatic Subroutine	26
#TCPERR Ethernet communication error automatic subroutine	27
\$ Hexadecimal	28
% Modulo Operator	29
& JS subroutine pass variable by reference	30
& Bitwise AND Operator	31
( , ) Parentheses (order of operations)	32
* Multiplication Operator	33
/ Division Operator	34
; Semicolon (Command Delimiter)	35
@ABS Absolute value	36
@ACOS Inverse cosine	37
@AN Analog Input Query	38
@ASIN Inverse sine	39
@ATAN Inverse tangent	40
@COM Bitwise complement	41
@COS Cosine	42
@FRAC Fractional part	43
@IN Read digital input	44
@INT Integer part	45
@OUT Read digital output	46
@RND Round	47
@SIN Sine	48
@SQR Square Root	49
@TAN Tangent	50
[,] Square Brackets (Array Index Operator)	51
^ JS subroutine stack variable	52
^L^K Lock program	53
^R^S Master Reset	54
^R^V Revision Information	55
_ GP Gearing Phase Differential Operand	56
_ LF Forward Limit Switch Operand	57
_ LR Reverse Limit Switch Operand	58
Bitwise OR Operator	59
~ Variable Axis Designator	60
+ Addition Operator	61
< Less than comparator	62

<= Less than or Equal to comparator	63
<> Not Equal to comparator	64
= Assignment Operator	65
= Equal to comparator	66
> Greater than comparator	67
>= Greater than or Equal to comparator	68
AB Abort	69
AC Acceleration	70
AD After Distance	71
AF Analog Feedback Select	72
AG Amplifier Gain	73
AI After Input	74
AL Arm Latch	75
AM After Move	77
AO Analog Output	78
AP After Absolute Position	79
AQ Analog Input Configuration	80
AR After Relative Distance	81
AS At Speed	82
AT At Time	83
AU Set amplifier current loop	84
AV After Vector Distance	86
AW Amplifier Bandwidth	87
BA Brushless Axis	88
BC Brushless Calibration	89
BD Brushless Degrees	90
BG Begin	91
BI Brushless Inputs	92
BK Breakpoint	93
BL Reverse Software Limit	94
BM Brushless Modulo	95
BN Burn	96
BO Brushless Offset	97
BP Burn Program	98
BQ Brushless Offset dual DAC	99
BR Brush Axis	100
BT Begin PVT Motion	101
BV Burn Variables and Array	102
BW Brake Wait	103
BX Sine Amp Initialization	104
BZ Brushless Zero	106
CA Coordinate Axes	107
CB Clear Bit	108
CC Configure Communications Port 2	109
CD Contour Data	110
CE Configure Encoder	111
CF Configure Unsolicited Messages Handle	112
CI Configure Communication Interrupt	113
CM Contour Mode	114
CN Configure	115
CR Circle	116
CS Clear Sequence	118
CW Copyright information and Data Adjustment bit on off	119
DA Deallocate Variables and Arrays	120

DC Deceleration	121
DE Dual (Auxiliary) Encoder Position	122
DF Dual Feedback (DV feedback swap)	123
DH DHCP Server Enable	124
DL Download	125
DM Dimension	126
DP Define Position	127
DR Configures I O Data Record Update Rate	128
DT Delta Time	129
DV Dual Velocity (Dual Loop)	131
EA Choose ECAM master	132
EB Enable ECAM	133
EC ECAM Counter	134
ED Edit	135
EG ECAM go (engage)	137
EI Event Interrupts	138
ELSE Else function for use with IF conditional statement	140
EM Cam cycles (modulus)	141
EN End	142
ENDIF End of IF conditional statement	143
EO Echo	144
EP Cam table master interval and phase shift	145
EQ ECAM quit (disengage)	146
ER Error Limit	147
ES Ellipse Scale	148
ET Electronic cam table	149
EW ECAM Widen Segment	150
EY ECAM Cycle Count	151
FA Acceleration Feedforward	152
FE Find Edge	153
FI Find Index	154
FL Forward Software Limit	155
FV Velocity Feedforward	156
GA Master Axis for Gearing	157
GD Gear Distance	158
GM Gantry mode	159
GR Gear Ratio	160
HM Home	161
HS Handle Assignment Switch	162
HV Homing Velocity	163
HX Halt Execution	164
IA IP Address	165
ID Identify	166
IF IF conditional statement	167
IH Open IP Handle	168
II Input Interrupt	170
IK Block Ethernet ports	171
IL Integrator Limit	172
IP Increment Position	173
IT Independent Time Constant - Smoothing Function	174
JG Jog	175
JP Jump to Program Location	176
JS Jump to Subroutine	177
KD Derivative Constant	180

KI Integrator	181
KP Proportional Constant	182
KS Step Motor Smoothing	183
LA List Arrays	184
LC Low Current Stepper Mode	185
LD Limit Disable	186
LE Linear Interpolation End	187
LI Linear Interpolation Distance	188
LL List Labels	189
LM Linear Interpolation Mode	190
LS List	191
LV List Variables	192
LZ Inhibit leading zeros	193
MB Modbus	194
MC Motion Complete	198
MF Forward Motion to Position	199
MG Message	200
MO Motor Off	202
MR Reverse Motion to Position	203
MT Motor Type	204
MU Multicast Address	205
MW Modbus Wait	206
NB Notch Bandwidth	207
NF Notch Frequency	208
NO No Operation	209
NZ Notch Zero	210
OA Off on encoder failure	211
OB Output Bit	212
OC Output Compare	213
OE Off-on-Error	214
OF Offset	216
OP Output Port	217
OT Off on encoder failure time	218
OV Off on encoder failure voltage	219
P2CD Serial port 2 code	220
P2CH Serial port 2 character	221
P2NM Serial port 2 number	222
P2ST Serial port 2 string	223
PA Position Absolute	224
PF Position Format	225
PL Pole	226
PR Position Relative	228
PT Position Tracking	229
PV PVT Data	230
PW Password	231
QD Download Array	232
QH Hall State	233
QR I O Data Record	234
QS Error Magnitude	235
QU Upload Array	236
QZ Return Data Record information	237
RA Record Array	238
RC Record	239
RD Record Data	240

RE Return from Error Routine	241
REM Remark	242
RI Return from Interrupt Routine	243
RL Report Latched Position	244
RP Reference Position	245
RS Reset	246
SA Send Command	247
SB Set Bit	248
SC Stop Code	249
SD Switch Deceleration	250
SH Servo Here	251
SI Configure the special Galil SSI feature	252
SL Single Step	254
SM Subnet Mask	255
SP Speed	256
SS Configure the special Galil BiSS feature	257
ST Stop	259
SY Serial encoder BiSS active level	260
TA Tell Amplifier error status	261
TB Tell Status Byte	262
TC Tell Error Code	263
TD Tell Dual Encoder	265
TE Tell Error	266
TH Tell Ethernet Handle	267
TI Tell Inputs	268
TIME Time Operand	269
TK Peak Torque Limit	270
TL Torque Limit	271
TM Update Time	272
TN Vector Tangent	273
TP Tell Position	274
TR Trace	275
TS Tell Switches	276
TT Tell Torque	277
TV Tell Velocity	278
TW Timeout for IN Position (MC)	279
TZ Tell I O Configuration	280
UI User Interrupt	281
UL Upload	282
VA Vector Acceleration	283
VD Vector Deceleration	284
VE Vector Sequence End	285
VF Variable Format	286
VM Vector Mode	287
VP Vector Position	288
VR Vector Speed Ratio	290
VS Vector Speed	291
VV Vector Speed Variable	292
WH Which Handle	293
WT Wait	294
XQ Execute Program	295
YA Step Drive Resolution	296
YB Step Motor Resolution	297
YC Encoder Resolution	298

YR Error Correction	299
YS Stepper Position Maintenance Mode Enable, Status	300
ZA User Data Record Variables	301
ZS Zero Subroutine Stack	302

## COMMAND REFERENCE OVERVIEW

This command reference is a supplement to the Galil User Manual.

Resources on <a href="http://www.galilmc.com">www.galilmc.com</a>
<a href="#"> Printable version</a>
<a href="#">Product Manuals</a>
<a href="#">Application Notes</a>
<a href="#">Newest Firmware</a>
<a href="#">Sample DMC code</a>
<a href="#">Learning Center</a>
<a href="#">Support and Downloads</a>

## HOW TO USE THIS REFERENCE

At left is a table of contents organizeable in two ways, alphabetically and by category. These views are settable respectively by clicking on the following images:



In category view, clicking on a category will link to a subset of commands relevant to the chosen category. In alphabetic view, each command is listed in alphabetical order. In both views, clicking a command will load its reference.

## COMMAND REFERENCE LAYOUT

The following section dissects the LS command and describes the various parts of the command reference layout. The command reference described is printed below with a gray blue background.

<b>LS</b>   <i>List</i>	<b>INTERROGATION</b>
The LS command returns a listing of the programs in memory	
 	
LS n ... Arguments specified with an implicit comma-separated order	

### Command Header

The command header lists the command, a short description, and the first sentence of the full description. In the top right corner a listing of the relevant command categories are displayed. Next are usage icons which describe various characteristics of the command, including one or more of the following

Icon	Meaning and mouse-over text
	Parameter is burnable with BN
	Command can be used from the terminal
	Command is valid while running
	Command is valid in an embedded program
	Command is a trippoint that holds up execution until a condition is satisfied
	Axis assignment via variable axis supported

Following the usage icons is a table that describes the command's [syntax](#) and lists any operands, if applicable.

### Command Description

The description section provides the "executive summary" of what the command does.

### Command Arguments

The arguments section provides a table which describes the valid arguments for the command. Symbols are used in a command prototype to denote the various command constituents including one or more of the following

Symbol	Meaning	Example
n	Value. Same range, meaning and default for every instance of n in the prototype	KP n,n <a href="#">KP 6, 12</a>
n0,n1,n2,n3,n4,n5,n6,n7	Value. Unique range, meaning and or default	IA n0,n1,n2,n3 <a href="#">IA 192,168,1,123</a>
o,p	Value. Unique range, meaning, and or default. Arguments are located in &lt;to>&gt; delimiter structure. o and p arguments may be optional, see command Remarks for more detail	VP n0,n1 &lt;o&gt;p <a href="#">VP 2048,256&lt;1000&gt;25000</a>

m	Single-axis mask. Used to specify a single axis or channel.	KPm=n [KPB= 12]
mm	Multi-axis mask. Used to specify one or more axes	ST mm [ST ABD]
str	String. If double quotes are needed, the prototype will denote them.	PW str,str [PW passwd,passwd] MG "str" MG "Hello"
ex	An expression of Galil code. Can be a conditional or a mathematical statement.	IF(ex) [IF (var1 = var2)]
s	Generic one-letter symbol, usually a,b,c,d,e,f,g or h	~s=n [~a= 2]

## REMARKS

- n0 < n1 must always be true
- If n0 or n1 is omitted, default values are used
- n0 and n1 can also specify a label, for example:
  - "LS #label,20" would print out program lines from #label to line 20.

## EXAMPLES

```
:LS #a,6;      ' List program starting at #A through line 6
2 #a
3 PR 500
4 BG A
5 AM
6 WT 200
'Hint: Remember to quit the Edit Mode Q prior to giving the LS command. (DOS)
```

LS is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

## Command Remarks

The Remarks section will contain supplemental and detailed information about the command. This can include extended use cases, operand usage, step by step setup instructions, etc.

## Command Examples

The Examples section includes terminal-based and/or embedded program examples of the command. The following are some tips for reading examples.

- Terminal-based examples start with a colon (:
- Hover the mouse over commands to see a description of the code
- Click on many commands to launch a browser window to the reference for that command

## Command Footer

The command footer lists all Galil products for which the command is used. An email address is supplied for customer feedback.

## WHAT IS DMC CODE?

DMC (Digital Motion Controller) code is the programming language used for all Galil hardware. It is a high-level, interpreted language which is simple to learn and use, yet is surprisingly powerful. Actively developed and refined since 1983, DMC code provides functionality that is particularly well suited to motion control and PLC applications.

DMC code can be used manually from a terminal, programmatically from an external device or customer application, and can be fully embedded into a Galil controller's memory to leverage powerful "embedded-only" features and for stand-alone applications.

DMC code of course provides symbolic variables, arrays, and math support. The elegance of DMC coding is particularly evident when writing code for embedded applications. When running on the controller, the DMC language supports if-then-else conditionals, code branching, subroutines, a call stack (with parameter passing and local variable scope on some models), multi-threading, and automatic subroutines (i.e. event-driven programming).

DMC code runs on the Galil Real Time Operating System (RTOS) which is specifically designed for Galil hardware and for motion control.

The learning curve on DMC code is quite fast, usually less than one hour to basic motion, so called, "spinning motors". It is the fastest to learn, the easiest, the simplest, and one of the most flexible and powerful languages in the industry. Don't forget, Galil's [Applications Support Team](#) is available to assist you; from the most basic question to the most complicated needs.

## TOP DOWN: HOW IS A GALIL SYSTEM NORMALLY STRUCTURED?

However you want, there are three general approaches to Galil programming:

### Embedded/Galil-centric Programming

In this approach, a host computer is only used during development to program the controller. The program is then downloaded and burned to non-volatile flash using the #AUTO automatic subroutine to indicate where code execution should start on boot-up. The Galil controller will now run "standalone," not requiring any intervention from the host. Note that for serial and Ethernet controllers, the standalone controller can still actively work with other controllers in a network, without host intervention.

PCI and other PC bus-based controllers support this approach, although still require the PCI bus for power.

[GalilTools \(GT\)](#) is provided as a programming environment for developing embedded applications.

### Host-centric Programming

If a GUI or other frontend is desired to be run on a host, all development can be conducted on the host PC, with the architecture, operating system, and programming language of choice. In this approach, the controller receives every command from the host PC, nothing is running embedded. Many Galil firmware features are available to facilitate host-centric programming including mode-of-motion buffers, data logging buffers, asynchronous data record updates from the controller, PCI and UDP interrupt events, and more.

[GalilTools \(GT\)](#) is bundled with a programming library (API) for programming applications from a host. Many popular operating systems and languages are supported.

### Hybrid Programming

Perhaps the most versatile approach to Galil system design, the Hybrid approach allows for both embedded code and host-side code to work in tandem. Typically an application is developed for embedded use in DMC code. The code incorporates all of the detail of an application but relies on the host to provide it data. Through variables, arrays, and other commands, the host is able to define the bounds of the embedded algorithms. The host plays a supervisory role, interrogating status, receiving asynchronous updates from the controller, starting and stopping threads, and so on. The controller takes care of the motion and I/O responsibilities based on its embedded program, and the controller's real time operating system (RTOS) ensures

that the application won't suffer from indeterminacy which is common on general purpose PC operating systems (e.g. Microsoft Windows). Because the controller takes care of the details, the host is able to use its resources on other tasks, such as complicated number crunching or user interface.

It is noteworthy that Galil Standalone controllers (e.g. DMC-40x0, DMC-41x3, DMC-21x3, RIO-47xxx) can leverage the Ethernet to provide powerful modularity. Using any of the above three system approaches, multiple controllers can work in concert to achieve an application's requirements. Networked controllers also provide easy scalability. Need some more digital or analog I/O? Add an RIO. Need another axis of control? Add another DMC to the network. Both the Galil firmware and the Galil software libraries provide features which allow easy use of multiple controllers on an Ethernet network. RS232/422/485 networks are also possible.

## BOTTOM UP: ANATOMY OF DMC CODE

### Classification

DMC language can be broken up into the following general classifications

Classification	Description	Examples	Example Comments
<a href="#">Explicit Only</a>	The command receives its arguments only by assignment with the "=" operator.	IHC=192,168,1,101<1070>2	Create a TCP connection on Ethernet handle C to a device at IP address 192.168.1.101 on port 1070
<a href="#">Implicit Only</a>	The command receives its arguments only by an implicit argument order.	IA 192,168,1,102	Set the local IP address to 192.168.1.102
Explicit or Implicit	The command receives its arguments either by an explicit assignment using the "=" symbol, or an implicit argument order.	KPA=64;KPB=32;KPH=128 KP 64.32,...,128	Assign the proportional constant (KP) of the PID filter to three different axes.
<a href="#">Accepts Axis Mask</a>	The command receives its arguments as a string of valid axis names.	ST ADF	Stop (ST) axes A, D and F. Leave other axes running
<a href="#">Two Letter Only</a>	The command accepts no arguments	BN	Burn (BN) controller parameters to flash memory
<a href="#">Operator or Comparator</a>	Operators take two arguments and produce a result. Comparators take two values and return a Boolean (1 or 0).	+,-,*/ =<>,<=>,<>	Operators Comparators
<a href="#">At Function</a>	Starting with the @ character, these functions take one argument and perform a function, returning its result	@SIN,@ASIN @AN,@IN @RND,@FRAC,@COM	Trig functions Sine and ArcSine I/O functions Analog in and Digital in Numerical functions Round, Fractional Part, Bitwise complement
<a href="#">Embedded Only</a>	Not valid from the terminal, or from PC-side code, these commands are used in embedded DMC code only	IF,ELSE,ENDIF JS,JP EN,RE	IF Conditionals Jump commands End program, Return from Error
<a href="#">Operand</a>	Operands hold values, and are not valid on their own. They can be used as arguments to commands, operators or comparators	_TPA _LFC _TC	Current position of axis A encoder Forward limit state on C axis Current Error code
<a href="#">Trippoint</a>	Trippoints hold up a thread's execution (block) until a certain condition occurs. These are a special case of Embedded Only type commands.	WT 1000 AMA AII	Wait 1000 ms Wait until axis A completes profiled motion Wait for input one to go high
<a href="#">Comment</a>	Comments are used to document code.	'This is a comment	There are three types of comments: REM ', and NO

DMC code is case sensitive. All Galil commands are uppercase. User variables and arrays can be upper-case or lower case. Galil recommends that array and variable names contain at least one lower-case character to help distinguish them from commands.

### Explicit Notation

These commands specify data using an axis designator followed by an equals sign. The \* symbol can be used in place of the axis designator. The \* defines data for all axes to be the same. For example:

Syntax	Description
PRB=1000	Sets B axis data at 1000
PR*=1000	Sets all axes to 1000

### Implicit Notation

These commands require numerical arguments to be specified following the instruction. Values may be specified for any axis separately or any combination of axes. The comma delimiter indicates argument location. For commands that affect axes, the order of arguments is axis A first, followed by a comma, axis B next, followed by a comma, and so on. Omitting an argument will result in two consecutive commas and doesn't change that axis' current value. Examples of valid syntax are listed below.

Valid Syntax	Description
AC n	Specify argument for A axis only
AC n,n	Specify argument for A and B only
AC n,,n	Specify argument for A and C only
AC n,n,n,n	Specify arguments for A,B,C,D axes
AC .n,,n	Specify arguments for B and E axis only
AC ,,,n,n	Specify arguments for E and F

Where n is replaced by actual values.

### Accepts Axis Mask

These commands require the user to identify the specific axes to be affected. These commands are followed by uppercase X,Y,Z and W or A,B,C,D,E,F,G and H. In DMC code, X,Y,Z,W and A,B,C,D are synonyms, respectively.

No commas are used and the order of axes is not important. When an argument is not required and is not given, the command is executed for all axes.

Valid Syntax	Description
SH A	Servo Here, A only
SH ABD	Servo Here, A,B and D axes
SH ACD	Servo Here, A,C and D axes
SH ABCD	Servo Here, A,B,C and D axes
SH XYZW	Identical to SH ABCD
SH BCAD	Servo Here, A,B,C and D axes
SH ADEG	Servo Here, A,D,E and G axes
SH H	Servo Here, H axis only
SH	Servo Here, all axes

### Two Letter Only

These commands have no options or arguments. Some examples follow.

Valid Syntax	Description
--------------	-------------

BN	Burn parameters
BV	Burn Variables
BP	Burn Programs (not applicable on the DMC30000)
ID	Identify hardware configuration
LA	List arrays

## Operator or Comparator

Operators and Comparators take two arguments and return one value. All comparison and operations occur left to right. That is, multiplication and addition have the same order-of-operation priority, and operations and comparisons are performed as encountered on a left to right search. Parenthesis should be used to indicate order of operation precedence. Some examples follow.

Valid Syntax	Description
var = 1 + 1	Variable var is assigned value 2
var = 2 + 1 * 3	Variable var is assigned value 9
var = 2 + (1 * 3)	Variable var is assigned value 5
IF ((a=b) & (a=c))	Checks if a=b=c
IF (a = b = c)	Invalid syntax to check if a=b=c
var = (a=1)	var is assigned with Boolean value (true/false) based on comparison a=1

## At Function

At functions take one value or evaluated expression and return a result. Some examples follow.

Valid Syntax	Description
var = @SIN[90]	Variable var is assigned value 1. Sine of 90 degrees.
var = @ASIN[1]	Variable var is assigned value 90. Inverse Sine of 1
var = @IN[1]	Variable var is assigned 1 or 0, based on current state of digital input 1
var = @RND[1 + 0.6]	Variable var is assigned 2, 1.6 round to the nearest integer

## Embedded Only

Embedded commands make sense only in the context of an embedded application. These commands include jumps, if-then-else syntax, subroutines, etc. Some examples follow.

Valid Syntax	Description
#go	Labels can be called by name in order to jump code to specific lines
JP#go	Jump to line number indicated by #go label
#AUTO	Automatic subroutine. #AUTO is the entry point for execution on bootup. See entries starting with # for other automatic subroutines.
RI	Return from interrupt. This is the termination for certain automatic subroutines (event handlers)
IF (a=5);MG"Five";ELSE;MG"Not Five";ENDIF	If statement. ; can be replaced by carriage return for better readability

Automatic subroutines operate very similarly to event handlers in event-driven languages. When an event occurs, execution of code jumps to the automatic subroutine. Once the end of the automatic subroutine is reached, code execution continues where it left off.

## Operand

Many commands have corresponding operands that can be used for interrogation or for use within mathematical or other expressions. Operands are not valid alone, and must be used inside a valid DMC code expression. For example, to print the value of the *TIME* operand the following command is issued.

```
:MG TIME
13779546.0000
:
```

To assign *TIME* to a variable and then print it, the following is used.

```
:var= TIME
:MG var
13909046.0000
:
```

All DMC codes starting with the underscore \_ character are operands. The servo loop counter, *TIME*, is an operand without an underscore.

Variables and array elements act similarly to operands. Whereas operands are read-only, variables and array elements are read-write. Operands, variables, and array elements can be arguments to commands, are valid in mathematical expressions, and can be used in assignments to other variables and array elements.

## Tripoints

The controller provides several commands that can be used to pause execution of code until certain conditions are met. Commands of this type are called "trippoints." Such tripoints may wait for an elapsed time, wait for a particular input, or in motion controllers wait for particular motion event to occur.

When a tripoint command is executed, the program halts execution of the next line of code until the status of the tripoint is cleared. Note that the tripoint only halts execution of the thread from which it is commanded while all other independent threads are unaffected. Additionally, if the tripoint is commanded from a subroutine, execution of the subroutine, as well as its calling thread, is halted.

**Tripoints are intended for use only within embedded DMC code and should not be sent from a terminal or a host application program executing from a PC.**

### Popular Tripoints

Trippoint	Short Description	Supported On
WT	wait for a time period (sleep)	All Galil Hardware
AI	wait for a digital input	All Galil Hardware
AM	after move	Motion Controllers
MC	motion complete, in position	Motion Controllers
AT	At time, time from reference	All Galil Hardware
AD	after distance	Motion Controllers
AS	At speed	Motion Controllers
AV	After Vector Distance	Motion Controllers
AA	After Analog	RIO-47xxx only

## Comments

Comments are used to document code, and to disable lines of code while debugging. There are three ways to comment.

**REM** REM stands for "Remark." When a line begins with the REM command, the entire line is stripped by Galil software before downloading to the controller. REM is NOT a recognized Galil command; it is a keyword recognized by Galil software as data that is to be skipped during program download. When program speed and code length are at a premium, use REM comments.

**NO** NO stands for "No Operation." Lines beginning with NO are downloaded to the controller and incur a non-zero processing overhead as a result. If the developer desires the comments

to stay in code so that uploaded code will still be noted, use NO or '. NO comments are not stripped when code is compressed by software.

' The single quote character is similar to NO. Lines beginning with ' are downloaded to the controller and incur a non-zero processing overhead as a result. If the developer desires the comments to stay in code so that uploaded code will still be noted, use NO or '. ' comments ARE stripped when code is compressed by software.

When commenting inline, NO and ' are valid when preceded by a ; character. REM is only valid as the start of a line. Some examples follow.

```
BG ;' This is a comment. semicolon and ' precede, followed by spaces, and then the comment
ST ;NO Same as above, except on compression, this data will remain, less spaces
REM This is a remark. It will not be downloaded to the controller by Galil software
NO This is an NO comment starting a line
NOTE This is also an NO comment
' This is a single quote comment starting a line
REM PRX=1000;BGX;' This line of code has been disabled with a leading REM
```

## Special characters ; and '

;  
The semicolon is used to separate individual commands on a single line of embedded code or in a single interrogation from the host. When running multi-threaded, embedded code, all commands on a single line will be executed before the program counter switches to the next thread\*. Using multiple commands on a single line therefore allows for increased thread priority.

\* Certain commands such as tripoints will cause the program counter to continue to the next thread before a line has completed.

'  
On the RIO series of PLCs and the DMC3000, the backtick (ascii 96) is a line continuation character. If a line of code passes the controller's 40 character length limit, the ' character can be used to continue the code on the next line.

## Interrogation

Most commands accept a question mark (?) as an argument. This argument causes the controller to return parameter information. Type the command followed by a ? for each axis requested. The syntax format is the same as the parameter arguments described above except "?" replaces the values.

Syntax	Description
PR ?	The controller will return the PR value for the A axis
PR _,_? ?	The controller will return the PR value for the D axis
PR ?,?,? ?	The controller will return the PR value for the A,B,C and D axes
PR ,,,,_? ?	The controller will return the PR value for the H axis
PR*=?	The controller will return the PR value for all axes

## Data Types

### Galil4.2

There is only one native data type in DMC language, the Galil4.2 format. Galil4.2 is a signed, fixed-point, decimal number with 4 bytes of integer and 2 bytes of fraction. Bit encoding of Galil4.2 is 2's complement.

Integer values range from -2,147,483,648 to 2,147,483,647

Fractional values range from 0.999985 to .000015 in increments of .000015 (one part in 65535). When working with very small fractional values, use the \$ formatter to display the number in hex

```
:v= 1-$0.0001;'subtract the smallest fractional value
:v= ?
1.0000
:v= ?{$1.4};'hex display has higher resolution
$0.FFFF
:v= v+$0.0001
:v= ?
1.0000
:v= ?{$1.4}
$1.0000
:
:
```

## Strings

Galil "strings" are still variables in 4.2 format, with each byte printed as the ASCII representation of the number. Galil strings are max 6 characters. The left most character of a string is the most significant byte in the Galil4.2 number.

## Boolean

A Boolean is represented in the Galil language as a Galil4.2 value. 0.0 is false. All other values are true.

```
:a= 1
:b= 2
:c= (a=b);' (a=b) returns a Galil Boolean
:LV
a= 1.0000
b= 2.0000
c= 0.0000
:a= 2
:c= (a=b)
:LV
a= 2.0000
b= 2.0000
c= 1.0000
:
```

## Units of Distance

The units of distance in a Galil controller are either in "counts" or "steps". A count is a single unit of feedback, such as a quadrature count, an SSI or BiSS bit, or an Analog to Digital converter bit. Counts are typical with servos. Steps are used for stepper-type motors. Steps are open-loop units and refer to a single level transition sent to a stepper amplifier. In general for a unit of real distance, 1 step is NOT equal in distance to 1 count. See the "Stepper Position Maintenance Mode" in the user manual for more information.

Each axis of a Galil motion controller can be configured to control either a servo or a stepper. In this documentation, servo motors are generally assumed. Differences between functionality in stepper and servo operation are noted in each command. Where not explicitly noted otherwise, when using stepper motors, the unit "count" can be exchanged with the unit "step" (e.g. steps per second instead of counts per second).

## Flash Memory

Each Galil controller has a flash memory provided for saving parameters and user data. The flash is divided into three sectors, Parameters, Variables and Arrays, and Program. Each sector has an associated burn command which burns the entire sector.

Flash Sector	Data Storage	Burn Command
Parameters	Stores the controller parameters such as PID filter coefficients, IP address, motion kinematic values, I/O configurations	BN
Variables and Array	Stores the currently allocated variable table (LV) and each of the arrays in the array table (LA)	BV
Program	Stores program currently downloaded on the controller*	BP

*\*The DMC30000 downloads and runs programs directly out of flash. BP is not applicable.*

### **Resetting the Controller to Factory Defaults**

When a master reset occurs, the controller will reset all setup parameters to their default values and the non-volatile memory is cleared to the factory state. A master reset is executed by the command, <ctrl R> <ctrl S> <Return> OR by powering up or resetting the controller with the MRST jumper on.

'	Comment	PROGRAMMING
The ' allows for a user to insert in a comment on a blank line after a command following a semicolon ";"		
		

## DESCRIPTION

The ' allows for a user to insert in a comment on a blank line after a command following a semicolon ;;. See examples for valid uses of'.

## ARGUMENTS

' str

Argument	Value	Description	Notes
str	String	Comments added into program	Comment strings are restricted to the maximum row size for a program. This will vary per controller.

## REMARKS

- Comments will be downloaded to controller, thus taking up program space.
  - See REM for comments that will not download to controller

## EXAMPLES

```
'Include an example like this one in the program.  
SH AB;'Comments following a command MUST be proceeded by a semi-colon.  
KP 10'This is NOT valid use of the '.
```

' is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

-   <i>Subtraction Operator</i>	PROGRAMMING, MATH FUNCTIONS
<b>Subtraction operator</b>	

variable = (value1 - value2) | Performs an operation between two values or evaluated statements

## DESCRIPTION

Subtraction operator. Takes as arguments any two values and returns a value equal to the difference of the arguments.

## ARGUMENTS

### n0 - n1

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to subtract from	
<b>n1</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to subtract	

## REMARKS

- An operator is not a command and is not valid individually.
- Evaluation occurs left to right. Use parenthesis for operator precedence.
- n0 and n1 may also be variables, array elements, operands, or @ functions (e.g. @SIN[]).

## EXAMPLES

```
:var1 = 10-4
:var2 = var1 - 3
:MG var2 - 1
2.0000
:
```

'It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.  
'Example:  
var = ((10\*30)+(60/30));' evaluates as 302  
var = 10\*30+60/30;' evaluates as 12

- is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

#	<i>Label Designator</i>	PROGRAMMING, SUBROUTINE
	Denotes the name of a program label	
		

## DESCRIPTION

Denotes the name of a program label. For example, #move. Labels can be up to seven characters long and are often used to implement subroutines or loops. Labels are either user-defined or are automatic subroutines that are run automatically by the firmware when a particular event occurs.

DMC40x0, DMC41x3, DMC21x3, DMC18x6

There is a maximum of 510 labels available.

## ARGUMENTS

#str

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	7 chars	N/A	String	Name of label	

## REMARKS

- A label can only be defined at the beginning of a new line
- The number of labels available can be queried with MG\_DL
- LL returns the current label table in the controller
- Galil recommends that at least the first character be lowercase to differentiate from Automatic subroutines.
- Label must be the first element on a line of code

## EXAMPLES

```
'A simple example of iteration. The loop will run 10 times
i= 0;'          Create a counter
#loop; '         Label
    i= i+1; '    Increment counter
JP #loop, i<10; ' Spin in #Loop until i >= 10
EN; '           End the subroutine or thread
```

# is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#AMPERR</b>	<i>Amplifier error automatic subroutine</i>	PROGRAMMING, ERROR CONTROL, SUBROUTINE
Automatic subroutine used to run code when a fault occurs on a Galil amplifier		
		

## DESCRIPTION

Automatic subroutine used to run code when a fault occurs on a Galil amplifier. See the TA command and individual amplifier information in the controller user manual.

DMC40x0, DMC41x3, DMC30010

Thread 0 does not need to be running for #AMPERR to be raised. This was a requirement on earlier products.

## ARGUMENTS

### #AMPERR

Label must be the first element on a line of code.

## REMARKS

- Use RE to return from the AMPERR subroutine
- See the TA command for more information

DMC40x0, DMC41x3

- When an external servo driver is used on an axis where the AMP-430x0 is also installed, the axis should be setup as a brushed motor (BRm=1), otherwise the lack of hall inputs will cause an amplifier error.

## EXAMPLES

```
'this code will run in the event of an amplifier error,
'setting a digital output and notifying the operator.

#AMPERR
'Set a digital bit to signal an amplifier error to peripheral hardware
SB 4

'Send a message to the user
MG "An amplifier error has occurred"

'Return from the AMPERR subroutine, restoring trippoints that were running
RE 1
```

#AMPERR is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#AUTO</b>	<i>Subroutine to run automatically upon power up</i>	PROGRAMMING, SUBROUTINE
Defines the automatic entry point of embedded DMC code		
		

## DESCRIPTION

Defines the automatic entry point of embedded DMC code. When power is applied to the controller, or after the controller is reset, the program will automatically begin executing at this label. When no host software is used with the controller, #AUTO is required to run an application program on the controller stand-alone.

## ARGUMENTS

### #AUTO

Label must be the first element on a line of code.

## REMARKS

- Use EN to end the routine
- Thread 0 is used to execute #AUTO on startup

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

- The BP command must be used to burn a program into EEPROM for the #AUTO to function.

## EXAMPLES

```
'On startup, this code will create a 50% duty cycle square wave on output 1 with a period of 1 second.
#AUTO;          Start on powerup
SB 1;          Set bit 1
WT 500;        Wait 500msec
CB 1;          Clear bit 1
WT 500;        Wait 500msec
JP #AUTO;      Jump back to #AUTO
```

#AUTO is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#AUTOERR</b>	<i>Bootup Error Automatic Subroutine</i>	PROGRAMMING, ERROR CONTROL, SUBROUTINE
#AUTOERR will run code upon power up if the firmware detects errors		
		

## DESCRIPTION

#AUTOERR will run code upon power up if the firmware detects errors. If the EEPROM is corrupted, #AUTOERR will run. The EEPROM is considered corrupt if the checksum calculated on the bytes in the EEPROM do not match the checksum written to the EEPROM.

### SER

#AUTOERR will also run if the time to acquire serial position data exceeds 90% of the hardware sample loop. This type of error is very rare and should never occur in normal operation.

## ARGUMENTS

### #AUTOERR

Label must be the first element on a line of code.

## REMARKS

- Use EN to end the routine
- The type of checksum error can be queried with MG\_RS

### SER

- In the event of a serial position acquisition timeout, the following will occur:
  - The controller will reset
  - The controller servo loop will not run, TM will be set to zero
  - TCI will return "143 TM timed out"
  - The automatic subroutine #AUTOERR will run, if present
  - The Error output will be set
- When using serial encoders (SSI or BiSS), the #AUTOERR should follow these guidelines
  - IF\_TC=143 do not employ any trippoints in following code because the timer interrupt is suspended
  - Serial encoders can be disabled with the commands SIn=0 or SSn=0 where n is the axis indicator ABCDEFG or H
  - In order to re-enable the timer interrupt issue "TM n" where n is the servo update period in us (usually n=1000). See TM for more details

## EXAMPLES

```
'Code detects a checksum error and notifies the user
#AUTOERR
  MG "EEPROM ERROR ",_RS
EN
```

### SER

```
'Distinguishing between a serial timeout
' condition and an EEPROM condition
#AUTOERR
IF _TC=143
REM BiSS or SSI timeout
REM No trippoints in this clause
REM Print message to DMC-4020 LCD
LU 0
MG "BiSS"\{L1}
MG "Timeout"\{L2}
SSA= 0
SSB= 0
ELSE
REM Checksum error
REM trippoints ok here
REM Print message to DMC-4020 LCD
LU 0
MG "EEPROM:"\{L1}
MG \{Z10.0\}_RS\{L2}
ENDIF
EN
```

#AUTOERR is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#CMDERR</b>	<i>Command error automatic subroutine</i>	PROGRAMMING, ERROR CONTROL, SUBROUTINE
Automatic subroutine that runs code when a DMC code error occurs		
		

## DESCRIPTION

Automatic subroutine that runs code when a DMC code error occurs. Without #CMDERR defined, if an error (see TC command) occurs in an application program running on the Galil controller, the program (and all threads) will stop.

## ARGUMENTS

### #CMDERR

Label must be the first element on a line of code.

## REMARKS

- Use EN to end the routine
- #CMDERR will only run from errors generated within embedded DMC code, not from the terminal or host
- In a single threaded application (Thread 0 only), the EN command in the #CMDERR routine will restart thread 0 where it left off.
- In a multi-threaded application, the thread that has an error will be halted when a command error occurs. Thread 0 will be interrupted to run the #CMDERR routine but other threads will continue to run.
  - In order to restart the thread that encountered the error, see the example in Chapter 7 of the User Manual and the \_ED operand.

DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010

- Thread 0 does not need to be running in order for the #CMDERR routine to execute.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
'This code will put the motion controller in Position Tracking mode.
'Variable "target" is updated from the terminal or from a host program
'to specify a new target. #CMDERR is used to detect a bad target value.

#start
DPA= 0;          Define current position as zero
PTA= 1;          Turn on position tracking
target= 0;        Initialize target variable
#track;          Start tracking
PAA= target;    Track to current value of target
WT 500;          Wait 500 ms
JP #track;       Continue to track
'

#CMDERR; ' runs if an error occurs
JP #done,_TC<>6 ;'check that an out of range occurred (See TC)
MG "Value ",target," is out of range for Position Tracking"
target= _PAA ;' reset target
#done
EN 1 ;'return to tracking logic
```

#CMDERR is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#COMINT</b>	<i>Communication interrupt automatic subroutine</i>	PROGRAMMING, SUBROUTINE
Automatic subroutine to provide interrupt driven communications from the serial port		
		

## DESCRIPTION

Automatic subroutine to provide interrupt driven communications from the serial port. #COMINT can be configured by the CI command to run either when any character is received, or when a carriage return is received over the com port. The auxiliary port is used if equipped.

## ARGUMENTS

### #COMINT

Label must be the first element on a line of code.

## REMARKS

- Use EN to end the routine
- #COMINT runs in thread 0, and an application must be running in thread 0 in order for #COMINT to be enabled.
- Code running in thread zero will be interrupted by the #COMINT subroutine.

## EXAMPLES

DMC40x0, DMC41x3

```
#a;                                'Program Label
CC 9600,0,1,0
CI 2;                                'interrupt on any character
#loop
MG "Loop";
WT 1000
JP #loop
#COMINT
MG "COMINT:", P2CH(S1);      'print a message and the received character
EN 1,1;                                ' End this subroutine, re-arming trip points that
                                         were running and re-enabling the CI mask
```

#COMINT is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#ININT</b>	<i>Input interrupt automatic subroutine</i>	PROGRAMMING, IO, SUBROUTINE
<b>Automatic subroutine that runs upon a state transition of digital inputs</b>		
		

## DESCRIPTION

Automatic subroutine that runs upon a state transition of digital inputs. #ININT is configured with II. #ININT runs in thread 0.

## ARGUMENTS

### #ININT

Label must be the first element on a line of code.

## REMARKS

- Use the II command to enable the routine.
- Use RI to exit the routine.
- To make an unconditional jump from #ININT, there are two methods for re-enabling the interrupt capability
  - Issue a ZS and then re-issue the command II before the JP
  - or, use a "null" routine. The "null" routine allows for the execution of the RI command before the unconditional jump. For more information see Application Note #2418, <http://www.galilmc.com/support/appnotes/optima/note2418.pdf>

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
II 1;          'arm digital input 1
EN;           'End thread zero
'
#ININT;        'Automatic sub. Runs on input event
MG "Inputs:",_TIO; 'Display status of inputs 1-8
WT 100;        'Debounce input
RI;           'Return from interrupt
```

#ININT is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

**DESCRIPTION**

Automatic sub for running user-defined code on a limit switch event. Without #LIMSWI defined, the controller will issue ST on the axis when it's limit switch is tripped during motion in the direction of the switch. With #LIMSWI defined, code is executed in addition to the stop.

**ARGUMENTS****#LIMSWI**

Label must be the first element on a line of code.

**REMARKS**

- Use RE to terminate the subroutine
- See \_LF and \_LR for switch state operands

DMC40x0, DMC41x3, DMC18x6, DMC30010

- #LIMSWI runs on thread 0. Code does not need to be running in thread 0 for #LIMSWI to be enabled.
- LD can be used to disable the limit operation
- SD can be used to set the deceleration speed on the limit.

**EXAMPLES**

DMC40x0, DMC41x3, DMC18x6

```
#main      ;'print a message every second
  MG "Main"
  WT 1000
  JP #main
EN
'
#LIMSWI ;'runs when a limit switch is tripped
MG "Limit switch:{N}
IF ((_LFA = 0) | (_LFA = 0))
  MG "Axis A"
ENDIF
IF ((_LFB = 0) | (_LFB = 0))
  MG "Axis B"
ENDIF
RE 1; ' RE used to exit the #LIMSWI sub
```

#LIMSWI is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#MCTIME</b>	<i>MC command timeout automatic subroutine</i>	PROGRAMMING, SUBROUTINE
Automatic sub used to run user-code if a Motion Complete (MC) trippoint times out		
		

## DESCRIPTION

Automatic sub used to run user-code if a Motion Complete (MC) trippoint times out. If the motor position does not reach or pass the target within the specified timeout (TW), #MCTIME will run if present.

MC uses position from TP for servos, or TD for steppers.

## ARGUMENTS

### #MCTIME

Label must be the first element on a line of code.

## REMARKS

- Use EN to terminate the subroutine

## EXAMPLES

```

#begin;          Begin main program
TWA= 1000;      Set the time out to 1000 ms
PRA= 10000;     Position relative
BG A;           Begin motion
MC A;           Motion Complete trip point
EN;             End main program
'
#MCTIME;        Motion Complete Subroutine
MG "A fell short"; Send out a message
EN 1;           End subroutine

```

#MCTIME is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#POSERR</b>	<i>Position error automatic subroutine</i>	SUBROUTINE
#POSERR is an automatic subroutine that runs user code when a position error event occurs		

## DESCRIPTION

#POSERR is an automatic subroutine that runs user code when a position error event occurs. The factory default behavior of the Galil controller upon a position error ( $\_TEn > \_ERn$ ) is to drive the error signal low only, turning on the red error LED. If OE is set to 1, the motor whose position error (TE) equals or exceeds its threshold (ER) will be turned off (MO). #POSERR is used to run code upon a position error, for example to notify a host computer.

## ARGUMENTS

### #POSERR

Label must be the first element on a line of code.

## REMARKS

- Use RE to end the routine.

DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010

- #POSERR runs on thread 0. Code does not need to be running in thread 0 for #LIMSWI to be enabled.
- #POSERR will also run when OE1 is set for an axes and that axis is also setup for encoder failure detection (see OA, OT, OV commands).

## EXAMPLES

```
#main;          main program
'
JP #main

REM simple example of #POSERR
#POSERR
  MG "#POSERR"
RE

REM example of #POSERR that checks for position error on each axis
#POSERR
  ~a= 0;        axis designator
  IF ((\_TE~a>\_ER~a) & (\_OE~a))
    MG "Position Error occured on ",~a{F1.0}, " axis"
  ENDIF
  ~a= ~a+1
JP #POSERR, ~a<_BV; ' loop until axes have been checked
  AI 1;         wait until input 1 goes high (ex. safety switch)
  SH
RE 1;          return to main program
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
REM #POSERR example for checking to see if encoder failure occurred
REM The stop code will only update of the profiler is running at the time
REM the encoder failure is detected.
#POSERR
  ~a= 0
  #loop
  IF _MO~a=1
    IF ((\_TE~a<\_ER~a) & (\_OE~a) & (\_OA~a))
      MG "possible encoder failure on ",~a{Z1.0}, " axis"
    ENDIF
  ENDIF
  ~a= ~a+1
JP #loop, ~a<_BV
  AI 1;         wait for input 1 to go high
  SH ;          enable all axes
RE
```

#POSERR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#SERERR</b>	<i>Serial Encoder Error Automatic Subroutine</i>	SUBROUTINE
#SERERR is an automatic subroutine that allows user code to run when there is a serial encoder fault		

#SERERR Only valid from within embedded code. No terminal.

## DESCRIPTION

#SERERR is an automatic subroutine that allows user code to run when there is a serial encoder fault.

DMC40x0, DMC41x3, DMC30010

This subroutine is only used with controllers equipped with hardware featuring the -BiSS encoder upgrade.

## ARGUMENTS

### #SERERR

Label must be the first element on a line of code.

## REMARKS

- Use the RE command to end this routine.
- #SERERR runs on thread 0
- The following are the fault conditions which will cause #SERERR to interrupt.

DMC40x0, DMC41x3, DMC30010

*Serial Encoder Faults*

BiSS
Encoder timeout (bit 0 of _SS)
CRC error (bit 1 of _SS)
Error bit* (bit 2 of _SS)
Warning bit* (bit 3 of _SS)

- The active level of the Error and Warning bits for BiSS must be configured with SY.
- For the encoder timeout condition, TC1 will also return "140 Serial encoder missing"
- Note: The encoder manufacturer may name the Error and Warning bits differently. Consult the encoder documentation for the naming convention.
- Galil defines the Warning bit as the bit directly preceding the CRC. The Error bit is defined as the bit directly preceding the Warning bit.

## EXAMPLES

DMC41x3, DMC30010

```
#SERERR ;' display error, shutdown axis
LU 0
MG "SERERR"
MG _SSA
REM disable axis A
OEA= 1,ERA= 0
REM disable axis serial encoder
SSA= 0
RE
```

#SERERR is supported on DMC40x0, DMC41x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>#TCPERR</b>	<i>Ethernet communication error automatic subroutine</i>	SUBROUTINE
#TCPERR is an automatic subroutine which allows execution of user code when an TCP error event occurs		
		

## DESCRIPTION

#TCPERR is an automatic subroutine which allows execution of user code when an TCP error event occurs. #TCPERR allows the application programmer to run code (for example to reestablish the connection) when error code 123 occurs.

## ARGUMENTS

### #TCPERR

Label must be the first element on a line of code.

## REMARKS

- Use RE to exit this subroutine.
- Error code 123 (TCP lost sync or timeout) occurs when a message is sent out a handle, and no acknowledgement is received.
  - When this occurs, the handle the message was sent out is closed.
  - #TCPERR can be used to reestablish the handle

DMC40x0, DMC41x3, RIO, DMC30010

- Code does not need to be running in thread 0 for #TCPERR to run.

## EXAMPLES

```

#loop
  MG {EA} "L"
  WT 1000
JP #loop

#TCPERR
  MG {P1} "TCPERR.  Dropped handle", _IA4
RE

'example of reestablishing connection after TCPERR
'

#main
IHE= 192,168,1,30;   connect to 192,168,1,30
WT 100;               wait for handle to be established
ipe= _IHE0;           save IP for reconnection use
n= 0;                 connection counter
#loop;                endless message loop
MG "hello"
WT 1000
JP #loop
EN

#TCPERR
IHE= >-3;             make sure handle E is clear
JP #TCPERR, _IHE2<>0; wait for clear handle
IHE= ihe;              set handle with saved IP var
WT 100
n= n+1;               increment counter
JP #end,n>5;          try at least 5 times
JP #TCPERR, _IHE2<>-2; repeat if handle failed
#end
IF (n>5)
  MG "failed connection"
  HX 0;                  stop code if connection lost
ELSE
  MG "Reconnected"
  n = 0;                 reset connection counter
ENDIF
RE

```

#TCPERR is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

The \$ operator denotes that the following string is in hexadecimal notation



## DESCRIPTION

The \$ operator denotes that the following string is in hexadecimal notation.

ARGUMENTS

Sn

Argument	Min	Max	Default	Resolution	Description	Notes
n	\$80000000.0000	\$7FFFFFFF.FFFF	N/A	\$0.0001	Value of hexadecimal number	32 bits of integer and 16 bits of fraction in total

## **REMARKS**

- None

## EXAMPLES

```
x = $ffffffff.0000 ;'store 2147483647 in x  
y = x & $0000ffff.0000 ;'store lower 16 bits of x in y  
z = x & $ffff0000.0000 / $10000 ;'store upper 16 bits of x in z
```

\$ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>%</b>	<i>Modulo Operator</i>	MATH FUNCTIONS
The %symbol is the modulo operator		

variable = (value1 % value2) Performs an operation between two values or evaluated statements

## DESCRIPTION

The % symbol is the modulo operator. It takes as arguments any two values, variables, array elements, operands, or At functions (@SIN[]) and returns a value equal to the modulo of the arguments.

Mathematical operations are calculated left to right rather than multiplication and division calculations performed prior to addition and subtraction.

Example:

1+2\*3 = 9, not 7

It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.

Example:

var=((10\*30)+(60/30));' evaluates as 302

var = 10\*30+60/30;' evaluates as 12

## ARGUMENTS

**n % n**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n</b>	-2147483648.0000	-2147483647.9999	N/A	1/65536	Value to use in modulo operation	

## REMARKS

- This is a binary operator (takes two arguments and returns one value). The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.
- Mathematical operations are calculated left to right rather than multiplication and division calculations performed prior to addition and subtraction.
  - Example: 1+2\*3 = 9, not 7
- It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.
  - Example: var=((10\*30)+(60/30));' evaluates as 302
  - var = 10\*30+60/30;' evaluates as 12

## EXAMPLES

```
'Determine the day of week in n days
DM name[7];'Strings for day of week
name[0]= "SUN"
name[1]= "MON"
name[2]= "TUE"
name[3]= "WED"
name[4]= "THU"
name[5]= "FRI"
name[6]= "SAT"
today= 2; 'Tuesday
days= 123; 'Days from now
dow= ((days + today)%7);'calculate future day of week
MG "The day of week in ",days{Z10.0}, " days will be ", name[dow]{S3.0}
EN
```

REM Code Returns: The day of week in 123 days will be SAT

% is supported on DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

&	<i>JS subroutine pass variable by reference</i>	SUBROUTINE
The & symbol is used to pass a variable by reference on the subroutine stack		
		

## DESCRIPTION

The & symbol is used to pass a variable by reference on the subroutine stack. When passed by reference, a change to the local-scope variable changes the global value.

## ARGUMENTS

DMC40x0, DMC41x3, DMC18x6

**JS#str0(&str1,&str1,&str1,&str1,&str1,&str1,&str1)**

Argument	Min	Max	Default	Resolution	Description	Notes
str0	1 char	7 chars	N/A	String	Name of label to use for subroutine call	
str1	1 char	8 chars	N/A	String	Name of variable to pass by reference to the subroutine	

## REMARKS

- Variables sent to a subroutine must be global variables that are already dimensioned.
- To ensure that the global variable does not get changed, omit the & symbol to send a local copy of the variable to the stack.

## EXAMPLES

Pass By Reference Example:

```
#main
value= 5;          a value to be passed by reference
global= 8;          a global variable
JS #sum(&value,1,2,3,4,5,6,7);' note first arg passed by reference
MG value;          message out value after subroutine.
MG _JS;'          message out returned value
EN
'
#sum;'           (* ^a,^b,^c,^d,^e,^f,^g)
^a= ^b+^c+^d+^e+^f+^g+^h+global
EN ,^a
'notes-
'do not use spaces when working with ^
'If using global variables, they MUST be created before the subroutine is run
'From Terminal
:
Executed program from program2.dmc
36.0000
36.0000
```

& is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>&amp;</b>	<i>Bitwise AND Operator</i>	<b>Misc</b>
The & symbol is the bitwise AND operator used with IF, JP, and JS decisions, and also to perform bitwise ANDING of values		

`variable = (value1 & value2)` Performs an operation between two values or evaluated statements

## DESCRIPTION

The & symbol is the bitwise AND operator used with IF, JP, and JS decisions, and also to perform bitwise ANDING of values.

ARGUMENTS

n & n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use with AND operator	

## REMARKS

- The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.
  - For IF, JP, and JS, the values used for n are typically the results of logical expressions such as  $(x > 2) \& (y = 8)$

## EXAMPLES

```
'Bitwise use
:var1= $F;'00001111
:var2= $FO;'1111000
:MG (var1 & var2)
0.0000
:MG var1
15.0000
:MG var2
240.0000
:

'Conditional Use
var1= $F;'00001111
var2= $FO;'1111000
IF (var1 = $F) & (var2 = $F1)
  MG "True"
ELSE
  MG "False"
ENDIF
EN

REM Returned: False
```

& is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

(	Parentheses (order of operations)	MATH FUNCTIONS
The parentheses denote the order of math and logical operations		

## DESCRIPTION

The parentheses denote the order of math and logical operations.

## ARGUMENTS

(n)

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2147483648.0000	2147483647.9999	N/A	1/65536	Math or logical expression for evaluation	

## REMARKS

- Note that the controller DOES NOT OBEY STANDARD MATHEMATICAL OPERATOR PRECEDENCE.
  - For example, multiplication is NOT evaluated before addition.
- Instead, the controller follows left-to-right precedence.
  - Therefore, it is required to use parentheticals to ensure intended precedence.

## EXAMPLES

```
:MG 1+2*3
9.0000
:MG 1+(2*3)
7.0000
```

```
:var1= $1F
:var2= $F
:MG var1&var2/$10
0.9375      ($0.F000)
:MG var1&(var2/$10)
0.0000      ($0.0000)
```

( , ) is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)


variable = (value1 \* value2) | Performs an operation between two values or evaluated statements

MATH FUNCTIONS

## DESCRIPTION

The \* symbol is the multiplication operator. It takes as arguments any two values, variables, array elements, operands, or At functions (@SIN[]) and returns a value equal to the product of the arguments.

## ARGUMENTS

n \* n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	-2,147,483,647	N/A	1/65,536	Value to use in multiplication operation	

## REMARKS

- This is a binary operator (takes two arguments and returns one value). The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.
- Mathematical operations are calculated left to right rather than multiplication and division calculations performed prior to addition and subtraction.
  - Example: 1+2\*3 = 9; not 7
- It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.
  - Example: var =((10\*30)+(60/30)); evaluates as 302
  - var = 10\*30+60/30; evaluates as 12

## EXAMPLES

```
:var1 = (2 + 3) * 2
:var2 = var1 * 10
:MG var2 * 0.5
50.0000
:
```

\* is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

/	<i>Division Operator</i>	MATH FUNCTIONS
<b>The / symbol is the division operator</b>		
		

variable = (value1 / value2) | Performs an operation between two values or evaluated statements

## DESCRIPTION

The / symbol is the division operator. It takes as arguments any two values, variables, array elements, operands, or At functions (@SIN[]) and returns a value equal to the quotient of the arguments.

## ARGUMENTS

### n0 / n1

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2,147,483,648	2,147,483,647	N/A	1/65,536	Numerator of divide operation	
<b>n1</b>	-2,147,483,648	2,147,483,647	N/A	1/65,536	Denominator of divide operation	

## REMARKS

- This is a binary operator (takes two arguments and returns one value). The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.
- Mathematical operations are calculated left to right rather than multiplication and division calculations performed prior to addition and subtraction.
  - Example:  $1+2*3=9$ ; not 7
- It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.
  - Example: var = ((10\*30)+(60/30)); evaluates as 302
  - var = 10\*30+60/30; evaluates as 12

## EXAMPLES

```
:var1 = 100/10
:var2 = var1/2
:MG var2 + 1
6.0000
:
```

/ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

The semicolon operator allows multiple Galil commands to exist on a single line



## DESCRIPTION

The semicolon operator allows multiple Galil commands to exist on a single line.

## ARGUMENTS

**arg;arg;arg;arg**

arg represents any valid Galil command

## REMARKS

- The semicolon operator is used for the following three reasons:
  - (1) To put comments on the same line as the command (STX ;'stop)
  - (2) To compress DMC programs to fit within the program line limit (Note: use a compression utility to do this. Do not program this way because it is hard to read.)
  - (3) To give higher priority to a thread. All commands on a line are executed before the thread scheduler switches to the next thread.

## EXAMPLES

```
SB 1;WT 500;CB 1;# multiple commands separated by semicolons with a comment
```

```
#high;#High priority thread executes twice as fast as
a = a + 1; b = b + 1
JP #high
```

```
#low;#Low when run in parallel
c = c + 1
d = d + 1
JP #low
```

; is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@ABS</b>	Absolute value	MATH FUNCTIONS
The @ABS[] operation takes the absolute value of the given number		

variable = @ABS[value] Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @ABS[] operation takes the absolute value of the given number. Returns the value if positive, and returns -1 times the value if negative.

## ARGUMENTS

**@ABS[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,535	Number to display as absolute value	

## REMARKS

- @ABS[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @ABS[-2147483647]
2147483647.0000
```

@ABS is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@ACOS</b>	<i>Inverse cosine</i>	MATH FUNCTIONS
The @ACOS operator returns in degrees the arc cosine of the given number		
		

variable = @ACOS[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @ACOS operator returns in degrees the arc cosine of the given number.

## ARGUMENTS

**@ACOS[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-1	1	N/A	1/65,536	Value used for arc cosine operation	

## REMARKS

- @ACOS[] is an operand, not a command. It can only be used as an argument to other commands and operators
- @ACOS[] is also referred to as the inverse cosine function

## EXAMPLES

```
:MG @ACOS [-1]
180.0000
:MG @ACOS [0]
90.0000
:MG @ACOS [1]
0.0001
```

@ACOS is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@AN</b>	Analog Input Query	<b>IO</b>
The @AN[] operator returns the value of the given analog input in volts		

variable = @AN[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @AN[] operator returns the value of the given analog input in volts.

## ARGUMENTS

**@AN[n]**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	8	N/A	1	Analog input to query	

## REMARKS

- @AN[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @AN[1] ;'print analog input 1
1.7883
:x = @AN[1] ;'assign analog input 1 to a variable
```

@AN is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@ASIN</b>	<i>Inverse sine</i>	MATH FUNCTIONS
The @ASIN operator returns in degrees the arc sine of the given number		
		

variable = @ASIN[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @ASIN operator returns in degrees the arc sine of the given number.

## ARGUMENTS

**@ASIN[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-1	1	N/A	1/65,536	Value used for arc sine operation	

## REMARKS

- @ASIN[] is an operand, not a command. It can only be used as an argument to other commands and operators
- @ASIN[] is also referred to as the inverse sine function

## EXAMPLES

```
:MG @ASIN[-1]
-90.0000
:MG @ASIN[0]
0.0000
:MG @ASIN[1]
90.0000
```

@ASIN is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@ATAN</b>	<i>Inverse tangent</i>	MATH FUNCTIONS
--------------	------------------------	----------------

The @ATAN operator returns in degrees the arc tangent of the given number



variable = @ATAN[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @ATAN operator returns in degrees the arc tangent of the given number.

## ARGUMENTS

**@ATAN[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,638	2,147,483,647	N/A	1/65,536	Value used for arc tangent operation	

## REMARKS

- @ATAN[] is an operand, not a command. It can only be used as an argument to other commands and operators
- @ATAN[] is also referred to as the inverse tangent function

## EXAMPLES

```
:MG @ATAN [-10]
-84.2894
:MG @ATAN [0]
0.0000
:MG @ATAN [10]
84.2894
```

@ATAN is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@COM</b>	<i>Bitwise complement</i>	MATH FUNCTIONS
The @COM[] operation performs the bitwise complement (NOT) operation to the given number		

variable = @COM[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @COM[] operation performs the bitwise complement (NOT) operation to the given number.

## ARGUMENTS

### @COM[n]

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1	Value to perform bitwise complement operation.	Integer interpreted as a 32-bit field

## REMARKS

- @COM[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG {$8.0} @COM[0]
$FFFFFFF
:MG {$8.0} @COM[$FFFFFFF]
$00000000
```

@COM is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@COS</b>	Cosine	MATH FUNCTIONS
The @COS[] operation returns the cosine of the given angle in degrees		

variable = @COS[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @COS[] operation returns the cosine of the given angle in degrees

## ARGUMENTS

**@COS[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-32,768	32,767	N/A	1/65,536	Value in degrees to use for cosine operation	

## REMARKS

- @COS[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @COS[0]
1.0000
:MG @COS[90]
0.0000
:MG @COS[180]
-1.0000
:MG @COS[270]
0.0000
:MG @COS[360]
1.0000
```

@COS is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@FRAC</b>	Fractional part	MATH FUNCTIONS
The @FRAC operation returns the fractional part of the given number		
		

variable = @FRAC[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @FRAC operation returns the fractional part of the given number

## ARGUMENTS

**@FRAC[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use in fractional operation	

## REMARKS

- The sign of the number input to the operation will be maintained in the fractional output.
- @FRAC[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @FRAC[1.2]
0.2000
:MG @FRAC[-2.4]
-0.4000
```

@FRAC is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@IN</b>	Read digital input	INTERROGATION, IO
The @IN operand returns the value of the given digital input (either 0 or 1)		

variable = @IN[value] Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @IN operand returns the value of the given digital input (either 0 or 1)

## ARGUMENTS

**@IN[n]**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n</b>	1	16	N/A	1	General input to query	Inputs 9-16 only valid for 5-8 axis controller
	81	96	N/A	1	Aux encoder input to query	Used when repurposing aux encoder inputs as digital inputs

## REMARKS

- @IN[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @IN[1]
1.0000
:x = @IN[1]
:x = ?
1.000 print digital input 1
```

@IN is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@INT</b>	Integer part	MATH FUNCTIONS
The @INT operation returns the integer part of the given number		

variable = @INT[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @INT operation returns the integer part of the given number. Note that the modulus operator can be implemented with @INT (see example below).

## ARGUMENTS

**@INT[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use in integer operation	

## REMARKS

- @INT[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @INT[1.2]
1.0000
:MG @INT[-2.4]
-2.0000
```

```
#AUTO; '      modulus example
x = 10; '    prepare arguments
y = 3
JS #mod; '    call modulus
MG z; '       print return value
EN

'subroutine: integer remainder of x/y (10 mod 3 = 1)
'arguments are x and y. Return is in z
#mod
z = x - (y * @INT[x/y])
EN
```

@INT is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@OUT</b>	Read digital output	IO
Returns the value of the given digital output (either 0 or 1)		
		

variable = @OUT[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

Returns the value of the given digital output (either 0 or 1)

## ARGUMENTS

**@OUT[n]**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	16	N/A	1	General output to query	Outputs 9-16 only valid for 5-8 axis controller

## REMARKS

- @OUT[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @OUT[1];'      print state of digital output 1
1.0000
:x = @OUT[1];'    assign state of digital output 1 to a variable
```

@OUT is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@RND</b>	<i>Round</i>	MATH FUNCTIONS
The @RND operation rounds the given number to the nearest integer		
		

variable = @RND[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @RND operation rounds the given number to the nearest integer.

## ARGUMENTS

**@RND[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use in rounding operation	

## REMARKS

- @FRAC[] is an operand, not a command. It can only be used as an argument to other commands and operators
- The sign of the number input to the operation will be maintained in the rounded output.

## EXAMPLES

```
:MG @RND[1.2]
1.0000
:MG @RND[1.6]
2.0000
:MG @RND[-1.2]
-1.0000
:MG @RND[5.7]
6.0000
:MG @RND[-5.7]
-6.0000
:MG @RND[5.5]
6.0000
:MG @RND[-5.5]
-5.0000
```

@RND is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@SIN</b>	Sine	MATH FUNCTIONS
The @SIN[] operation returns the sine of the given angle in degrees		

variable = @SIN[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @SIN[] operation returns the sine of the given angle in degrees

## ARGUMENTS

### @SIN[n]

Argument	Min	Max	Default	Resolution	Description	Notes
n	-32,768	32,767	N/A	1/65,536	Value in degrees to use for sine operation	

## REMARKS

- @SIN[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @SIN[0]
0.0000
:MG @SIN[90]
1.0000
:MG @SIN[180]
0.0000
:MG @SIN[270]
-1.0000
:MG @SIN[360]
0.0000
```

@SIN is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@SQR</b>	Square Root	MATH FUNCTIONS
The @SQR operation takes the square root of the given number		
  		

variable = @SQR[value] | Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @SQR operation takes the square root of the given number.

## ARGUMENTS

**@SQR[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use in square root operation	If n < 0, the absolute value is taken first.

## REMARKS

- @SQR[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @SQR[2]
1.4142
:MG @SQR[-2]
1.4142
```

@SQR is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>@TAN</b>	Tangent	MATH FUNCTIONS
The @TAN[] operation returns the tangent of the given angle in degrees		
		

variable = @TAN[value] Performs a function on a value or evaluated statement and returns a value

## DESCRIPTION

The @TAN[] operation returns the tangent of the given angle in degrees.

## ARGUMENTS

**@TAN[n]**

Argument	Min	Max	Default	Resolution	Description	Notes
n	-32,768	32,767	N/A	1/65,536	Value in degrees to use for tangent operation	

## REMARKS

- @TAN[] is an operand, not a command. It can only be used as an argument to other commands and operators

## EXAMPLES

```
:MG @TAN[23]
0.4245
:
```

@TAN is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

The square brackets are used to denote the array index for an array, or to denote an array name



## DESCRIPTION

The square brackets are used to denote the array index for an array, or to denote an array name.

They are also used to designate the argument to a function, such as @ABS[n].

## ARGUMENTS

### str[n]

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	8 chars	N/A	String	Name of array to access	Must be a valid dimensioned array name.
n	-1	15,999	N/A	1	Element of array to query	n = -1 returns the array length

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	8 chars	N/A	String	Name of array to access	Must be a valid dimensioned array name.
n	0	399	N/A	1	Element of array to query	For RIO-47xx0
	0	999	N/A	1	Element of array to query	For RIO-47xx2

## REMARKS

DMC40x0, DMC41x3, DMC18x6, DMC30010

- If the array will be passed by reference on the subroutine stack (JS), the array name MUST be 6 characters or less.

## EXAMPLES

```
DM a[50]      ;'define a 50 element array
a[0] = 3      ;'set first element to 3
MG a[0]       ;'print element 0
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
#array
DM a[5];'          define a 5 element array
a[0] = 3;''        set first element to 3
MG "A[0]=",a[0];'  print element 0
len=a[-1];'        len now contains the length of A[]
QU a[],0,len-1,1;MG "";'  print entire array
MG "A[] length=",len;'  display Variable len
EN
```

```
'Example Output from terminal
:XQ #array
:
A[0]= 3
3, 4320, 216666, 217522, 607950
A[] length= 5
:
```

[,] is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>^</b>	<i>JS subroutine stack variable</i>	PROGRAMMING
The ^ character provides local subroutine access for variables passed on the subroutine stack		

## DESCRIPTION

The ^ character provides local subroutine access for variables passed on the subroutine stack. Passing values on the stack is advanced DMC programming, and is recommended for experienced DMC programmers familiar with the concept of passing arguments by value and by reference.

## ARGUMENTS

**^s**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
s	a	h	N/A	N/A	Stack variable name	a,b,c,d,e,f,g,h supports

## REMARKS

- See the JS command for a full explanation of passing stack variables.
- Passing parameters has no type checking so it is important to exercise good programming style when passing parameters. See examples below for recommended syntax.
- Do not use spaces in expressions containing ^.
- Global variables MUST be assigned prior to any use in subroutines where variables are passed by reference.
- Arrays passed on the stack must have names no longer than 6 chars.
- Stack zero has no local-scope variables. Accessing these variables from stack zero writes to stack 1's variable table.

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
#add
JS #sum{1,2,3,4,5,6,7,8} ;' call subroutine, pass values
MG _JS ;' print return value
EN
'
#sum ;NO(^a,^b,^c,^d,^e,^f,^g,^h) Sums the values ^a to ^h and returns the result
EN ,,(^a+^b+^c+^d+^e+^f+^g+^h) ;' return sum

'Output from the previous program
:XQ #add
36.0000
```

**^** is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>^L^K</b>	<i>Lock program</i>	SYSTEM CONFIG
<b>LK</b> locks user access to the application program		
		

**^L^K n ...** Arguments specified with an implicit comma-separated order

## DESCRIPTION

<control>L<control>K locks user access to the application program. When locked, the ED, UL, LS, and TR commands will give privilege error #106. The application program will still run when locked. Once the program is unlocked, it will remain accessible until a lock command or a reset (with the locked condition burned in) occurs.

## ARGUMENTS

<control>L<control>K str,n

Argument	Min	Max	Default	Resolution	Description	Notes
str	0 char	8 chars	""	String	Controller password string	Password assigned with the PW command.
n	0	1	0	1	Set lock/unlock state for controller	n = 1 locks the application program. n = 0 unlocks the application program.

## REMARKS

- The PW command can only be set while the application program is unlocked.

## EXAMPLES

```
:PW test,test;          Set password to "test"
:^L^K test,1;          Lock the program
:LS;                  Attempt to list the program
?
:TC 1
106 Privilege violation
:
```

^L^K is supported on DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>^R^S</b>	Master Reset	SYSTEM CONFIG
<b>The Master Reset command resets the controller to factory default settings and erases EEPROM</b>		
		

**^R^S** Command takes no arguments

## DESCRIPTION

The Master Reset command resets the controller to factory default settings and erases EEPROM. A master reset can also be performed by installing a jumper at the location labeled MRST and resetting the board (power cycle or pressing the reset button). Remove the jumper after this procedure.

## ARGUMENTS

<control>R <control>S

^R^S has no parameters

## REMARKS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- Sending a ^R^S over an Ethernet connection will cause the IP address to be cleared from the controller and will result in a timeout.

## EXAMPLES

Example burns-in a non-default value for KP, does a standard reset with the RS command, then performs a master reset with ^R^S.

```
:KP ?
6.00
:KP 10
:BN
:RS

:KP ?
10.00
:^R^S

:KP ?
6.00
:
```

^R^S is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>^R^V</b>	Revision Information	INTERROGATION
-------------	----------------------	---------------

The Revision Information command causes the controller to return the firmware revision information



**^R^V** Command takes no arguments

## DESCRIPTION

The Revision Information command causes the controller to return the firmware revision information.

## ARGUMENTS

<control>R <control>V

^R^V has no arguments

## REMARKS

- Do not use ^ symbols to send ^R^V command. ^ symbols denote using the control (Ctrl) key when pressing the characters.

## EXAMPLES

DMC41x3

```
:^R^V
DMC4143 Rev 1.1a1
:
```

^R^V is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilinc.com](mailto:documentation@galilinc.com)

<b>_GP</b>	Gearing Phase Differential Operand	ECAM/GEARING, OPERAND ONLY
The <b>_GP</b> operand contains the value of the "phase differential" accumulated on the most current change in the gearing ratio between the master and the slave axes		

variable= <b>_GP</b>	Holds a value
<b>_GPm</b>	Operand has special meaning see Remarks

## DESCRIPTION

The **\_GP** operand contains the value of the "phase differential" accumulated on the most current change in the gearing ratio between the master and the slave axes. The value does not update if the distance over which the slave will engage is set to 0 with the GD command.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### **\_GPm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis of interest	

## REMARKS

- An operand is not valid individually. Instead, **\_GP** would be used in an expression. See example below.
- Phase Differential is a term that is used to describe the lead or lag between the master axis and the slave axis due to gradual gear shift
  - $Pd = GR * Cm - Cs$  where
    - Pd is the phase differential
    - GR is the gear ratio
    - Cm is the number of encoder counts the master axis moved
    - Cs is the number of encoder counts the slave moved.

## EXAMPLES

```
GA da; Sets the A axis auxs encoder as the gearing master for the A axis.
GD 1000; Set the distance that the master will travel to 1000
' counts before the gearing is fully engaged for the A
' axis slave.
AI -1; Wait for input 1 to go low. In this example, this
' input is representing a sensor that senses an object
' on a conveyor. This will trigger the controller to
' begin gearing and synchronize the master and slave
' axes together.
GR 1; Engage gearing between the master and slave
p1=_TDA; Sets the current A axis position to variable P1. This
' variable is used in the next command
#wait
' Wait for the aux encoder to move forward 1000
' encoder counts so the gearing engagement period is
' complete. Then the phase difference can be adjusted
' for. Note this example assumes forward motion.
JP #wait, (_TDA < (p1+1000))
IP _GPA; Increment the difference to bring the master/slave in
' position sync from the point that the GR1 command was
' issued.
EN; End Program
```

**\_GP** is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>_LF</b>	<i>Forward Limit Switch Operand</i>	<b>IO, OPERAND ONLY</b>
<b>The _LF operand contains the state of the forward limit</b>		

variable= _LF	Holds a value
_LFn	Operand has special meaning, see Remarks

## DESCRIPTION

The \_LF operand contains the state of the forward limit.

## ARGUMENTS

### **\_LFm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis of forward limit switch	

## REMARKS

- \_LF is an operand only with the following output:
  - \_LFm = 1 when the limit switch state will allow motion in the positive direction.
  - \_LFm = 0 when the limit switch state will not allow motion in the positive direction.
- This operand is not a direct readout of the digital input and is affected by the command CN.

*Values of LF*

Digital Input activation	_LF value for CN-1	_LF value for CN1
On. Grounded for TTL, or sufficient activation current flowing for optos.	0 (forward motion prohibited)	1 (forward motion allowed)
Off. Pullup for TTL, or insufficient activation current flowing for optos.	1 (forward motion allowed)	0 (forward motion prohibited)

## EXAMPLES

MG \_LFA display the status of the a axis forward limit switch  
 \*See Connecting Hardware in User Manual for active/inactive state

\_LF is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>_LR</b>	<i>Reverse Limit Switch Operand</i>	<b>IO, OPERAND ONLY</b>
------------	-------------------------------------	-------------------------

The **\_LR** operand contains the state of the reverse limit

variable= <b>_LR</b>	Holds a value
<b>_LRn</b>	Operand has special meaning see Remarks

## DESCRIPTION

The **\_LR** operand contains the state of the reverse limit.

## ARGUMENTS

### **\_LRm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis of reverse limit switch	

N/A

## REMARKS

- **\_LR** is an operand with the following output
  - **\_LRm= 1** when the limit switch state will allow motion in the reverse direction.
  - **\_LRm= 0** when the limit switch state will not allow motion in the reverse direction.
- This operand is not a direct readout of the digital input and is affected by the command CN.

*Values of LR*

Digital input activation	<b>_LR</b> value for CN-1	<b>_LR</b> value for CN1
On. Grounded for TTL, or sufficient activation current flowing for optos.	0 (reverse motion prohibited)	1 (reverse motion allowed)
Off. Pullup for TTL, or insufficient activation current flowing for optos.	1 (reverse motion allowed)	0 (reverse motion prohibited)

## EXAMPLES

MG **\_LR** display the status of the a axis reverse limit switch  
 \*See Connecting Hardware in User Manual for active/inactive state

**\_LR** is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<i>Bitwise OR Operator</i>	Misc
The   symbol is the bitwise OR operator used with IF, JP, and JS decisions, and also to perform bitwise ORING of values	

variable = (value1 | value2) Performs an operation between two values or evaluated statements

## DESCRIPTION

The | symbol is the bitwise OR operator used with IF, JP, and JS decisions, and also to perform bitwise ORING of values.

## ARGUMENTS

n | n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use with OR operator	

## REMARKS

- For IF, JP, and JS, the values used for m are typically the results of logical expressions such as (x>2) | (y=8)
- The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.

## EXAMPLES

```
'Bitwise use
var1= $F; '00001111
var2= $F0; '1111000
MG (var1 | var2)
EN

REM Returned: 255.0000 (same as 11111111)
```

```
'Conditional Use
var1= $F; '00001111
var2= $F0; '1111000
IF (var1 = $F) | (var2 = $F1)
  MG "True"
ELSE
  MG "False"
ENDIF
EN

REM Returned: True
```

| is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

		Variable Axis Designator		Misc

## DESCRIPTION

Variable axis designator. Each variable can be assigned an individual axis, a vector plane, or a virtual axis. Motion commands on the variable will then apply to the assigned axis.

Commands supporting variable axes are denoted in this command reference with the following icon.



## ARGUMENTS

**~s= "str"**

**~s= n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>s</b>	a	h	N/A	N/A	Variable axis name	a,b,c,d,e,f,g,h supported
<b>str</b>	"A"	"H"	N/A	String	Name of axis	"A", "B", "C", "D", "E", "F", "G", "H" supported
	"M"	"N"	N/A	String	Virtual axis	"M", "N" supported
	"S"	"T"	N/A	String	Coordinate System	"S", "T" supported
<b>n</b>	0	7	N/A	1	Index of the axis	A=0, B=1, C=2, etc.
	8	9	N/A	1	Coordinate System	S=8, T=9
	10	11	N/A	1	Virtual Axis	M=11, N=10

## REMARKS

- ~s contains the axis number as defined by n and can be used in expressions (see example)

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```

~a= 2;~b= 6;      Sets ~a to 2(Z axis). Sets ~b to 6 (G axis)
MG "~a=",~a;"   Print axis number
MG "~b=",~b;"   Print axis number
PR~a= 1000;"     Relative position move 1000 counts on ~a variable (set as Z axis)
JG~b= 9000;"     Set jog speed of ~b variable (set as G axis) to 9000 cts/sec
BG ~a~b;"        Begin motion on ~a and ~b variables (Z and G)

```

~ is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

	+ <i>Addition Operator</i>	MATH FUNCTIONS
<b>The + symbol is the addition operator</b>		

variable = (value1 + value2) Performs an operation between two values or evaluated statements

## DESCRIPTION

The + symbol is the addition operator. It takes as arguments any two values, variables, array elements, operands, or At functions (@SIN[]) and returns a value equal to the sum of the arguments.

## ARGUMENTS

### n + n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	N/A	1/65,536	Value to use in addition operation	

## REMARKS

- This is a binary operator (takes two arguments and returns one value). The result of this operation is a value, which is not valid on its own. It must be coupled with a command. See examples below.
- Mathematical operations are calculated left to right rather than multiplication and division calculations performed prior to addition and subtraction.
  - Example: 1+2\*3 = 9; not 7
- It is recommended that parenthesis be used when more than one mathematical operation is combined in one command.
  - Example: var =((10\*30)+(60/30)); evaluates as 302
  - var = 10\*30+60/30; evaluates as 12

## EXAMPLES

```
:var1 = 1+2
:var2 = var1 + 1
:MG var2 + 2
6.0000
:
```

+ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

"Less than" comparator for testing if one value is less than another



variable = (value1 < value2) | Performs an operation between two values or evaluated statements

## DESCRIPTION

"Less than" comparator for testing if one value is less than another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

*Comparators in DMC Code*

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to

## ARGUMENTS

### n0 < n1

Argument	Min	Max	Default	Resolution	Description	Notes
n0	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
n1	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

## REMARKS

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0 < n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

## EXAMPLES

```
:bool= (1<2)
:MG bool
1.0000
:bool= (1<0)
:MG bool
0.0000
:
```

```
REM Example to find the largest
REM value in an array
REM ****
REM Create an array and fill it
len= 5
DM array[len]
array[0]= 5
array[1]= 100.0001
array[2]= 42
array[3]= 3.14
array[4]= 100
JS #max;' call max subroutine
MG "Max value is ", max
EN
REM ****
REM Find max element in array
#max
i= 0
max = -2147483648; ' start at min
#max_h
IF (array[i] > max)
  max = array[i]
ENDIF
i= i+1
JP #max_h, (i < len)
EN
REM ****
REM Program output
REM :XQ
REM :
REM Max value is 100.0001
```

< is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>&lt;=</b>	Less than or Equal to comparator	PROGRAMMING, MATH FUNCTIONS
--------------	----------------------------------	-----------------------------

"Less than or Equal to" comparator for testing if one value is less than or equal to another



variable = (value1 <= value2) | Performs an operation between two values or evaluated statements

## DESCRIPTION

"Less than or Equal to" comparator for testing if one value is less than or equal to another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

Comparators in DMC Code

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to

## ARGUMENTS

**n0 <= n1**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
<b>n1</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

## REMARKS

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0 <= n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

## EXAMPLES

```
:bool= (1 <= 2)
:MG bool
1.0000
:bool= (2 <= 2)
:MG bool
1.0000
:bool= (3 <= 2)
:MG bool
0.0000
:

max= 2.05
min= 1.47
value = 0.025
JS #check
value = 1.471
JS #check
EN
REM
REM ****
REM Determine if in range
#check
inrange= 0
IF ((value >= min) & (value <= max))
  inrange= 1
ENDIF
IF (inrange)
  MG "Value ",value," in range"
ELSE
  MG "Value ",value," NOT in range"
ENDIF
EN
REM
REM ****
REM Program output
REM :XQ
REM :
REM Value 0.0250 NOT in range
REM Value 1.4710 in range
```

<= is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

	<i>Not Equal to comparator</i>	PROGRAMMING, MATH FUNCTIONS
<b>"Not Equal to" comparator for testing if one value is not equal to another</b>		
		

variable = (value1  $\diamond$  value2) Performs an operation between two values or evaluated statements

## DESCRIPTION

"Not Equal to" comparator for testing if one value is not equal to another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

*Comparators in DMC Code*

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
$\leq$	Less than or equal to
$\geq$	Greater than or equal to
$\diamond$	Not equal to

## ARGUMENTS

### n0 $\diamond$ n1

Argument	Min	Max	Default	Resolution	Description	Notes
n0	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
n1	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

## REMARKS

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0  $\diamond$  n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

## EXAMPLES

```
:bool= (1 <> 2)
:MG bool
1.0000
:bool= (2 <> 2)
:MG bool
0.0000
```

```
REM Lock out code until
REM a particular digital
REM input pattern is detected
#AUTO
JS #lock; 'block until pattern
REM
REM
REM Rest of code here
REM
REM
EN
REM
REM
REM ****
#lock
JP #lock, (_TI0 <> 170)
EN
```

$\diamond$  is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

=	Assignment Operator	Misc
---	---------------------	------

The = operator is the assignment operator for the controller



## DESCRIPTION

The = operator is the assignment operator for the controller. The assignment operator is used for three reasons:

- (1) to define and initialize a variable ( $x = 0$ ) before it is used
- (2) to assign a new value to a variable ( $x = 5$ )
- (3) to print a variable or array element ( $x=$  which is equivalent to MG  $x$ ). MG is the preferred method of printing

## ARGUMENTS

### str = n

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	8 chars	N/A	String	Variable name to access	
n	-2,147,483,648	2,147,483,647	see Notes	1/65,536	Value to assign to specified variable	Default n, or n = null results in a query of the value of variable

## REMARKS

- None

## EXAMPLES

```
:x= 5
:x=
5.0000
:MG x
5.0000
'define and initialize x to 5
'print x two different ways
```

= is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

=	<i>Equal to comparator</i>	PROGRAMMING, MATH FUNCTIONS
<b>"Equal to"</b> comparator for testing if one value is equal to another		

variable = (value1 = value2) Performs an operation between two values or evaluated statements

## DESCRIPTION

"Equal to" comparator for testing if one value is equal to another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

*Comparators in DMC Code*

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
<=	Less than or equal to
>=	Greater than or equal to
◊	Not equal to

## ARGUMENTS

### n0 = n1

Argument	Min	Max	Default	Resolution	Description	Notes
n0	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
n1	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

## REMARKS

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0 = n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

## EXAMPLES

```
:bool= (1=0)
:MG bool
0.0000
:bool= (3.14=3.14)
:MG bool
1.0000
:
```

```
REM Checks for a digital
REM input pattern and
REM sets a bit if matched
#loop
IF (_TI0 = 170)
SB 1
ELSE
CB 1
ENDIF
JP #loop
```

= is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

'Greater than' comparator for testing if one value is greater than another



variable = (value1 &gt; value2) | Performs an operation between two values or evaluated statements

**DESCRIPTION**

"Greater than" comparator for testing if one value is greater than another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

Comparators in DMC Code

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to

**ARGUMENTS****n0 > n1**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
<b>n1</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

**REMARKS**

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0 > n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

**EXAMPLES**

```
:bool= (1>2)
:MG bool
0.0000
:bool= (1>0)
:MG bool
1.0000
:
```

```
REM Example to find the largest
REM value in an array
REM
REM ****
REM Create an array and fill it
len= 5
DM array[len]
array[0]= 5
array[1]= 100.0001
array[2]= 42
array[3]= 3.14
array[4]= 100
JS #max;' call max subroutine
MG "Max value is ", max
EN
REM
REM ****
REM Find max element in array
#max
i= 0
max = -2147483648;' start at min
#max_h
IF (array[i] > max)
  max = array[i]
ENDIF
i= i+1
JP #max_h, (i < len)
EN
REM
REM ****
REM Program output
REM :XQ
REM :
REM Max value is 100.0001
```

> is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

"Greater than or Equal to" comparator for testing if one value is greater than or equal to another



variable = (value1 >= value2) | Performs an operation between two values or evaluated statements

## DESCRIPTION

"Greater than or Equal to" comparator for testing if one value is greater than or equal to another. Comparators are used in mathematical expressions, IFs, and in conditional jumps. The result is a boolean.

*Comparators in DMC Code*

Symbol	Comparator
<	Less than
>	Greater than
=	Equal to
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to

## ARGUMENTS

**n0 >= n1**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	
<b>n1</b>	-2147483648.0000	2147483647.0000	N/A	1/65536	Value to test	

## REMARKS

- A comparator is not a command and is not valid individually. Instead, the above expression would be used as part of a jump (JP,JS), IF expression, or assignment. See examples below.
- If n0 >= n1, the expression will evaluate to 1.0000. If the comparison is false, it will evaluate to 0.0000.
- Evaluation occurs left to right. Use parenthesis for operator precedence.

## EXAMPLES

```
:bool= (1 >= 2)
:MG bool
0.0000
:bool= (2 >= 2)
:MG bool
1.0000
:bool= (3 >= 2)
:MG bool
1.0000
:
```

```
max= 2.05
min= 1.47
value = 0.025
JS #check
value = 1.471
JS #check
EN
REM
REM ****
REM Determine if in range
#check
inrange= 0
IF ((value >= min) & (value <= max))
    inrange= 1
ENDIF
IF (inrange)
    MG "Value ",value," in range"
ELSE
    MG "Value ",value," NOT in range"
ENDIF
EN
REM
REM ****
REM Program output
REM :XQ
REM :
REM Value 0.0250 NOT in range
REM Value 1.4710 in range
```

>= is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AB</b>	<i>Abort</i>	PROGRAMMING
-----------	--------------	-------------

The AB command is a command to issue an abort to controller operation

AB n ...	Arguments specified with an implicit comma-separated order
_ABm	Operand has special meaning see Remarks

## DESCRIPTION

The AB command is a command to issue an abort to controller operation.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

AB (Abort) stops motion instantly without a controlled deceleration. If there is a program operating AB can also be specified to abort the program and all running threads. The command, AB, will shut off the motors for any axis in which the off on error function is enabled (see command "OE").

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

Argument	Value	Description	Notes
n	0	Abort motion and the program operation	Default if omitted
	1	Abort motion only	

## REMARKS

- \_AB gives state of Abort Input, 1 inactive and 0 active.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- AB aborts motion on all axes in motion and cannot stop individual axes.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:AB; ' Stops motion
:OE*= 1; ' Enable off on error on axes
:AB; ' Shuts off motor command and stops motion
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a; ' Label - Start of program
JG 20000; ' Specify jog speed on A-axis
BG A; ' Begin jog on A-axis
WT 5000; ' Wait 5000 msec
AB 1; ' Stop motion without aborting program
WT 5000; ' Wait 5000 milliseconds
SH ; ' Servo Here
JP #a; ' Jump to Label A
EN; ' End of the routine
'Remember to use the parameter 1 following AB if you only want the motion to be aborted
'Otherwise, your application program will also be aborted.
```

AB is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AC</b>	<i>Acceleration</i>	INDEPENDENT MOTION
Sets the linear acceleration rate of the motors for independent moves, such as PR, PA and JG moves		

ACA= n	Arguments specified with an axis mask and an assignment (=)
AC n ...	Arguments specified with an implicit comma-separated order
ACm	Operand holds the value set in the command

## DESCRIPTION

Sets the linear acceleration rate of the motors for independent moves, such as PR, PA and JG moves. The acceleration rate may be changed during motion.

## ARGUMENTS

**ACm= n**

**AC n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Virtual axis to assign value	
n	1,024	1,073,740,800	256,000	1,024	Value of proportional term	Resolution and Min depends on TM, see remarks

## REMARKS

- The DC command is used to designate deceleration
- Specify realistic acceleration rates based on your physical system such as
  - motor torque rating
  - loads
  - amplifier current rating
- Specifying an excessive acceleration will cause large following error during acceleration and the motor will not follow the commanded profile.
- The acceleration feedforward command (FA) will help minimize the error for aggressive accelerations

DMC40x0, DMC41x3, DMC18x6, DMC30010

## Resolution

DMC40x0, DMC41x3, DMC18x6, DMC30010

- The Min and Resolution depends upon the update rate setting (TM). The equation to calculate these values is:
  - Resolution = Min =  $1024 * (1000/TM)^2$
  - example:
    - With TM 500 the minimum AC setting and resolution is 4096 cnts/second<sup>2</sup>
    - resolution =  $1024 * (1000/500)^2 = 4096$

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
REM Set A-axis acceleration to 150000, B-axis to 200000 counts/sec2, the C axis to 300000 counts/sec2, and the D-axis to 400000 count/sec2.
AC 150000,200000,300000,400000
a= ACB;! Assigns the B acceleration to the variable a
```

AC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AD</b>	After Distance	TRIPPOINT
Trippoint to block command execution until a given distance is traversed		

ADA=n	Arguments specified with an axis mask and an assignment (=)
AD n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

Trippoint to block command execution until a given distance is traversed. This is a profiled trippoint which means it depends on the motion profiler and not the actual motor encoder. AD can only be used when there is commanded motion on the axis.

## ARGUMENTS

**ADm= n**

**AD n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	2,147,483,647	N/A	1	Distance of motion	Cannot specify more than 1 argument at a time

## REMARKS

- AD will hold up the execution of the following command until one of the following conditions have been met
  - The commanded motor position crosses the specified relative distance from the start of the move
  - The motion profiling on the axis is complete
  - If in jog (JG) mode, the commanded motion is in the direction which moves away from the specified position
- Not valid for a slave during ECAM or Gearing, use MF and MR
- If the direction of motion is reversed when in PT mode, the starting position for AD is reinitialized to the position at which the motor is reversed
- The AD command is accurate to the number of counts that occur in 2\*TM msec
- AD command will be affected when the motion smoothing time constant, IT, is not 1. See IT command for further information
- AD measures incremental distance from start of move on one axis

## EXAMPLES

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
#a
DP 0,0;          Zero position
PR 10000,20000; Specify position relative moves
BG ;'           Begin motion
AD 5000;        After A reaches 5000
MG "Halfway to A";TP A; Send message
AD ,10000;      After B reaches 10000
MG "Halfway to B";TP B; Send message
EN;            End Program
```

AD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AF</b>	Analog Feedback Select	FILTER/CONTROL
The AF command configures analog feedback mode for the PID filter		

AFA= n	Arguments specified with an axis mask and an assignment (=)
AF n ...	Arguments specified with an implicit comma-separated order
AFm	Operand holds the value set in the command

## DESCRIPTION

The AF command configures analog feedback mode for the PID filter.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

The controller ADC can be used as position feedback for the axis control law. The analog input used for feedback is fixed and uses the input that corresponds with the axis letter. For example, Analog input 1 is used for the A axis.

## ARGUMENTS

**AFm= n**

**AF n,n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	Use the controller ADC as servo feedback	1=analog 0=digital feedback
	-1	-1	0	0	Analog hardware sampled in the servo interrupt	This provides evenly sampled analog data for both the data record and the RA/RD/RC function.

## REMARKS

- Below is the feedback in counts decoded by the controller hardware when reading in analog feedback for certain analog input ranges.

	12 Bit ADC	16 Bit ADC
+/-5 V, +/-10 V	-2048 to 2047 counts	-32768 to 32767 counts
0-5 V, 0-10 V	0 to 4095 counts	0 to 65535 counts

DMC40x0, DMC41x3, DMC21x3

- The analog voltage range is set using the AQ command. AQ must be set prior to setting AF
- The analog feedback is decoded by a 12-bit A/D converter. An upgrade option is available for 16-bits.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- When using Sin/Cos encoders (AF5-12), the encoder must be connected to the controller prior to issuing the AF command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
AF 1;          Analog feedback on A axis
v1=_AFa;      Assign feedback type to variable
KP 1;          Assigns PID's for motor using analog feedback on A-axis
KD 10;         %
KI 0.5;        %
```

AF is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AG</b>	<i>Amplifier Gain</i>	Filter/Control
The AG command sets the amplifier current/voltage gain for the internal amplifier		

AGA=n	Arguments specified with an axis mask and an assignment (=)
AG n ...	Arguments specified with an implicit comma-separated order
_AGm	Operand holds the value set in the command

## DESCRIPTION

The AG command sets the amplifier current/voltage gain for the internal amplifier.

## ARGUMENTS

**AGm= n**

**AG n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	3	1	1	Gain setting	See table for gain settings

## Servo Amplifiers

DMC40x0, DMC41x3

*Gain settings by Amplifier (Amps/Volt)*

Gain Setting, n=	0	1	2	3
AMP-43040	0.4	0.7	1	N/A
AMP-43240	0.5	1	2	N/A
AMP-43540	0.4	0.8	1.6	N/A

## Stepper Amplifiers

DMC40x0, DMC41x3

*Gain settings by Amplifier (Amps per phase)*

Gain Setting, n=	0	1	2	3
SDM-44140	0.5	1	2	3
SDM-44040	0.5	0.75	1	1.4

## REMARKS

DMC40x0, DMC41x3, DMC21x3

- The MT command must be issued prior to the AG command to set the proper range
- The axis must be in the motor off state (MO) before setting AG

## EXAMPLES

```
ST ;' Stop any motion
AM ;' Wait for motion to decel and stop
MO ;' Turn motor off
MT 1; Set the A axis as a servo
AG 2; Sets the highest amplifier gain for A axis on servo amplifier
BN; Save AG setting to EEPROM
```

AG is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AI</b>	After Input	TRIPPOINT
The AI command is a trippoint used in motion programs to wait until after a specified input has changed state		

AI n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The AI command is a trippoint used in motion programs to wait until after a specified input has changed state. This command can be configured such that the controller will wait until the input goes high or the input goes low.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### AI n

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	16	N/A	1	General input to use for trippoint	+n = High trigger. -n = low trigger. 9-16 only valid for 5-8 axis controller
	81	96	N/A	1	Aux encoder input to use for trippoint	

## REMARKS

- Hint: The AI command actually halts execution until specified input is at desired logic level. Use the conditional Jump command (JP) or input interrupt (II) if you do not want the program sequence to halt.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a;' Begin Program
AI 8;' Wait until input 8 is high
SP 10000;' Speed is 10000 counts/sec
AC 20000;' Acceleration is 20000 counts/sec2
PR 400;' Specify position
BG A;' Begin motion
EN;' End Program
```

AI is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AL</b>	Arm Latch	<b>IO</b>
The AL command enables the latch function (high speed main or auxiliary position capture) of the controller		

<b>AL A</b>	Argument is an axis mask
<b>_ALm</b>	Operand has special meaning, see Remarks

## DESCRIPTION

The AL command enables the latch function (high speed main or auxiliary position capture) of the controller. When the position latch is armed, the main or auxiliary encoder position will be captured upon a low going signal from the specified digital input.

## ARGUMENTS

### AL mm

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGH	N/A	Multi-Axis Mask	Encoder to latch	Latch main encoder
<b>mm</b>	SA	SASBSCSDSESFSGSH	N/A	Multi-Axis Mask	Encoder to latch	Latch aux encoder
<b>mm</b>	TA	TATBTCTDTETFTGTH	N/A	Multi-Axis Mask	Input to trigger latch	Main encoder is latched from the index pulse instead of a digital input

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

*Latch input by Axis*

Axis	Latch Input
A	Input 1
B	Input 2
C	Input 3
D	Input 4
E	Input 9
F	Input 10
G	Input 11
H	Input 12

- The command RL returns the latched position
- \_ALm** contains the state of the specified latch. 0 = not armed, 1 = armed
- The CN command can be used to change the polarity of the latch function
- The latch function is available on incremental quadrature encoder inputs only. For other position capture methods contact Galil.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#start
AL A; '           Arm A-axis latch
JG 50000; '       Set up jog at 50000 counts/sec
BG A; '           Begin the move
#loop; '          Loop until latch has occurred
JP #loop, (_ALA=1)
RL A; '           Transmit the latched position
EN; '             End of program
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
REM Homing routine using the AL command to detect the Motor's index position
#start
AL TA; '           Arm A-axis latch. Latch will trigger off the index pulse
JG 50000; '         Set up jog at 50000 counts/sec
BG A; '             Begin the move
#loop; '            Loop until latch has occurred
JP #loop, (_ALA=1)
ST A; '             Stop the jog
AM A
PAA=_RIA; '         Set up a move to return to the latched position
BG A
AM A
WT 100; '           Allow for settling.
REM Checking that KI has eliminated error (TE) would be more thorough
DP 0; '             Zero position
MG "A Homed"; '     Report status
EN; '               End of program
```

DMC40x0, DMC41x3, DMC18x6

```
REM manual find index using latch off of index pulse using variable axes
#index
'settings
~a= 0; 'axis number
```

```

s1spd= 10000;'stage 1 speed
s2spd= 1000;'stage 2 speed
'jog until index is latched
JG~a= s1spd;BG ~a
WT 1000
AL T~a
#a;JP #a,_AL~a=1
ST ~a;AM ~a
'Return to latched position
SP~a= s2spd
PA _RL~a;BG ~a;MC ~a
JS #stl;DP~a= 0
EN

'wait for axis to settle
#stl
WT 2;JP #stl,@ABS[_TE~a]>2
WT 2;JP #stl,@ABS[_TE~a]>0
WT 2;JP #stl,@ABS[_TE~a]>0
EN

```

AL is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AM</b>	After Move	TRIPPOINT
<b>The AM command is a trippoint used to control the timing of events</b>		

AM A Argument is an axis mask

## DESCRIPTION

The AM command is a trippoint used to control the timing of events. This command will hold up execution of the following commands until the current move on the specified axis or axes is completed. Any combination of axes or a motion sequence may be specified with the AM command.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

For example, AM AB waits for motion on both the A and B axis to be complete. AM with no parameter specifies that motion on all axes to be complete.

## ARGUMENTS

**AM mm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to wait for profiled motion to complete	
	S	T	N/A	Multi-Axis Mask	Vector plane to wait for profiled motion to complete	

## REMARKS

- AM is a very important command for controlling the timing between multiple move sequences.
  - For example, if the A-axis is in the middle of a position relative move (PR) you cannot make a position absolute move (PAA, BGA) until the first move is complete. Use AMA to halt the program sequences until the first profiled motion is complete.
  - AM tests for profile completion only. The actual motor may still be moving. To halt program sequence until the actual physical motion has completed, use the MC command.
  - Another method for testing motion complete is to check for the internal variable \_BGn, being equal to zero (see BG command).

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#move;           'Program MOVE
PR 5000,5000,5000,5000; 'Position relative moves
BG A;           'Start the A-axis
AM A;           'After the move is complete on A,
BG B;           'Start the B-axis
AM B;           'After the move is complete on B,
BG C;           'Start the C-axis
AM C;           'After the move is complete on C
BG D;           'Start the D-axis
AM D;           'After the move is complete on D
EN;             'End of Program
```

AM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AO</b>	Analog Output	ETHERNET, IO
The AO command sets the analog outputs on the Galil or for a Modbus Slave		
		

AO n ...	Arguments specified with an implicit comma-separated order
_AO1, _AO2	Operand holds the value set in the command

## DESCRIPTION

The AO command sets the analog outputs on the Galil or for a Modbus Slave.

## ARGUMENTS

### AO n0,n1

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
n0	1,000	8,999	N/A	1	Set Analog Output on Modbus Slave	See "Using AO with a Modbus Slave" in Remarks
n1	-9.9998	9.9998	N/A	20/65,536	Analog Output Voltage	

## REMARKS

### Using AO with a Modbus Slave

- RIO as Modbus Slave
- 3rd Party Modbus Slave Device
- n0 is the I/O number calculated using the following equations:
- n0 = (HandleNum\*1000) + ((Module-1)\*4) + (Bitnum-1)
  - HandleNum is the handle specifier from A to H.
  - Handle must be assigned to port 502 for Modbus comms (See IH)
  - Module is the position of the module in the rack from 1 to 16.
  - BitNum is the I/O point in the module from 1 to 4

## EXAMPLES

AO 3005,3.2;	Outputs 3.2 Volts on Channel 5 of the Device connected to Handle C
--------------	--

AO is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AP</b>	After Absolute Position	TRIPPOINT
-----------	-------------------------	-----------

The AP command will hold up the execution of the following command until the actual motor position crosses the specified position



APA= n	Arguments specified with an axis mask and an assignment (=)
AP n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The AP command will hold up the execution of the following command until the actual motor position crosses the specified position. This trippoint does not rely on the profiler, but on actual encoder position.

## ARGUMENTS

**APm= n**

**AP n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	N/A	1	Position trippoint value	Only one axis may be specified at a time.

## REMARKS

- For AP command to clear, one of the following conditions have been met:
  - The actual motor position crosses the specified absolute position.
  - The motion profiling on the axis is complete.
  - The commanded motion is in the direction which moves away from the specified position.
- The units of the command are quadrature counts.
- When using a stepper motor, the AP trippoint condition is satisfied when the stepper position (TD) has crossed the specified position.
  - For further information see Chapter 6 of the User Manual "Stepper Motor Operation".
- Not valid for a slave during ECAM or Gearing - use MF and MR.
- The motion profiler must be active before the AP command is used.
- AP is accurate to the number of counts that occur in 2\*TM msec
- AP tests for absolute position. Use the AD command to measure incremental distances.

## EXAMPLES

```
#test; ' Program B
DP 0; ' Define zero
JG 1000; ' Jog mode (speed of 1000 counts/sec)
BG A; ' Begin move
AP 2000; ' After passing the position 2000
v1=_TPA; ' Assign V1 A position
MG "Position is", v1=' Print Message
ST ;' Stop
EN; ' End of Program
```

AP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AQ</b>	Analog Input Configuration	<b>IO</b>
The AQ command is used to set the behavior of the analog inputs		

AQ n ...	Arguments specified with an implicit comma-separated order
_AQ0, _AQ1, _AQ2, _AQ3, _AQ4, _AQ5, _AQ6, _AQ7, _AQ8	Operand has special meaning see Remarks

## DESCRIPTION

The AQ command is used to set the behavior of the analog inputs. This command will set the analog range and operation for the specified input.

## ARGUMENTS

### AQ n0,n1

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	1	8	N/A	1	Analog input channel	
<b>n1</b>	1	4	2	1	Analog range setting	See Table Below
	-4	-1	N/A	1	Specify analog input is differential	See Remarks

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

Argument	Value	Description	Notes
<b>n1</b>	1	+/- 5v	
	2	+/- 10v	Default
	3	0-5v	
	4	0-10v	

## REMARKS

DMC40x0, DMC41x3, DMC21x3

- Default resolution for analog inputs is 12bits. 16 bit is optional.
- Operands \_AQ1 through \_AQ8 return the setting for the specified input
- Setting a negative n1 for inputs 1,3,5 or 7, configures those inputs as the differential input relative to input 2,4,6 and 8 respectively.

DMC40x0, DMC41x3, RIO, DMC21x3

### Differential Input Mapping (-n1)

DMC40x0, DMC41x3, DMC21x3

Input (n0)	Compliment (n0 + 1)
1	2
3	4
5	6
7	8

DMC40x0, DMC41x3, DMC21x3, DMC30010

Position Range when in Analog Feedback by AQ

Argument	Value	Analog Range	Position Range (12 bit)	Position Range (16 bit)
<b>n1</b>	1	+/- 5 V	-2048 to 2047	-32,768 to 32767
	2	+/- 10 V	-2048 to 2047	-32,768 to 32767
	3	0-5 V	0 to 4095	0 to 65535
	4	0-10 V	0 to 4095	0 to 65535

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:AQ 2,3; Specify analog input 2 as 0-5V
:AQ 1,-3; Specify analog input 1 as 0-5V and the differential input to analog input 2
:MG _AQ2
3.0000
```

AQ is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AR</b>	After Relative Distance	TRIPPOINT
The After Relative (AR) command is a trippoint used to control the timing of events		

ARA= n	Arguments specified with an axis mask and an assignment (=)
AR n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The After Relative (AR) command is a trippoint used to control the timing of events. This command will hold up the execution of the following command until one of the following conditions have been met:

1. The commanded motor position crosses the specified relative distance from either the start of the move or the last AR or AD command.
2. The motion profiling on the axis is complete.
3. If in jog (JG) mode, the commanded motion is in the direction which moves away from the specified position.

## ARGUMENTS

**ARm= n**

**AR n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-2,147,483,648	2,147,483,647	N/A	1	Relative position for trippoint	Only one axis may be specified at a time.

## REMARKS

- The units of the command are quadrature counts.
- When using a stepper motor, this condition is satisfied when the stepper position (as determined by the output buffer) has crossed the specified Relative Position.
  - For further information see Chapter 6 of the User Manual "Stepper Motor Operation".
- If the direction of the motion is reversed when in position tracking mode (see PT command), the starting point for the trippoint is reinitialized to the point at which the motion reversed.
- The motion profiler must be active before the AR command is issued.
- Not valid for a slave during ECAM or Gearing - use MF and MR.
- Note: AR will be affected when the motion smoothing time constant, IT, is not 1. See IT command for further information.
  - AP is accurate to the number of counts that occur in  $2^*TM$  msec
- AR is used to specify incremental distance from last AR or AD command.
- Use AR if multiple position trippoints are needed in a single motion sequence.

## EXAMPLES

```
#a; ' Begin Program
DP 0
JG 50000; ' Specify speed
BG A; ' Begin motion
#b; ' Label
AR 25000; ' After passing 25000 counts of relative distance on A-axis
MG "Passed",TP A; ' Send message on A-axis
JP #b; ' Jump to Label #B
EN; ' End Program
```

AR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AS</b>	<i>At Speed</i>	TRIPPOINT
The AS command is a trippoint that occurs when the generated motion profile has reached the specified speed		

ASA | Argument is an axis mask

## DESCRIPTION

The AS command is a trippoint that occurs when the generated motion profile has reached the specified speed. This command will hold up execution of the following command until the commanded speed has been reached. The AS command will operate after either accelerating or decelerating.

## ARGUMENTS

**AS mm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGHST	ABCDEFGH	Multi-Axis Mask	Axes to use for AS trippoint	

## REMARKS

- If the speed is not reached, the trippoint will be triggered after the speed begins diverging from the AS value.
- The AS command applies to a trapezoidal velocity profile only with linear acceleration. AS used with Smoothing profiling will be inaccurate.

## EXAMPLES

```
#speed; ' Program
PR 100000; ' Specify position
SP 10000; ' Specify speed
BG A; ' Begin A
AS A; ' After speed is reached
MG "At Speed"; ' Print Message
EN; ' End programm
```

AS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AT</b>	<i>At Time</i>	TRIPPOINT
The AT command is a trippoint which is used to hold up execution of the next command until after the specified time has elapsed		

AT n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The AT command is a trippoint which is used to hold up execution of the next command until after the specified time has elapsed. The time is measured with respect to a defined reference time. AT 0 establishes the initial reference. AT n specifies n msec from the reference. AT -n specifies n msec from the reference and establishes a new reference after the elapsed time period.

DMC40x0, DMC41x3, DMC30010

AT n,1 specifies n samples from the reference. This is useful when TM is lowered and faster application loop times are required.

## ARGUMENTS

DMC40x0, DMC41x3, DMC30010

**AT n0,n1**

DMC40x0, DMC41x3, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2,147,483,648	2,147,483,647	0	2	Specify a wait time for AT trippoint	See Remarks
<b>n1</b>	0	1	0	1	Specify time in samples or msec	n1=0 for msec. n1=1 for samples

## REMARKS

DMC40x0, DMC41x3, DMC30010

- n0 = 0 sets the reference time for AT to the current time.
- n0 > 0 specifies the wait time as the absolute value of n0 from the reference time
- n0 < 0 specified the wait time as the absolute value of n0 from the reference time, and resets the reference time when the trippoint is complete to the current time.
  - AT -n0 is equivalent to AT n0; AT (old reference +n0)

## EXAMPLES

DMC40x0, DMC41x3, DMC30010

```
'The following commands are sent sequentially
AT 0;' Establishes reference time 0 as current time
AT 50;' Waits 50 msec from reference 0
AT 100;' Waits 100 msec from reference 0
AT -150;' Waits 150 msec from reference 0 and sets new reference at 150
AT 80;' Waits 80 msec from new reference (total elapsed time is 230 msec)
```

DMC40x0, DMC41x3, DMC30010

```
' jog proportional to analog input example with AT in ms
'AT -n
#main0
AT 0;'           set time reference for AT command
JG 0;BG A;       start Jog mode
gain= 1
#atloop
jgspd=gain*@AN[1]
JG jgspd
AT -100;'        wait 100 ms from last time reference (last AT-n or AT0)
REM same functionality would be:
REM AT -100,0
REM -or-
REM AT 100,0;AT0
JP #atloop
```

DMC40x0, DMC41x3, DMC30010

```
' jog proportional to analog input example with AT in samples
' AT n,1
#main1
AT 0;'           set time reference for AT command
JG 0;BG A;       start Jog mode
gain= 1
#atloop
jgspd=gain*@AN[1]
JG jgspd
AT -100,1;'      wait 100 samples from last time reference (AT0)
JP #atloop
```

AT is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AU</b>	<i>Set amplifier current loop</i>	Filter/Control
The AU command sets the amplifier current loop gain for internal amplifiers		
		

AUA=n	Arguments specified with an axis mask and an assignment (=)
AU n ...	Arguments specified with an implicit comma-separated order
_AUm	Operand holds the value set in the command

## DESCRIPTION

The AU command sets the amplifier current loop gain for internal amplifiers.

**DMC40x0, DMC41x3, DMC21x3**

For Galil Trap amplifiers, the current loop is available in one of two settings. AU also sets the switching mode where available, Chopper vs. Inverter.

**DMC40x0, DMC41x3, DMC30010**

For Galil Sine amplifiers, the optimal current loop gain setting is determined by the bus voltage supplied to the amplifier and the phase to phase inductance of the motor. The table in the Arguments section provides ideal AU settings for common bus voltages and phase to phase inductance.

## ARGUMENTS

**AU<sub>m</sub>= n**

**AU n,n,n,n,n,n,n,n**

**DMC40x0, DMC41x3**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	see Notes	see Notes	0	see Notes	Set amplifier current loop gain setting	See table below for setting for your amplifier model

**DMC40x0, DMC41x3**

**AMP-43040/43020/43240 (-D3040/-D3020/-D3240)**

**DMC40x0, DMC41x3, DMC21x3**

Argument	Value	Description	Notes
<b>n</b>	0	Inverter mode, Normal current loop gain	Default
	0.5	Chopper mode, Normal current loop gain	
	1	Inverter mode, Higher current loop gain	
	1.5	Chopper mode, Higher current loop gain	

**DMC40x0, DMC41x3**

**AMP-43540 (-D3540)**

**DMC40x0, DMC41x3, DMC30010**

Argument	Value	(24VDC Bus) Current loop setting	(48VDC Bus) Current loop setting
<b>n</b>	0	Minimum Current Loop Gain	Minimum Current Loop Gain
	1	For inductance < 1mH	For inductance < 2.4mH
	2	For inductance > 1mH and < 2.3mH	For inductance > 2.4mH and < 4.2mH
	3	For inductance > 2.3mH and < 4.2mH	For inductance > 4.2mH and < 7mH
	4	For inductance > 4.2mH	For inductance > 7mH

## REMARKS

**DMC40x0, DMC41x3, DMC30010**

- The AU settings for Galil sine drives are only recommended values for the given bus voltages. For other bus voltages and their recommended settings, contact Galil.

**DMC40x0, DMC41x3, DMC21x3**

### High Current Loop

**DMC40x0, DMC41x3, DMC21x3**

- Use the higher current loop gain (AU 1 or 1.5) when the phase to phase inductance of the motor is > 5mH with a 24VDC supply, or if the inductance is > 10mH with a 48VDC supply.

**DMC40x0, DMC41x3, DMC21x3**

### Chopper Mode

**DMC40x0, DMC41x3, DMC21x3**

- The chopper mode is in contrast to the normal inverter mode in which the amplifier sends PWM power to the motor of +/-Vs.
  - In chopper mode, the amplifier sends a 0 to +VS PWM to the motor when moving in the forward direction, and a 0 to -VS PWM to the motor when moving in the negative direction.
- Chopper mode should be used in 2 different scenarios
  - 1 - The inductance of the motor is 200uH to 500uH
  - 2 - The application requires a continuous operation at >= 4 Amps of continuous torque at a duty cycle of >= 50%.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3

```
'settings for trap amps
:AU 1,0;' Sets X-axis to higher loop gain and Y-axis to normal loop gain
:AUB= ?;' Query Y-axis current loop gain
0
:MG _AUA;' Query A axis current loop gain
1
```

DMC40x0, DMC41x3, DMC30010

```
'setting for sine amps
'BLM inductance = 2.6mH
:AU 3;' Sets A-axis for standard Galil BLM motor at 24V
:MG _AUA;' Query A axis current loop gain
3
:AU 2;' Sets A axis for standard Galil BLM motor at 24V
```

AU is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AV</b>	After Vector Distance	VECTOR/LINEAR, TRIPPOINT
The AV command is used to hold up execution of the next command during coordinated moves such as VP,CR or LI		

AV n ...	Arguments specified with an implicit comma-separated order
_AVS, _AVT	Operand has special meaning see Remarks

## DESCRIPTION

The AV command is used to hold up execution of the next command during coordinated moves such as VP,CR or LI. This trippoint occurs when the path distance of a sequence reaches the specified value. The distance is measured from the start of a coordinated move sequence or from the last AV command.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

**AV n0,n1**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
n0	0	2,147,483,647	0	1	Vector distance to be executed in the S coordinate system	
n1	0	2,147,483,647	0	1	Vector distance to be executed in the T coordinate system	

## REMARKS

- The units of the command are quadrature counts.
- \_AVS contains the vector distance from the start of the sequence in the S coordinate system

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- \_AVT contains the vector distance from the start of the sequence in the T coordinate system.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#move; ' Label
DP 0,0
CA T; ' Specify the T coordinate system
LM AB; ' Linear move for A,B
LI 1000,2000; ' Specify distance
LI 2000,3000; ' Specify distance
LE
BG T; ' Begin motion in the T coordinate system
AV ,500; ' After path distance = 500,
MG "Path>500"
TP AB; ' Print position of A and B axes
EN; ' End Program
'Vector Distance is calculated as the square root of the sum of the
'squared distance for each axis in the linear or vector mode.'
```

AV is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>AW</b>	<i>Amplifier Bandwidth</i>	Filter/Control
The AW command calculates and reports the amplifier bandwidth based on specified motor parameters		

AWA=n Arguments specified with an axis mask and an assignment (=)

## DESCRIPTION

The AW command calculates and reports the amplifier bandwidth based on specified motor parameters. The AW command accepts the drive voltage (volts) and motor inductance (millihenries) and uses the current loop gain setting (AU) as the default and then reports the calculated bandwidth. The user can check how the amplifier bandwidth is affected by changing the n2 parameter.

DMC40x0, DMC41x3

The AU command uses the transfer function for the AMP-430x0 for the calculation of the bandwidth.

## ARGUMENTS

**AWm=n0,n1,n2**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n0</b>	0	80	0	1/65536	Drive voltage in volts	
<b>n1</b>	0	25	0	1/65536	Motor inductance in millihenries	
<b>n2</b>	0	1	0	1	Optional current loop setting 1=High current loop, 0=normal current loop	

where

m = Axis designator A,B,C,D,E,F,G or H

v = Drive voltage in Volts

l = Motor inductance in millihenries

n = optional current loop gain setting(1 or 0)

## REMARKS

DMC40x0, DMC41x3

- The AU command uses the transfer function for the AMP-430x0 for the calculation of the bandwidth.

## EXAMPLES

```
:AWB=60,5,0; Sets a 60 volt drive, motor with 5 millihenries inductance and normal current loop gain
4525.732 Is the bandwidth in hertz
```

AW is supported on DMC40x0, DMC41x3, DMC21x3 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BA</b>	<i>Brushless Axis</i>	SINE COMMUTATION
<b>BA is used to configure the controller for sinusoidal operation</b>		

BA A	Argument is an axis mask
_BAm	Operand has special meaning, see Remarks

## DESCRIPTION

BA is used to configure the controller for sinusoidal operation.

DMC40x0, DMC41x3

### Galil Sine Drive Use

DMC40x0, DMC41x3

For axes equipped with a Galil sine drive, BA is used to configure the axis for sinusoidal operation. In addition to BA, BM and BX or BZ must be used to initialize the drive commutation. When using a Galil sine drive, one axis of control is required for one axis of drive. This is in contrast to the paired behavior below.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

### Third-Party Sine Drives Requiring Dual Analog Inputs (Rare)

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

In rare cases, some third-party sinusoidal drives require two analog signals to perform commutation. In this case, the BA command configures the controller axes for sinusoidal commutation and reconfigures the controller to reflect the actual number of motors that can be controlled. In this configuration, each axis requires 2 motor command signals. The second motor command signals will always be associated with the highest axis on the controller. For example a 3 axis controller with A and C configured for sinusoidal commutation will require 5 command outputs (a 5 axis controller), where the second outputs for A and C will be the D and E axes respectively.

## ARGUMENTS

### BA mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axes to initialize for sine amps	mm = "" removes all axes configured for sine commutation
	N	N	N/A	Multi-Axis Mask	Disable sine initialization for all axes.	

## REMARKS

DMC40x0, DMC41x3

### Galil Sine Drive Use

DMC40x0, DMC41x3, DMC30010

- \_BAm will contain a 1 if the BA command has been issued for the specified axis, or a 0 if it has not.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

### Third-Party Sine Drives Requiring Dual Analog Inputs (Rare)

- \_BAm indicates the axis number of the auxiliary DAC used for the second phase of the selected sinusoidal axis. The axis numbers start with zero for the A axis DAC. If the motor is configured as standard servo or stepper motor, \_BAn contains 0.

## EXAMPLES

```
BA A; ' Configure axis A for sine amp
```

BA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BC</b>	<i>Brushless Calibration</i>	SINE COMMUTATION
The BC command is used to initialize a motor for sine commutation using hall sensors		

BC A	Argument is an axis mask
_BCm	Operand has special meaning, see Remarks

## DESCRIPTION

The BC command is used to initialize a motor for sine commutation using hall sensors.

The function BC monitors the status of the Hall sensors of a sinusoidally commutated motor, and resets the commutation phase upon detecting the first hall sensor. This procedure replaces the estimated commutation phase value with a more precise value determined by the hall sensors.

## ARGUMENTS

### BC mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to initialize with hall commutation	

## REMARKS

DMC40x0, DMC41x3, DMC30010

- The BC command is one of several ways to initialize a Galil sine drive. The table below lists the various methods:

DMC40x0, DMC41x3, DMC30010

*Commutation of a Galil Sine Drive*

Command	Description
BC/BI	Uses hall sensors to commutate until a hall transition is encountered. Drive then commutes sinusoidally.
BX	Uses an algorithm to determine phase angle with minimal motion.
BZ	Drives the motor to a known magnetic phase. Drive then commutes sinusoidally.

### Steps for BC sine initialization

- Specify the axis/axes for initialization with the BA command
- Specify the number of encoder counts per magnetic phase of the motor with the BM command (see command for examples)
- Issue BI to select the inputs to use as hall inputs.
- Ensure the wiring is correct using the BS command.
- Servo the motor and verify it holds position
  - If the motor will not servo, verify encoder is functional. If it is, then re-verify hall wiring
- Issue the BC command, then issue a small jog until a hall transition occurs.
- The motor is now fully commutated based off of the hall sensor feedback.
- (Optional) Use the BB command to correct for hall offsets from true magnetic 0 of the motor.

### Operand Usage

- \_BCm contains the state of the Hall sensor inputs. This value should be between 1 and 6. 0 and 7 are invalid hall states.

## EXAMPLES

```
BA A; ' Enable sine drive
BMA= 2000; ' Set brushless modulus to 2000 cnts
BIA= 4; ' Hall inputs on IN4,5, and 6
MG _BCA; ' Read hall state
EN
```

DMC40x0, DMC41x3, DMC30010

```
REM Example for use with internal sine amp
#ex
BA A
BMA= 2000
BIA= -1; ' use hall sensor inputs on the Galil
BC A; ' enable brushless calibration
bc= _BCA; ' store hall state
JGA= 500
BG A; ' begin jog
#hall;JP #hall,_BCA=bc; 'wait for a hall transition
ST A
MG "Commutation Complete"
EN
```

BC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BD</b>	<i>Brushless Degrees</i>	SINE COMMUTATION
The BD command sets the commutation phase of a sinusoidally commutated motor manually		
		

BDA= n	Arguments specified with an axis mask and an assignment (=)
BD n ...	Arguments specified with an implicit comma-separated order
BDm	Operand has special meaning see Remarks

## DESCRIPTION

The BD command sets the commutation phase of a sinusoidally commutated motor manually. When using hall effect sensors, a more accurate value for this parameter can be set by using the command, BC. This command should not be used except when the user is creating a specialized phase initialization procedure.

## ARGUMENTS

**BDm= n**

**BD n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	360	6	1/32	Brushless motor angle in degrees	

## REMARKS

- Using BD to set a brushless degree overrides the current brushless degrees set by the BZ/BX/BI initialization routines.
- Once initialized, BD is updated by the firmware to the current brushless degree value.
- n = ? queries the current brushless degrees
- \_BDm contains the commutation phase of the specified axis.

## EXAMPLES

```
BDA= 100; ' Set Brushless degrees for A axis to 100
MG BDA; ' Report the brushless degrees for A axis
```

BD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BG</b>	Begin	INDEPENDENT MOTION
The BG command starts a motion on the specified axis or sequence		
		

BG A	Argument is an axis mask
_BGm	Operand has special meaning, see Remarks

## DESCRIPTION

The BG command starts a motion on the specified axis or sequence.

## ARGUMENTS

### BG mm

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to begin motion	Any combination of axes is acceptable. BG with no arguments begins motion on all axes
	S	T	N/A	Multi-Axis Mask	Vector plane axes to begin motion	Any combination of axes is acceptable
	M	N	N/A	Multi-Axis Mask	Virtual axis to begin motion	Any combination of axes is acceptable

## REMARKS

- Any combination of Axes, Vector Planes, and Virtual Axes may be mixed to begin motion
- A BG command cannot be executed for any axis in which motion has not completed
  - Slaving to a master in gearing mode is an exception. Gearing does not require the axis to profile a motion and therefore Independent moves may be superimposed on top of gearing
- Use the AM trippoint to wait for motion complete between moves from embedded code.
- From host code, use one of the following methods to determine motion is complete
  - Poll MG \_BGm
  - Use the data record (DR/QR)
  - Use interrupts (EI), if available

## Operands

- \_BGm contains a '0' if motion complete on the specified axis or coordinate system, otherwise contains a '1'
- \_BGm can be used from host programs to determine if motion is complete by polling the axes of interest

## EXAMPLES

```
PR 2000,3000,,5000; Set up for a relative move
BG ;' Start the A,B and D motors moving
```

```
HM ;' Set up for the homing
BG A; ' Start only the A-axis moving
```

```
JG 1000,4000; Set up for jog
BG B; ' Start only the B-axis moving
```

```
bstate=_BGB; ' Assign a 1 to bstate if the B-axis is performing a move
```

```
VM AB; ' Vector Mode
VP 1000,2000; ' Specify vector position
VS 20000; ' Specify vector velocity
BG S; ' Begin coordinated sequence
VP 4000,-1000; ' Specify vector position
VE; ' Vector End
```

BG is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BI</b>	<i>Brushless Inputs</i>	SINE COMMUTATION
The BI command is used to define the inputs which are used when Hall sensors have been wired for sinusoidally commutated motors		

BIa=n	Arguments specified with an axis mask and an assignment (=)
BI n ...	Arguments specified with an implicit comma-separated order
Blm	Operand holds the value set in the command

## DESCRIPTION

The BI command is used to define the inputs which are used when Hall sensors have been wired for sinusoidally commutated motors. See the BC command for more information about initialization of sine amplifiers via hall inputs

## ARGUMENTS

**BI<sub>m</sub>=n**

**BI n,n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-1	14	0	1	Select starting General input for hall sensor use	n = -1 uses dedicated hall inputs. n = 0 clears configuration. Inputs 9-16 only valid for 5-8 axis controller
	81	94	N/A	1	Select starting auxiliary encoder input for hall sensor use.	

## REMARKS

DMC41x3

- The inputs can be the general use inputs or the auxiliary encoder inputs. The Hall sensors of each axis must be connected to consecutive input lines,
  - For example: BI 3 indicates that inputs 3,4 and 5 are used for halls sensors.

DMC40x0, DMC41x3, DMC30010

- The Hall A, Hall B and Hall C inputs on the Encoder connector may be specified by setting the BI command to -1.

DMC40x0, DMC41x3

- With the AMP-43540 or the AMP-43640, for a motor wired to work with the AMP-43020/43040, the following wiring to the general inputs is used for commutating the halls with the BI n command.
  - @IN[n] = Hall B
  - @IN[n+1] = Hall C
  - @IN[n+2] = Hall A
- The brushless setup command, BS, can be used to determine the proper wiring of the hall sensors.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
BI , 5; ' The Hall sensor of the Y axis are on inputs 5, 6 and 7.
```

DMC40x0, DMC41x3

```
REM Example for use with AMP-43540
#ex
BA A
BMA= 2000
BIA= -1; ' use hall sensor inputs on the Galil
BC A; ' enable brushless calibration
bc= _BCA; ' store hall state
JGA= 500
SH A; ' enable servo
BG A; ' begin jog
#hall;JP #hall,_BCA=bc;'wait for a hall transition
ST A
MG "Commutation Complete"
EN
```

BI is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BK</b>	<i>Breakpoint</i>	PROGRAMMING
The BK command causes the controller to pause execution of the given thread at the given program line number		

BK n ...	Arguments specified with an implicit comma-separated order
_BK	Operand has special meaning see Remarks

## DESCRIPTION

The BK command causes the controller to pause execution of the given thread at the given program line number. When that line is reached, program execution halts before the line is executed, while all other threads continue running. After a breakpoint is encountered, a new breakpoint can be armed (to continue execution to the new breakpoint) or BK will resume program execution. The SL command can be used to single step from the breakpoint.

## ARGUMENTS

### BK n0,n1

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	1,999	N/A	1	Line number to set breakpoint	n = null resumes normal code operation
<b>n1</b>	0	7	0	1	Thread number to set breakpoint	If n omitted, default value used.

## REMARKS

- Only one breakpoint may be armed at any time.
- BK can be armed before or during thread execution.

### Operand Usage

- \_BK will tell whether a breakpoint has been armed, whether it has been encountered, and the program line number of the breakpoint:
  - = -LineNumber: breakpoint armed
  - = LineNumber: breakpoint encountered
  - = -2147483648: breakpoint not armed

## EXAMPLES

```
BK 3; Pause at line 3 (the 4th line) in thread 0
BK 5; Continue to line 5
SL; Execute the next line
SL 3; Execute the next 3 lines
BK; Resume normal execution
```

BK is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BL</b>	<i>Reverse Software Limit</i>	ERROR CONTROL
The BL command sets the reverse software limit		
		

BLA=n	Arguments specified with an axis mask and an assignment (=)
BL n ...	Arguments specified with an implicit comma-separated order
BLm	Operand holds the value set in the command

## DESCRIPTION

The BL command sets the reverse software limit. If this limit is exceeded during motion, motion on that axis will decelerate to a stop. Reverse motion beyond this limit is not permitted.

## ARGUMENTS

**BLm= n**

**BL n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	-2,147,483,648	1	Position for reverse soft limit	

## REMARKS

- The reverse limit is activated at the position n-1. n = -2147483648 effectively disables the reverse soft limit
- The units are in quadrature counts.
- When the reverse software limit is activated, the automatic subroutine #LIMSWI will be executed if it is included in the program.

## EXAMPLES

```
#test; ' Test Program
AC 1000000; ' Acceleration Rate
DC 1000000; ' Deceleration Rate
BL -15000; ' Set Reverse Limit
JG -5000; ' Jog Reverse
BG A; ' Begin Motion
AM A; ' After Motion (limit occurred)
TP A; ' Tell Position
EN; ' End Program
'
'Galil Controllers also provide hardware limits.
```

BL is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BM</b>	<i>Brushless Modulo</i>	SINE COMMUTATION
-----------	-------------------------	------------------

The BM command defines the length of the magnetic cycle in encoder counts



BM=n	Arguments specified with an axis mask and an assignment (=)
BM n ...	Arguments specified with an implicit comma-separated order
BMm	Operand holds the value set in the command

## DESCRIPTION

The BM command defines the length of the magnetic cycle in encoder counts.

## ARGUMENTS

**BMm= n**

**BM n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	1	10,000,000	2,000	1/65,536	Encoder counts per magnetic cycle	

## REMARKS

- For rotary motors, the magnetic cycle (BM value) is calculated by:
  - BM = encoder counts per revolution / # of pole pairs

## EXAMPLES

```
BM ,60000;' Set brushless modulo for B axis to be 60000
BMC= 100000/3; ' Set brushless modulo for C axis to be 100000/3 (33333.333)
BM ,,,?;' Interrogate the Brushless Module for the D axis

'example calculating brushless modulus using calculated integers
cts= 4096;' Counts per rev
pp= 3;' Pole pairs
BMA= cts/pp

'Changing the BM parameter causes an instant change in the commutation phase.
```

BM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BN</b>	<i>Burn</i>	SYSTEM CONFIG
The BN command saves certain board parameters in non-volatile EEPROM memory		
		

BN	Command takes no arguments
_BN	Operand has special meaning, see Remarks

## DESCRIPTION

The BN command saves certain board parameters in non-volatile EEPROM memory. This command typically takes 1 second to execute and must not be interrupted. The controller returns a colon (:) when the Burn is complete.

## ARGUMENTS

### BN

The BN command has no arguments

## REMARKS

- The following table shows the commands that have their parameters saved with the BN command:

DMC41x3, DMC30010

*Parameters saved during burn*

AC	BO	EO	IK	MO	OT	TM
AF	BR	ER	IL	MT	OV	TR
AG	BW	FA	IT	MU	PF	VA
AQ	CB	FL	KD	NB	PL	VD
AU	CE	FV	KI	NF	PW	VF
BA	CN	GA	KP	NZ	SB	VS
BB	CW	GM	KS	OA	SM	YA
BI	DC	GR	LC	OE	SP	YB
BL	DH	HV	LD	OF	TK	YC
BM	DV	IA	LZ	OP	TL	

### Operand Usage

- \_BN contains the serial number of the processor board.

## EXAMPLES

```
SB 1;' Set bit 1
CB 2;' Clear bit 2
CW 1;' Set data adjustment bit
BN;' Burn all parameter states
```

BN is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BO</b>	<i>Brushless Offset</i>	SINE COMMUTATION
The BO command sets a fixed offset on the command signal for sinusoidally commutated motors		

BOA= n	Arguments specified with an axis mask and an assignment (=)
BO n ...	Arguments specified with an implicit comma-separated order
BOm	Operand holds the value set in the command

## DESCRIPTION

The BO command sets a fixed offset on the command signal for sinusoidally commutated motors. This may be used to offset any bias in the amplifier, or can be used for phase initialization.

## ARGUMENTS

**BOm= n**

**BO n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-5	5	0	20/65,536	Offset applied to DAC output in volts	

## REMARKS

DMC40x0, DMC41x3

### Internal Sine Drive

DMC40x0, DMC41x3, DMC30010

- When using an internal Galil sine drive, each axis has two DACs (Digital to Analog Converter). BO sets the first DAC offset. BQ sets the second.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

### External Sine Drive

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- When using a third party, external sine drive, each motor axis requires two control axes. Therefore, for 4 axes of external sine control, an 8 axis controller is required.
- In this configuration, BO sets the offset for both DACs. Each member of a pair of axes has its own BO value.
- When measuring DAC output voltage, to assure that the output voltage equals the BO parameters, set the PID and OF parameters to zero.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Assume a two axis controller
BA A;` BA allows the control of an external sine drive with the use of two axis. This is now a one axis controller.
' Axis B is used as the secondary DAC for axis A commutation.
'
BO -2,1;` Generates the DAC voltage -2 on the first DAC A, and 1 on the second DAC B of a sinusoidally commutated drive.
```

DMC40x0, DMC41x3, DMC30010

```
'Assume internal Sine drive
BO 1 ;`set A axis first DAC to 1v offset
BO 2 ;`set the A axis second DAC to 2v offset
```

BO is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BP</b>	<i>Burn Program</i>	SYSTEM CONFIG
<b>The BP command saves the application program in non-volatile EEPROM memory</b>		
		

BP Command takes no arguments

## DESCRIPTION

The BP command saves the application program in non-volatile EEPROM memory. This command may take several seconds to execute and must not be interrupted. The controller returns a : when the Burn is complete.

## ARGUMENTS

### BP

The BP command has no arguments

## REMARKS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

- Legacy Software Note: This command may cause the Galil software to issue the following warning "A time-out occurred while waiting for a response from the controller". This warning is normal and is designed to warn the user when the controller does not respond to a command within the timeout period.
- The timeout can be changed in the Galil software but this warning does not affect the operation of the controller or software.

## EXAMPLES

```
:BP;: Burn in program to controller
:': Get colon response when done
```

BP is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BQ</b>	<i>Brushless Offset dual DAC</i>	SINE COMMUTATION
The BQ command sets a fixed offset on the command signal for sinusoidally commutated motors when using an internal Galil sine drive		
		

BQA= n	Arguments specified with an axis mask and an assignment (=)
BQ n ...	Arguments specified with an implicit comma-separated order
BQm	Operand holds the value set in the command

## DESCRIPTION

The BQ command sets a fixed offset on the command signal for sinusoidally commutated motors when using an internal Galil sine drive. This may be used to offset any bias in the amplifier, or can be used for phase initialization.

## ARGUMENTS

**BQm= n**

**BQ n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-5	5	0	20/65,536	Offset applied to DAC output in volts.	

## REMARKS

- When using an internal Galil sine drive, each axis has two DACs (Digital to Analog Converter). BO sets the first DAC offset. BQ sets the second.

## EXAMPLES

```
'Assume internal Sine drive
BO 1;'    set A axis first DAC to 1v offset
BQ 2;'    set the A axis second DAC to 2v offset
```

BQ is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilinc.com](mailto:documentation@galilinc.com)

<b>BR</b>	<i>Brush Axis</i>	SYSTEM CONFIG
The BR command configures the motor configuration and type for an axis		
		

BRA= n	Arguments specified with an axis mask and an assignment (=)
BR n ...	Arguments specified with an implicit comma-separated order
BRm	Operand holds the value set in the command

## DESCRIPTION

The BR command configures the motor configuration and type for an axis.

DMC40x0, DMC41x3, DMC30010

The BR command is used with internal Galil amplifiers to enable which axes will be set as brush-type servos or to configure the firmware to use external drives instead of the internal channel.

## ARGUMENTS

**BRm= n**

**BR n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	

DMC40x0, DMC41x3

Argument	Value	Description	Notes
n	-1	Configured for external drive	Use for external drives with internal sine amps -D3640, -D3540 and -D3520
	0	Configured for Brushless servo	Default
	1	Configured for Brush-type servo	Use for external drives with -D3040 and -D3020

## REMARKS

- If an axis has Off-On-Error(OE) set to 1, an amplifier error will occur on an axis if there are no halls and BR is set to 0. Set BR to 1 to avoid an amplifier error state.
  - The hall error bits cannot cause #AMPERR events if an axis is configured as brush-type.
- With BR1, the hall inputs are available for general use via the QH command.

DMC40x0, DMC41x3, DMC30010

- Note: If the controller has been previously configured with the BA command for sinusoidal commutation with a Galil internal amplifier, the command "BA N" must be issued prior to setting the axis to brushed mode.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3

```
BR 1,0,0; Sets X-axis to brush-type, Y and Z to brushless
```

```
BR 1; Set to brush type, ignore hall errors
BR -1; Set to external amp
```

BR is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BT</b>	Begin PVT Motion	PVT MODE
The BT command begins PVT motion on the specified axes		
		
BT A	Argument is an axis mask	
_BTm	Operand has special meaning, see Remarks	

## DESCRIPTION

The BT command begins PVT motion on the specified axes. All axes specified will begin at the same time. For more details on PVT mode see the user manual.

## ARGUMENTS

### BT mm

DMC40x0, DMC41x3

Min	Max	Default	Resolution	Description	Notes
A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to begin PVT motion	

## REMARKS

- For more details on PVT mode see the user manual.
- \_BTm contains the number of PV segments that have executed.

## EXAMPLES

```
:MG _BT A; '           Query number of PVT segments executed
0.0000
:PVA= 100,200,100; ' Command X axis to move 100 counts reaching an ending speed of 200c/s in 100 samples
:PVA= 100,0,100; '   Command X axis to move another 100 counts reaching an ending speed of 0c/s in 100 samples
:PVA= ,,0; '         Command X axis to exit PVT mode
:BT A; '             Begin PVT mode
:MG _BT A; '           Query number of PVT segments executed
3.0000
:
```

BT is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BV</b>	<i>Burn Variables and Array</i>	SYSTEM CONFIG
The BV command saves the controller variables and arrays in non-volatile EEPROM memory		
		

BV	Command takes no arguments
_BV	Operand has special meaning, see Remarks

## DESCRIPTION

The BV command saves the controller variables and arrays in non-volatile EEPROM memory. This command typically takes up to 2 seconds to execute and must not be interrupted. The controller returns a : when the Burn is complete.

## ARGUMENTS

### BV

The BV command has no arguments

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- \_BV returns the number of controller axes.
- This command will store the ECAM table values in non-volatile EEPROM memory.
- Note: This command may cause the Galil software to issue the following warning "A time-out occurred while waiting for a response from the controller".
- This warning is normal and is designed to warn the user when the controller does not respond to a command within the timeout period.
- This occurs because this command takes more time than the default timeout period. The timeout can be changed in the Galil software.
- This warning does not affect the operation of the board or software.

## EXAMPLES

```
:BV;^ burn in variables
:colon response returned
```

BV is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BW</b>	<i>Brake Wait</i>	SYSTEM CONFIG
The BW command sets the delay between when the brake is turned on and when the amp is turned off		

BWA= n	Arguments specified with an axis mask and an assignment (=)
BW n ...	Arguments specified with an implicit comma-separated order
BWm	Operand holds the value set in the command

## DESCRIPTION

The BW command sets the delay between when the brake is turned on and when the amp is turned off. When the controller goes into a motor-off (MO) state, this is the time (in samples) between when the brake digital output changes state and when the amp enable digital output changes state. The brake is actuated immediately upon MO and the delay is to account for the time it takes for the brake to engage mechanically once it is energized electrically. The brake is released immediately upon SH.

## ARGUMENTS

**BWm= n**

**BW n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	4096	0	1	Specify brake wait time, in samples.	0 = Turn brake function off

## REMARKS

DMC40x0, DMC41x3

- Outputs 1-8 are used for Axes A-H, where output 1 is the brake for axis A and output 2 is the brake for axis B and so on.
- The Brake Wait does not apply when the motor is shut off due to OE1 (Off on Error). In this case (position error exceeded or Abort triggered) the motor off and brake output will be applied simultaneously.

DMC41x3, DMC30010

- When using the brake outputs, it is recommended to order the controller with 500mA sourcing output option (HSRC).

## EXAMPLES

DMC40x0, DMC41x3, DMC30010

```
BW 100; ' Set brake delay to 100 ms (TM1000) for the A axis
```

BW is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BX</b>	Sine Amp Initialization	SINE COMMUTATION
The BX command uses a method to initialize an axis with limited movement of the hardware		

BXA= n	Arguments specified with an axis mask and an assignment (=)
BX n ...	Arguments specified with an implicit comma-separated order
BXm	Operand has special meaning see Remarks

## DESCRIPTION

### THIS COMMAND IS STILL IN BETA. ITS IMPLEMENTATION IS SUBJECT TO CHANGE.

The BX command uses a method to initialize an axis with limited movement of the hardware. The BX uses a limited motion algorithm to determine the proper location of the motor within the magnetic cycle. It is expected to move no greater than 10 degrees of the magnetic cycle. The last stage of the BX command will lock the motor into the nearest 15 degree increment.

## ARGUMENTS

**BXm= n**

**BX n,n,n,n,n,n,n**

**BX< o**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-4.998	4.998	0	20/65,536	Voltage to be applied during amp initialization	-n = end BX with SH. +n = end BX with MO
<b>o</b>	100	5,000	1,000	1	Number of samples for BZ to hold final torque pulse.	o should be set before BXm= n command.

## REMARKS

DMC40x0, DMC41x3

- The BX command is only valid with the AMP-43540 or the AMP-43640
- \_BXm contains 0 if axis m is not a Galil sine amp axis, contains 1 if axis m is an uninitialized sine amp axis, and contains 3 if axis m is an initialized sine amp axis
- An axis with a Galil sine amp powers up in MO state and SH will generate an error for that axis until it is initialized.
- While the BX command is executing communication to and from the controller will be halted. This may result in a timeout if the BX command is sent from the host\*. Embedded code execution will also pause during BX operation.
  - The long timeout (-l) for GalilTools 1.5.0 has been increased to prevent a timeout while using the BX command.
- If the BX command fails to initialize an axis, it will return an error code of 160. TC1 will return "160 BX Command Failure".
- There are several methods to initialize a motor with the Galil sine amplifier. They are listed below:

DMC40x0, DMC41x3, DMC30010

*Commutation of a Galil Sine Drive*

Command	Description
BC/BI	Uses hall sensors to commute until a hall transition is encountered. Drive then commutes sinusoidally.
BX	Uses an algorithm to determine phase angle with minimal motion.
BZ	Drives the motor to a known magnetic phase. Drive then commutes sinusoidally.

## BX Initialization Steps

- Set axes enabled for sine amp with the BA command
- Set motor modulo with the BM command.
- Set OE1 for motor runaway.
- Issue BX1 to test at smaller voltage
  - If error code 160 occurs, try a larger voltage. If motion is occurring then check that the encoder is working. Ensure that the timeout time is long enough for BX (BX
  - If BX is successful, issue SH. Ensure the motor holds position.
  - Attempt a jog. If the motor jogs, then the initialization is complete.
    - If the motor shuts off due to position error, retry BX. Invert the encoder direction with CE if that hasn't been attempted.

## EXAMPLES

```
REM Simple Example
BA A
BMA= 2000
BXA= -3
#bxa;JP #bxa,_BXA>>3
ENDIF
```

```
REM Detailed Example
#com
~a= 0; '0 = A axis, 1 = B axis . . .
BA ~a; 'enable brushless mode
BM~a= 2000; 'must be set per individual motor specifications
BX <1000; 'set pulse duration to 1000 samples
bx_i= 0; 'number of tries for the BX command
#com_h
tc= 0; 'response from TC command if an error occurs
MO ~a; 'start in motor off state
#tv;JP #tv, TV~a>>500; 'make sure axis is not moving
BX~a= -3; 'command the BX command
REM loop until BX passes or error occurs
#loop;JP #loop, (( BX~a>>3) & (tc=0))
REM try again if an error occurred and the number of tries < 5
JP #com_h, ((tc>>0) & (bx_i<5))
```

```

REM if the number of tries is < 5 then BX passed
REM else, try BZ command
IF (bx_i<5)
  MG "Commutation complete"
ELSE
  MG "BX failed to complete"
  MG "attempting BZ command"
  tc= 0;BZ~a= -3
IF tc=0
  MG "BZ command complete"
ELSE
  MG "BZ command failed"
  MG "check motor and encoder wiring"
  MG "try setting CE 2 or swapping 2 motor leads"
ENDIF
ENDIF
EN

#OMDERR
tc= _TC
TC 1
REM if 160 error, increase BX
IF tc=160
  MG "Retry BX"
  bx_i= bx_i+1
  BX <(bx_i*1000);'increase pulse time on failure
ENDIF
RE

```

BX is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>BZ</b>	Brushless Zero	SINE COMMUTATION
The BZ command is used for axes which are configured for sinusoidal commutation to initialize the motor at zero magnetic phase		

BZA= n	Arguments specified with an axis mask and an assignment (=)
BZ n ...	Arguments specified with an implicit comma-separated order
BZm	Operand has special meaning see Remarks

## DESCRIPTION

The BZ command is used for axes which are configured for sinusoidal commutation to initialize the motor at zero magnetic phase. To do this, the command drives the motor to zero magnetic phase and then sets the commutation phase to zero.

## ARGUMENTS

**BZm= n**

**BZ n,n,n,n,n,n,n**

**BZ<o**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-4.998	4.998	0	20/65,536	Voltage to be applied during amp initialization	-n = end BZ with SH. +n = end BZ with MO
o	100	32,767	1,000	1	Number of samples for BZ to hold torque.	o should be set before BZm= n command.

## REMARKS

- BZm contains the distance in encoder counts from the motor's current position and the position of commutation zero for the specified axis.
  - This can be useful to command a motor to move to the commutation zero position for phase initialization.
- This command may be given when the motor is off.
- The BZ command causes instantaneous movement of the motor. It is recommended to start with small voltages and increase as needed. The BZ command voltage must be large enough to move the motor.
- Always use the Off On Error function (OE command) to avoid motor runaway whenever testing sinusoidal commutation.

DMC40x0, DMC41x3, DMC30010

- There are several methods to initialize a motor with the Galil sine amplifier. They are listed below:

DMC40x0, DMC41x3, DMC30010

*Commutation options with a Galil sine drive*

Command	Description
BC/BI	Uses hall sensors to commutate until a hall transition is encountered. Drive then commutes sinusoidally.
BX	Uses an algorithm to determine phase angle with minimal motion.
BZ	Drives the motor to a known magnetic phase. Drive then commutes sinusoidally.

DMC40x0, DMC41x3, DMC30010

## BZ Initialization Steps

DMC40x0, DMC41x3, DMC30010

1. Set axes enabled for sine amp with the BA command
2. Set motor modulo with the BM command.
3. Set OE1 for motor runaway. ER must be greater than BM brushless modulus.
4. Issue BZ1 to test at smaller voltage
  1. If error code 113 occurs, try a larger voltage. If motion is occurring then check that the encoder is working
  2. If error code 114 occurs, ensure that the timeout time is long enough for BZ (BZ)
5. If BZ is successful, issue SH. Ensure the motor holds position.
6. Attempt a jog. If the motor jogs, then the initialization is complete.
  1. If the motor shuts off due to position error, retry BZ. Invert the encoder direction with CE if that hasn't been attempted.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
BZ , -3; ' Drive B axis to zero phase with 3 volt signal, and end with motor enabled.
```

```
:BZ 2; ' Drive A axis to zero phase with 3V torque, and end with Motor off
```

BZ is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CA</b>	<i>Coordinate Axes</i>	VECTOR/LINEAR
-----------	------------------------	---------------

The CA command specifies the coordinate system to apply proceeding vector commands



CA A	Argument is an axis mask
CAm	Operand has special meaning, see Remarks

## DESCRIPTION

The CA command specifies the coordinate system to apply proceeding vector commands. The following commands apply to the active coordinate system as set by the CA command:

*Table Caption Here*

CR	ES	LE	LI	LM
TN	VE	VM	VP	

## ARGUMENTS

### CA m

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	S	T	S	Axis	Coordinate plane to specify	

where

m=S specifies that proceeding vector commands shall apply to the S coordinate system

m=T specifies that proceeding vector commands shall apply to the T coordinate system

CA ? returns a 0 if the S coordinate system is active and a 1 if the T coordinate system is active.

## REMARKS

- CA ? returns a 0 if the S coordinate system is active and a 1 if the T coordinate system is active.
- \_CA contains a 0 if the S coordinate system is active and a 1 if the T coordinate system is active.

## EXAMPLES

```
CA T; ' Specify T coordinate system
VM AB; ' Specify vector motion in the A and B plane
VS 10000; ' Specify vector speed
CR 1000,0,360; ' Generate circle with radius of 1000 counts, start at 0 degrees and complete one circle in counterclockwise direction.
VE; ' End Sequence
BG T; ' Start motion of T coordinate system
```

CA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CB</b>	<i>Clear Bit</i>	<b>IO</b>
<b>The CB command clears a particular digital output</b>		
		

CB n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The CB command clears a particular digital output. The SB and CB (Clear Bit) instructions can be used to control the state of output lines.

## ARGUMENTS

### CB n

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	16	N/A	1	General output bit to be set	Max value is 8 for 1-4 axis controllers
n	1,000	8,999	N/A	1	Set Modbus slave bit	See "CB via Modbus Slave" in Remarks

## REMARKS

- The state of the output can be read with the @OUT command

### CB via Modbus Slave

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- n0 = (SlaveAddress\*10000) + (HandleNum\*1000) + ((Module-1)\*4) + (Bitnum-1)
  - Slave Address is used when the ModBus device has slave devices connected to it and specified as Addresses 0 to 255. Please note that the use of slave devices for modbus are very rare and this number will usually be 0.
  - HandleNum is the handle specifier from A to H.
  - Module is the position of the module in the rack from 1 to 16.
  - BitNum is the I/O point in the module from 1 to 4

## EXAMPLES

```
#main
SB 5;' Set digital output 5
SB 1;' Set digital output 1
CB 5;' Clear digital output 5
CB 1;' Clear digital output 1
EN
```

```
#modbus
REM connect to modubs slave at IP address 192.168.1.50
IHH= 192,168,1,50<502>2
WT 100
SB 8001;'set bit 1 on modbus slave
WT 10
CB 8003;'set bit 3 on modbus slave
EN
```

For detailed information on connecting to a Modbus slave, see:  
<http://www.galilmc.com/techtalk/io-control/setting-up-and-rio-as-extended-io-for-a-controller/>

CB is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CC</b>	<i>Configure Communications Port 2</i>	SYSTEM CONFIG
The CC command configures baud rate, handshake, mode, and echo for the AUX SERIAL PORT, referred to as Port 2		
		

CC n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The CC command configures baud rate, handshake, mode, and echo for the AUX SERIAL PORT, referred to as Port 2. This command must be given before using the MG, or CI commands with Port 2.

## ARGUMENTS

**CC n0,n1,n2,n3**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	9,600	19,200	N/A	9,600	Baud rate	9600 and 19200 are valid baud rates
<b>n1</b>	0	1	N/A	1	Handshake setting	n1=0 turns off handshaking. n1=1 turns handshaking on
<b>n2</b>	0	1	N/A	1	Enable aux serial port	n2=0 disables port. n2=1 enables port
<b>n3</b>	0	1	N/A	1	Echo setting	n3=0 for echo off. n3=1 for echo on

## REMARKS

- None

## EXAMPLES

```
:CC 9600,0,1,0; ' 9600 baud, no handshake, enable, echo off.  
: ' Typical setting with TERM-P or TERM-H.
```

CC is supported on DMC40x0, DMC41x3 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CD</b>	<i>Contour Data</i>	CONTOUR MODE
<b>The CD command specifies the incremental position on contour axes</b>		

CDA= n	Arguments specified with an axis mask and an assignment (=)
CD n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The CD command specifies the incremental position on contour axes. This command is used only in the Contour Mode (CM). The incremental position will be executed over the time period specified by the command DT (ranging from 2 to 256 servo updates)

## ARGUMENTS

**CDm= n**

DMC40x0, DMC41x3, DMC18x6

**CD n,n,n,n,n,n,n=t**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-32,768	32,767	0	1	Contour position segment	Incremental position move
<b>t</b>	1	8	0	1	Time override option	t = 1-8 specifies $2^n$ samples for the given interval.
	0	0	0	0	Time override option	t=0 with n=0 disables Contour mode. See Remarks
	-1	-1	0	0	Time override option	Pauses contour buffer at the segment with t=-1. Reissue DT to re-engage contour mode.

## REMARKS

- The units of the command are in encoder counts.

DMC40x0, DMC41x3, DMC18x6, DMC30010

- CS is the Segment Counter for Contour mode
- The = operator can be used to override the global DT time by transmitting the time in a CD with the position data.
- n=t=0 terminates Contour mode similar to VE or LE for vector mode and linear interpolation mode

DMC40x0, DMC41x3, DMC18x6

- Example. CMBC is terminated with CD 0,0=0

DMC40x0, DMC41x3, DMC18x6, DMC30010

- The user must have a space after CD in order to terminate the Contour Mode correctly.
  - The command CD0=0 (no space) will assign a variable CD0 the value of 0 rather than terminate Contour mode.

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
#contour;
  CM AB;          Program Label
  DT 4;           Enter Contour Mode
  CD 1000,2000;  Set time interval
  CD 2000,4000;  Specify data
  CD 0,0=0;       Next data
  #wait;          End of Contour Buffer
  WT 16,1;        Wait for all segments to process (buffer to empty)
  JP #wait,(_CM<>511) wait for 1 DT time segment (2^4)
  EN;             End Program
```

```
#contour;
  CM A;          Program Label
  DT 4;           Enter Contour Mode
  CD 1000;        Set time interval
  CD 2000;        Specify data
  CD 0=0;         Next data
  #wait;          End of Contour Buffer
  WT 16,1;        Wait for all segments to process (buffer to empty)
  JP #wait,(_CM<>31) wait for 1 DT time segment (2^4)
  EN;             End Program
```

CD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CE</b>	<i>Configure Encoder</i>	SYSTEM CONFIG
The CE command configures the encoder to quadrature type or pulse and direction type		
		
CEA=n	Arguments specified with an axis mask and an assignment (=)	
CE n ...	Arguments specified with an implicit comma-separated order	
CEm	Operand holds the value set in the command	

## DESCRIPTION

The CE command configures the encoder to quadrature type or pulse and direction type. It also allows inverting the polarity of the encoders which reverses the direction of the feedback. The configuration applies independently to the main axes encoders and the auxiliary encoders.

## ARGUMENTS

**CEm=n**

**CE n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	15	0	1	Encoder configuration setting	n is the sum of 2 integers M and N which configure main and auxiliary encoders. See table below for configuration description.

*Configure Encoder Types*

Margument	Main Encoder Type	N argument	Auxiliary Encoder Type
0	Normal quadrature	0	Normal quadrature
1	Normal pulse and direction	4	Normal pulse and direction
2	Reversed quadrature	8	Reversed quadrature
3	Reversed pulse and direction	12	Reversed pulse and direction

For example: n = 10 implies M = 2 and N = 8, thus both encoders are reversed quadrature.

## REMARKS

- When using a servo motor, changing the CE type can cause the motor to run away.
- When the MT command is configured for a stepper motor, the auxiliary encoder (used to count stepper pulses) will be forced to pulse and direction.
- When using pulse and direction encoders, the pulse signal is connected to CHA and the direction signal is connected to CHB.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:CE 0, 3, 6, 2;!  
:CE ?, ?, ?, ?;!  
0,3,6,2  
:v = _CEB;!  
:v = ?  
3  
:
```

CE is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CF</b>	<i>Configure Unsolicited Messages Handle</i>	SYSTEM CONFIG
The CF command sets the port for unsolicited messages		

CF A	Argument is an axis mask
_CFm	Operand has special meaning, see Remarks

## DESCRIPTION

The CF command sets the port for unsolicited messages. The CF command directs the controller to send unsolicited responses to the Main or Aux Serial Port (If equipped), or to an Ethernet handle. An unsolicited message is data generated by the controller which is not in response to a command sent by the host.

## ARGUMENTS

### CF m

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	S	Handle	Ethernet Handle to assign as unsolicited message port	See Remarks
	I	I	S	Handle	Set the port that sent the command as the unsolicited message port	Not valid in program
	S	T	S	Handle	Set serial port as unsolicited message port	m=S is Main serial port. m=T is Aux Serial port

## REMARKS

- Examples of application code commands that will generate unsolicited messages follow.

```
MG "Hello";'      A message (MG)
TC 1;'           A command that returns a response
TP ;'            "
RP A;'           "
var= ?;'         A variable interrogation
var= ;'          "
thisIsAnError;' A dmc error will generate an error message
```

## Ethernet Handle as Unsolicited Message Port

- When communicating over Ethernet, two Ethernet handles should be used:
  - 1.) The first handle should be used for command-and-response traffic. This is the primary handle that the host uses to communicate to the controller.
  - 2.) The second handle should be used for unsolicited traffic. This is the primary handle that the controller uses to asynchronously communicate to the host. Use CF to point unsolicited traffic to this handle.
- It is NOT recommended to use one Ethernet handle for both command-and-response, and unsolicited messages.
- GalilTools will by default establish a two handle connection when using Ethernet, and set CF to the second handle.

## Operand Usage

- \_CF contains the decimal value of the ASCII letter where unsolicited messages are currently routed.

## EXAMPLES

```
:CF I;' send unsolicited traffic to the terminal that sent the command
```

```
'Demonstrates from GalilTools terminal that the
'main handle is separate from the unsolicited handle
192.168.1.3, RIO47102 Rev 1.0c, 1480, IHA IHB
:TH
CONTROLLER IP ADDRESS 192,168,1,3 ETHERNET ADDRESS 00-50-4C-28-05-C8
IHA TCP PORT 23 TO IP ADDRESS 192,168,1,100 PORT 2420
IHB UDP PORT 60007 TO IP ADDRESS 192,168,1,100 PORT 2421
IHC AVAILABLE
IHD AVAILABLE
IHE AVAILABLE
:WH
IHA
:'Main handle is A
:MG _CF
66.0000
:'Unsolicited handle. 66 is ASCII for "B"
:'
```

CF is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CI</b>	<i>Configure Communication Interrupt</i>	SYSTEM CONFIG
The CI command configures program interrupts based on input of characters over the communication port		

CI n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The CI command configures program interrupts based on input of characters over the communication port.

DMC40x0, DMC41x3

The command configures a program interrupt based on characters received on communications port 2, the AUX serial port. An interrupt causes program flow to jump to the #COMINT subroutine. If multiple program threads are used, the #COMINT subroutine runs in thread 0 and the remaining threads continue to run without interruption. The characters received can be accessed via the operands P2CH, P2ST, P2NM, P2CD.

## ARGUMENTS

DMC40x0, DMC41x3

**CI n**

DMC40x0, DMC41x3

Argument	Value	Description	Notes
<b>n</b>	-1	Clear interrupt data buffer	
	0	Do not interrupt	Default
	1	Interrupt on carriage return	
	2	Interrupt on any character	

Argument	Value	Description	Notes
<b>n0</b>	-1	Clear interrupt data buffer	
	0	Do not interrupt	Default
	1	Interrupt on carriage return	
	2	Interrupt on any character	
<b>n1</b>	0	Main serial port configured as standard port for input of Galil commands.	Default. CI execution disabled.
	1	Main serial port configured for CI execution	Data received will not be interpreted as a command.

## REMARKS

- For more, see Operator Data Entry Mode in the user manual.

## EXAMPLES

DMC40x0, DMC41x3

```
:CI 1; Interrupt when the key is received on port 2
:CI 2; Interrupt on a single character received on Port 2
:
```

CI is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CM</b>	Contour Mode	CONTOUR MODE
<b>The Contour Mode is initiated by the instruction CM</b>		

CM A	Argument is an axis mask
_CMm	Operand has special meaning see Remarks

## DESCRIPTION

The Contour Mode is initiated by the instruction CM. This mode allows the generation of an arbitrary motion trajectory with any of the axes. The CD command specified a position increment, and the DT command specifies the time interval between subsequent increments.

## ARGUMENTS

### CM mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axes to initialize to Contour mode	Disabled by default

## REMARKS

- mm=? Returns a 1 if the contour buffer is full and 0 if the contour buffer is empty.

DMC40x0, DMC41x3, DMC18x6, DMC30010

- mm=? Returns a 0 if the contour buffer is full and 511 if the contour buffer is empty.
- \_CM contains a '0' if the contour buffer is full; otherwise it contains the number of available contour segments.
- Issuing the CM command will clear the contour buffer when contour mode is not running.

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
#cont0;
    Define label #cont0
    CM ABCD;          Specify Contour Mode Axes ABCD
    DT 4;             Specify time increment for contour (2^4 servo loops, 16ms at TM1000)
    CD 200,350,-150,500; ' Specify incremental positions on A,B,C and D axes
                           A-axis moves 200 counts B-axis moves 350 counts C-
                           axis moves -150 counts D-axis moves 500 counts
                           '
                           '
                           '
CD 100,200,300,400; ' Next position data
CD 0,0,0,0=0;         Special syntax to terminate Contour mode
#wait;JP #wait,_CM>>511; ' Spin on #Wait label until buffer is empty
                           End of Contour Buffer/Sequence
EN;                  End program
                           '
                           '
#cont1;
    Define label #cont1
    CM ABC;           Specify Contour Mode
    DT 8;             Specify time increment for contour (2^8 servo loops, 256ms at TM1000)
    CD 100,100,100;   New position data
    CD 100,100,100;   New position data
    CD 0,0,0 =-1;     Pause contour buffer set DT to resume
    CD 100,100,100;   New position data
    CD 100,100,100;   New position data
    CD 0,0,0,0=0;     Special syntax to terminate Contour mode
#wait2;JP #wait2,_CM>>511; 'Spin on #wait2 label until buffer is empty
                           End of Contour Buffer/Sequence
EN
```

CM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CN</b>	<i>Configure</i>	SYSTEM CONFIG, IO
The CN command configures the polarity of the limit switches, home switches, latch inputs, the selective abort function, and the program termination behavior of the abort input		

CN n ...	Arguments specified with an implicit comma-separated order
CN0, CN1, CN2, CN3, CN4	Operand holds the value set in the command

## DESCRIPTION

The CN command configures the polarity of the limit switches, home switches, latch inputs, the selective abort function, and the program termination behavior of the abort input.

## ARGUMENTS

### CN n0,n1,n2,n3,n4

Argument	Value	Description	Notes
<b>n0</b>	1	Limit switches active high	Default
	-1	Limit switches active low	
<b>n1</b>	1	HM will drive motor forward when Home input is high. See HM and FE commands.	Default
	-1	HM will drive motor backward when Home input is high. See HM and FE commands	
<b>n2</b>	1	Latch input is active high	Default
	-1	Latch input is active low	
<b>n3</b>	1	Configures inputs 5,6,7,8,13,14,15,16 as selective abort inputs for axes A,B,C,D,E,F,G, and H respectively.	Will also trigger #POSERR automatic subroutine if program is running
	0	Inputs 5,6,7,8,13,14,15,16 are configured as general use inputs	
<b>n4</b>	1	Abort input will not terminate program execution	Default
	0	Abort input will terminate program execution	

## REMARKS

- n0 is useful for testing the operation of the #LIMSWI automatic subroutine. See example below.

## EXAMPLES

```
CN 1,1; Sets limit and home switches to active high
CN ,,-1; Sets input latch active low
```

REM n0 is useful for testing the operation of the #LIMSWI automatic subroutine

```
#test
CN -1; Switches are active low
JCA= 100
BG A; Start a slow jog move
WT 1000
CN 1; Cause a limit fault by inverting the limit polarity
EN
'
#LIMSWI; Automatic sub will automatically launch on limit detection
MG "Limit Switch Routine"
WT 100
CN -1; Return to correct polarity
RE
```

CN is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CR</b>	<i>Circle</i>	VECTOR/LINEAR
-----------	---------------	---------------

When using the vector mode (VM), the CR command specifies a 2-dimensional arc segment



CR n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

When using the vector mode (VM), the CR command specifies a 2-dimensional arc segment. The VE command must be used to denote the end of the motion sequence after all CR and VP segments are specified. The BG (Begin Sequence) command is used to start the motion sequence. Parameters for radius, starting angle and traverse angle must all be entered for each CR command.

## ARGUMENTS

**CR n0, n1, n2 < o > p**

DMC41x3, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	10	6000000	N/A	1	Radius of circle segment	
<b>n1</b>	-32000	32000	N/A	1	Starting angle of circle segment	
<b>n2</b>	-32000	32000	N/A	1/65536	Degrees to traverse for circle segment	
<b>o</b>	0	15000000	See Notes	2	Speed override at execution of segment	For MT 1,-1,1.5,-1.5. If omitted, use VS
	0	3000000	See Notes	2	Speed override at execution of segment	For MT 2,-2,2.5,-2.5. If omitted, use VS
<b>p</b>	0	8000000	See Notes	2	Speed override achieved at end of segment.	If omitted, use VS

## REMARKS

- The product of n0 \* n2 must be less than 450,000,000
- A positive n2 denotes counterclockwise traverse, -n2 denotes clockwise.
- n0 units are in quadrature counts.
- n1 and n2 have units of degrees.

## EXAMPLES

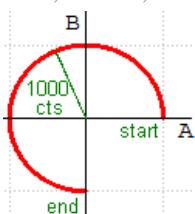
DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

'A starting position of zero degrees denotes that the radius lies along  
'a vector following the positive X axis, on a 2D Cartesian space:

```
VM AB
CR 1000,0,270
VE
BG S
EN
```

'The 2-d map out the position output can be seen below

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2



DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
VM AB;' Specify vector motion in the A and B plane
VS 1000;' Specify vector speed
CR 1000,0,360;' Generate circle with radius of 1000 counts, start at
' 0 degrees and complete one circle in counterclockwise
' direction.
CR 1000,0,360 < 40000;' Generate circle with radius of 1000 counts, start
' at 0 degrees and complete one circle in counterclockwise
' direction and use a vector speed of 40000.
VE;' End Sequence
BG S;' Start motion
```

```
'Generate a sine wave output on the A axis
VM AN;' Specify vector motion in the A and N plane
VS 1000;' Specify vector speed
CR 1000,0,360;' Generate sine wave with amplitude of 1000 counts
' start at 0 degrees and complete one cycle
CR 1000,0,360<40000;' Generate same sine wave with same amplitude
' but run at faster speed (higher frequency)
'
VE;' End Sequence
BG S;' Start motion
```



<b>CS</b>	<i>Clear Sequence</i>	VECTOR/LINEAR
The CS command will remove VP, CR or LI commands stored in a motion sequence for a coordinated axis		
		

CS A	Argument is an axis mask
_CSm	Operand has special meaning, see Remarks

## DESCRIPTION

The CS command will remove VP, CR or LI commands stored in a motion sequence for a coordinated axis. After a sequence has been executed, the CS command is not necessary to put in a new sequence. This command is useful when you have incorrectly specified VP, CR or LI commands.

## ARGUMENTS

### CS m

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	S	T	N/A	Axis	Coordinate plane specified to clear buffer	

## REMARKS

- \_CSm contains the segment number in the sequence specified by m, S or T.
- This operand is valid in the Linear mode, LM, Vector mode, VM, and Contour mode, CM.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#clear; ' Label
CA T; ' Specify the T coordinate system vector points
VP 1000,2000; ' Vector Position
VP 4000,8000; ' Vector Position
CS T; ' Clear vectors specified in T coordinate system
CA S; ' Specify the S coordinate system vector points
VP 1000,5000; ' New vector
VP 8000,9000; ' New vector
CS S; ' Clear vectors specified in S coordinate system
```

CS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>CW</b>	<i>Copyright information and Data Adjustment bit on off</i>	SYSTEM CONFIG
<b>The CW command will return the copyright information when the argument, n, is 0 or is omitted</b>		

CW n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The CW command will return the copyright information when the argument, n, is 0 or is omitted. Otherwise, the CW command is used as a communications enhancement for use by the Galil terminal software programs. When turned on, the most significant bit of unsolicited ASCII characters is set to 1. Unsolicited ASCII characters are characters that are returned from a program running on the controller (usually from the MG command). This command does not affect solicited characters, which are characters that are returned as a response to a command sent from a host PC (e.g. TP).

If using Galil drivers, CW will be automatically configured - the user should not change the CW settings.

## ARGUMENTS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

**CW n0,n1**

DMC41x3

Argument	Value	Description	Notes
n0	0	Causes controller to return a copyright information string	Equivalent to n0 = ?
	1	Controller will set the MSB of unsolicited message characters	
	2	Controller will not set the MSB of unsolicited message characters	
n1	0	Non-functional argument. Preserved for compliance with third party software	Default. Must be set when viewing unsolicited messages from non-Galil software
	1	Non-functional argument. Preserved for compliance with third party software	

## REMARKS

- Galitools automatically sends CW 1 during connection to a controller.
  - If also reading unsolicited data through a non-Galil software (eg Hyperterminal), issue CW 2

## EXAMPLES

```
CW 1; Set CW to Galil Driver mode (MSB set on unsolicited characters)
'
' The CW command can cause garbled (non-ASCII) characters to be returned
' by the controller when using third-party software. Use CW2.
CW 2; Set CW to third-party device mode (normal ASCII on unsolicited characters)
```

CW is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DA</b>	<i>Deallocate Variables and Arrays</i>	PROGRAMMING
<b>The DA command frees the array and/or variable memory space</b>		

DA n ...	Arguments specified with an implicit comma-separated order
DAm	Operand has special meaning, see Remarks

## DESCRIPTION

The DA command frees the array and/or variable memory space. In this command, more than one array or variable can be specified for memory de-allocation. Different arrays and variables are separated by comma when specified in one command.

## ARGUMENTS

### DA str[],str

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	8 chars	N/A	String	Array name to deallocate	If str = *, deallocate all arrays
	1 char	8 chars	N/A	String	Variable name to deallocate	If str = *, deallocate all variables

where

- c[] - Defined array name
- d - Defined variable name
- d = \* deallocates all the variables
- c = \*[] - Deallocates all the arrays
- DA? Returns the number of arrays available.

## REMARKS

- DA contains the total number of arrays available.
- Since this command deallocates the spaces and compacts the array spaces in the memory it is possible that execution of this command may take longer time than a standard command.
- Variables and arrays that are deallocated are not set to zero. A routine that writes zeros to the array and/or variables should be created if this is desired.

## EXAMPLES

```
'Cars' and 'Salesmen' are arrays, and 'Total' is a variable.
DM cars[40],salesmen[50];'           Dimension 2 arrays
total= 70;                            Assign 70 to the variable Total
DA cars[0],salesmen[0],total;'        Deallocate the 2 arrays & variable
DA *[0];'                            Deallocate all arrays
DA *,*[0];'                          Deallocate all variables and all arrays
```

DA is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DC</b>	<i>Deceleration</i>	INDEPENDENT MOTION
The Deceleration command (DC) sets the linear deceleration rate of the motors for independent moves such as PR, PA and JG moves		

DCA= n	Arguments specified with an axis mask and an assignment (=)
DC n ...	Arguments specified with an implicit comma-separated order
DCm	Operand holds the value set in the command

## DESCRIPTION

The Deceleration command (DC) sets the linear deceleration rate of the motors for independent moves such as PR, PA and JG moves. The parameters will be rounded down to the nearest factor of 1024 and have units of counts per second squared.

## ARGUMENTS

**DCm= n**

**DC n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	1,024	1,073,740,800	256,000	see Notes	Deceleration rate	See Remarks for resolution details

## REMARKS

- DC may be changed during a move in Jog mode, but not in a PA or PR move.
  - However, directly following an axis stop (ST m or a limit switch, #LIMSWI), the DC value of a PA or PR move may be changed while the axis is still decelerating

DMC40x0, DMC41x3, DMC18x6, DMC30010

## Resolution

DMC40x0, DMC41x3, DMC18x6, DMC30010

- The resolution of the DC command is dependent upon the update rate setting (TM). With the default rate of TM 1000 the resolution is 1024 cnts/second^2. The equation to calculate the resolution of the DC command is:
  - resolution =  $1024 * (1000/TM)^2$
- example: With TM 500 the minimum DC setting and resolution is 4096 cnts/second^2
  - resolution =  $1024 * (1000/500)^2 = 4096$

## EXAMPLES

```
PR 10000; ' Specify position
AC 2000000; ' Specify acceleration rate
DC 1000000; ' Specify deceleration rate
SP 5000; ' Specify slew speed
BG ;' Begin motion
```

DC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DE</b>	<i>Dual (Auxiliary) Encoder Position</i>	SYSTEM CONFIG
<b>The DE command defines the position of the auxiliary (dual) encoders</b>		

DEA= n	Arguments specified with an axis mask and an assignment (=)
DE n ...	Arguments specified with an implicit comma-separated order
DEm	Operand holds the value set in the command

## DESCRIPTION

The DE command defines the position of the auxiliary (dual) encoders.

Dual encoders are useful when you need an encoder on the motor and on the load. The encoder on the load is typically the auxiliary encoder and is used to verify the true load position. Any error in load position is used to correct the motor position.

## ARGUMENTS

**DEm= n**

**DE n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-2,147,483,648	2,147,483,647	0	1	Position set for auxiliary encoders	For MT 1,-1,1.5,-1.5
	-2,147,483,648	2,147,483,647	0	1	Position set for main encoders	For MT 2,-2,2.5,-2.5

## REMARKS

- When using stepper motors, the DE command defines the main encoder position.
- The auxiliary encoders are not available for the stepper axis or for any axis where output compare is active.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
DE 0,100,200,400; Set the current auxiliary encoder position to 0,100,200,400 on A,B,C and D axes
DE ?,?,?,?; Return auxiliary encoder positions
duala= DEA; Assign auxiliary encoder position of A-axis to the variable duala
```

DE is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilinc.com](mailto:documentation@galilinc.com)

<b>DF</b>	<i>Dual Feedback (DV feedback swap)</i>	SYSTEM CONFIG
The DF command allows configuration of BiSS or SSI feedback in Dual Loop mode as the load encoder		

DF n ...	Arguments specified with an implicit comma-separated order
_DFm	Operand holds the value set in the command

## DESCRIPTION

The DF command allows configuration of BiSS or SSI feedback in Dual Loop mode as the load encoder. For users wishing to operate with SSI or BiSS in Dual Loop mode (DV), the DF command can be used to configure a load-side serial encoder and a motor-side incremental encoder with DVI.

## ARGUMENTS

### DF n,n,n,n,n,n,n

Argument	Value	Description	Notes
n	0	Disable feedback swap	Default
	1	Enable feedback swap	

## REMARKS

- Wire the motor's incremental encoder per normal to the DMC-4xxx main encoder inputs. The load SSI encoder should be wired to the axis aux encoder lines:  
*SSI Signals for DMC-4xxx*

Nominal Signal Name	Signal Reassignment with SSI	Signal Reassignment with BiSS
AA+	Clock+	MA+
AA-	Clock-	MA-
AB+	Data+	SLO+
AB-	Data-	SLO-

- Once wired, configure the serial encoder as an auxiliary encoder.
  - See SI or SS for configuration information.
- Once configured for Dual Loop (DVI), DF1 will swap the serial encoder to become the load (Main) encoder. The incremental encoder wired to the main encoder inputs becomes the motor (Auxiliary) encoder. TP will now report the serial encoder position, and TD will report the incremental encoder position.

## EXAMPLES

```
MO A; ' Disable motor on X
SIA= 2,25,15,0<13>2; ' Setup SSI encoder to fill the Aux encoder register
DF 1; ' Enable Dual Feedback Swap
DV 1; ' Enable Dual Loop mode
SH A; ' Enable servo with new configuration
```

DF is supported on DMC40x0, DMC41x3 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DH</b>	<i>DHCP Server Enable</i>	ETHERNET
The DH command configures the DHCP or BOOT-P functionality on the controller for Server IP addressing		
		

DH n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The DH command configures the DHCP or BOOT-P functionality on the controller for Server IP addressing.

## ARGUMENTS

### DH n

Argument	Value	Description	Notes
n	0	Enable BOOT-P and disable DHCP	Allows IP assignment through IA command.
	1	Disable BOOT-P and enable DHCP	Default. Allows IP assignment through DHCP server.

## REMARKS

- DH 0 must be set to manually assign and burn in an IP address. With DH 1 set, IP address cannot be burned.

## EXAMPLES

DH 1; ' Sets the DHCP function on. IA assignment will no longer work.
DH 0; ' Sets the DHCP function off, and the Boot-P function on.

DH is supported on DMC40x0, DMC41x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DL</b>	<i>Download</i>	<b>PROGRAMMING</b>
<b>The DL command transfers a data file from the host computer to the controller</b>		
		

DL n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The DL command transfers a data file from the host computer to the controller. It is recommended to use the program download functions available through the GalilTools software and drivers rather than directly using the DL command. Instructions in the file will be accepted as a data stream without line numbers. The file is terminated using <control> Z, <control> Q, <control> D, or \.

## ARGUMENTS

### DL n

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

### DL #str

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	1,999	0	1	Line number to begin program download	
str	1 char	8 chars	""	String	Name of label in RAM to begin download from	If str = "", download begins at the end of the current program in RAM

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	999	0	1	Line number to begin program download	

## REMARKS

- DO NOT insert spaces before each command.

DMC40x0, DMC41x3, DMC18x6

\_DL gives the number of available labels (510 maximum)

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:DL
#A;PR 4000;BGA
AMA;MG DONE
EN
Z
'End download
```

DL is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DM</b>	<i>Dimension</i>	PROGRAMMING
The DM command defines a single-dimensional array with a name and n total elements		

DM n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The DM command defines a single-dimensional array with a name and n total elements. The first element of the defined array starts with element number 0 and the last element is at n-1.  
Typing in array name with [-1] element marked reports the number of elements for that array.

## ARGUMENTS

**DM str[n]**

**DM str**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	8 chars	N/A	String	Name of array or variable to dimension	
n	1	16,000	N/A	1	Number of array elements to assign to dimensioned array	

where

c is a array name of up to eight alphanumeric characters, starting with an alphabetic character.

i is the number of array elements.

n = ? returns the number of array elements available.

## REMARKS

DMC40x0, DMC41x3, DMC30010

- Typing in array name with [-1] element marked reports the number of elements for that array.
- The first character of str must be alphabetic. The rest can be any alphanumeric characters.
- When assigning array elements, the number specified must be less than the current available array space
- \_DM contains the available array space.

## EXAMPLES

```
DM pets[5],dogs[2],cats[3];' Define dimension of arrays, Pets with 5 elements, Dogs with 2 elements, Cats with 3 elements
DM tests[1600];' Define dimension of array Tests with 1600 elements
```

DMC40x0, DMC41x3, DMC30010

```
:DM ?
16000
:DM myarray[1000]
:DM ?
15000
'DMC-4xxx and 30010 provide length of array with array[-1]
:MG "MyArray contains",myarray[-1]," elements"
MyArray contains 1000.0000 elements
:
```

DM is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DP</b>	<i>Define Position</i>	SYSTEM CONFIG
The DP command sets the current motor position and current command positions to a user specified value		

DPA= n	Arguments specified with an axis mask and an assignment (=)
DP n ...	Arguments specified with an implicit comma-separated order
DPm	Operand holds the value set in the command

## DESCRIPTION

The DP command sets the current motor position and current command positions to a user specified value. The units are in quadrature counts. This command will set both the TP and RP values.

## ARGUMENTS

**DPm= n**

**DP n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Virtual axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	Value assigned to motor/commanded position (RP and TP registers)	For MT 1,-1,1.5,-1.5
	-2,147,483,648	2,147,483,647	0	1	Value assigned to step/commanded position (RP and TD registers)	For MT 2,-2,2.5,-2.5

## REMARKS

- The DP command sets the commanded reference position for axes configured as steppers. The units are in steps.
  - Example: "DP 0" This will set the registers for TD and RP to zero, but will not effect the TP register value. When equipped with an encoder, use the DE command to set the encoder position for stepper mode.
- The DP command is useful to redefine the absolute position.
  - For example, you can manually position the motor by hand using the Motor Off command, MO. Turn the servo motors back on with SH and then use DP0 to redefine the new position as your absolute zero.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:DP 0,100,200,400; Sets the current position of the A-axis to 0, the B-axis to 100, the C-axis to 200, and the D-axis to 400
:DP ,-50000; Sets the current position of B-axis to -50000. The B,C and D axes remain unchanged.
:DP ?,?,?,?; Interrogate the position of A,B,C and D axis.
0, -50000, 200, 400
:DP ?; Interrogate the position of A axis
0
:
```

```
:DP 0; Sets the current position of the A-axis to 0
:DP -50000; Sets the current position of A-axis to -50000.
:DP ?; Interrogate the position of A
-50000
```

DP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DR</b>	<i>Configures I O Data Record Update Rate</i>	SYSTEM CONFIG
<b>DR</b> specifies and enables the rate for the controller to output its data record		
		

DR n ...	Arguments specified with an implicit comma-separated order
_DR	Operand has special meaning see Remarks

## DESCRIPTION

DR specifies and enables the rate for the controller to output its data record.

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

For ethernet-based controllers, the controller creates a QR record and sends it to the unsolicited UDP Ethernet Handle at the specified rate. See the User Manual for the data record map

## ARGUMENTS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

**DR n0, n1**

DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	8	30,000	0	2	Data update rate specified in samples between packets.	
	0	0	0	0	Turn off data record output	
<b>n1</b>	0	7	see Notes	1	Ethernet handle to output data record packet	0=A,1=B,2=C,3=D,4=E,5=F,6=G,7=H.

## REMARKS

- If a small sample period and a small update rate is used, the controller may become noticeably slower as a result of maintaining a high update rate.

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- If n1 is omitted, then the CF unsolicited message port is used by default.
- The DR port specified with n1 must be a UDP handle.
- \_DR0 contains the data record update rate (n)
- \_DR1 contains the specified handle (m). Will return an integer 0-7 for handles A-H.

## EXAMPLES

```
:DR 8,0
Gx~P
`@~P
-H`~P
0~P
:DR 0

'Note: The data record is in a binary, non-printable format
'(the output above is normal when printing to the terminal)
```

DR is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>DT</b>	<i>Delta Time</i>	CONTOUR MODE
<b>The DT command sets the time interval for Contour Mode</b>		

**DT n ...** Arguments specified with an implicit comma-separated order

## DESCRIPTION

The DT command sets the time interval for Contour Mode.

DMC40x0, DMC41x3, DMC18x6, DMC30010

Sending the DT command once will set the time interval for all contour data until a new DT command (or CDm=n) is sent.

## ARGUMENTS

### DT n

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n</b>	1	8	1	1	Set time interval for contour mode in $2^n$ samples.	
	-1	-1	N/A	0	$n=-1$ to pause the contour mode	See Remarks.

## REMARKS

- By default the sample period is 1 msec (set by the TM command); with n=1, the time interval would be 2 msec
- n = -1 allows a pre-load of the contour buffer or to asynchronously pause the contour buffer. DT-1 during contour mode will pause the contour buffer (and commanded movement).
- A positive DT will resume contour mode from paused position of buffer.
- DT can be overridden with the =t parameter within a CD segment.

## EXAMPLES

```
:DT 4; '           Specifies time interval to be 16 msec (TM1000)
:DT 7; '           Specifies time interval to be 128 msec
:
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
REM basic contour example
#cont0; '          Define label #Cont0
CM ABCD; '          Specify Contour Mode
DT 4; '              Specify time increment for contour
CD 200,350,-150,500; '  Specify incremental positions on A,B,C and C axes
'                  A-axis moves 200 counts B-axis moves 350 counts C-
'                  axis moves -150 counts C-axis moves 500 counts
CD 100,200,300,400 ;'  New position data
CD 0,0,0,0=0; '       End of Contour Buffer/Sequence
#wait; '             Wait for all segments to process (buffer to empty)
WT 16,1; '            wait for 1 DT time segment ( $2^4$ )
JP #wait,(_CM<>511)  End program
EN; '
```

DMC40x0, DMC41x3, DMC18x6

```
REM contour example for pre-loading of contour buffer
#cont1; '          Define label #Cont1
CM AB; '            Specify Contour Mode
DT -1; '             Pause Contour Mode to allow pre-load of buffer
CD 100,200; '        Countour Data pre-loaded in buffer
CD 400,200; '        Countour Data pre-loaded in buffer
CD 200,100; '        Countour Data pre-loaded in buffer
CD 300,50; '         Countour Data pre-loaded in buffer
AI -1; '             Wait for Analog input 1 to go low
DT 8; '              Set positive DT to start contour mode
CD 0,0,0,0=0; '       End of Contour Buffer/Sequence
#wait; '             Wait for all segments to process (buffer to empty)
WT 16,1; '            wait for 1 DT time segment ( $2^4$ )
JP #wait,(_CM<>511) End program
EN; '
```

```
REM contour example for pre-loading of contour buffer
#cont1; '          Define label #Cont1
CM A; '             Specify Contour Mode
DT -1; '             Pause Contour Mode to allow pre-load of buffer
CD 100; '            Countour Data pre-loaded in buffer
CD 400; '            Countour Data pre-loaded in buffer
CD 200; '            Countour Data pre-loaded in buffer
CD 300; '            Countour Data pre-loaded in buffer
AI -1; '             Wait for Analog input 1 to go low
DT 8; '              Set positive DT to start contour mode
CD 0=0; '             End of Contour Buffer/Sequence
#wait; '             Wait for all segments to process (buffer to empty)
WT 16,1; '            wait for 1 DT time segment ( $2^4$ )
JP #wait,(_CM<>31) End program
EN; '
```



<b>DV</b>	<i>Dual Velocity (Dual Loop)</i>	FILTER/CONTROL
-----------	----------------------------------	----------------

The DV function changes the operation of the PID filter to work off of dual encoders



DVA= n	Arguments specified with an axis mask and an assignment (=)
DV n ...	Arguments specified with an implicit comma-separated order
DVm	Operand holds the value set in the command

## DESCRIPTION

The DV function changes the operation of the PID filter to work off of dual encoders. DV enabled causes the KD (derivative) term to operate on the dual encoder instead of the main encoder. This results in improved stability in the cases where there is a backlash between the motor and the main encoder, and where the dual encoder is mounted on the motor.

## ARGUMENTS

**DVm= n**

**DV n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	State of dual loop mode	n = 0 disables Dual loop, n = 1 enables Dual loop

## REMARKS

- The DV command is useful in backlash and resonance compensation.

DMC40x0, DMC41x3, DMC18x6, DMC30010

### Using DV with Large motor/load encoder ratio

DMC40x0, DMC41x3, DMC18x6, DMC30010

- When using Dual Loop mode with a large motor:load ratio and/or running at high velocities where low position error at speed is required, FV should be used to compensate for the derivative contribution from the higher resolution motor encoder.
- The FV value is calculated by the equation:
  - FV = (KD/4)\*(motor/load)
  - motor/load = effective motor to load ratio
- For example: KD = 200, motor encoder changes 5000 counts per 1000 counts of load encoder (motor/load = 5/1)
  - FV = (200/4)\*(5/1) = 250

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6

```
DV 1,1,1,1;' Enables dual loop on all axes
DV 0;' Disables DV on A axis
DV ,,1,1;' Enables dual loop on C axis and D axis. Other axes remain unchanged.
DV 1,0,1,0;' Enables dual loop on A and C axis. Disables dual loop on B and D axis.
MG _DVA;' Returns state of dual velocity mode for A axis
```

DV is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EA</b>	Choose ECAM master	ECAM/GEARING
The EA command selects the master axis for the electronic cam mode		
		

EA A Argument is an axis mask

## DESCRIPTION

The EA command selects the master axis for the electronic cam mode. Any axis may be chosen.

## ARGUMENTS

**EA m**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign as ECAM master	
	M	N	N/A	Axis	Virtual axis to assign as ECAM master	

## REMARKS

- The ECAM mode runs off of the master's main encoder (TP) even when the axis is running in stepper mode.
- When using the M or N imaginary axes, the commanded position is used.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#camone
master= 400
slave= 8192
EB 0; 'Disable ECAM Mode
ET[0]= ,0
ET[1]= ,2048
ET[2]= ,4096
ET[3]= ,6144
ET[4]= ,8192
EA A; 'Set Master Axis as X
EM master,slave
EP master/4,0
DP 0,0
SH AB
'NOTE: (EP Value)*(# of Cam Points) must be >= to Master Modulus
JG 100;BG A
EB 1
EG ,0; 'Start ECAM profile
EN
```

```
#camone
master= 400
slave= 8192
EB 0; 'Disable ECAM Mode
ET[0]= 0
ET[1]= 2048
ET[2]= 4096
ET[3]= 6144
ET[4]= 8192
EA N; 'Set Master Axis as N
MM master
EM slave
EP master/4,0
DPN= 0;DP 0
SH A
REM (EP Value)*(# of Cam Points) must be >= to Master Modulus
JGN= 100;BG N
EB 1; 'Enable ECAM
EG 0; 'Engage slave at 0 master position
EN
```

EA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EB</b>	<i>Enable ECAM</i>	ECAM/GEARING
<b>The EB function enables or disables the cam mode</b>		
		

EB n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The EB function enables or disables the cam mode. In this mode, the starting position of the master axis is specified within the cycle.

## ARGUMENTS

### EB n

Argument	Value	Description	Notes
n	0	Stop ECAM mode	Default
	1	Start ECAM mode	

## REMARKS

- When the EB command is given, the master axis position is modularized.

## EXAMPLES

```
EB 1;' Starts ECAM mode
EB 0;' Stops ECAM mode
b = EB;' Return status of cam mode
```

EB is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EC</b>	<i>ECAM Counter</i>	ECAM/GEARING
-----------	---------------------	--------------

The EC function sets the index into the ECAM table

EC n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The EC function sets the index into the ECAM table. This command is only useful when entering ECAM table values without index values and is most useful when sending commands in binary. See the command, ET.

## ARGUMENTS

### EC n

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	256	0	1	Set the ECAM table index	

## REMARKS

- None

## EXAMPLES

```
EC 0;' Set ECAM index to 0
ET 200,400;' Set first ECAM table entries to 200,400
ET 400,800;' Set second ECAM table entries to 400,800
var= _EC;' Set the ECAM index value to a variable
```

EC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ED</b>	Edit	PROGRAMMING
<b>The ED command puts the controller into the Edit subsystem</b>		



ED n ...	Arguments specified with an implicit comma-separated order
ED, _ED1, _ED4	Operand has special meaning, see Remarks

## DESCRIPTION

The ED command puts the controller into the Edit subsystem. The ED command is used when using Telnet style interface (not Galil Software). In the Edit subsystem, programs can be created, changed, or destroyed.

## ARGUMENTS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

### ED n

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	1,999	see Notes	1	Line number to begin editing	Default n is the last line of program space with commands.

## REMARKS

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

- The commands in the Edit subsystem are the following

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

*ED Commands*

Key Combination	Function
D	Deletes a Line
I	Inserts a line before the current
P	Displays the previous line
Q	Exits the ED subsystem
Enter	Saves a line and moves cursor to next

### Operand Usage

- \_ED contains the line number of the last line to have an error.
- \_ED1 contains the number of the thread where the error occurred (for multitasking).
- \_ED4 when evaluated in an embedded code thread, this operand will contain the thread id of the calling thread. This is useful for DMC code to determine which thread it is running in. See example below.

## EXAMPLES

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
:ED
0 #START
1 PR 2000
2 BGA
3 SLKJ Bad line
4 EN
5 #CMDERR Routine which occurs upon a command error
6 V=_ED
7 MG "An error has occurred" {n}
8 MG "In line", V{F3.0}
9 ST
10 ZS0
11 EN
Hint: Remember to quit the Edit Mode prior to executing or listing a program.
```

```
'Using _ED4
XQ #id,1
XQ #id,2
XQ #id,3
XQ #id,4
XQ #id,5
XQ #id,6
XQ #id,7
#id
MG {Z10.0}"This message is from thread",_ED4
EN
```

```
' Returns...
' :XQ
' This message is from thread 1
' This message is from thread 2
' This message is from thread 3
' This message is from thread 4
' This message is from thread 5
' This message is from thread 6
' This message is from thread 7
' This message is from thread 0
```



<b>EG</b>	<i>ECAM go (engage)</i>	ECAM/GEARING
The EG command engages an ECAM slave axis at a specified position of the master		

EGA= n	Arguments specified with an axis mask and an assignment (=)
EG n ...	Arguments specified with an implicit comma-separated order
EGm	Operand has special meaning see Remarks

## DESCRIPTION

The EG command engages an ECAM slave axis at a specified position of the master. Once a slave motor is engaged, its position is redefined to fit within the cycle.

## ARGUMENTS

**EGm= n**

**EG n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	- 2,147,483,648	2,147,483,647	0	1	Master position to engage ECAM slave	n = outside of master axis position range causes slave to engage immediately.

## REMARKS

- EGm contains ECAM status for specified slave axis. 0 = axis is not engaged, 1 = axis is engaged.
- n=? Returns 1 if specified axis is engaged and 0 if disengaged.
- This command is not a trippoint. This command will not hold the execution of the program flow. If the execution needs to be held until master position is reached, use MF or MR command.

## EXAMPLES

```
EG 700,1300; ' Engages the A and B axes at the master position 700 and 1300 respectively.
b = EGB; ' Return the status of B axis, 1 if engaged
```

EG is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EI</b>	<i>Event Interrupts</i>	SYSTEM CONFIG
The EI command is used to enable interrupts on events		

EI n ...	Arguments specified with an implicit comma-separated order
EI	Operand has special meaning see Remarks

## DESCRIPTION

The EI command is used to enable interrupts on events. EI enables interrupts for the predefined event conditions in the table below. When a condition (e.g. Axis A profiled motion complete) occurs after EI is armed, a particular status byte value (e.g. \$D0 or 208) is delivered to the host PC along with the interrupt.

DMC40x0, DMC41x3, DMC30010

Interrupts are issued as automatically dispatched UDP packets. GalilTools version 1.2.1.0 or newer required for software support.

## ARGUMENTS

DMC40x0, DMC41x3, DMC30010

**EI n1,n2,n3**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n1</b>	0	65,535	0	1	16-bit interrupt mask	0 turns off interrupts. See Remarks for bit mask
<b>n2</b>	0	255	0	1	8-bit input mask	Used to select the specific digital input trigger. Bit 15 of n1 must be set for the n2 mask to be used.
<b>n3</b>	-1	7	-1	1	Preconfigured UDP handle for interrupt transmission	-1 disabled, 0-7 indicate Handles A-H, respectively

## REMARKS

- EI contains the interrupt mask n1
- n1 = 0 means "don't interrupt" and clears the queue when issued
- The interrupts marked with \* in the table below must be re-enabled with EI after each occurrence
- Bit 15 of n1 must be set for the n2 input mask to be used

DMC40x0, DMC41x3, DMC30010

- If the handle specified by n3 is not UDP or is not initialized, an error will occur
- GalilTools software will auto configure n3, allowing the user to ignore its use in most cases

## n1 Bit Mask

DMC40x0, DMC41x3

*Interrupt Bits*

bit	n1=2^bit Hex (decimal)	Status Byte Hex (decimal)	Condition
<b>0</b>	\$0001 (1)	\$D0 (208)	Axis A profiled motion complete _BGA = 0
<b>1</b>	\$0002 (2)	\$D1 (209)	Axis B profiled motion complete _BGB = 0
<b>2</b>	\$0004 (4)	\$D2 (210)	Axis C profiled motion complete _BGC = 0
<b>3</b>	\$0008 (8)	\$D3 (211)	Axis D profiled motion complete _BGD = 0
<b>4</b>	\$0010 (16)	\$D4 (212)	Axis E profiled motion complete _BGE = 0
<b>5</b>	\$0020 (32)	\$D5 (213)	Axis F profiled motion complete _BGF = 0
<b>6</b>	\$0040 (64)	\$D6 (214)	Axis G profiled motion complete _BGG = 0
<b>7</b>	\$0080 (128)	\$D7 (215)	Axis H profiled motion complete _BGH = 0
<b>8</b>	\$0100 (256)	\$D8 (216)	All axes profiled motion complete _BGI = 0
<b>9</b>	\$0200 (512)	\$C8 (200)	* Excess position error _TEn >= _ERn
<b>10</b>	\$0400 (1024)	\$C0 (192)	* Limit switch _LFn = 0 Must be profiling motion in direction of activated limit switch for interrupt to occur.
<b>11</b>	\$0800 (2048)	\$D9 (217)	Watchdog timer
<b>12</b>	\$1000 (4096)		Reserved
<b>13</b>	\$2000 (8192)	\$DB (219)	Application program stopped _XQn = -1
<b>14</b>	\$4000 (16384)	\$DA (218)	PC command done, colon response sent
<b>15</b>	\$8000 (32768)	\$E1-\$E8 (225-232)	* Digital input(s) 1-8 low (use n2 for mask)
UI, user interrupt command		\$F0-SFF (240-255)	User Interrupt, See UI command

DMC40x0, DMC41x3

*Interrupt Bits*

bit	n1=2^bit Hex (decimal)	Status Byte Hex (decimal)	Condition
<b>0</b>	\$0001 (1)	\$D0 (208)	Axis A profiled motion complete _BGA = 0
<b>1</b>	\$0002 (2)	\$D1 (209)	Axis B profiled motion complete _BGB = 0
<b>2</b>	\$0004 (4)	\$D2 (210)	Axis C profiled motion complete _BGC = 0
<b>3</b>	\$0008 (8)	\$D3 (211)	Axis D profiled motion complete _BGD = 0
<b>4</b>	\$0010 (16)	\$D4 (212)	Axis E profiled motion complete _BGE = 0
<b>5</b>	\$0020 (32)	\$D5 (213)	Axis F profiled motion complete _BGF = 0
<b>6</b>	\$0040 (64)	\$D6 (214)	Axis G profiled motion complete _BGG = 0
<b>7</b>	\$0080 (128)	\$D7 (215)	Axis H profiled motion complete _BGH = 0
<b>8</b>	\$0100 (256)	\$D8 (216)	All axes profiled motion complete
<b>9</b>	\$0200 (512)	\$C8 (200)	* Excess position error _TEn >= _ERn
<b>10</b>	\$0400 (1024)	\$C0 (192)	* Limit switch _LFn=0 / _LRn=0 Must be profiling motion in direction of activated limit switch for interrupt to occur.

11	\$0800 (2048)	\$D9 (217)	Reserved
12	\$1000 (4096)		Reserved
13	\$2000 (8192)	\$DB (219)	Application program stopped XQn = -1
14	\$4000 (16384)	\$DA (218)	Reserved
15	\$8000 (32768)	SE1-\$E8 (225-232)	* Digital input(s) 1-8 low (use n2 for mask)
	UI, user interrupt command	\$F0-\$FF (240-255)	User Interrupt, See UI command

## n2 Bit Mask

*Input Interrupts*

bit	n2 = 2^bit hex (decimal)	Status Byte hex (decimal)	Condition
0	\$01 (1)	\$E1 (225)	* Digital input 1 is low @IN[1] = 0
1	\$02 (2)	\$E2 (226)	* Digital input 2 is low @IN[2] = 0
2	\$04 (4)	\$E3 (227)	* Digital input 3 is low @IN[3] = 0
3	\$08 (8)	\$E4 (228)	* Digital input 4 is low @IN[4] = 0
4	\$10 (16)	\$E5 (229)	* Digital input 5 is low @IN[5] = 0
5	\$20 (32)	\$E6 (230)	* Digital input 6 is low @IN[6] = 0
6	\$40 (64)	\$E7 (231)	* Digital input 7 is low @IN[7] = 0
7	\$80 (128)	\$E8 (232)	* Digital input 8 is low @IN[8] = 0

DMC40x0, DMC41x3, DMC30010

## UDP Interrupts Framing

DMC40x0, DMC41x3, DMC30010

*The UDP packet can contain up to 16 individual status bytes and is framed as follows*

Format	Header (Fixed Byte)	Status Byte (1-16 bytes)	Payload Byte Count (0x03 - 0x12) [Includes header and footer in count]
Example	0x01	0xD0F1DBE1	0x06
Example Decoded	Interrupt Packet Indicator	Axis A Profiled Motion Complete; User Interrupt 1; Application Program Stopped; Digital Input 1 is low	6 bytes in payload

## EXAMPLES

```
'Interrupt when motion is complete on all axes OR if a limit switch is hit:  
'From the table, enable bits 8 and 10. n1 = 256 + 1024 = 1280  
EI 1280  
'  
'Interrupt when digital input 3 is low.  
'Enable bit 15 of n1 and bit 2 of n2.  
EI 32768, 4
```

EI is supported on DMC40x0, DMC41x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ELSE</b>	<i>Else function for use with IF conditional statement</i>	PROGRAMMING
<b>The ELSE command is an optional part of an IF conditional statement</b>		
		

ELSE | Only valid from within embedded code. No terminal.

## DESCRIPTION

The ELSE command is an optional part of an IF conditional statement. The ELSE command must occur after an IF command and it has no arguments. It allows for the execution of a command only when the argument of the IF command evaluates False. If the argument of the IF command evaluates false, the controller will skip commands until the ELSE command. If the argument for the IF command evaluates true, the controller will execute the commands between the IF and ELSE command.

## ARGUMENTS

### ELSE

ELSE is a command with no parameters

## REMARKS

- None

## EXAMPLES

```

IF (@IN[1]=0);'          IF conditional statement based on input 1
  IF (@IN[2]=0);'        2nd IF conditional statement executed if 1st IF conditional true
    MG "INI AND IN2 ARE ACTIVE";' Message to be executed if 2nd IF conditional is true
    ELSE ;'               ELSE command for 2nd IF conditional statement
      MG "ONLY IN1 IS ACTIVE";' Message to be executed if 2nd IF conditional is false
    ENDIF ;'              End of 2nd conditional statement
  ELSE ;'                ELSE command for 1st IF conditional statement
    IF (@IN[2]=0);'        3rd IF conditional statement executed if 1st IF conditional false
      MG "ONLY IN2 IS ACTIVE";' Message to be executed if 3rd IF conditional statement is true
    ELSE ;'               ELSE command for 3rd conditional statement
      MG "INI AND IN2 INACTIVE";' Message to be executed if 3rd IF conditional statement is false
    ENDIF ;'              End of 3rd conditional statement
  ENDIF ;'              End of 1st conditional statement

```

ELSE is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EM</b>	<i>Cam cycles (modulus)</i>	ECAM/GEARING
The EM command defines the change in position over one complete cycle of the master		

EMA= n	Arguments specified with an axis mask and an assignment (=)
EM n ...	Arguments specified with an implicit comma-separated order
EMm	Operand holds the value set in the command

## DESCRIPTION

The EM command defines the change in position over one complete cycle of the master.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

The field for the master axis is the cycle of the master position. For the slaves, the field defines the net change in one cycle.

## ARGUMENTS

**EMn= n**

**EM n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	1	8,388,607	N/A	1	Position change over one full ECAM cycle	For defining master axis
	0	2,147,483,647	N/A	1	Position change over one full ECAM cycle	For defining slave axis

## REMARKS

- If a slave will return to its original position at the end of the cycle, then n=0.
- If the change is negative, specify the absolute value for n.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
EA C; ' Select C axis as master for ECAM.
EM 0,3000,2000; ' Define the changes in A and B to be 0 and 3000 respectively. Define master cycle as 2000.
v = EM; ' Return cycle of A
```

EM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EN</b>	<i>End</i>	PROGRAMMING
The EN command is used to designate the end of a program or subroutine		
		

EN Only valid from within embedded code. No terminal.

## DESCRIPTION

The EN command is used to designate the end of a program or subroutine. If a subroutine was called by the JS command, the EN command ends the subroutine and returns program flow to the point just after the JS command.

DMC40x0, DMC41x3, DMC18x6, DMC30010

A return parameter can be specified to EN from a subroutine to return a value from the subroutine to the calling stack.

## ARGUMENTS

DMC40x0, DMC41x3, DMC18x6, DMC30010

**EN n0,n1,n2**

DMC40x0, DMC41x3, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	1	0	1	Specify trippoint status when returning from subroutine	n0=1 restores trippoints. n0=0 does not restore trippoints
<b>n1</b>	0	1	0	1	Set status of CI interrupt when returning from #COMINT	n1=1 restores CI interrupt. n1=0 does not restore CI interrupt
<b>n2</b>	-2,147,483,648	2,147,483,647	0	1	Return a value from a subroutine.	Accessible from the calling program with _JS. See JS for more information

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC30010

- The EN command is used to end the automatic subroutines #MCTIME #COMINT and #CMDERR.
  - Use the RE command to end the #POSERR and #LIMSWI subroutines.
  - Use the RI command to end the #ININT subroutine

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a;' Program A
PR 500;' Move A axis forward 500 counts
BG A;' Begin motion
AM A;' Pause the program until the A axis completes the motion
EN;' End of Program
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#example
'test program showing restoring trippoints with EN
XQ #err,1;' Execute thread to generate error
AI 1;' Wait for input 1 to trigger
MG "hello%;" After input, message out
EN

#err
'dummy thread that runs to cause an error
XX123;' Invalid command
'causes CMDERR to be called, interrupting thread 0
EN

#CMDERR
'error subroutine running on thread 0
tc=_TC;' Save error code
EN 1;' End routine, restore AI trippoint.
```

EN is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ENDIF</b>	<i>End of IF conditional statement</i>	PROGRAMMING
The ENDIF command is used to designate the end of an IF conditional statement		
		

ENDIF Only valid from within embedded code. No terminal.

## DESCRIPTION

The ENDIF command is used to designate the end of an IF conditional statement. An IF conditional statement is formed by the combination of an IF and ENDIF command. An ENDIF command must always be executed for every IF command that has been executed. It is recommended that the user not include jump commands inside IF conditional statements since this causes re-direction of command execution. In this case, the command interpreter may not execute an ENDIF command.

## ARGUMENTS

### ENDIF

ENDIF is a command with no parameters

## REMARKS

- None

## EXAMPLES

```

IF (@IN[1]=0);' IF conditional statement based on input 1
IF (@IN[2]=0);' 2nd IF conditional statement executed if 1st IF conditional true
  MG "INI AND IN2 ARE ACTIVE";' Message to be executed if 2nd IF conditional is true
  ELSE ;' ELSE command for 2nd IF conditional statement
    MG "ONLY IN1 IS ACTIVE";' Message to be executed if 2nd IF conditional is false
  ENDIF ;' End of 2nd conditional statement
  ELSE ;' ELSE command for 1st IF conditional statement
  IF (@IN[2]=0);' 3rd IF conditional statement executed if 1st IF conditional false
    MG "ONLY IN2 IS ACTIVE";' Message to be executed if 3rd IF conditional statement is true
    ELSE ;' ELSE command for 3rd conditional statement
      MG "INI AND IN2 INACTIVE";' Message to be executed if 3rd IF conditional statement is false
    ENDIF ;' End of 3rd conditional statement
  ENDIF ;' End of 1st conditional statement

```

ENDIF is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EO</b>	<i>Echo</i>	SYSTEM CONFIG
The EO command turns the echo on or off		

EO n ...	Arguments specified with an implicit comma-separated order
EO	Operand holds the value set in the command

## DESCRIPTION

The EO command turns the echo on or off. If the echo is off, characters input over the bus will not be echoed back.

## ARGUMENTS

### EO n

Argument	Value	Description	Notes
n	0	Echo Off	
	1	Echo On	Default

## REMARKS

- This command is defaulted to EO1. Galil software upon connection will set EO0

DMC40x0, DMC41x3, RIO, DMC21x3, DMC3001

- The EO command is accepted over the serial port only.
  - The ethernet port will not echo commands

## EXAMPLES

```
EO 0; ' Turns echo off
EO 1; ' Turns echo on
```

EO is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EP</b>	<i>Cam table master interval and phase shift</i>	<b>ECAM/GEARING</b>
<b>The EP command defines the ECAM table intervals and offset</b>		
		

EP n ...	Arguments specified with an implicit comma-separated order
<u>EP</u>	Operand holds the value set in the command

## DESCRIPTION

The EP command defines the ECAM table intervals and offset. The offset is the master position of the first ECAM table entry. The interval is the difference of the master position between 2 consecutive table entries. This command effectively defines the size of the ECAM table. Up to 257 points may be specified.

## ARGUMENTS

### EP n0,n1

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	1	32,767	256	1	Master position interval	Cannot be changed while ECAM is running
<b>n1</b>	-2,147,483,648	2,147,483,647	0	1	ECAM table phase shift	Can be modified during ECAM

## REMARKS

- EP contains the value of the interval n0.
- The offset parameter 'n1' can also be used to instantaneously phase shift the graph of the slave position versus the master position. This can be used to make on-the-fly corrections to the slaves.
  - See application note #2502 for more details. <http://www.galilmc.com/support/application-notes.php>

## EXAMPLES

```
EP 20;          Sets the cam master points to 0,20,40 . . .
d = _EP;        Set the variable d equal to the ECAM internal master interval
EP ,100;       Phase shift all slaves by 100 master counts
```

EP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EQ</b>	<i>ECAM quit (disengage)</i>	ECAM/GEARING
The EQ command disengages an electronic cam slave axis at the specified master position		

EQA= n	Arguments specified with an axis mask and an assignment (=)
EQ n ...	Arguments specified with an implicit comma-separated order
EQm	Operand has special meaning see Remarks

## DESCRIPTION

The EQ command disengages an electronic cam slave axis at the specified master position. Separate points can be specified for each axis. If a value is specified outside of the master's range, the slave will disengage immediately.

## ARGUMENTS

**EQm= n**

**EQ n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	- 2,147,483,648	2,147,483,647	N/A	1	Master position to disengage the slave axis specified.	If n = outside of master position range, disengage slave axis immediately.

## REMARKS

- EQn contains 1 if engage command issued and axis is waiting to engage, 2 if disengage command issued and axis is waiting to disengage, and 0 if ECAM engaged or disengaged.
- n = ? Returns 1 if engage command issued and axis is waiting to engage, 2 if disengage command issued and axis is waiting to disengage, and 0 if ECAM engaged or disengaged.
- This command is not a trippoint. This command will not hold the execution of the program flow.
  - If the execution needs to be held until master position is reached, use MF or MR command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
EQ 300,700; ' Disengages the A and B motors at master positions 300 and 700 respectively.
```

```
EQ 300; ' Disengages the A motor at master position 300.
```

EQ is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ER</b>	Error Limit	ERROR CONTROL
The ER command sets the magnitude of the position errors for each axis that will trigger an error condition		



ERA= n	Arguments specified with an axis mask and an assignment (=)
ERn ...	Arguments specified with an implicit comma-separated order
ERm	Operand holds the value set in the command

## DESCRIPTION

The ER command sets the magnitude of the position errors for each axis that will trigger an error condition. When the limit is exceeded, the Error output will go low (true) and the controller's red light will be turned on. If the Off On Error (OE1) command is active, the motors will be disabled.

## ARGUMENTS

**ERm= n**

**ER n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-1	2147483647	16384	1	Set the position error limit in counts	n=0 enables Error output. n=-1 disables Error output.

## REMARKS

- The error limit specified by ER should be high enough as not to be reached during normal operation.
  - Examples of exceeding the error limit would be a mechanical jam, or a fault in a system component such as encoder or amplifier
- For debugging purposes, ER0 and ER-1 can be used to turn the red LED on and off.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:ER 200,300,400,600;' Set the A-axis error limit to 200, the B-axis error limit to 300, the C-axis error limit to 400, and the D-axis error limit to 600.
:ER ,1000;' Sets the B-axis error limit to 1000, leave the A-axis error limit unchanged.
:ER ?,?,?;' Return A,B,C and D values
00200,00100,00400,00600
:ER ?;' Return A value
00200
:v1=_ERA;' Assigns V1 value of ERA
:v1=';' Returns V1
00200
```

ER is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ES</b>	<i>Ellipse Scale</i>	VECTOR/LINEAR
<b>The ES command divides the resolution of one of the axes in a vector mode (VM)</b>		
		

ESn ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The ES command divides the resolution of one of the axes in a vector mode (VM). This function allows for the generation of circular motion when encoder resolutions differ. It also allows for the generation of an ellipse instead of a circle. The resolution change applies for the purpose of generating the VP and CR commands, effectively changing the axis with the higher resolution to match the coarser resolution.

## ARGUMENTS

### ES n0,n1

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	1	65535	1	1	First value used for resolution scaling	See Remarks for usage
<b>n1</b>	1	65535	1	1	Second value used for resolution scaling	See Remarks for usage

## REMARKS

- For VM xy
  - When n0 > n1, the resolution of x will be multiplied by n0/n1
  - When n0 < n1, the resolution of y will be multiplied by n1/n0

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The ES command will apply to the selected coordinate system, S or T. To select the coordinate system, use the command CAS or CAT.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
VM AB;ES 3,4;' Divide B resolution by 4/3
VM CA;ES 2,3;' Divide A resolution by 3/2
VM AC;ES 3,2;' Divide A Resolution by 3/2
'Note: ES must be issued after VM.
```

```
VM AN;ES 3,2;' Divide A Resolution by 3/2
'Note: ES must be issued after VM.
```

ES is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)

Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ET</b>	<i>Electronic cam table</i>	ECAM/GEARING
The ET command sets the ECAM table entries for the slave axes		

ET n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The ET command sets the ECAM table entries for the slave axes.. The values of the master axes are not required. The slave entry (n) is the position of the slave axes when the master is at the point ( $m_i + o$ , where  $i$  is the interval and  $o$  is the offset as determined by the EP command.

## ARGUMENTS

**ET n,n,n,n,n,n,n**

**ET[n0] = n,n,n,n,n,n,n**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	256	N/A	1	Index of the ECAM table entry	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	Position of the slave axis at the specified table point.	

## REMARKS

- [n0] can be omitted only if EC has initialized the index count. In this case, each ET command will increment the index counter by 1.
- n=? Returns the slave position for the specified point.

## EXAMPLES

<code>ET[0]= 0,,0;</code>	Specifies the position of the slave axes A and C to be synchronized with the starting point of the master.
<code>ET[1]= 1200,,400;</code>	Specifies the position of the slave axes A and C to be synchronized with the second point of the master
<code>EC 0;'</code>	Set the table index value to 0, the first element in the table
<code>ET 0,,0;</code>	Specifies the position of the slave axes A and C to be synchronized with the starting point of the master.
<code>ET 1200,,400;</code>	Specifies the position of the slave axes A and C to be synchronized with the second point of the master

ET is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)

Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EW</b>	<i>ECAM Widen Segment</i>	ECAM/GEARING
The EW command allows widening the length of one or two ECAM segments beyond the width specified by EP.		

EW0, EW1, EW2, EW3, Operand has special meaning see Remarks

## DESCRIPTION

The EW command allows widening the length of one or two ECAM segments beyond the width specified by EP. For ECAM tables with one or two long linear sections, this allows placing more points in the curved sections of the table. There are only two widened segments, and if used they are common for all ECAM axes.

## ARGUMENTS

**EW n0=n1,n2=n3**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	1	255	-1	1	Index of first widened segment	If n0 = -1, no segment is widened
<b>n1</b>	1	2147483647	0	1	Length of first widened segment	In master counts
<b>n2</b>	3	255	-1	1	Index of second widened segment	If n2 = -1, no segment is widened. n2 must be > n0
<b>n3</b>	1	2147483647	0	1	Length of second widened segment	In master counts

## REMARKS

- Remember that the widened segment lengths must be taken into account when determining the modulus (EM) for the master.
- The second widened segment cannot be used unless the first widened segment is also being used.
- The segments chosen should not be the first or last segments, or consecutive segments.

## Operand Usage

- \_EW0 contains n0, the index of the first widened segment.
- \_EW1 contains n1, the length of the first widened segment.
- \_EW2 contains n2, the index of the second widened segment
- \_EW3 contains n3, the length of the second widened segment.

## EXAMPLES

<code>ew 41= 688;</code>	Widen segment 41 to 688 master counts
<code>ew 41= 688, 124=688;</code>	Widen segments 41 and 124 to 688 master counts

EW is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMCI8x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>EY</b>	<i>ECAM Cycle Count</i>	<b>ECAM/GEARING</b>
The EY command sets or gets the ECAM cycle count		

EY n ...	Arguments specified with an implicit comma-separated order
_EY	Operand holds the value set in the command

## DESCRIPTION

The EY command sets or gets the ECAM cycle count. This is the number of times that the ECAM axes have exceeded their modulus as defined by the EM command. EY will increment by one each time the master exceeds its modulus in the positive direction, and EY will decrement by one each time the master exceeds its modulus in the negative direction.

## ARGUMENTS

### EY n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-2,147,483,648	2,147,483,647	0	1	Current ECAM cycle count	

## REMARKS

- \_EY returns the current cycle count
- EY can be used to calculate the absolute position of an axis with the following equation:
  - Absolute position = EY \* EM + TP

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
MG EY * EMB + TPB; ! print absolute position of master (Y)
```

EY is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>FA</b>	<i>Acceleration Feedforward</i>	FILTER/CONTROL
<b>The FA command sets the acceleration feedforward coefficient</b>		

FAA=n	Arguments specified with an axis mask and an assignment (=)
FA n ...	Arguments specified with an implicit comma-separated order
_FAm	Operand holds the value set in the command

## DESCRIPTION

The FA command sets the acceleration feedforward coefficient. This coefficient is scaled by the set acceleration and adds a torque bias voltage during the acceleration phase and subtracts the bias during the deceleration phase of a motion.

## ARGUMENTS

**FAm=n**

**FA n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	8,191	0	1/4	Value of proportional term	

## REMARKS

- The Feedforward Bias product is limited to 10 Volts.
- If the feedforward coefficient is changed during a move, then the change will not take effect until the next move.
- FA operates on PA, PR, IP, JG and PVT mode.
- FA does not operate in:
  - Contour Mode (CM)
  - Axis is Gearing or ECAM slave
  - Coordinated motion (LM, VM)

DMC40x0, DMC41x3, DMC18x6, DMC30010

- Acceleration Feedforward Bias = FA \* AC \* (1.5 10-7) \* ((TM/1000)^2)
- Deceleration Feedforward Bias = FA \* DC \* (1.5 10-7) \* ((TM/1000)^2)
- FA is enabled during the PVT mode of motion.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Set feedforward coefficient to 10 for the A-axis
'and 15 for the B-axis. The effective bias will
'be 0.75V for A and 2.25V for B.
```

```
:AC 500000,1000000
:FA 10,15
:MG _FAA,_FAB
10 15
```

FA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>FE</b>	Find Edge	INDEPENDENT MOTION
The FE command moves a motor until a transition is seen on the homing input for that axis		

FE A Argument is an axis mask

## DESCRIPTION

The FE command moves a motor until a transition is seen on the homing input for that axis. The direction of motion depends on the initial state of the homing input (use the CN command to configure the polarity of the home input). Once the transition is detected, the motor decelerates to a stop. This command is useful for creating your own homing sequences.

## ARGUMENTS

### FE mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axes to Find Edge	

## REMARKS

- Find Edge only searches for a change in state on the Home Input. Use FI (Find Index) to search for the encoder index. Use HM (Home) to search for both the Home input and the Index.
- Remember to specify BG after each of these commands
- Speed of Find Edge is set with the SP command and should be low enough to allow for a minimum of a 2 sample period pulse width on the home signal. With TM 1000, the pulse width must be at least 2ms.

## EXAMPLES

```
:FE ;' Set find edge mode
:BG ;' Begin find edge
:FE A; ' Only find edge on A
:BG A
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:FE B; ' Only find edge on B
:BG B
:FE CD; ' Find edge on C and D
:BG CD
```

FE is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>FI</b>	Find Index	INDEPENDENT MOTION
The FI and BG commands move the motor until an encoder index pulse is detected		
		

FI A Argument is an axis mask

## DESCRIPTION

The FI and BG commands move the motor until an encoder index pulse is detected.

## ARGUMENTS

### FI mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axes to Find Index	

## REMARKS

DMC40x0, DMC41x3, DMC18x6, DMC30010

- The controller looks for a transition from low to high. There are 2 stages to the FI command. The first stage jogs the motor at the speed and direction of the JG command until a transition is detected on the index line. When the transition is detected, the position is latched and the motor will decelerate to a stop. In the second stage, the motor will reverse direction and move to the latched position of the index pulse at the speed set by the HV command. At the conclusion of FI, the position is defined as zero.
- Find Index only searches for a change in state on the Index. Use FE to search for the Home. Use HM (Home) to search for both the Home input and the Index. Remember to specify BG after each of these commands.

## EXAMPLES

```
#home;          Home Routine
JG 500;         Set speed and forward direction
FI A;           Find index
BG A;           Begin motion
AM A;           After motion
MG "FOUND INDEX"; Print message
EN
```

FI is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>FL</b>	Forward Software Limit	ERROR CONTROL
The FL command sets the forward software position limit		
		

FLA= n	Arguments specified with an axis mask and an assignment (=)
FL n ...	Arguments specified with an implicit comma-separated order
FLm	Operand has special meaning, see Remarks

## DESCRIPTION

The FL command sets the forward software position limit. If this limit is exceeded during motion, motion on that axis will decelerate to a stop. Forward motion beyond this limit is not permitted.

## ARGUMENTS

**FLm= n**

**FL n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	2,147,483,647	1	Value of software forward limit	2147483647 turns off forward limit

## REMARKS

- The forward limit is activated at n+1. n = 2147483647 effectively disables the forward soft limit.
- The units are in quadrature counts.
- When the forward software limit is activated, the automatic subroutine #LIMSWI will be executed if it is included in the program.

## EXAMPLES

```
#test; ' Test Program
AC 1000000; ' Acceleration Rate
DC 1000000; ' Deceleration Rate
FL 15000; ' Forward Limit
JG 5000; ' Jog Forward
BG A; ' Begin
AM A; ' After Limit
RP A; ' Tell Position
EN; ' End

'Hint: Galil controllers also provide hardware limits.
```

FL is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>FV</b>	Velocity Feedforward	Filter/Control
The FV command sets the velocity feedforward coefficient		

FVA=n	Arguments specified with an axis mask and an assignment (=)
FV n ...	Arguments specified with an implicit comma-separated order
_FVm	Operand holds the value set in the command

## DESCRIPTION

The FV command sets the velocity feedforward coefficient. This coefficient generates an output bias signal in proportion to the sample to sample change in reference position (RP).

## ARGUMENTS

**FVm= n**

**FV n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	8,191	0	1	Value of proportional term	

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010

- Velocity feedforward bias =  $FV * (\text{Velocity [cts/s]} * (1.22 \cdot 10^{-6}) * (TM/1000))$ 
  - With FVA=10, TM 1000 and the velocity is 200,000 count/s, the velocity feedforward bias equals 2.44 volts

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Set feedforward coefficients to 10 and 20 for A and B respectively.  
'This effective bias will be 0.366 volts for A and 1.95 volts for B.
```

```
:FV 10,20  
:JG 30000,80000  
:MG _FVA,_FVB  
10 20
```

FV is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>GA</b>	Master Axis for Gearing	ECAM/GEARING
The GA command specifies the master axes for electronic gearing		

GAA=n	Arguments specified with an axis mask and an assignment (=)
GA n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The GA command specifies the master axes for electronic gearing. Multiple masters for gearing may be specified. A slave axis may have only one master. The masters may be the main encoder input, auxiliary encoder input, or the commanded position of any axis. The master may also be the commanded vector move in a coordinated motion of LM or VM type. When the master is a simple axis, it may move in any direction and the slave follows. When the master is a commanded vector move, the vector move is considered positive and the slave will move forward if the gear ratio is positive, and backward if the gear ratio is negative. The slave axes and ratios are specified with the GR command and gearing is turned off by the command GR0.

## ARGUMENTS

**GA**m0=m

**GA** m,m,m,m,m,m,m,m

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m0</b>	A	H	N/A	Axis	Slave axis to assign master	<b>m0&gt;m</b>
<b>m</b>	A	H	N/A	Axis	Master axis main encoder as the slave's master	
	CA	CH	N/A	Axis	Master axis commanded position as the slave's master	Valid arguments: CA,CB,CC,CD,CE,CF,CG,CH
	DA	DH	N/A	Axis	Master axis aux encoder as the slave's master	Valid arguments: DA,DB,DC,DD,DE,DF,DG,DH
S	T	N/A	Axis		Vector plane as the slave's master	
M	N	N/A	Axis		Virtual axis as the slave's master	

## REMARKS

- m=? returns the GA setting
- When the geared motors must be coupled "strongly" to the master, use the gantry mode GM.
- When gearing is used in a gantry application, gearing off of the commanded position is recommended.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
REM setup gearing where B axis is master for A and C axes.
#gear
MO B; ' Turn off servo to B motor
GA b,,b; ' Specify master axis as B on A and C
GR .25,, -5; ' Specify A and C gear ratios
SH B; ' Enable B axis
PRB= 1000;BG B; ' Move B axis 1000 counts
' A axis will be commanded to move 250 counts positive
' C axis will be commanded to move -5000 counts
EN; ' End program
```

```
REM imaginary axis example
#imag
GAA= n; ' set the imaginary N axis as the master of the A axis
GRA= 2.5; ' set the gear ratio for the A axis as 2.5
PRN= 1000;BG N; ' Move N axis 1000 counts
' (C axis will be commanded to move 2500 counts positive)
EN; ' End Program
```

GA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>GD</b>	<i>Gear Distance</i>	ECAM/GEARING
The GD command sets the distance of the master axis over which the specified slave will be engaged, disengaged or changed to a new gear setting		

GD n ...	Arguments specified with an implicit comma-separated order
_Gdm	Operand holds the value set in the command

## DESCRIPTION

The GD command sets the distance of the master axis over which the specified slave will be engaged, disengaged or changed to a new gear setting.

## ARGUMENTS

**GDm= n**

**GD n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	32,767	0	1	Absolute Value of Gearing Distance	0 engages gearing instantly

## REMARKS

- The distance is entered as an absolute value, the motion of the master may be in either direction.
- If the distance is set to 0, then the gearing will engage instantly.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#a
GA ,a;' Sets the A axis as the gearing master for the B axis
GD ,5000;' Set distance over which gearing is engaged to 5000 counts of the master axis.
JG 5000;' Set the A axis jog speed to 5000 cts/sec
BG A;' Begin motion on the A axis
AS A;' Wait until A axis reaches the set speed of 5000 counts/sec
GR ,1;' Engage gearing on the B axis with a ratio of 1:1, the
'distance to fully engage gearing will be 5000 counts of the master axis
WT 1000;' Wait 1 second
GR ,3;' Set the gear ratio to three. The ratio will be changed
'over the distance set by the GD command
WT 1000;' Wait 1 second
GR ,0;' Disengage the gearing between the B axis slave and the
'master. The gearing will be disengaged over the number of
'counts of the master specified with the GD command above
EN;' End program
```

```
#a
GA da;' Set the aux encoder input as the gearing master
GD 5000;' Set distance over which gearing is engaged to 5000 counts of the master axis.
GR 1;' Set a gear ratio of 1:1, the distance to fully
'engage gearing will be 5000 counts of the master axis
WT 1000;' Wait 1 second
GR 3;' Set the gear ratio to three. The ratio will be changed
'over the distance set by the GD command
WT 1000;' Wait 1 second
GR 0;' Disengage the gearing between the axis aux encoder
'The gearing will be disengaged over the number of
'counts of the master specified with the GD command above
EN;' End program
```

GD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>GM</b>	Gantry mode	ECAM/GEARING
-----------	-------------	--------------

The GM command specifies the axes in which the gearing function is performed in the Gantry mode



GMA=n	Arguments specified with an axis mask and an assignment (=)
GM n...	Arguments specified with an implicit comma-separated order
GMm	Operand holds the value set in the command

## DESCRIPTION

The GM command specifies the axes in which the gearing function is performed in the Gantry mode. In this mode, the geared slaves will not be stopped by the ST command or by limit switches.

## ARGUMENTS

**GMm= n**

**GM n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	Value of GM command	1 Enables Gantry Mode, 0 disables Gantry Mode

## REMARKS

- The GM command is useful for driving heavy load on both sides with two motors (Gantry Style)
- Only setting Gantry Mode of the slave to 0 (GMm= 0) will disable Gantry Mode

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
GM 1,1,1,1;          Enable GM on all axes
GM 0;                Disable GM on A-axis, other axes remain unchanged
GM ,1,1;              Enable GM on C-axis and D-axis, other axes remain unchanged
GM 1,0,1,0;           Enable GM on A and C-axis, disable GM on B and D axis
```

```
GA da;' Set master for A axis to the A axis Aux encoder input
GM 1;          Enable Gantry Mode on A axis
GR 1;          Set Gear Ratio to 1
WT 1000
ST ;'          Axis will still be in gearing Mode
WT 1000
GM 0;          Disable Gantry Mode (Axis still gearing)
WT 1000
ST ;'          Will clear gearing mode
EN
```

GM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>GR</b>	Gear Ratio	ECAM/GEARING
GR specifies the Gear Ratios for the geared axes in the electronic gearing mode		

GRA= n	Arguments specified with an axis mask and an assignment (=)
GR n ...	Arguments specified with an implicit comma-separated order
GRm	Operand holds the value set in the command

## DESCRIPTION

GR specifies the Gear Ratios for the geared axes in the electronic gearing mode. The master axis is defined by the GA command.

## ARGUMENTS

**GRm= n**

**GR n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Slave axis to assign gear ratio	
<b>n</b>	-127	127	0	1/65,536	Value of Gear Ratio of Slave	n = 0 disables gearing

## REMARKS

- The gear ratio may be different for each geared axis.
- The master can go in both directions.
- When the geared motors must be coupled "strongly" to the master, use the gantry mode GM.
- Unless the GM command is set to 1, gearing is disabled in the following conditions:
  - The gear ratio is set to 0
  - A limit switch is reached
  - The axis is commanded to stop with the ST command

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
REM setup gearing where B axis is master for A and C axes.
#gear
MO B;' Turn off servo to B motor
GA b,,b; ' Specify master axis as B
GR .25,, -5;' Specify A and C gear ratios
SH B;' Enable B axis
PRB= 1000;BG B;' Move B axis 1000 counts
' A axis will be commanded to move 250 counts positive
' C axis will be commanded to move 5000 counts negative (-5000)
EN; ' End program
```

REM setup gearing where B axis is master for A and C axes.

```
#gear
GA n; ' Specify master axis as N (imaginary Axis)
GR -2;' Specify gear ratio or -2
PRN= 1000;BG N;' Move N axis 1000 counts
WT 1000
MG _RPA,_RPN;' will indicate -2000 on A and 1000 on N
EN; ' End program

:'execution of gearing example
:XQ
:
:
-2000.0000 1000.0000
:
```

GR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>HM</b>	<a href="#">Home</a>	<a href="#">INDEPENDENT MOTION</a>
The HM command performs a three stage homing sequence for servo systems and a two stage sequence for stepper motors		

<b>HM A</b>	Argument is an axis mask
<b>_HMm</b>	Operand has special meaning see Remarks

## DESCRIPTION

The HM command performs a three stage homing sequence for servo systems and a two stage sequence for stepper motors.

## ARGUMENTS

### HM mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axis to performing Homing Routine	No argument homes all axes

## REMARKS

- The FE command is derived of FE and FI commands and therefore you can create your own custom homing sequence by using the FE (Find Edge) and FI (Find Index) commands.
- The sequence of FE and FI commands varies depending upon if the axis is configured for a stepper or servo

### Step One. Servos and Steppers

- During the first stage of the homing sequence, the motor moves at the user-programmed speed until detecting a transition on the homing input for that axis. The speed for step one is set with the SP command.
- The direction for this first stage is determined by the initial state of the homing input. The state of the homing input can be configured using the second field of the CN command.
- Once the homing input changes state, the motor decelerates to a stop.

### Step Two. Servos and Steppers

DMC40x0, DMC41x3, DMC18x6, DMC30010

- At the second stage, the motor changes directions and approaches the transition again at the speed set with the HV command. When the transition is detected, the motor is stopped instantaneously.

### Step Three. Servos only

DMC40x0, DMC41x3, DMC18x6, DMC30010

- At the third stage, the motor moves forward at the speed set with the HV command until it detects an index pulse via latch from the encoder. It returns to the latched position and defines it as position 0.

## Operand

*HMn state as a function of CN,n and Home digital input*

CN1 value	Home input digital state	HMn state	Direction of travel if HM begun in this state
-1	1 (pull-up or non-active opto)	1	Backward
-1	0 (grounded or active opto)	0	Forward
1	1 (pull-up or non-active opto)	0	Forward
1	0 (grounded or active opto)	1	Backward

## EXAMPLES

```
:HM ;'      Set Homing Mode for all axes
:BG ;'      Home all axes
:HM A;'      Set Homing Mode for axis A
:BG A;'      Home only the A-axis
```

HM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>HS</b>	Handle Assignment Switch	ETHERNET
<b>The HS command is used to switch the ethernet handle assignments between two handles</b>		

HSA= n | Arguments specified with an axis mask and an assignment (=)

## DESCRIPTION

The HS command is used to switch the ethernet handle assignments between two handles. Handles are opened when a connection is established by an external client (TCP or UDP), or when a handle is assigned explicitly with the IH command. Should those assignments need modifications, the HS command allows the handles to be reassigned.

## ARGUMENTS

**HSm= m0**

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	First handle to switch	
	S	S	N/A	Handle	First handle to switch	S = current handle sending command. Not valid in program
<b>m0</b>	A	H	N/A	Handle	Second handle to switch	
	S	S	N/A	Handle	Second handle to switch	S = current handle sending command. Not valid in program

## REMARKS

- A handle encapsulates the following 4 pieces of information:
  - 1. Local IP address (same for all handles)
  - 2. Remote IP address
  - 3. Local Port
  - 4. Remote Port
- Handles are used as a pointer to the network socket in commands such as SAh, MBh, {Eh}, and IHh where h is the handle letter

## EXAMPLES

```
:HSC=d;' Connection for handle C is assigned to handle D. Connection for handle D is assigned to handle C.  
:HSS=e;' Executing handle connection is assigned to handle E. Connection for handle E is assigned to executing handle.
```

HS is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>HV</b>	<i>Homing Velocity</i>	INDEPENDENT MOTION
Sets the slew speed for the FI final move to the index and all but the first stage of HM		

HVA= n	Arguments specified with an axis mask and an assignment (=)
HV n ...	Arguments specified with an implicit comma-separated order
HVm	Operand holds the value set in the command

## DESCRIPTION

Sets the slew speed for the FI final move to the index and all but the first stage of HM.

## ARGUMENTS

**HVm= n**

**HV n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	15,000,000	256	2	Value of Homing Velocity in cnts/second	For MT settings of 1,-1,1.5 and -1.5 (Servos)
	0	3,000,000	256	2	Value of Homing Velocity in cnts/second	For MT settings of 2,-2,2.5 and -2.5 (Steppers)

## REMARKS

- None

## EXAMPLES

```
HVA= 1000; ' set homing speed
HM A; ' home to home switch then index
BG A; ' begin motion
AM A; ' wait for motion complete
EN; ' end program
```

HV is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>HX</b>	<i>Halt Execution</i>	PROGRAMMING
The HX command halts the execution of any program that is running		
		

HX n ...	Arguments specified with an implicit comma-separated order
_HXm	Operand has special meaning, see Remarks

## DESCRIPTION

The HX command halts the execution of any program that is running. The parameter n specifies the thread to be halted.

## ARGUMENTS

### HX n

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	7	0	1	Thread number to halt	If n omitted, default value used.

## REMARKS

- When used as an operand, \_HXn contains the running status of thread n with:
  - 0 Thread not running
  - 1 Thread is running
  - 2 Thread has stopped at tripoint

## EXAMPLES

```
XQ #a; Execute program #A, thread zero
XQ #b,3; Execute program #B, thread three
HX 0; Halt thread zero
HX 3; Halt thread three
```

HX is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IA</b>	<i>IP Address</i>	ETHERNET
<b>The IA command assigns the controller IP address and the TCP time out</b>		
IA n ...	Arguments specified with an implicit comma-separated order	
_IA0, _IA1, _IA2, _IA3, _IA4, _IA5	Operand has special meaning see Remarks	

## DESCRIPTION

The IA command assigns the controller IP address and the TCP time out. The IP address can also be assigned via Galil software or from an external server.

DMC40x0, DMC41x3, RIO, DMC30010

The controller defaults to DHCP and will receive an IP address from a DHCP server if present. To manually set an IP address over the serial connection, send DH0 to disable DHCP prior to setting the new IP address with IA.

GalilTools and GalilSuite software packages feature a DHCP/BOOTP capability to assign the IP address to the controller. Please refer to the user manuals for those products for more information.

## ARGUMENTS

**IA n0, n1, n2, n3**

**IA n4**

**IA < o**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	255	0	1	Byte 3 of the IP address	
<b>n1</b>	0	255	0	1	Byte 2 of the IP address	
<b>n2</b>	0	255	0	1	Byte 1 of the IP address	
<b>n3</b>	0	255	0	1	Byte 0 of the IP address	
<b>n4</b>	-2,147,483,648	2,147,483,647	0	1	The full IP address specified as a signed 32 bit two's complement integer	
<b>o</b>	1	2,147,483,647	250	1	The time in update samples between TCP retries	Up to 5 retries occur

## REMARKS

- When specifying the IP address with IA, remember to use commas as delimiters instead of periods
- n4 = ? will return the IP address of the controller in comma separated format
- Setting the IP address over Ethernet to a new value will cause an immediate disconnect/timeout. Reconnect to the controller on the new IP address and issue a BN to save the new value to flash

DMC40x0, DMC41x3, RIO, DMC30010

- To change the IP address manually over Ethernet on a controller which was initially assigned via DHCP, send "DH 0;IA n0,n1,n2,n3" as one command line. Reconnect on the new IP and issue BN to save

## Operands

- \_IA0 contains the IP address representing a 32 bit signed number (Two's complement). See the example below.
- \_IA1 contains the value for o (retry time)
- \_IA2 contains the number of available handles
- \_IA3 contains the number of the handle using this operand where the number is 0 to 7. 0 represents handle A, 1 handle B, etc. This is used by a remote device to detect its outgoing handle.
- \_IA4 contains the number of the handle that lost communication last, contains a -1 on reset to indicate no handles lost

DMC40x0, DMC41x3, RIO, DMC30010

- \_IA5 returns autonegotiation Ethernet speed. Returns 10 for 10-Base T and returns 100 for 100-Base T, it will return -1 if there is no physical link

## EXAMPLES

```
IA 151,12,53,89; Assigns the controller with the address 151.12.53.89
IA 2534159705; Assigns the controller with the address 151.12.53.89
IA < 500; Sets the timeout value to 500 msec
```

```
REM The individual IP address bytes can be derived within imbedded code using _IA0
a= @INT[( _IA0&($FF000000))/$1000000]&$FF
b= @INT[( _IA0&($00FF0000))/$10000]
c= @INT[( _IA0&($0000FF00))/$100]
d= @INT[( _IA0&($000000FF))]
REM IP address = a.b.c.d
```

IA is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ID</b>	<i>Identify</i>	<b>INTERROGATION</b>
<b>The ID command is used to query the controller for the accessories that are attached</b>		
		

ID Command takes no arguments

## DESCRIPTION

The ID command is used to query the controller for the accessories that are attached.

DMC41x3

The following is an example response to the ID command and a description of each line.

## ARGUMENTS

### ID

ID is a command with no arguments

## REMARKS

- The following are example responses to the ID command and a description of each line.

### Example ID

DMC41x3

```
'This ID response is applicable to the following part number:  
'DMC-4183-BOX8-D3040-D3040
```

```
:ID  
DMC4103 rev 1  
Servo Amplifier Board AMP-43040 500 watt rev 1  
Servo Amplifier Board AMP-43040 500 watt rev 1  
:
```

## Description of response

DMC41x3

DMC4103 rev [number]

[amp1]

[amp2]

where

[number] - indicates the main board CPLD revision

[amp1] - indicates the amplifier configuration, if equipped, for axes ABCD

[amp2] - indicates the amplifier configuration, if equipped, for axes EFGH

Note that the rev number at the end of each line of the ID command indicates the hardware revision of that board.

Newer board revisions will have a higher revision value.

## EXAMPLES

DMC41x3

```
:ID  
DMC4103 rev 1  
Stepper Amplifier Board AMP-44140 rev 0  
Stepper Amplifier Board AMP-44140 rev 0  
:
```

ID is supported on DMC40x0, DMC41x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IF</b>	<i>IF conditional statement</i>	PROGRAMMING
The IF command is used in conjunction with an ENDIF command to form an IF conditional statement		
		

<b>IF</b>	Only valid from within embedded code. No terminal.
-----------	--

## DESCRIPTION

The IF command is used in conjunction with an ENDIF command to form an IF conditional statement. The arguments consist of one or more conditional statements and each condition must be enclosed with parenthesis (). If the conditional statement(s) evaluates true, the command interpreter will continue executing commands which follow the IF command. If the conditional statement evaluates false, the controller will ignore commands until the associated ENDIF command or an ELSE command occurs in the program.

## ARGUMENTS

### IF (ex)

Argument	Min	Max	Default	Resolution	Description	Notes
<b>ex</b>	N/A	N/A	N/A	Expression	Conditional statement for IF statement	See Remarks

## REMARKS

- Conditions are tested with the following logical operators:
  - < less than or equal to
  - > greater than
  - = equal to
  - <= less than or equal to
  - >= greater than or equal to
  - $\neq$  not equal
- Bit wise operators | and & can be used to evaluate multiple conditions.
- A true condition = 1 and an false condition = 0.
- Each condition must be placed in parenthesis for proper evaluation by the controller.

```
IF ((var0=1) & (var1=2));' valid IF statement
IF var0=1&var1=2;'      invalid IF statement
IF (var0=1&var1=2);'    invalid IF statement
```

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a
IF ( _TEA<1000);' IF conditional statement based on a motor position
  MG "Motor is within 1000 counts of zero";' Message to be executed for true
ENDIF ;'           End of IF conditional statement
EN; '             End Program
```

```
#input
IF (@IN[1]=0);'   IF conditional statement based on input 1
  MG "Input 1 is Low";' Message to be executed if "IF" statement is true
ENDIF ;'           End of IF conditional statement
EN
```

```
#var
v1= @AN[1]*5;'      some calculation for variable v1
IF ((v1>25)&(@IN[4]=1));' Conditions based on V1 variable and input 4 status
  MG "Conditions met";' Message to be executed if "IF" statement is true
ENDIF ;'           End of IF statement
EN
```

REM The conditions of an if statement can be simplified with the fact that  
 REM a true condition = 1 and a false condition = 0.

```
#true
v1= 1
IF (v1)
  MG "True v1=",v1
ENDIF
#false
v1= 0
IF (v1)
  'if statement evaluates false
ELSE
  MG "False v1=",0
ENDIF
EN
```

IF is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IH</b>	<i>Open IP Handle</i>	ETHERNET
The IH command is used when the controller is operated as a master (client) to open a handle and connect to a slave (server)		
		

IHA=n	Arguments specified with an axis mask and an assignment (=)
<u>IHm0</u> , <u>IHm1</u> , <u>IHm2</u> , <u>IHm3</u> , <u>IHm4</u>	Operand has special meaning, see Remarks

## DESCRIPTION

The IH command is used when the controller is operated as a master (client) to open a handle and connect to a slave (server).

## ARGUMENTS

**IHm= n0,n1,n2,n3 <o>p**

**IHm= n <o>p**

**IHm=>p**

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to assign connection	
	S	T	N/A	Handle	Special handle designator used when closing handles	See Remarks
<b>n0</b>	0	255	0	1	Byte 3 of the IP address	
<b>n1</b>	0	255	0	1	Byte 2 of the IP address	
<b>n2</b>	0	255	0	1	Byte 1 of the IP address	
<b>n3</b>	0	255	0	1	Byte 0 of the IP address	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	IP address in its 32 bit value	
<b>o</b>	0	65,535	see Notes	1	Specify the port to connect over	If o is omitted, the controller selects the port starting at 1000
<b>p</b>	1	2	2	1	Specify the connection type to open	n = 2 is TCP. n = 1 is UDP.
	-3	-1	N/A	1	Specify the connection type to close when closing a handle	See Remarks

## REMARKS

- All 4 bytes must be assigned for an IP address to be valid.
- IHm=? returns the IP address as 4 1-byte numbers
- Use the following equation to change the 4 byte IP (n0,n1,n2,n3) to a single 32 bit number, n
  - n = (n0\*2^24) + (n1\*2^16) + (n2\*2^8) + n3

### Opening a Handle

- To open a handle, the user must specify:
  - The IP address of the slave
  - (optional) The port number of the slave. If not specified, the board will specify the port value as 1000.
    - Modbus connections always specify port 502
  - (optional) The connection type as TCP/IP or UDP/IP. If not specified, the controller will make a TCP connection.
- Once it is known, issue the IH command on an open handle with the correct settings for IP (n0-n3), port (o) and connection type (p)

### Closing a Handle

DMC40x0, DMC41x3, RIO, DMC21x3

- Closing a handle is done with the S and T handle identifiers, along with connection type p selector.
  - IHS=>p closes the handle that sent the command with connection type matching >p
  - IHT=>p closes all handles except for the one sending the command with connection type matching >-p
  - For closing handles, use >p where p = -1 closes UDP handles, p = -2 closes TCP handles, and p = -3 closes all handle types

## Operand Usage

Operand	Reported Value	Description of Value	Notes
<u>IHm0</u>	-2147483648 to 2147483648	IP address of handle m as a 32 bit number (n)	
<u>IHm1</u>	0 to 65535	Slave port number for handle m	
<u>IHm2</u>	0	Handle is free	Handle 'Available' in TH
	1	Handle connected as UDP slave	
	2	Handle connected as TCP slave	
	-1	Handle connected as UDP master	
	-2	Handle connected as TCP master	
	-5	Attempting to establish UDP handle	
	-6	Attempting to establish TCP handle	
<u>IHm3</u>	0	ARP was successful	
	1	ARP failed or still in progress	
<u>IHm4</u>	1	Waiting for ACK from slave controller after issuing a command	
	2	Received ":" as response to a command	
	3	Received "?" as response to a command	
	4	Connection timed-out waiting for a response to a command	

## EXAMPLES

```
IHA= 251,29,51,1; Open handle A at IP address 251.29.51.1
'TCP is used as default
IHA= -2095238399; Open handle A at IP address 251.29.51.1
```

```
'When the IH command is given,  
'the controller initializes an ARP  
'on the slave device before opening a handle.  
'This operation can cause a small time delay  
'before the controller responds
```

```
'setting up a modbus handle  
MW 1;          setup modbus wait  
IHE= 192,168,100,200<502>2;  setup a modbus handle to slave  
#wt;           wait for handle to be connected  
WT 2;          before issuing a command  
JP #wt, IHE2<>-2;  Set output 3 on slave  
SB 5003;       1 second wait  
WT 1000;       Clear output 3 using MB command  
MBE= ,5,3,0;  EN
```

IH is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>II</b>	<i>Input Interrupt</i>	<b>IO</b>
The II command enables the input interrupt function for the specified inputs		

II n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The II command enables the input interrupt function for the specified inputs.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

If any of the specified inputs are activated during program execution, the program will jump to the subroutine with label #ININT. Any tripoints set by the program will be cleared but can be re-enabled by the proper termination of the interrupt subroutine using RI.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	8	0	1	Lowest input to use for interrupt trigger	n0=0 disables input interrupt.
<b>n1</b>	1	8	N/A	1	Highest input to use for interrupt trigger	n1 must be >= n0. If omitted, n1=n0
<b>n2</b>	1	255	N/A	1	Use bitmask as alternative selection of input interrupt triggers	If n0 and n1 used, n2 is ignored. See Remarks
<b>n3</b>	0	255	0	1	Bitmask specifying required input state for interrupt trigger	Default=interrupt triggers on low inputs. See Remarks

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- The argument n2 is an integer value and represents a binary number showing the inputs selected for the input interrupt function.
  - For example, if n2 = 15, the binary equivalent is 0000111 where the bottom 4 bits are 1 (bit 0 through bit 3) and the top 4 bits are 0 (bit 4 through bit 7). Each bit represents an interrupt to be enabled - bit0 for interrupt 1, bit 1 for interrupt 2, etc. If o=15, the inputs 1,2,3 and 4 would be enabled.
- This argument n3 is an integer value and represents a binary number showing which inputs will trigger on a logic '1' and which on a logic '0'. This binary number is used to logically "AND" with the inputs which have been specified by the parameters n1 and n2 or the parameter n3.
  - For example, if n1=1 and n2=4, the inputs 1,2,3 and 4 have been activated. If the value for n3 is 2 (the binary equivalent of 2 is 00000010), input 2 will be activated by a logic '1' and inputs 1,3, and 4 will be activated with a logic "0".
- The RI command is used to return from the #ININT routine.

DMC41x3, DMC21x3, DMC18x2, DMC30010

- Note: An application program must be running on the controller for the interrupt function to work.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a;'                                Program A
II 1;'                                Specify interrupt on input 1
JG 5000;BG A;'                         Specify jog and begin motion on A axis
#loop;JP #loop;'                        Loop to keep thread zero active, only necesary on Ecnono (21x3/18x2)
EN;'                                     End Program
#ININT;'                                Interrupt subroutine
                                         Stop A, print message, wait for motion to complete
ST A;MG "INTERRUPT";AM A;'              Check for interrupt clear
AI 1;'                                    Begin motion
BG A;'                                    Return to main program, don't re-enable tripoints
```

II is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IK</b>	<i>Block Ethernet ports</i>	ETHERNET
<b>The IK command blocks client connections to the controller on most ports below port number 1000</b>		
		

IK n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The IK command blocks client connections to the controller on most ports below port number 1000. Specific port numbers and ports above 1000 are unaffected.

## ARGUMENTS

### IK n

DMC40x0, DMC41x3

Argument	Value	Description	Notes
n	0	Allow controller to receive Ethernet packets on any port	
	1	Blocks Ethernet packets on ports lower than 1000.	Default. Ports 0,23,68, and 502 are unaffected.

## REMARKS

- A Galil Ethernet controller simultaneously operates as a server (listening for Ethernet connections from a client) and a client (able to create connections to a server).
- Ports 0, 23, 68 and 502 are used for standard client connections to the controller.

## EXAMPLES

```
:IK 1;' Blocks undesirable port communication
:IK 0;' Allows all Ethernet ports to be used
:
```

IK is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IL</b>	<i>Integrator Limit</i>	FILTER/CONTROL
<b>The IL command limits the effect of the integrator gain in the filter to a certain voltage</b>		

ILA= n	Arguments specified with an axis mask and an assignment (=)
IL n ...	Arguments specified with an implicit comma-separated order
ILm	Operand holds the value set in the command

## DESCRIPTION

The IL command limits the effect of the integrator gain in the filter to a certain voltage.

## ARGUMENTS

**ILm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**IL n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-9.998	9.998	9.998	20/65,536	Value of Integrator limit in volts	n<0 (negative value) freezes the effect of the integrator during the move

## REMARKS

- IL is the absolute value of the integrator limit. For example:
  - ILA= 2 limits the output of the integrator of the A-axis to the +/-2 Volt range.
  - KD and KP terms remain active in any case. The output from the KD and KP terms is not affected.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- A negative parameter will freeze the effect of the integrator during the move. For Example:
  - ILA= -3 limits the integrator output of the A axis to +/-3V but freezes the contribution of the Integrator loop during motion.
- If, at the start of the motion, the integrator output is 1.6 Volts, that level will be maintained through the move and the integrator will not accumulate during the move.
- Once the profiled move has completed (RP has reached final commanded position), the integrator loop will be enabled.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
KI 2,3,5,8;' Integrator constants
IL 3,2,7,2;' Integrator limits
IL ?;' Returns the A-axis limit
```

IL is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>IP</b>	<i>Increment Position</i>	INDEPENDENT MOTION
The IP command allows for a change in the command position while the motor is moving		

IPA=n	Arguments specified with an axis mask and an assignment (=)
IP n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The IP command allows for a change in the command position while the motor is moving. This command does not require a BG.

## ARGUMENTS

**IPm= n**

**IP n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Imaginary axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	N/A	1	Value of incremental move	

## REMARKS

- IPm contains the current position of the motor
- The IP command has four effects depending on the mode of motion being executed.

*IP operation based upon modes of motion*

Case	Equivalent Commands	Description
Motor is standing still	IPm=n Equivalent to PRm=n;BGm	Motor will move to specified position with the predefined AC,DC,SP values.
Motor is moving toward position n	PRm=n0; BGm; IPm=n1 Equivalent to PRm=(n0+n1); BGm	Motor will move a relative move of (n0+n1).
Motor is in Jog Mode	JGm=n0;BGm;IPm=n1 Equivalent to Continuing jog from (current position + n1)	The motor will instantly try to servo to a position which is the current instantaneous position plus the specified IP position. SP and AC parameters have no effect. This command is useful when synchronizing 2 axes in which one of the axis' speed is indeterminate due to a variable diameter pulley.
Motor is a slave in gearing mode	GAm=m0; GRm=n0; IPm=n1 Equivalent to GAm=m0; GRm=n0; PRm=n1; BGm	The motor will move with the predefined AC,DC,SP values superimposed on top of the existing gearing motion.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
IP 50;' 50 counts with set acceleration and speed
#correct;' Label
AC 100000;' Set acceleration
JG 10000;BG A;' Jog at 10000 counts/sec rate
WT 1000;' Wait 1000 msec
IP 10;' Move the motor 10 counts instantaneously
ST A;' Stop Motion
EN
```

IP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

The IT command filters the acceleration and deceleration functions of independent moves such as JG, PR, PA to produce a smooth velocity profile



ITA=n	Arguments specified with an axis mask and an assignment (=)
IT n ...	Arguments specified with an implicit comma-separated order
_ITm	Operand holds the value set in the command

## DESCRIPTION

The IT command filters the acceleration and deceleration functions of independent moves such as JG, PR, PA to produce a smooth velocity profile. The resulting profile, known as smoothing, has continuous acceleration and results in reduced mechanical vibrations. IT sets the bandwidth of the filter where 1 means no filtering and 0.004 means maximum filtering.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010

The IT command also filters the individual axes during Vector Mode (VM) and Linear Interpolation Mode (LM).

## ARGUMENTS

**ITm= n**

**IT n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0.004	1	1	1/256	Value of independent smoothing function	1 = no filtering 0.004 = maximum filtering

## REMARKS

- The IT filtering results in longer motion time.
- The use of IT will not effect the tripoints AR and AD.
  - The tripoints AR & AD monitor the profile prior to the IT filter and therefore can be satisfied before the actual distance has been reached if IT is NOT 1.
- Details on the IT filtering can be found in Application Note #3412
  - <http://www.galilmc.com/support/appnotes/optima/note3412.pdf>

## EXAMPLES

```
:IT 0.8, 0.6, 0.9, 0.1;' Set independent time constants for a,b,c,d axes
:IT ?;' Return independent time constant for A-axis
0.8000
```

REM example showing increased time due to IT filtering

```
#move
IT 1
t= TIME;'store time reference
PR 1000
BG A;AM A
MG TIME-t;'display move time
IT 0.01
t= TIME;'store time reference
PR 1000
BG A;AM A
MG TIME-t;'display move time
EN

: 'program execution output
:XQ
:
508.0000
1112.0000
:
```

IT is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

**JG**Jog  
The JG command sets the jog mode and the jog slew speed of the axes

INDEPENDENT MOTION



JGA= n	Arguments specified with an axis mask and an assignment (=)
JG n ...	Arguments specified with an implicit comma-separated order
JGm	Operand has special meaning, see Remarks

**DESCRIPTION**

The JG command sets the jog mode and the jog slew speed of the axes.

**ARGUMENTS****JGm= n****JG n,n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	N	N	N/A	Axis	Virtual axis to assign value	
<b>n</b>	-15000000	15000000	25000	2	Value of jog speed in cnts/second	For MT settings of 1,-1,1.5 and -1.5 (Servos)
	-3000000	3000000	25000	2	Value of jog speed in cnts/second	For MT settings of 2,-2,2.5 and -2.5 (Steppers)

**REMARKS**

- When jogging, the motion controller profiles a continuous move at the commanded speed.
- To stop the motion, use the ST command.
- JG 2 is the minimum non-zero speed
- \_JGm contains the absolute value of the jog speed for the specified axis.

**EXAMPLES**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#ja
JG 100,500,2000,5000
' Sets for jog mode with a slew speed of 100 counts/sec for the A-axis,
' 500 counts/sec for the B-axis,
' 2000 counts/sec for the C-axis,
' and 5000 counts/sec for D-axis.
BG ;
WT 1000; Begin Motion
Wait one second
JG ,,-2000; Change the C-axis to slew in the negative direction at -2000 counts/sec.
EN
```

JG is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>JP</b>	<i>Jump to Program Location</i>	PROGRAMMING
The JP command causes a jump to a program location on a specified condition		
		

JP Only valid from within embedded code. No terminal.

## DESCRIPTION

The JP command causes a jump to a program location on a specified condition. The program location may be any program line number or label. A jump is taken if the specified condition is true. Multiple conditions can be used in a single jump statement.

## ARGUMENTS

**JP #str,(ex)**

**JP n,(ex)**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>str</b>	1 char	7 chars	N/A	String	Label name for jump destination	Must be a valid label in application code
<b>n</b>	0	see Notes	N/A	1	Line number for jump destination	Maximum is number of lines of controller program memory - 1
<b>ex</b>	N/A	N/A	N/A	Expression	Conditional statement/s that must evaluate true for jump to occur	If omitted, JP automatically evaluates as true

## REMARKS

- The logical operators that can be used in the conditional statement are:
  - < less than
  - > greater than
  - = equal to
  - <= less than or equal to
  - >= greater than or equal to
  - $\neq$  not equal to
- The conditional statements are combined in pairs using the operands "&" and "|".
  - The "&" operand between any two conditions, requires that both statements must be true for the combined statement to be true.
  - The "|" operand between any two conditions, requires that only one statement be true for the combined statement to be true.
- Each condition must be placed in parenthesis for proper evaluation by the controller.

```
JP #a, ((var0=1) & (var1=2));' valid conditional jump
JP #a,var0=1&var1=2;' invalid conditional jump
```

## EXAMPLES

```
JP #pos1,(v1<5);' Jump to label #POS1 if variable V1 is less than 5
JP #a,((v7*v8)=0);' Jump to #A if V7 times V8 equals 0
JP #b,(@IN[1]=1);' Jump to #B if input 1 = 1
JP #c;' Jump to #C unconditionally
```

JP is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>JS</b>	<i>Jump to Subroutine</i>	PROGRAMMING
Allows the program to jump to a subroutine and return back after completion		
		

JS	Only valid from within embedded code. No terminal.
_JS	Operand has special meaning see Remarks

## DESCRIPTION

Allows the program to jump to a subroutine and return back after completion. This command is often used to call reusable code.

## ARGUMENTS

**JS #str, (ex)**

**JS n, (ex)**

DMC40x0, DMC41x3, DMC18x6

**JS #str(arg,arg,arg,arg,arg,arg,arg,arg), (ex)**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>str</b>	1 char	7 chars	N/A	String	Label Name for jump destination	Must be a valid label in application code
<b>n</b>	0	1,999	N/A	1	Line number for jump destination	
<b>ex</b>	N/A	N/A	N/A	N/A	Conditional statement/s that must evaluate true for jump to occur	If omitted, the jump is taken
<b>arg</b>	N/A	N/A	N/A	N/A	A value, variable, or array to pass to the subroutine being called	referenced from within the subroutine as ^a-^h, respectively. See Remarks for a table of valid args

## REMARKS

- JS can be nested, called up to 16 deep

DMC40x0, DMC41x3, DMC18x6, DMC30010

- When used after JS is called, the \_JS operand contains the returned value of the subroutine called by JS

DMC40x0, DMC41x3, DMC18x6, DMC30010

## Basic Usage

- The JS command will change the sequential order of execution of commands in a program
- If the jump is taken, program execution will continue at the line specified by the destination parameter, which can be either a line number or label. A variable holding a line number or an expression resulting in the calculation of a line number can also be used
- The line number of the calling JS command is saved and after an EN command is encountered (End of subroutine), program execution will continue with the instruction following the calling JS command.
- A jump is taken if the specified condition is true. Each condition must be placed in parenthesis for proper evaluation by the controller.
- Code flexibility/reuse. A single subroutine can be written and called many times and from various locations in code. The stack "remembers" where to return when completed. This is opposite from a "blind jump" (JP).

### Conditional Syntax

Condition	Validity
JS#A,(var1=0)&(var2=1)	This conditional statement is valid
JS#A,var1=0&var2=1	This conditional statement is not valid

DMC40x0, DMC41x3, DMC18x6, DMC30010

## Passing Values on the Stack

DMC40x0, DMC41x3, DMC18x6, DMC30010

- Parameters can be passed on the subroutine stack
- Passing parameters in a subroutine has many advantages including the following
  - Variable Scope/ Local variables. A subroutine can run with a protected variable space. Local variables exist only in the extent of the subroutine, and no external thread or stack level can access local variables. Local variables can be used for counters, indices, and other helper variables
  - Each thread has its own stack, therefore subroutines are reentrant. In other words, multiple threads can be running the same subroutine simultaneously at various stack depths.
  - Support for recursion. Although the subroutine stack is only 16 deep, recursion is possible. A stack depth of 16 is sufficient for many recursive tasks. E.G. recursing axes, handles, and thread status.
  - Parameter passing. A calling command can explicitly specify the inputs to a subroutine. The subroutine can pass one value back to the calling command. More returns are possible with pass by reference and array passing
- Constants, Variables, and Arrays may be passed up a subroutine stack.
- Variables may be passed by value or by reference. If passed by value, a copy is made in the subroutine stack, leaving the original variable immutable. If passed by reference, the original variable's value will be changed when the subroutine writes to its local variable. This is similar, but not exactly analogous to a C pointer.
- A variable passed by reference is automatically dereferenced; the variable pointer is not exposed to the user. Following the C syntax, a by-reference pass is accomplished with the ampersand (&) in the invoking call.
  - IMPORTANT NOTE: When passing a variable by reference, do not allocate any new variables in the called subroutine.
- Arrays can be passed in the stack, though only by reference. No "&" is used when passing arrays, by-reference is assumed. To pass an array, use its name in quotations.
  - IMPORTANT NOTE: Arrays to be passed must have names that are 6 characters or less.
- The number of elements in an array is returned by reading index -1, e.g. array[-1].
- To return a value on the stack, write the value in the EN command upon ending the subroutine. The parent stack can access this value via \_JS.

DMC40x0, DMC41x3, DMC18x6, DMC30010

Examples of valid args (see examples for demo of each concept)

What to pass	arg	Example
--------------	-----	---------

Value	the value	JS #square(7)
Variable's value	variable name	JS #sub(var)
Variable by reference	ampersand + variable name	JS #sub(&var)
Array by reference	array name	JS#sub("array")

## EXAMPLES

```
JS #square, (v1<5);' Jump to subroutine #SQUARE if V1 is less than 5
JS #loop, (v1>0);' Jump to #LOOP if V1 is not equal to 0
JS #a;' Jump to subroutine #A (unconditionally)
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

### Advanced Usage Examples

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
REM Run all examples
#all
JS #val
JS #var
JS #varref
JS #array
EN

REM Example for each way to pass to a subroutine
REM ****
REM Pass a Value
#val
JS #square(3)
MG _JS
EN
#square
REM Return the passed value squared
EN , (^a^a)
REM ****
REM Pass a variable's value
#var
val= 7
REM call the same sub above
JS #square(val)
MG _JS
EN
REM ****
REM Pass a variable by reference
#varref
val= 9
JS #square2(&val)
MG val
EN
#square2
REM change the value of the variable
^a= ^a^a
REM don't return anything
EN
REM ****
REM Pass an array by reference
#array
DM array[100]
array[42]= 11
JS #square3("array")
MG array[42]
EN
#square3
REM change the array element
^a[42]= ^a[42]^a[42]
REM don't return anything
EN
REM ****
REM Controller Response
REM :XQ#all
REM :
REM 9.0000
REM 49.0000
REM 81.0000
REM 121.0000
```

DMC40x0, DMC41x3, DMC18x6

```
#add
JS #sum(1,2,3,4,5,6,7,8);' Call subroutine, pass values
MG _JS;' Print return value, will print 36.0000
EN

#sum;' Sums values passed to it. Expects 8 numbers
EN , ^a+^b+^c+^d+^e+^f+^g+^h;'
```

DMC40x0, DMC41x3, DMCI8x6, DMC30010

```
'Dimension two arrays
DM array1[10]
DM array2[100]
'Zero the contents of each array
JS #zeroary("array1", 0)
JS #zeroary("array2", 0)
EN

'Zero the contents of an array
#zeroary; (^a array, ^b starting index)
^a[^b]= 0
^b= (^b+1)
JP #zeroary, (^b < ^a[-1])
EN
```

DMC40x0, DMC41x3, DMCI8x6, DMC30010

```
REM Using dynamic destinations in a jump table
i= 1; Counter
#loop
offset= #spell+i; Calculate offset
JS offset; Jump to offset
i= i+1; Increment Counter
JP #loop,i<=3; Loop through 3 states
EN
#
#spell; Subroutine containing various words
MG "One";EN; Prints "One" if this line is called (i=1)
MG "Two";EN; Prints "Two" if this line is called (i=2)
MG "Three";EN; Prints "Three" if this line is called (i=3)

REM Controller responds with:
REM One
REM Two
REM Three
```

JS is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>KD</b>	<i>Derivative Constant</i>	FILTER/CONTROL
<b>KD designates the derivative constant in the control filter</b>		

KDA=n	Arguments specified with an axis mask and an assignment (=)
KD n ...	Arguments specified with an implicit comma-separated order
KDm	Operand holds the value set in the command

## DESCRIPTION

KD designates the derivative constant in the control filter. The derivative gain outputs a voltage based on the rate of change of the error. The filter transfer function follows:

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010

$$D(z) = KP + KD \frac{z-1}{z} + KI \frac{z}{z-1}$$

## ARGUMENTS

**KDm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**KD n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	4,095.875	64	1/8	Value of derivative term	

## REMARKS

- n=? will return the currently set value of KD
- m=\* will set the KD value for all axes/channels

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- For further details see the section "Theory of Operation" in the controller user manual.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:KD 12,14,16,20; ' Implicit notation to set A,B,C,D axis derivative term
:KDC= 8; ' Explicit notation to set C
:KD ,8; ' Implicit notation to set C
:KD ?,?,?,?; ' Return A,B,C,D values
12, 14, 8, 20
:KDC= ?; ' Return C value
8
:MG _KDA; ' Message the operand for the A axis
12
:
```

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
REM Zeroing the PID filter allows the
REM motor command signal to be
REM used as a programmable DAC
KI*= 0; ' Zero KI
KP*= 0; ' Zero KP
KD*= 0; ' Zero KD
ER -1,-1; ' Turn off position error limit
OF 1,2; ' Set one volt on A and two volts on B
EN
```

KD is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>KI</b>	<i>Integrator</i>	FILTER/CONTROL
The KI command sets the integral gain of the control loop		

KIA= n	Arguments specified with an axis mask and an assignment (=)
KI n ...	Arguments specified with an implicit comma-separated order
KIm	Operand holds the value set in the command

## DESCRIPTION

The KI command sets the integral gain of the control loop. The integrator term will reduce the position error at rest to zero. It fits in the control equation as follows:

DMC40x0, DMC41x3, DMC18x6, DMC30010

$$D(z) = KP + KD \frac{z-1}{z} + KI \frac{z}{z-1}$$

## ARGUMENTS

**KIm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**KI n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	255.999	0	1/1,024	Value of Integral term	

## REMARKS

- n=? will return the currently set value of KD
- m=\* will set the KD value for all axes/channels

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- For further details see the section "Theory of Operation" in the controller user manual.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:KIC= 8;          Explicit notation to set C
:KI , ,8;        Implicit notation to set C
:KI ?,?,?,?;    Return A,B,C,D values
7, 14, 8, 20
:KIC= ?;         Return C value
8
:MG _KIA;        Message the operand for the A axis
7
:
```

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
REM Zeroing the PID filter allows the
REM motor command signal to be
REM used as a programmable DAC
KI*= 0;          Zero KI
KP*= 0;          Zero KP
KD*= 0;          Zero KD
OF 1,2;          Set one volt on A and two volts on B
EN
```

KI is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>KP</b>	<i>Proportional Constant</i>	FILTER/CONTROL
KP designates the proportional constant in the controller filter		
KPA=n	Arguments specified with an axis mask and an assignment (=)	
KP n ...	Arguments specified with an implicit comma-separated order	
KPm	Operand holds the value set in the command	

## DESCRIPTION

KP designates the proportional constant in the controller filter. The proportional gain outputs a control signal proportional to the amount of error. The filter transfer function follows.

DMC40x0, DMC41x3, DMC18x6, DMC30010

$$D(z) = KP + KD \frac{z-1}{z} + KI \frac{z}{z-1}$$

## ARGUMENTS

**KPm= n**

**KP n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, RIO, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1,023,875	6	1/8	Value of proportional term	

## REMARKS

- n=? will return the currently set value of KP

DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010

- KP now has four times more resolution as prior controllers, and thus the same value as that of an Optima controller is four times less effective

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- For further details see the section "Theory of Operation" in the controller user manual.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:KP 12,14,16,20; Implicit notation to set a,b,c,d axis proportional term
:KPC= 8; Explicit notation to set C
:KP ,8; Implicit notation to set C
:KP ?,?,?,?; Return A,B,C,D values
7, 14, 8, 20
:KPC= ?; Return C value
8
:MG _KPA; Message the operand for the A axis
12
:
```

```
REM Zeroing the PID filter allows the
REM motor command signal to be
REM used as a programmable DAC
KI*= 0; Zero KI
KP*= 0; Zero KP
KD*= 0; Zero KD
OF 1,2; Set one volt on A and two volts on B
EN
```

KP is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>KS</b>	<i>Step Motor Smoothing</i>	STEPPER MOTOR
The KS parameter sets the amount of smoothing of stepper motor pulses		
		

KSA=n	Arguments specified with an axis mask and an assignment (=)
KS n ...	Arguments specified with an implicit comma-separated order
KSm	Operand holds the value set in the command

## DESCRIPTION

The KS parameter sets the amount of smoothing of stepper motor pulses. Larger values of KS provide greater smoothness. KS adds a single pole low pass filter onto the output of the motion profiler.

## ARGUMENTS

**KSm= n**

**KS n,n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0.5	64	2	1/32	Value of smoothing constant	

## REMARKS

- This is most useful when operating in full or half step mode.
- KS effect on timing
  - This parameter will increase the time to complete a motion time by 3KS sampling periods.
  - KS will cause an overall delay in the generation of output steps.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:KSC= 8;          Explicit notation to set C
:KS , ,8;        Implicit notation to set C
:KS ?,?,?,?;    Return A,B,C,D values
7, 14, 8, 20
:KSC= ?;         Return C value
8
:MG _KSA;        Message the operand for the A axis
7
:
```

KS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LA</b>	<i>List Arrays</i>	PROGRAMMING, INTERROGATION
<b>The LA command returns a list of all arrays in memory</b>		
  		

LA Command takes no arguments

## DESCRIPTION

The LA command returns a list of all arrays in memory. The size of each array will be included next to each array name in square brackets.

## ARGUMENTS

### LA

LA is an interrogation command with no parameters

## REMARKS

- The listing will be in alphabetical order.

## EXAMPLES

```
:DM gold[100],silver[50],plat[200]; Dimensions arrays with given name and the number of array elements in square brackets
:LA; Commands the controller to list arrays in alphabetical order
gold[100]
plat[200]
silver[50]
:DA *[]; Dialocates all arrays
:LA; List arrays now returns with no arrays
:
```

LA is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LC</b>	Low Current Stepper Mode	STEPPER MOTOR
The LC command enables low current mode for stepper motors		
		

LCA= n	Arguments specified with an axis mask and an assignment (=)
LC n ...	Arguments specified with an implicit comma-separated order
LCm	Operand holds the value set in the command

## DESCRIPTION

The LC command enables low current mode for stepper motors. Low current mode reduces the holding torque of the stepper motors while at rest.

## ARGUMENTS

**LCm= n**

**LC n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	n= 0, Default, stepper drive providing 100% torque at rest; n= 1, 25% holding torque when motor at rest.	
	2	32,767	0	1	Specifies "n" samples after move before going to 0% holding current	

## REMARKS

DMC40x0, DMC41x3, DMC21x3

- The MT command must be issued prior to the LC command.

DMC40x0, DMC41x3, DMC21x3, DMC30010

- Using LC with an internal Galil Stepper drive (SDM)

DMC40x0, DMC41x3, DMC21x3

- Setting LC to 0 for each axis may be necessary to shut off all current to the motors in the "motor off" (MO) state

DMC40x0, DMC41x3, DMC21x3, DMC30010

- Using LC will reduce current consumption, but there will be a reduction of holding torque at rest
- Consult the user manual for more details regarding your specific amplifier
- Using LC with external amplifiers
  - When using external amplifiers low current mode will simply disable the motors by toggling the amplifier enable line during rest
  - Using LC will reduce current consumption, but there will be no holding torque at rest

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#ex
MTA= -2;' Specify stepper mode for the C axis
LCC= 1;' Specify low current mode for the C axis
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
#ex
MTA= -2;'specify stepper mode for A axis
LCA= 15;'specify motor to go to low current
' 15 samples after motion has completed
EN
```

LC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LD</b>	<i>Limit Disable</i>	ERROR CONTROL
Allows user to disables forward and/or reverse limit switches		

LDA= n	Arguments specified with an axis mask and an assignment (=)
LD n ...	Arguments specified with an implicit comma-separated order
LDm	Operand holds the value set in the command

## DESCRIPTION

Allows user to disables forward and/or reverse limit switches.

## ARGUMENTS

**LDm= n**

**LD n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	4	0	1	Sets limit disable state	See table below for details

Argument	Value	Description	Notes
<b>n</b>	0	Both limit switches are enabled	Default
	1	Forward limit switch disabled	
	2	Reverse limit switch disabled	
	3	Both limit switches disabled	

## REMARKS

- n=? will return the current setting of LD
- When this feature should be used:
  - To gain additional digital inputs if limit switches are not being utilized.
  - To prevent noise from causing the limit switch conditions even though no limit switches are connected.
- LD does not disable software limits set by BL and FL.

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
:LD 3,1,2;           'Implicit notation to set channel A, B, and C
:MG _LDA; '         'Message the operand for the A channel
 3.0000
:LDC= 3; '          'Explicit notation to set channel C-only
:LD*= ?; '          'Queries the value of LD for all channels
 3, 1, 3, 0
:
```

```
REM use forward limit switch as an extra I/O point
#io
:LDA= 1; 'disable forward limit switch
:io= _LFA; 'set state of limit switch to variable "io"
'Use "io" in an IF statement
IF io=1
  MG "Input On"
ELSE
  MG "Input Off"
ENDIF
EN
```

LD is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LE</b>	<i>Linear Interpolation End</i>	VECTOR/LINEAR
The LE command signifies the end of a linear interpolation sequence		
		

LE	Command takes no arguments
LEm	Operand has special meaning, see Remarks

## DESCRIPTION

The LE command signifies the end of a linear interpolation sequence. This command follows the final LI command.

## ARGUMENTS

### LE n

Argument	Value	Description	Notes
n	?	Returns the total vector move length in encoder counts for the current coordinate system	When used to end a move sequence, n should be omitted

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- LEN will return the total move length in encoder counts for the selected coordinate system, where n is S or T.
- If not specified, the LE command will apply to the last selected coordinate system, S or T.
- To select the coordinate system, use the command CA S or CA T.
- The VE command is interchangeable with the LE command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
CA S;           'Specify S coordinated motion system
LM CD;         'Specify linear interpolation mode for C and D axes
LI ,100,200;   'Specify linear distance
LE;            'Ends linear interpolation distance
BG S;          'Begin motion of the S-coordinate system
```

LE is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LI</b>	<i>Linear Interpolation Distance</i>	VECTOR/LINEAR
The LI command specifies the incremental distance of travel for each axis in the Linear Interpolation (LM) mode		
		

LIA=n	Arguments specified with an axis mask and an assignment (=)
LI n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The LI command specifies the incremental distance of travel for each axis in the Linear Interpolation (LM) mode. LI parameters are relative distances given with respect to the current axis positions.

## ARGUMENTS

**LI** m<o>p

**LI** n,n,n,n,n,n,n <o>p

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-8,388,607	8,388,607	0	1	Assigns linear interpolation point for that axis	
<b>o</b>	2	15,000,000	N/A	2	Specifies the vector speed to be achieved at the beginning of the linear segment	
	-1	-1	N/A	0	Specifies vector speed to be set by Vector Speed Variable (VV command)	See VV command
<b>p</b>	2	15,000,000	N/A	2	Specifies the vector speed to be achieved at the end of the linear segment	

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The CA command is used to set the coordinated system (S or T) for which an LI segment is executed. The default is the S coordinate system (CAS).
- The controller always uses the axis specifications from LM, not LI, to compute the speed.
  - For example: if LM specifies that A-, B-, and C-axis are to be used in linear interpolation mode, but LI only specifies positions for B- and C-, the A-axis will still be used in calculating the overall vector speed.
  - The maximum independent speed of any axis configured as a stepper must not exceed the maximum value allowable via the SP setting
- The slew speed, set by VS, 'o' or 'p' for linear interpolation mode, is the vector speed based on the axes specified in the LM mode. For example, if LM ABC designates linear interpolation for the A,B and C axes the speed of these axes (Va, Vb, and Vc respectively) will be computed from:

$$VS = \sqrt{V_A^2 + V_B^2 + V_C^2}$$

- The Linear End (LE) command must be given after the last LI segment in a sequence. LE tells the controller to decelerate to a stop at the last LI command.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The BG S or BG T command should be issued before the total LI distance reaches 1,073,741,824 (2^30) encoder counts.

## Linear Interpolation Mode Buffer

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- Up to 511 LI segments may be given ahead of the begin sequence (BG S or BG T) command.
- Additional LI commands may be sent during motion when the controller sequence buffer frees additional space for new vector segments.
- It is the responsibility of the user to keep enough LI segments in the controller's sequence buffer to ensure continuous motion.
- \_LMm(\_LMS and \_LMT) contains the available spaces for LI segments that can be sent to the buffer.
  - 511 returned means the buffer is empty and 511 LI segments can be sent.
  - A 0 returned means the buffer is full and no additional segments can be sent.
  - See the LM command for full details.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
LM ABC;          'Specify linear interpolation mode between A-, B-, and C- axis
LI 500,,400;    'Specifies linear interpolation point, B-axis remains stagnat but is still part of the interpolation.
LI 1000,2000,3000; 'Specify linear interpolation point
LE;            'Last segment of sequence
BG S;          'Begin sequence
```

LI is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LL</b>	<i>List Labels</i>	INTERROGATION
The LL command returns a listing of all of the program labels in memory		
		

LL Command takes no arguments

## DESCRIPTION

The LL command returns a listing of all of the program labels in memory.

## ARGUMENTS

### LL

LL is an interrogation command with no arguments

## REMARKS

- The LL command label listing will be in alphabetical order.

DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010

- The LL command returns all of the program labels in memory and their associated line numbers

## EXAMPLES

DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010

```
:LL
#FIVE=5
#FOUR=4
#ONE=1
#THREE=3
#TWO=2
```

LL is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LM</b>	<i>Linear Interpolation Mode</i>	VECTOR/LINEAR
<b>The LM command specifies the linear interpolation mode and specifies the axes for linear interpolation</b>		

LM A	Argument is an axis mask
LMm	Operand has special meaning see Remarks

## DESCRIPTION

The LM command specifies the linear interpolation mode and specifies the axes for linear interpolation.

## ARGUMENTS

### LM mm

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axes to use for linear interpolation mode	

## REMARKS

- Any set of axis may be used for linear interpolation.
- LI commands are used to specify the travel distances between various linear interpolation moves.
- Several LI commands may be given as long as the controller sequence buffer has room for additional segments
  - See the LI command for more information regarding the Linear Interpolation Buffer
- The LE command specifies the end of the linear interpolation sequence.
- Once the LM command has been given, it does not need to be given again unless the VM command has been used

## Operand/Queries

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- LMm contains the number of spaces available in the sequence buffer for the 'm' coordinate system, S or T.
- The LM command will apply to the selected coordinate system, S or T. To select the coordinate system, use the command CA S or CA T.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
LM ABCD;          'Specify linear interpolation mode
VS 10000;VA 100000;VD 1000000;  'Specify vector speed, acceleration and deceleration
LI 100,200,300,400;            'Specify linear distance
LI 200,300,400,500;          'Specify linear distance
LE; BG S;                    'Last vector, then begin motion
```

LM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LS</b>	<i>List</i>	INTERROGATION
The LS command returns a listing of the programs in memory		
		

LS n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The LS command returns a listing of the programs in memory.

## ARGUMENTS

**LS n0,n1**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	1998	0	1	Specifies the line in the program for which the listing will start	
<b>n1</b>	1	1999	1999	1	Specifies the line at which the listing will end	

## REMARKS

- n0 < n1 must always be true
- If n0 or n1 is omitted, default values are used
- n0 and n1 can also specify a label, for example:
  - "LS #label,20" would print out program lines from #label to line 20.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:LS #a,6;          ' List program starting at #A through line 6
2 #a
3 PR 500
4 BG A
5 AM
6 WT 200
'Hint: Remember to quit the Edit Mode Q prior to giving the LS command. (DOS)
```

LS is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LV</b>	<i>List Variables</i>	INTERROGATION
The LV command returns a listing of all of the program variables in memory		

LV Command takes no arguments

## DESCRIPTION

The LV command returns a listing of all of the program variables in memory. The listing will be in alphabetical order.

## ARGUMENTS

### LV

LV is an interrogation command with no parameters

## REMARKS

- Use the \_UL operand for total number of variables available for your controller.
  - See the UL command for more details.

## EXAMPLES

```
:LV
APPLE = 60.0000
BOY = 25.0000
ZEBRA = 37.0000
:
```

LV is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>LZ</b>	Inhibit leading zeros	SYSTEM CONFIG
-----------	-----------------------	---------------

The LZ command is used for formatting the values returned from interrogation commands, variables, and arrays



LZ n ...	Arguments specified with an implicit comma-separated order
_LZ	Operand has special meaning see Remarks

## DESCRIPTION

The LZ command is used for formatting the values returned from interrogation commands, variables, and arrays. By enabling the LZ function, all leading zeros of returned values will be removed.

## ARGUMENTS

### LZ n

Argument	Value	Description	Notes
n	0	Does not remove leading zeros from interrogated values	
	1	Removes leading zeros from interrogated values	Default

## REMARKS

- \_LZ contains the state of the LZ function. '0' is disabled and '1' is enabled.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:LZ 0;          'Disable the LZ function
:varl= 10;      'Sets variable varl to the value of 10.
:TP A;          'Interrogate the controller for current position of A-axis
0000021645.0000
:varl= ?,       'Request value of variable varl
0000000010.0000
:LZ 1;          'Enable LZ function
:TP A;          'Interrogate the controller for current position of A-axis
21645.0000
:varl= ?,       'Request value of variable varl
10.0000
```

```
:LZ 0;          'Disable the LZ function
:TB;            'Tell status bits
001
:LZ 1;          'Inhibit leading zeros
:TB;            'Tell status
1
```

LZ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MB</b>	Modbus	ETHERNET
The MB command is used to communicate with I/O devices using the Modbus TCP/IP protocol		
		

MBA= n Arguments specified with an axis mask and an assignment (=)

## DESCRIPTION

The MB command is used to communicate with I/O devices using the Modbus TCP/IP protocol. The MB command supports the first two levels of Modbus commands. The function code - 1 designates that the first level of Modbus is used (creates raw packets and receives raw data). The other codes are the 10 major function codes of the second level. The format of the command varies depending on each function code.

## ARGUMENTS

*Level 2 Modbus Function Codes*

Function Code	Modbus Definition	Slaved Galil Description (RIO only)
01	Read Coil Status (Read Bits)	Read Digital Outputs (RIO only)
02	Read Input Status (Read Bits)	Read Digital Inputs (RIO only)
03	Read Holding Registers (Read Words)	Read Analog Inputs (RIO only)
04	Read Input Registers (Read Words)	Read Analog Outputs (RIO only)
05	Force Single Coil (Write One Bit)	Write Digital Output (RIO only)
06	Preset Single Register (Write One Word)	Write Digital Outputs (RIO only)
07	Read Exception Status (Read Error Code)	Read Digital Outputs (RIO only)
15	Force Multiple Coils (Write Multiple Bits)	Write Digital Outputs (RIO only)
16	Preset Multiple Registers (Write Words)	Write Analog Outputs (RIO only)
17	Report Slave ID	

### 01: MBm= n0, 1, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Read Coil Status (Read Bits)*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	1	see Notes	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	0	9,999	N/A	1	Address of first coil	
n2	0	99	N/A	1	Quantity of coils	Or, number of IO points to read
str	1 char	8 chars	N/A	String	Name of array to store values	str[0] holds the first value.

```
MBC= ,1,2,8,example[];' Read inputs 2-9 from handle C, save to example[]
'equivalent to reading Digital Outputs or registers mapped to 100xxx
```

### 02: MBm= n0, 2, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Read Input Status (Read Bits)*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	1	see Notes	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	0	9,999	N/A	1	Address of first input	
n2	0	99	N/A	1	Quantity of inputs	Or, number of IO points to read
str	1 char	8 chars	N/A	String	Name of array to store values	str[0] holds the first value.

```
MBC= ,2,4,3,example[];' Read inputs 4,5 and 6 from handle C, save to example[]
'equivalent to reading Digital Inputs or registers mapped to 000xxx
```

### 03: MBm= n0, 3, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Read Holding Registers (Read Words)*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	0	9,999	N/A	1	Address of first register	
n2	0	99	N/A	1	Quantity of registers to read	
str	1 char	8 chars	N/A	String	Name of array to store values	str[0] holds the first value. 2 bytes per element. Array must be as large as the value for n2

```
MBB= ,3,1,4,example[];' Read registers 1 through 4 from handle B, save to example[]
'equivalent to reading Analog Outputs, or registers mapped to 400xxx
```

### 04: MBm= n0, 4, n1, n2, str[]

DMC40x0, DMC41x3

*Read Input Registers (Read Words)*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	0	9,999	N/A	1	Address of first register	
n2	1	99	N/A	1	Quantity of registers to read	

<b>str</b>	1 char	8 chars	N/A	String	Name of array to store values	str[0] holds the first value. 2 bytes per element. Array must be as large as the value for n2
------------	--------	---------	-----	--------	-------------------------------	---

```
MBB= ,4,1,2,example[];' Read registers 1 through 2 from handle B, save to example[]
'equivalent to reading Analog Inputs, or registers mapped to 300xxx
```

## 05: MBn= n0, 5, n1, n2

DMC40x0, DMC41x3, DMC21x3

*Force Single Coil (Write One Bit)*

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to send Modbus command	
<b>n0</b>	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
<b>n1</b>	0	9,999	N/A	1	Address of coil	
<b>n2</b>	0	1	0	1	Set coil status	0 = turn off coil. 1 = turn on coil

```
MBB= ,5,11,1;' Set coil 11 high
'equivalent to setting a Digital Output (SB/CB)
```

## 06: MBn= n0, 6, n1, n2

DMC40x0, DMC41x3, DMC21x3

*Preset Single Register (Write One Word)*

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to send Modbus command	
<b>n0</b>	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
<b>n1</b>	0	9,999	N/A	1	Address of holding register	
<b>n2</b>	0	65,535	0	1	Set register value	

```
MBC= ,6,10,128;' Write 128 to holding register 10 on handle C
'equivalent to setting digital outputs on the RIO, or setting registers addressed 400xxx
```

## 07: MBn= n0, 7, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Read Exception Status (Read Error Code)*

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to send Modbus command	
<b>n0</b>	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
<b>n1</b>	0	9,999	N/A	1	Address of first register	
<b>n2</b>	1	99	N/A	1	Quantity of registers	
<b>str</b>	1 char	8 chars	N/A	String	Name of array to store values	str[0] holds the first value. 2 bytes per element.

```
MBE= ,7,1,1,example[];' Read first register and store in example[]
'used to read error codes on RIO
```

## 15: MBn= n0, 15, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Force Multiple Coils (Write Multiple Bits)*

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to send Modbus command	
<b>n0</b>	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
<b>n1</b>	0	9,999	N/A	1	Address of first coil	
<b>n2</b>	1	16	N/A	1	Quantity of coils	
<b>str</b>	1 char	8 chars	N/A	String	Array to set values for coils	str[0] holds the first value. 16 bits per element

```
example[0]= 255;'
MBC= ,15,0,16,example[];' Set 1st byte of coils high and 2nd byte of coils low
'equivalent to setting digital outputs on RIO, or setting coils addressed 000xxx
```

## 16: MBn= n0, 16, n1, n2, str[]

DMC40x0, DMC41x3, DMC21x3

*Preset Multiple Registers (Write Words)*

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Handle	Handle to send Modbus command	
<b>n0</b>	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
<b>n1</b>	0	9,999	N/A	1	Address of first register	
<b>n2</b>	0	99	N/A	1	Quantity of registers	
<b>str</b>	1 char	8 chars	N/A	String	Array containing modbus data	str[0] holds the first value. 2 bytes per element. Array size must be > n2

```
example[0]= $AEAE
MBD= ,16,2,1,example[];' Set $AEAE to holding register 2 on handle D
'equivalent to setting analog outputs, or writing to holding registers addressed 400xxx
```

DMC40x0, DMC41x3, DMC21x3, DMC30010

## MBn= n0,17,str[]

*Report Slave ID*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	see Notes	1	Unit ID	Default to Handle number (A=1, B=2, etc.)
str	1 char	8 chars	N/A	String	Name of array to receive data	str[0] holds the value.

```
'MBB= ,17,example[];' store slave ID of device on handle B to example[]
```

**Raw Modbus Packet Send****MBm= n0,-1,n1,str[]***Raw Modbus Send*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	1	see Notes	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	0	999	N/A	1	Number of array bytes to send	
str	1 char	8 chars	N/A	String	Name of array containing outgoing data	Array size >= n1. See Remarks

**Raw Modbus Packet Send/Receive:****MBm= n0,-1,n1,str0[],n2,n3,str1[]***Raw Modbus Send/Receive*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to send Modbus command	
n0	0	255	1	see Notes	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	1	999	N/A	1	Number of array bytes to send	
str0	1 char	8 chars	N/A	String	Name of array containing outgoing data	Array size >= n1. See Remarks
n2	1	999	N/A	1	Number of bytes of incoming data to discard	
n3	1	999	N/A	1	Number of bytes of incoming data to store in str1[]	
str1	1 char	8 chars	N/A	String	Name of array storing incoming data.	Array size >= n3. See Remarks

*Raw Modbus Send/Receive*

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	F	N/A	Handle	Handle to send Modbus command	
n0	0	255	1	see Notes	Unit ID	Default to Handle number (A=1, B=2, etc.)
n1	1	999	N/A	1	Number of array bytes to send	
str0	1 char	8 chars	N/A	String	Name of array containing outgoing data	Array size >= n1. See Remarks
n2	1	999	N/A	1	Number of bytes of incoming data to discard	
n3	1	999	N/A	1	Number of bytes of incoming data to store in str1[]	
str1	1 char	8 chars	N/A	String	Name of array storing incoming data.	Array size >= n3. See Remarks

**REMARKS**

- For those command formats that have "addr", this is the slave address. The slave address may be designated or defaulted to the device handle letter.
- All the formats contain an m parameter. This designates the connection handle letter. Port 502 must be used in the Ethernet handle.
  - See the IH command for more info on how to open a handle with a specific port number, and proper connection handling

**Raw Modbus (Function code -1)**

- Each element of str[] arrays may contain only one byte of the modbus packet when performing a raw modbus command.
- Outgoing str[] must contain the entire modbus packet, including transaction identifiers, protocol identifiers, length field, modbus function code, and data specific to that function code.

**EXAMPLES**

```
'This example shows accessing the IO of a RIO from a DMC controller
'First, we will establish a connection
IHF= > -3;'           clear handle
WT 100
IHF= 192,168,1,2<502;' RIO address
WT 100
MW 1;'                  Set to wait for MB response before next command
#douts
'Examples of reading/writing digital outputs
DM test[16];'          setup modbus array
MBF= ,5,1,1;'           Turn DO1 on
MBF= ,5,7,1;'           Turn DO7 on
test= 15
MBF= ,15,0,8,test[];'   Test value means DO0-DO3 turn on, DO4-DO7 turn off
MBF= ,1,0,16,test[];'   Read all 16 DO states into test array
#dins
'Examples of reading digital inputs
MBF= ,2,4,4,test[];'   Read inputs 4-7 to test
MG test[0];'             Output status of input 4
#ain
```

```

'Examples of reading analog inputs, assume MV0 set on RIO
'MI1 also set, so each analog takes 2 array elements
'and must be converted from 32bit floating point back to voltages for reading
MBF= ,3,2,4,test[];'      Read AN1 and AN2 from RIO
an1= (test[0]*65536)+test[1];' recombine array data to 32bit floating point value
an2= (test[2]*65536)+test[3];' for analog 1 and 2
MG an1,an2;'              Return values in floating point notation
#aout
'Examples for reading/writing analog outputs, assume MV0 set on RIO
test[0]= $4040
test[1]= $0000;'           This gives $40400000, which is 3V in 32bit floating point
MBF= ,16,6,2,test[];'     Set AO3 to 3V
MBF= ,4,4,4,test[];'     Read both analog output 2 and 3
'must be converted back from floating point like analog inputs example

```

MB is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MC</b>	<i>Motion Complete</i>	TRIPPOINT
The MC command is a trippoint command that holds up execution until motion is complete on any one of a specified group of axes		

MCA | Argument is an axis mask

## DESCRIPTION

The MC command is a trippoint command that holds up execution until motion is complete on any one of a specified group of axes. The MC command, unlike the AM (after motion command) requires that both the motion profiler has completed motion AND that the motor encoder has reached the specified position before continuing execution.

## ARGUMENTS

### MC mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axis to assign value	Any combination of the axis is valid. If no axis is specified, command applies to all axis.

## REMARKS

- Although many axes can be specified, the MC command will continue execution if one of the specified axis motion is completed.

### Using MC with Stepper Motors

- In the case of stepper motors, MC will monitor the number of step pulses are generated to complete the move.
- The MC command is recommended when operating with stepper motors in lieu of AM since the generation of step pulses can be delayed due to the stepper motor smoothing function, KS. In this case, the MC command would only be satisfied after all steps are generated.

### Using MC as part of the #MCTIME error routine

- The command TW can be used to set an acceptable amount of time between when the motion profiler has completed and the encoder is in position; if this condition is not satisfied, a timeout error occurs.
  - When a timeout occurs, the trippoint will clear and the stop code will be set to 99.
  - Thread 0 of the DMC program will also jump to the special label #MCTIME, if present.
    - See the #MCTIME automatic subroutine, TW and SC commands for more information

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#move;           'Label #move
TW 1000,1000;   'Set motion complete timeout to 1000 milliseconds per axis
PR 2000,4000;   'Position relative Move on A- and B-axis
BG AB;          'Start the motion on A- and B-axis
MC AB;          'After the move is complete on A and B axes
MG "DONE";
EN;             'End of Program
'
#MCTIME;        'Motion Complete timeout Subroutine
MG "Motion Timeout";
SC ;            'Print failure message
    'Print stop codes
EN;             'End subroutine
```

MC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MF</b>	<i>Forward Motion to Position</i>	TRIPPOINT
-----------	-----------------------------------	-----------

This command will hold up the execution of the following command until the specified motor moves forward and crosses the position specified



MFA= n	Arguments specified with an axis mask and an assignment (=)
MF n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

This command will hold up the execution of the following command until the specified motor moves forward and crosses the position specified.

## ARGUMENTS

**MFm= n**

**MF n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	N/A	1	Position required to be crossed before subsequent commands will be executed.	

## REMARKS

- Although multiple positions can be specified, only one of the MF conditions must be satisfied for subsequent code execution.
- MF command references absolute position.
- The MF command only requires an encoder and does not require that the axis be under servo control.
- The accuracy of the MF command is the number of counts that occur in  $2 \times TM$  sec. Multiply the speed by  $2 \times TM$  sec to obtain the maximum error.
  - Example with speed of 20,000 counts/second and TM of 1000 (1000 us).
    - Maximum error =  $2 \times 1000 \text{ E-6 seconds} \times 20,000 \text{ counts/second} = 40 \text{ counts}$
- When using a stepper motor:
  - This condition is satisfied when the stepper position (as determined by the output buffer - TD) has crossed the specified Forward Motion Position.

## EXAMPLES

```
#test;
DP 0;
JG 1000;
BG A;
MF 2000;
v1=_TPA;
MG "Position is",v1;
ST A;
EN;
'Program Test
'Define zero
'Jog mode (speed of 1000 counts/sec)
'Begin move
'After passing the position 2000
'Assign V1 A position
'Print Message
'Stop
'End of Program
```

MF is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

## MG Message

INQUIRERATION

The MG command is used to send strings, operand, variable, and array values to a specified destination



MG n ... Arguments specified with an implicit comma-separated order

### DESCRIPTION

The MG command is used to send strings, operand, variable, and array values to a specified destination.

### ARGUMENTS

MG "str", {^n0}, n1

Argument	Value	Description	Notes
str	String	A string including alphanumeric characters to be displayed	Limited to 76 characters
n0	ASCII character in decimal	Allows users to print ASCII characters	Range of 0-255
n1	Numeric value	Prints the numerical value specified	See Examples for valid uses of n1.
	Variable name	Prints the numeric value stored by the variable	
	Operand	Prints the numeric value stored by the operand	
	Array element	Prints the numeric value stored by the array element	
	Mathematical expression	Prints the numeric value of the solved equation	

### REMARKS

- Multiple strings, variables, and ASCII characters may be used, each must be separated by a comma.
- Solicited Messages
  - From a host terminal, application code, or device, sending the MG command will return with the requested information. This is known as a solicited command, because the host sends the command and expects a response.
- Unsolicited Messages
  - From embedded DMC code, the MG command will send an unsolicited, asynchronous message from the controller to the host. This can be used to alert an operator, send instructions or return a variable value. This is known as an unsolicited command because the host is not explicitly requesting it.
  - The CW command controls the ASCII format of all unsolicited messages.

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- Unsolicited messages can go to any of the Ethernet handles, or serial ports.
- The CF command sets the default communication port for routing unsolicited messages.

### Formatting

- Formatters can be placed after each argument to modify how it is printed.
  - {Fm.n} Display variable in decimal format with m digits to left of decimal, and n to the right.
  - {Zm.n} Same as {Fm.n} but suppresses leading zeros.
  - {Sm.n} Display variable in hexadecimal format with m digits to left of decimal, and n to the right.
  - {Sn} Display variable as a string of length n where n is 1 through 6
  - {N} Suppress carriage return line feed (\r\n) at the end of the message.

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

### Message Routing

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- MG can override the default CF setting by using the following modifiers at the beginning of the message, right after MG.
  - {Pn} Sends the message out the Serial port n, where n is 1 or 2 denoting Main or Auxiliary (where equipped).

DMC40x0, DMC41x3, DMC21x3

- {Ex} Sends the message out the Ethernet handle x, where x is A,B,C,D,E,F,G or H

### EXAMPLES

#### Valid uses of n1 argument

```
:Values
:MG 1234.5678
1234.5678
:
:Variables
:var= 12345678.9101
:MG var
12345678.9101
:
:Operands
:MG @AN[1]
0.0121
:
:Array Elements
:DM arr[3]
:arr[0]= 0
:arr[1]= 1
:arr[2]= 2
:MG arr[0],arr[1],arr[2]
0.0000 1.0000 2.0000
:
:Mathematical Expressions
:MG 1+2
3.0000
:MG arr[2]+var
12345680.9101
:
```

#### General Use

```
:MG "Good Morning";           'Message command displays ASCII string
Good Morning
:total= 1234.5322;          'Assigns variable total with the value 1234.5322
:MG "The answer is...",total{F4.2}; 'Will print the message and the value of variable total formatted with 4 integer digits and 2 fractional digits
The answer is... 1234.53
:MG {^13}, {^10}, {^48}, {^055};   'Specifies carriage return, line feed, and the characters 0 and 7 in ASCII decimal values
```

```
07  
:MG_TIME;  
261928200.0000  
:variable= 10;  
:MG_variable+5;  
15.0000  
:MG_TIO;  
255.0000
```

'Messages the operand TIME  
'Sets the variable equal to 10  
'Messages out variable + 5  
'Messages the value stored in the operand \_TIO

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

```
CF_A:  
MG_{EB}var:
```

'Messages configured to go out Ethernet handle A  
'Override CF and send the value of variable var to B handle

MG is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MO</b>	<i>Motor Off</i>	SYSTEM CONFIG
The MO command turns off the motor command line and toggles the amplifier enable signal		

MO A	Argument is an axis mask
MOm	Operand has special meaning see Remarks

## DESCRIPTION

The MO command turns off the motor command line and toggles the amplifier enable signal.

## ARGUMENTS

### MO mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Specifies axis to turn off	

## REMARKS

- The controller will continue to monitor the motor position
  - See the TP command for more details
- To turn the motor back on use the SH (Servo Here) command.
- The MO command is useful for positioning the motors by hand.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
MO ;           'Turns off all motors
MO A;         'Turns off the A motor.
MO B;         'Turns off the B motor.
MO CA;        'Turns off the C and A motors.
SH ;          'Turns all motors on
axis= MOA;    'Sets variable axis equal to the A-axis servo status
```

MO is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

This command will hold up the execution of subsequent code specified motor moves backward and crosses the position specified



MRA=n	Arguments specified with an axis mask and an assignment (=)
MR n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

This command will hold up the execution of subsequent code specified motor moves backward and crosses the position specified.

## ARGUMENTS

**MRm=n**

**MR n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	N/A	1	Value of position that must be crossed in the reverse direction	

## REMARKS

- MR command references absolute position.
- Although multiple positions can be specified, only one of the MR conditions must be satisfied for subsequent code execution.
- The MR command only requires an encoder and does not require that the axis be under servo control.
- The accuracy of the MR command is the number of counts that occur in  $2^*TM$  usec. Multiply the speed by  $2^*TM$  usec to obtain the maximum error.
  - Example with speed of 20,000 counts/second and TM of 1000 (1000 us).
    - Maximum error =  $2 * 1000 E-6$  seconds \* 20,000 counts/second = 40 counts
- When using a stepper motor, this condition is satisfied when the stepper position (as determined by the output buffer - TD) has crossed the specified reverse motion position.

## EXAMPLES

```
#test;          Program Test
DP 0;          Define zero
JG -1000;      Jog mode (speed of 1000 counts/sec)
BG A;          Begin move
MR -3000;      After passing the position -3000
v1=_TPA;       Assign V1 A position
MG "Position is", v1; Print Message
ST ;           Stop
EN;           End of Program
```

MR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

MT	Motor Type	SYSTEM CONFIG
The MT command selects the type of the motor and the polarity of the drive signal		
MTA=n	Arguments specified with an axis mask and an assignment (=)	
MT n ...	Arguments specified with an implicit comma-separated order	
_MTm	Operand holds the value set in the command	

## DESCRIPTION

The MT command selects the type of the motor and the polarity of the drive signal. Motor types include standard servomotors, which require a voltage in the range of +/- 10 Volts, and step motors, which require pulse and direction signals. The polarity reversal inverts the analog signals for servomotors, and inverts logic level of the pulse train, for step motors.

## ARGUMENTS

**MTnr= n**

**MT n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	

DMC40x0, DMC41x3, DMC18x6

Argument	Value	Description	Notes
n	1	Specifies Servo motor	Default
	-1	Specifies Servo motor with reversed polarity	
	1.5	Specifies PWM/Sign servo drive	
	-1.5	Specifies PWM/Sign servo drive with reversed polarity	
	2	Specifies Step motor with active low step pulses	
	-2	Specifies Step motor with active high step pulses	Valid setting for all Galil SDM stepper drives
	2.5	Specifies Step motor with reversed direction and active low step pulses	
	-2.5	Specifies Step motor with reversed direction and active high step pulses	Valid setting for all Galil SDM stepper drives

## REMARKS

- n=? will return the value of the motor type for the specified axis
- For step and direction modes (n=2,-2,2.5,-2.5), the auxiliary encoder input for the axis is no longer available

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
MT 1,-1,2,2;  'Configure A as servo, B as reverse servo, C and D as steppers
MT ?,?;        'Interrogate motor type for A- and B-axis
```

MT is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MU</b>	<i>Multicast Address</i>	ETHERNET
<b>MU</b> sets the controller's multicast address		
MU n ...	Arguments specified with an implicit comma-separated order	
_MU	Operand has special meaning see Remarks	

## DESCRIPTION

MU sets the controller's multicast address. This address is used by Galil software to detect an available Ethernet controller on the network.

## ARGUMENTS

### MU n0,n1,n2,n3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	255	0	1	First field of the multicast address	
<b>n1</b>	0	255	0	1	Second field of the multicast address	
<b>n2</b>	0	255	0	1	Third field of the multicast address	
<b>n3</b>	0	255	0	1	Last field of the multicast address	

## REMARKS

### DMC41x3

- Supported on DMC-41x3 firmware rev 1.1b and above.
- MU ? returns the current multicast address setting in 4 byte format
- \_MU contains the 32-bit multicast address number in two's complement.

## EXAMPLES

```
:MU 239,255,19,57
:MU ?
239, 255, 019, 057
:MG _MU
-268496071.0000
:MG _MU{8.0}
$EFFF1339
:
```

MU is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>MW</b>	Modbus Wait	ETHERNET
Enabling the MW command causes the controller to hold up execution of the program after sending a Modbus command until a response from the Modbus device has been received		
MW n ...	Arguments specified with an implicit comma-separated order	
_MW0, _MW1	Operand has special meaning, see Remarks	

## DESCRIPTION

Enabling the MW command causes the controller to hold up execution of the program after sending a Modbus command until a response from the Modbus device has been received. The MW command ensures that the command that was sent to the Modbus device was successfully received before continuing program execution.

## ARGUMENTS

### MW n

Argument	Value	Description	Notes
n	0	Disables Modbus wait	
	1	Enables Modbus wait	Default

## REMARKS

- \_MW0 returns last function code received
- \_MW1 returns Modbus error code
- n=? returns the state of the Modbus wait, either 1 or 0
- If a Modbus response is never received, then thread 0 would jump to the #TCPERR subroutine if it exists and an error code of 123 will occur on \_TC.
- MW prevents the controller from sending multiple commands to the same Modbus device before it has a chance to execute them.

## EXAMPLES

```
MW 1;      'Enables Modbus Wait
SB 1001;  'Set Bit 1 on Modbus Handle A
CB 1001;  'Clear Bit 1 on Modbus Handle A
```

MW is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>NB</b>	<i>Notch Bandwidth</i>	FILTER/CONTROL
The NB command sets real part of the notch poles		
NBA= n	Arguments specified with an axis mask and an assignment (=)	
NB n ...	Arguments specified with an implicit comma-separated order	
_NBm	Operand holds the value set in the command	

## DESCRIPTION

The NB command sets real part of the notch poles. In other words, the NB controls the range of frequencies that will be attenuated.

## ARGUMENTS

**NBm= n**

**NB n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	62.5	0.5	1/2	Value of the notch bandwidth in Hz	Max value dependent upon TM setting, see Remarks

## REMARKS

- \_NBm contains the value of the notch bandwidth for the specified axis.
- NB also determines the ratio of NB/NZ which controls the attenuation, or depth, of the notch. See NZ for more details.
- See the NF command for recommendations on choosing NZ, NB, and NF values.
- See Application note #2431 for additional information on setting the NF, NB and NZ commands
  - <http://www.galilmc.com/support/appnotes/optima/note2431.pdf>

## Maximum Range

- The maximum n argument is specified in Hz and is calculated by the equation below:

$$\frac{1}{(16 \times TM \times 10^{-6})}$$

- where TM is specified in microseconds.
- The default TM is 1000, therefore default maximum NB value = 1/(16x1000E-6) = 62.5 Hz

## EXAMPLES

```
NBA= 10;          'Sets the real part of the notch pole to 10/2 Hz
notch = NBA;     'Sets the variable "notch" equal to the notch bandwidth value for the A axis
```

NB is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>NF</b>	<i>Notch Frequency</i>	FILTER/CONTROL
The NF command sets the frequency of the notch filter, which is placed in series with the PID compensation		

NFA= n	Arguments specified with an axis mask and an assignment (=)
NF n ...	Arguments specified with an implicit comma-separated order
_NFm	Operand holds the value set in the command

## DESCRIPTION

The NF command sets the frequency of the notch filter, which is placed in series with the PID compensation.

## ARGUMENTS

**NFm= n**

**NF n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	250	0	1	Sets the frequency of the notch filter	Max value dependent upon TM setting see Remarks

## REMARKS

- \_NFm contains the value of notch filter for the specified axis.
- n=? Returns the value of the Notch filter for the specified axis.
- See Application note #2431 for additional information on setting the NF, NB and NZ commands
  - <http://www.galilmc.com/support/appnotes/optima/note2431.pdf>

## Choosing NF, NB, and NZ

- A simple way for attaining NF, NB, and NZ parameters is to follow these simple rules:
  - Estimate the resonance frequency (GalilTools Scope with cursors or Galil's FAS software)
  - Set NF equal to the resonance frequency
  - Set NB = 1/2NF
  - Set NZ between 0 and 5
- The ratio of NB/NF is extremely important. See the NB command for more details.

## Maximum Range

- The maximum n argument is specified in Hz and is calculated by the equation below:

$$\frac{1 \times 10^6}{(4 \times TM)}$$

- Where TM is in microseconds.
  - Default TM is 1000, therefore default maximum value =  $1E6/(4*1000) = 250$  Hz

## EXAMPLES

NF , 20 ;' Sets the notch frequency of B axis to 20 Hz
--

NF is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>NO</b>	No Operation	PROGRAMMING
The NO command performs no action in a sequence and can be used as a comment in a program		

NO Operand has special meaning see Remarks

## DESCRIPTION

The NO command performs no action in a sequence and can be used as a comment in a program.

## ARGUMENTS

### NO str

Argument	Value	Description	Notes
str	String	A no action sequence used to document a program	Comments are limited to the maximum row size in a program. This will vary by controller.

## REMARKS

- \_NO returns a bit mask indicating which threads are running
  - For example:
    - 0 means no threads are running
    - 1 means only thread 0 is running
    - 3 means threads 0 and 1 are running
    - 255 means all 8 threads are running

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2  
 ▪ 255 means all 8 threads are running

## EXAMPLES

```
#a;           'Program A
NO;           'No Operation
NO This Program;   'No Operation
NO This Program ;  'No Operation
NO Does Absolutely; 'No Operation
NO Nothing;       'No Operation
EN;           'End of Program
```

NO is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>NZ</b>	Notch Zero	FILTER/CONTROL
<b>The NZ command sets the real part of the notch zero</b>		
     		

NZA= n	Arguments specified with an axis mask and an assignment (=)
NZ n ...	Arguments specified with an implicit comma-separated order
_NZm	Operand holds the value set in the command

## DESCRIPTION

The NZ command sets the real part of the notch zero. In other words, the NB/NZ ratio controls the amount of attenuation, or depth, of the notch filter.

## ARGUMENTS

**NZm= n**

**NZ n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0.5	62.5	0	0.5	Value of Notch Frequency in Hz	Max value dependent upon TM setting, see Remarks

## REMARKS

- See the NF command for recommendations on choosing NZ, NB, and NF values.
- The maximum n argument is determined by the following equation

$$\frac{1}{(16 \times TM \times 10^{-6})}$$

- Where TM is in microseconds, the default TM is 1000.
- See Application note #2431 for additional information on setting the NF, NB and NZ commands
  - <http://www.galilmc.com/support/appnotes/optima/note2431.pdf>

## The NB/NZ Ratio

- The ratio, NB/NZ controls the amount of attenuation, or depth of the notch.
  - The larger the ratio of NB/NZ, the larger the attenuation, and vice versa.
- If NB/NZ > 1 the signal will amplify the output signal causing a resonance.
- NB = NZ essentially eliminates the notch

## EXAMPLES

```
NZA = 10; ' Sets the real part of the notch pole to 10/2 Hz
```

NZ is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OA</b>	Off on encoder failure	ERROR CONTROL
The OA command turns on or off encoder failure detection		
OAA=n	Arguments specified with an axis mask and an assignment (=)	
OA n ...	Arguments specified with an implicit comma-separated order	
OA_m	Operand holds the value set in the command	

## DESCRIPTION

The OA command turns on or off encoder failure detection. The controller can detect a failure on either or both channels of the encoder. This is accomplished by checking on whether motion of at least 4 counts is detected whenever the torque exceeds a preset level (OV) for a specified time (OT).

## ARGUMENTS

**OA\_m=n**

**OA n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	1	0	1	Status of encoder failure detection	1 = enabled, 0 = disabled

## REMARKS

- The OA command works like the OE command: if OA is set to 1 and an encoder failure occurs, the axis goes into the motor off (MO) state and the stop code (SC) is set to 12 if detected during motion.
- The encoder failure detection will shut the motor off regardless of profiling status, but the stop code is not updated unless the axis is executing a profiled move at the time of the detection of the encoder failure.
- If included in the application program and OA is set to 1, #POSERR will run when an encoder failure is detected for the axis.
  - Note that for this function to work properly it is recommended to have a non-zero value for KI.

## EXAMPLES

```
OAA=1; ' enable A axis encoder error detection
MG OA; 'query OA value for A axis
```

DMC40x0, DMC41x3, DMC18x6

```
OA ,1; ' enable B axis encoder error detection
```

```
#setup
'setup the encoder error detection
OTA= 10; ' Set time to 10 milliseconds
OVA= 5; ' Set voltage to 5
OAA= 1; ' Enable encoder detection feature
EN
```

```
REM #POSERR example for checking to see if encoder failure occurred
REM This procedure is needed because the stop code will only update if
REM the profilier is running at the time the encoder failure is detected.
```

```
#POSERR
~a= 0
#loop
IF _MO~a=1
  IF (( _TE~a< ER~a) & ( _OE~a) & ( _OA~a))
    MG "possible encoder failure on ",~a(Z1.0)," axis"
  ENDIF
ENDIF
~a= ~a+1
JP #loop,~a<_BV
AI 1; ' wait for input 1 to go high
SH ; ' enable all axes
RE
```

OA is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OB</b>	<i>Output Bit</i>	<b>IO</b>
The OB command allows variable control of an output bit based on logical expressions		
		

OB n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The OB command allows variable control of an output bit based on logical expressions. The OB n, logical expression command defines output bit i as either 0 or 1 depending on the result from the logical expression.

## ARGUMENTS

### OB n, ex

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n</b>	1	16	0	1	Output bit specified	Outputs 9-16 only valid on 5-8 axis controller
<b>ex</b>	N/A	N/A	N/A	Expression	Expression that defines status of output	If ex is true/non-zero, set output to 1. If ex is false/zero, set output to 0

## REMARKS

- An expression is any valid logical expression, variable or array element.
- Any non-zero value of the expression results in a one set to the output bit.

## EXAMPLES

```
OB 1, pos;' If pos<=0, Bit 1 is high.  
' If pos=0, Bit 1 is low  
OB 2, @IN[1]&@IN[2] ;' If Input 1 and Input 2 are both high, then  
' Output 2 is set high  
OB 3, count[1];' If the element 1 in the array is zero, clear bit 3  
OB n, count[1];' If element 1 in the array is zero, clear bit n
```

OB is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OC</b>	<i>Output Compare</i>	<b>IO</b>
The OC command sets up the Output Compare feature, also known as Pulse on Position		

OCA= n	Arguments specified with an axis mask and an assignment (=)
_OC	Operand has special meaning, see Remarks

## DESCRIPTION

The OC command sets up the Output Compare feature, also known as Pulse on Position. The controller has a special digital output which can be configured to pulse on a specified absolute encoder position, and optionally on a delta encoder change after that. These operations are known as one-shot and circular compare, respectively.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Each set of 4 axes, ABCD and EFGH, has one digital output which can be configured to this mode of operation

## ARGUMENTS

**OCm = n0, n1**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to enable output compare	Axes A-D share one output compare, axes E-H share a second output compare output
<b>n0</b>	-2,147,483,648	2,147,483,647	N/A	1	Absolute encoder position of first pulse	n0 must be within 65535 counts of current position
<b>n1</b>	-65,536	65,535	N/A	1	Incremental encoder distance between pulses	0 indicates single-shot pulse in positive direction, -65536 indicates single shot when moving in the negative direction

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The OC function cannot work for an axis configured as a stepper, and the auxiliary encoder of the corresponding axis cannot be used when in this mode.
  - Dual loop mode (which uses the aux encoder input) will not operate when the OC command enabled.
- For controllers with 5-8 axes, two output compares are available. One for the A-D axes, the other for the E-H axes
- The OC function requires that the main encoder and auxiliary encoders be configured exactly the same (see the command, CE). For example: CE 0, CE 5, CE 10, CE 15.
- OC only requires an encoder, and is independent of axis tuning and motion profiling

### One shot Compare Mode:

- The output compare signal will go low, and stay low at a specified absolute encoder position.
- This is done by specifying n1 as 0 for positive motion, and -65536 for negative motion

### Circular Compare Mode:

- After the absolute position of the first pulse (n0), the circular compare can be configured to pulse low at a relative distance thereafter (n1).
- This is done by specifying n1 to a non-zero delta position (range of -65535 to 65535)

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- OCA = 0 will disable the Circular Compare function on axes A-D.
- OCE = 0 will disable the Circular Compare function on axes E-H.

DMC41x3, DMC30010

- The circular compare output is a low-going pulse with a duration of approximately 510 nanoseconds.

### Operand Usage

- \_OC contains the state of the OC function
  - \_OC = 0 : OC function has been enabled but not generated any pulses.
  - \_OC = 1: OC function not enabled or has generated the first output pulse.

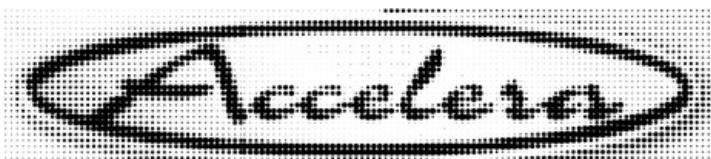
DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- On a 5-8 axis controller, \_OC is a logical AND of axes A-D and E-H

## EXAMPLES

```
  OCA= 300,100; ' Select A encoder as position sensor.
REM First pulse at 300. Following pulses at 400, 500, 600 ...
```

```
REM Output compare can be used to create raster scans.
REM By using circular compare on one axis, followed by an index move on a perpendicular axis
REM raster patterns are easily made.
REM The following image shows a rastered "dot matrix" type image easily created
REM with output compare and a laser on a two dimensional stage.
```



OC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OE</b>	Off-on-Error	ERROR CONTROL
The OE command sets the Off On Error function for the controller		

OEA= n	Arguments specified with an axis mask and an assignment (=)
OE n ...	Arguments specified with an implicit comma-separated order
OE <sub>m</sub>	Operand holds the value set in the command

## DESCRIPTION

The OE command sets the Off On Error function for the controller. The OE command causes the controller to shut off the motor command if a position error exceeds the limit specified by the ER command, an abort occurs from either the abort input or on AB command, or an amplifier error occurs based on the description of the TA command.

## ARGUMENTS

**OE<sub>m</sub>= n**

**OE n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	0	0	0	Disables the Off On Error Function	Default
	1	1	0	0	Motor shut off by position error, amplifier error or abort input	
	2	2	0	0	Motor shut off by hardware limit switch	
	3	3	0	0	Motor shut off by position error, amplifier error, abort input or by hardware limit switch	

## REMARKS

- For any value of OE  $\neq 0$ , the axis will be shut off due to amplifier faults. See the TA command for conditions of an amplifier fault.
- BR1 must be enabled when internal brushless servo amplifiers are installed but the axis is driven with an external amplifier. BR1 disables hall error checking when OE  $\neq 0$

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- Examples of brushless servo amps that require this consideration include the AMP-43040 (-D3040) or the AMP-20540
- Motion Behavior:
  - If an error or axis-specific abort is detected, and the motion was executing an independent move, only that axis will be shut off.
  - If the motion is a part of coordinated mode of the types GM, VM, LM or CM, all participating axes will be stopped.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:OE 1,1,1,1;' Enable OE on all axes
:OE 0;` Disable OE on A-axis, other axes remain unchanged
:OE ,,1,1;' Enable OE on C-axis and D-axis, other axes remain unchanged
:OE 1,0,1,0;' Enable OE on A and C-axis, Disable OE on B and D axis
:MG _OEA;' Query A axis OE setting
1.0000
```

DMC40x0, DMC41x3, DMC18x6

```
#main
'code to enable the OE command for all error conditions
'and setup the corresponding automatic subroutines
'to display relevant data
'no loop for abort input, as that stops code operation
OE 3,3,3,3
SH ABCD
JG*= 1000;' all jog at 1000
BG ABCD
#loop
'endless loop
WT 1000
JP #loop
EN

#AMPERR
MG "amplifier fault"
MG _TA0,_TA1,_TA2,_TA3
EN

#POSERR
MG "position error fault"
MG _TEA,_TEB,_TEC,_TED
EN

#LIMSWT
MG "limit switch fault"
MG _TSA,_TSB,_TSC,_TSD
EN
```



<b>OF</b>	<i>Offset</i>	Filter/Control
The OF command sets a bias voltage in the command output or returns a previously set value		

OFA= n	Arguments specified with an axis mask and an assignment (=)
OF n ...	Arguments specified with an implicit comma-separated order
OFm	Operand holds the value set in the command

## DESCRIPTION

The OF command sets a bias voltage in the command output or returns a previously set value.

## ARGUMENTS

**OFm= n**

**OF n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-9.9982	9.9982	0	20/65,536	Offset voltage applied to MCMD	

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- This can be used to counteract gravity or an offset in an amplifier.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:OF 1,-2,3,5; Set A-axis offset to 1, the B-axis offset to -2, the C-axis to 3, and the D-axis to 5
:OF -3; Set A-axis offset to -3 Leave other axes unchanged
:OF ,0; Set B-axis offset to 0 Leave other axes unchanged
:OF ?,?,?,?; Return offsets
-3.0000,0.0000,3.0000,5.0000
:OF ?; Return A offset
-3.0000
:OF ,?; Return B offset
0.0000
```

OF is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OP</b>	<i>Output Port</i>	<b>IO</b>
The OP command sets the output ports of the controller in a bank using bitmasks		

OP n ...	Arguments specified with an implicit comma-separated order
_OP0, _OP1, _OP2, _OP3, _OP4	Operand holds the value set in the command

## DESCRIPTION

The OP command sets the output ports of the controller in a bank using bitmasks. Arguments to the OP command are bit patterns (decimal or hex) to set entire banks (bytes) of digital outputs. Use SB, CB or OB to set bits individually.

## ARGUMENTS

**OP n0,n1,n2**

**OP n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	65535	0	1	Decimal representation: General Outputs 1-16	On a 1-4 axis controller, max is 255 (\$FF) for outputs 1-8 only.

## REMARKS

- Bit patterns for extended I/O banks (where available) configured as inputs have no affect on the IO status.
- At the time of release, there are no extended I/O options for the DMC-41x3. Contact Galil if extended I/O is required.

## Output Mapping Examples

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Example	Command Issued (Hex version)	Bits Set	Bits Cleared
1-4 axis Set all outputs	OP255 (OP\$FF)	1-8	-
5-8 axis Set all outputs	OP65535 (OP\$FFFF)	1-16	-
Clear all outputs	OP0 (OP\$0000)	-	1-16
Alternating on/off	OP43690 (OP\$AAAA)	2,4,6,8,10,12,14,16	1,3,5,7,9,11,13,15
Set High Byte	OP65280 (OP\$FF00)	9-16	1-8
Set Low Byte	OP255 (OP\$00FF)	1-8	9-16

## EXAMPLES

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
OP 0;!  
OP $85;!  
MG_OP0;!
```

Clear Output Port -- all bits  
Set outputs 1,3,8 and clear the others  
Returns the parameter "n0"

OP is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OT</b>	Off on encoder failure time	ERROR CONTROL
The OT command sets the timeout time for the encoder failure routine		
		

OTA= n	Arguments specified with an axis mask and an assignment (=)
OT n ...	Arguments specified with an implicit comma-separated order
OTm	Operand holds the value set in the command

## DESCRIPTION

The OT command sets the timeout time for the encoder failure routine. The command sets the time in samples that the encoder failure will wait for motion after the OV threshold has been exceeded. The controller can detect a failure on either or both channels of the encoder.

## ARGUMENTS

**OTm= n**

**OT n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	1	32,000	30	1	Number of samples for error detection	

## REMARKS

- Encoder error detection is based on whether motion of at least 4 counts is detected whenever the torque exceeds a preset level (OV) for a specified time (OT).
  - Note that for this function to work properly it is necessary to have a non-zero value for KI.
- See the OA command for more details on this error detection mode

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
OTD= 400; Set D axis encoder error timeout to 400 samples
OT 100,200; Set A axis to 100 and B axis to 200 sample timeouts
```

```
#setup
OTA= 10; Set time to 10 milliseconds
OVA= 5; Set voltage to 5
OAA= 1; Enable encoder detection feature
EN
```

```
REM #POSERR example for checking to see if encoder failure occurred
REM This procedure is needed because the stop code will only update if
REM the profilier is running at the time the encoder failure is detected.
```

```
#POSERR
~a= 0
#loop
IF _MO~a=1
  IF (( _TE~a < _ER~a) & ( _OE~a) & ( _OA~a))
    MG "possible encoder failure on ",~a{Z1.0}, " axis"
  ENDIF
ENDIF
~a= ~a+1
JP #loop,~a<_BV
AI 1; wait for input 1 to go high
SH ; enable all axes
RE
```

OT is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>OV</b>	Off on encoder failure voltage	ERROR CONTROL
The OV command sets the threshold voltage for detecting an encoder failure		

OVA= n	Arguments specified with an axis mask and an assignment (=)
OV n ...	Arguments specified with an implicit comma-separated order
_OVm	Operand holds the value set in the command

## DESCRIPTION

The OV command sets the threshold voltage for detecting an encoder failure. The controller can detect a failure on either or both channels of the encoder.

## ARGUMENTS

**OVm= n**

**OV n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	9.9982	0.9438	20/65,536	Torque voltage to trigger encoder error detection	

## REMARKS

- Encoder error detection is accomplished by checking on whether motion of at least 4 counts is detected whenever the torque exceeds a preset level (OV) for a specified time (OT).
  - Note that for this function to work properly it is recommended to have a non-zero value for KI.
- The value of OV should be high enough to guarantee that the motor would overcome any static friction in the system. If it is too low, there will be false triggering of the error condition.
- The OV value may not be higher than the TL value.
- See the OA command for more details on this error detection mode

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6

```
OVB= 1.2; ' Set B axis encoder detection torque value to 1.2V
OV 0.54; ' Set A axis encoder detection torque value to 0.54V
```

```
#setup
'setup the encoder error detection
OTA= 10; ' Set time to 10 milliseconds
OVA= 5; ' Set voltage to 5
OAA= 1; ' Enable encoder detection feature
EN
```

```
REM #POSERR example for checking to see if encoder failure occurred
REM This procedure is needed because the stop code will only update if
REM the profilier is running at the time the encoder failure is detected.
```

```
#POSERR
~a= 0
#loop
IF _MO~a=1
  IF (( _TE~a<_ER~a) & ( _OE~a) & ( _OA~a))
    MG "possible encoder failure on ",~a{Z1.0}, " axis"
  ENDIF
ENDIF
~a= ~a+1
JP #loop,~a<_BV
AI 1; '          wait for input 1 to go high
SH ; '          enable all axes
RE
```

OV is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>P2CD</b>	<i>Serial port 2 code</i>	<b>OPERAND ONLY</b>
<b>P2CD returns the status of the auxiliary serial port (port 2)</b>		
		

variable= P2CD	Holds a value
P2CD	Operand has special meaning see Remarks

## DESCRIPTION

P2CD returns the status of the auxiliary serial port (port 2). The value of P2CD returns zero after the corresponding string or number is read.

## ARGUMENTS

### P2CD

P2CD is an operand that holds a value cooresponding to status. See Examples for use in code.

## REMARKS

- P2CD contains the following status codes

*P2CD Status Codes*

Status Code	Meaning
-1	Mode disabled
0	Nothing received
1	Received character, but not carriage return
2	received a string, not a number
3	received a number

## EXAMPLES

```
:^R^V
DMC2240 Rev 1.00
:^R^S
:CC 9600,0,0,0
:MG "TEST" {P2};' send a message to the hand terminal
:MG P2CD;` no characters entered on hand terminal
0.0000
:MG P2CD;` the number 6 was pushed on the hand terminal
1.0000
:MG P2CD;` enter key pushed on hand terminal
3.0000
:MG P2CD;` the character B was pushed (shift f2) then enter
2.0000
```

P2CD is supported on DMC40x0, DMC41x3 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>P2CH</b>	<i>Serial port 2 character</i>	<b>OPERAND ONLY</b>
<b>P2CH returns the last character sent to the auxiliary serial port (port 2)</b>		
		

variable= P2CH	Holds a value
P2CH	Operand has special meaning see Remarks

## DESCRIPTION

P2CH returns the last character sent to the auxiliary serial port (port 2)

## ARGUMENTS

### P2CH

P2CH is an operand that holds a value cooresponding to ASCII characters sent over the serial port. See Examples for use in code.

## REMARKS

- None

## EXAMPLES

```
:^R^V
DMC2240 Rev 1.0o
:^R^S
:CC 9600,0,0,0
:MG "TEST" {P2} ;'send a message to the hand terminal
:MG P2CH {S1} ;'the 6 button was pushed on the hand terminal
6
:
```

P2CH is supported on DMC40x0, DMC41x3 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>P2NM</b>	<i>Serial port 2 number</i>	<b>OPERAND ONLY</b>
<b>P2NM returns the last number (followed by carriage return) sent to auxiliary serial port (port 2)</b>		
		

variable= P2NM	Holds a value
P2NM	Operand has special meaning see Remarks

## DESCRIPTION

P2NM returns the last number (followed by carriage return) sent to auxiliary serial port (port 2).

## ARGUMENTS

### P2NM

P2NM is an operand that holds a numerical value sent over the serial port. See Examples for use in code.

## REMARKS

- Converts from ASCII (e.g. "1234") to binary so that a number can be stored into a variable and math can be performed on it.
  - Numbers from -2147483648 to 2147483647 can be processed.

## EXAMPLES

```
:^R^V
DMC2240 Rev 1.0o
:^R^S
:CC 9600,0,0,0
:MG "TEST" {P2} ;'send a message to the hand terminal
:x = P2NM ;'the 1, 2, 3,  buttons were pushed
:MG x
123.0000
:
```

P2NM is supported on DMC40x0, DMC41x3 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>P2ST</b>	<i>Serial port 2 string</i>	<b>OPERAND ONLY</b>
<b>P2ST returns the last string (followed by carriage return) sent to auxiliary serial port (port 2)</b>		
		

variable= P2ST	Holds a value
P2ST	Operand has special meaning, see Remarks

## DESCRIPTION

P2ST returns the last string (followed by carriage return) sent to auxiliary serial port (port 2)

## ARGUMENTS

### P2ST

P2ST is an operand that contains a string. See Examples for usage.

## REMARKS

- No more than 6 characters can be assessed by this operand

## EXAMPLES

```
:CC 9600,0,1,0
:MG "TEST" {P2} ;'send a message to the hand terminal
:MG P2ST {S3} ;'the characters ABC were entered
ABC
```

P2ST is supported on DMC40x0, DMC41x3 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PA</b>	<i>Position Absolute</i>	INDEPENDENT MOTION
The PA command sets the end target of the Position Absolute Mode of Motion		
		

PAA= n	Arguments specified with an axis mask and an assignment (=)
PA n ...	Arguments specified with an implicit comma-separated order
_PAm	Operand has special meaning see Remarks

## DESCRIPTION

The PA command sets the end target of the Position Absolute Mode of Motion.

## ARGUMENTS

**PAm= n**

**PA n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Virtual axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	Absolute position target for independent move	n=? returns the commanded position at which motion last stopped

## REMARKS

- The position is referenced to the absolute zero position, defined as position 0.
- By default a new PA command may not be issued before the previous PA command has finished executing. This operation may be changed by running in Position Tracking Mode - See the PT command for more information.

## Operand Usage

- \_PAm contains the last commanded position at which motion stopped.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:PA 400,-600,500,200; A-axis will go to 400 counts B-axis will go to -600 counts
:BG ;' C-axis will go to 500 counts D-axis will go to 200 counts
:PA ?,?,?,?;' Execute Motion
400, -600, 500, 200 Returns the current commanded position after motion has completed
:PA 700; A-axis will go to 700 on the next move while the
:BG ;' B,C and D-axis will travel the previously set relative distance
:EN if the preceding move was a PR move, or will not move if the
preceding move was a PA move.
```

```
DP 10000; set current position to 10000
PA 3000; move to absolute position 3000, which is a -7000 count move
BG A; begin -7000 count move
EN
```

PA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PF</b>	<i>Position Format</i>	SYSTEM CONFIG
The PF command allows the user to format the position numbers such as those returned by TP		
		

PF n ...	Arguments specified with an implicit comma-separated order
_PF	Operand holds the value set in the command

## DESCRIPTION

The PF command allows the user to format the position numbers such as those returned by TP. The number of digits of integers and the number of digits of fractions can be selected with this command. An extra digit for sign and a digit for decimal point will be added to the total number of digits.

## ARGUMENTS

### PF n0.n1

Argument	Min	Max	Default	Resolution	Description	Notes
n0	-8	10	10	1	Number of places displayed preceding the decimal point	Negative numbers force data to display in hexadecimal format
n1	0	4	0	1	Number of places displayed after the decimal point	

## REMARKS

- If PF is minus, the format will be hexadecimal and a dollar sign will precede the characters. Hex numbers are displayed as 2's complement with the first bit used to signify the sign.
- If a number exceeds the format, the number will be displayed as the maximum possible positive or negative number (i.e. 999.99, -999, \$8000 or \$7FF).
- The PF command formats the values returned from the following commands:

BL?	IP ?	TD
DE?	LE?	TE
DP?	PA?	TN
EM?	PR?	TP
FL?	RL	VE
GP	RP	

## EXAMPLES

```
:TP A;` Tell position of X
0000000000 Default format
:PF 5.2;` Change format to 5 digits of integers and 2 of fractions
:TP A;` Tell Position
00021.00
:PF -5.2;` New format. Change format to hexadecimal
:TP A;` Tell Position
$00015.00 Report in hex
```

PF is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PL</b>	Pole	Filter/Control
The PL command adds a low-pass filter in series with the PID compensation		

PLA= n	Arguments specified with an axis mask and an assignment (=)
PL n ...	Arguments specified with an implicit comma-separated order
PLm	Operand holds the value set in the command

## DESCRIPTION

The PL command adds a low-pass filter in series with the PID compensation.

DMC40x0, DMC41x3, DMC30010

The crossover frequency is entered directly as an argument to PL. To maintain compatibility with earlier versions, a value less than 1 may also be specified.

## ARGUMENTS

**PLm= n**

**PL n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC30010

### Frequency Argument

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	250	0	1	Crossover frequency created by the PL command	'Max' is a function of TM. See Remarks

DMC40x0, DMC41x3, DMC30010

### Calculated Pole Argument (deprecated)

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	0.999	0	2/65536	Value used to generate pole filter crossover frequency	See Remarks for the equation used. n = 0 disables the Pole filter

## REMARKS

DMC40x0, DMC41x3, DMC30010

- At lower TM settings, the maximum pole frequency is increased. The maximum value of the PL command is determined by the value of TM according to the following equation
  - Max =  $(1/4 * 10^6) * (1/TM)$
- The digital transfer function of the filter is  $(1 - n) / (Z - n)$  and the equivalent continuous filter is  $A/(S+A)$  where A is the filter cutoff frequency:  $A = (1/T) \ln(1/n)$  rad/sec and T is the sample time.

### Calculated Pole

- To convert from the desired crossover (-3 dB) frequency in Hertz to the value given to PL, use the following formula

$$n = e^{-T \cdot f_c \cdot 2\pi}$$

- where
  - n is the argument given to PL
  - T is the controller's servo loop sample time in seconds (TM divided by 1,000,000)
  - Fc is the crossover frequency in Hertz
- Example: Fc=36Hz TM=1000 n= $e^{-0.001*36*2*\pi} = 0.8$
- The following shows several example crossover frequencies achieved with various values of PL

n	Fc (Hz)
0	Infinite (off)
0.2	256
0.4	145
0.6	81
0.8	36
0.999	0

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Set A-axis Pole to 0.95, B-axis to 0.9, C-axis to 0.8, D-axis pole to 0.822
:PL .95,.9,.8,.822
Query all Pole values
:PL ?,?,?,?
0.9527,0.8997,0.7994,0.8244
Return A Pole only
:PL ?
0.9527
```



<b>PR</b>	<i>Position Relative</i>	INDEPENDENT MOTION
The PR command sets the incremental distance and direction of the next move		

PRa=n	Arguments specified with an axis mask and an assignment (=)
PR n ...	Arguments specified with an implicit comma-separated order
_PRm	Operand holds the value set in the command

## DESCRIPTION

The PR command sets the incremental distance and direction of the next move. The move is referenced with respect to the current position. .

## ARGUMENTS

**PRm= n**

**PR n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Virtual axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	N/A	1	Incremental distance for independent move	<b>n = ?</b> returns the current incremental distance specified

## REMARKS

- \_PRm contains the current incremental distance for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:PR 100,200,300,400;' On the next move the A-axis will go 100 counts,
:BG ;' the B-axis will go to 200 counts forward, C-axis will go 300 counts and the D-axis will go 400 counts.
:PR ?,?,?;' Return relative distances
100,200,300
:PR 500;' Set the relative distance for the A axis to 500
:BG ;' The A-axis will go 500 counts on the next move while the B-axis will go its previously set relative distance.
```

```
'using PA/PR, you can query PR for the incremental distance
:DP 10000
:PA 8000
:PR ?
-2000
```

PR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PT</b>	<i>Position Tracking</i>	INDEPENDENT MOTION
The PT command will place the controller in the position tracking mode		

PTA=n	Arguments specified with an axis mask and an assignment (=)
PT n ...	Arguments specified with an implicit comma-separated order
_PTm	Operand holds the value set in the command

## DESCRIPTION

The PT command will place the controller in the position tracking mode. In this mode, the controller will allow the user to issue absolute position commands that begin motion immediately without requiring a BG command. The absolute position may be specified such that the axis will begin motion, continue in the same direction, reverse directions, or decelerate to a stop.

## ARGUMENTS

**PTm=n**

**PT n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	Setting for position tracking mode of motion	n = 1 enables PT mode, n = 0 disables PT mode

## REMARKS

- The PA command is used to give the controller an absolute position target. Motion commands other than PA are not supported in this mode.
- The motion profile is trapezoidal with the parameters controlled by acceleration, deceleration, and speed (AD, DC, SP).
- When in the PT mode the ST command will exit the mode.
- The AM and MC trip points are not valid in this mode.
  - MF and MR are recommended with this mode as they allow the user to specify both the absolute position, and the direction. The AP trip point may also be used.
- Position Tracking is not valid on virtual axes

## EXAMPLES

```
DPA= 0; ' Start position at absolute zero
PTA= 1; ' Start PT mode on A axis
PA 1000; ' Move to position 1000, motion starts right away
MF 500; ' Wait till position 500 reached
PA -1000; ' Reverse direction to move to position -1000
EN
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#a
PT 1,1,1,1; ' Enable the position tracking mode for axes A, B, C, and D
NOTE: The BG command is not used to start the PT mode.
#loop;
' Create label #LOOP in a program. This small program will
' update the absolute position at 100 Hz. Note that the
' user must update the variables v1, v2, v3 and v4 from the
' host PC, or another thread operating on the controller.

PA v1,v2,v3,v4; ' Command ABCD axes to move to absolute positions. Motion
' begins when the command is processed. BG is not used
' to begin motion in this mode. In this example, it is
' assumed that the user is updating the variable at a
' specified rate. The controller will update the new
' target position every 10 milliseconds (WT10).
WT 10; ' Wait 10 milliseconds
JP #loop; ' Repeat by jumping back to label LOOP
```

PT is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PV</b>	<i>PVT Data</i>	<b>PVT MODE</b>
The PV command is used to enter PVT data into the PVT buffer		

<b>PVA=n</b>	Arguments specified with an axis mask and an assignment (=)
<b>_PVm</b>	Operand has special meaning, see Remarks

## DESCRIPTION

The PV command is used to enter PVT data into the PVT buffer. Data is entered by specifying the target delta position, target velocity, and delta time for the segment duration.

## ARGUMENTS

**PVm= n0,n1,n2**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n0</b>	-30,000,000	30,000,000	0	1	Position target for PVT segment	
<b>n1</b>	-15,000,000	15,000,000	0	2	Velocity target for PVT segment	
<b>n2</b>	0	2048	0	2	Number of samples for PVT segment	t = -1 clears the PVT buffer, t = 0 exits PVT mode. See Remarks

## REMARKS

- n2 is in samples and sample time is defined by TM
  - With TM 1000 set, n2 = 1024 is equal to 1 second
- If t is omitted from the PVT command, the previous n2 value is used
- For more details on PVT mode of motion see the user manual.

## Operand Usage

DMC40x0, DMC41x3

- \_PVm contains the number of spaces available in the PV buffer for the specified axis. Each axis has a 255 segment PVT buffer

## EXAMPLES

```

PVA= 100,2000,256; ' Move 100 counts over 256 samples, end at 2000 cnts per sec
PVA= 500,1000,128; ' Move 500 counts over 128 samples, end at 1000 cnts per sec
PVA= 1000,2500; ' Move 1000 counts over 128 samples, end at 2500 cnts per sec
PVA= 0,0,0; ' End PVT mode

```

DMC40x0, DMC41x3

Desired X/Y Trajectory

XPosition (relative/absolute)	XSpeed at end of time period (c/s)	Time (ms at TM1000) (relative/time from start)	YPosition (relative/absolute)	YSpeed at end of time period (c/s)	Time (ms at TM1000) (relative/time from start)
0/0	0	0/0	0/0	0	0/0
100/100	200	256/256	-50/-50	500	100/100
200/300	200	50/306	-100/-150	-100	510/610
300/600	0	50/356	300/150	0	50/660

DMC40x0, DMC41x3

```

DP 0,0; ' Define zero position
PVA= 100,200,256; ' Command X axis to move 100 counts reaching an ending speed of 200c/s in 256 samples
PVB= -50,500,100; ' Command Y axis to move -50 counts reaching an ending speed of 500c/s in 100 samples
PVB= -100,-100,510; ' Command Y axis to move -100 counts reaching an ending speed of -100c/s in 510 samples
PVA= 200,200,50; ' Command X axis to move 200 counts reaching an ending speed of 200c/s in 50 samples
PVA= 300,0,50; ' Command X axis to move 300 counts reaching an ending speed of 0c/s in 50 samples
PVB= 300,0,50; ' Command Y axis to move 300 counts reaching an ending speed of 0c/s in 50 samples
PVB= ,0; ' Exit PVT mode on Y axis
PVA= ,0; ' Exit PVT mode on X axis
' When the PVT mode is exited, the axis will be in the "SH" state
' (assuming position error is not exceeded, etc)
BT AB; ' Begin PVT on X and Y axis
AM AB; ' Trip point will block until PVT motion on X AND Y is complete
EN; ' End program

```

PV is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)

Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>PW</b>	<i>Password</i>	SYSTEM CONFIG
The PW command sets the password used to lock the controller		

PW n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The PW command sets the password used to lock the controller. Locking the controller prevents interrogation of the controller program space.

## ARGUMENTS

**PW str,str**

**PW n,n**

Argument	Min	Max	Default	Resolution	Description	Notes
str	0 chars	8 chars	""	String	String to be used for password	Both parameters must match for the PW command to succeed

## REMARKS

- The password can only be changed when the controller is in the unlocked state. See the ^L^K for more details.
- The password is burnable but cannot be interrogated. If you forget the password and the controller is locked you must master reset the controller to gain access.

## EXAMPLES

```
:PW apple,orange
?
:TC 1
138 Passwords not identical
:PW apple,apple
:^L^K apple,1
```

```
:PW test,test;          Set password to "test"
:^L^K test,1;          Lock the program
:ED;                  Attempt to edit program
?
:TC 1
106 Privilege violation
```

PW is supported on DMC40x0, DMC41x3, RIO, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QD</b>	Download Array	SYSTEM CONFIG
The QD command transfers array data from the host computer to the controller		
		

QD n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The QD command transfers array data from the host computer to the controller. QD array[], start, end requires that the array name be specified along with the index of the first element of the array and the index of the last element of the array.

## ARGUMENTS

**QD str[],n0,n1**

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	7 chars	N/A	String	Name of array to receive data via download.	
n0	0	see Notes	0	1	Index of the first array element.	Value cannot exceed size of array - 2
n1	1	see Notes	see Notes	1	Index of the last array element.	Value cannot exceed size of array - 1. Defaults to size of array - 1.

## REMARKS

- Array name must be a valid, dimensioned array name followed by empty [] brackets.
- The array elements can be separated by a comma ( , ) or by CR/LF.
- The downloaded array is terminated by a \ character.
- QD is not supported in the GalilTools terminal
  - It is recommended to use the array download functions available through the GalilTools software and drivers rather than directly using the QD command.

## EXAMPLES

```
: 'From a character-buffered terminal such as Telnet or Hyperterm
:DM array[3]
:QD array[]
1,2,3\:LA
array[3]
:array[0]= ?
1.0000
:array[1]= ?
2.0000
:array[2]= ?
3.0000
:
```

QD is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QH</b>	Hall State	INTERROGATION
<b>The QH command transmits the state of the Hall sensor inputs</b>		

QH A	Argument is an axis mask
_QHm	Operand has special meaning see Remarks

## DESCRIPTION

The QH command transmits the state of the Hall sensor inputs. The value is decimal and represented by a 3 bit value (see Remarks).

## ARGUMENTS

### QH mm

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to return hall status	

## REMARKS

- The 3 bit value returned by QH is defined in the table below:

Bit	Status
07	Undefined (set to 0)
06	Undefined (set to 0)
05	Undefined (set to 0)
04	Undefined (set to 0)
03	Undefined (set to 0)
02	Hall C State
01	Hall B State
00	Hall A State

- QH should return a value from 1 through 6 as valid hall combinations. A value of 0 or 7 is invalid when using halls and will generate a hall error with OE set.
  - The valid sequence for hall inputs is a grey code output (only one bit changes at a time):
    - 1,3,2,6,4,5 (or 5,4,6,2,3,1)
  - To disable hall error checking, set the axis to brushed with a BR 1 command.
- When using an internal sine amplifier, the BA command must be issued before QH will report the hall state status.

## Operand Usage

- \_QHm Contains the state of the Hall sensor inputs for the specified axis

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3

```
:QH A; ' Query the A axis hall state
var=_QHb; ' Set a variable var equal to the B axis hall state
```

```
:QH A; ' Query A axis hall status
7
:TA 1; ' Check for hall errors in the amp
1
:'A 1 indicates hall error on axis A
```

QH is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QR</b>	<i>I O Data Record</i>	INTERROGATION
The QR command causes the controller to return a record of information regarding controller status		
		

QR A Argument is an axis mask

## DESCRIPTION

The QR command causes the controller to return a record of information regarding controller status.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

This status information includes 4 bytes of header information and specific blocks of information as specified by the command arguments. The details of the status information is described in Chapter 4 of the user's manual.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**QR mm**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGHSTI	ABCDEFGHSTI	Multi-Axis Mask	Axes/Coordinated/IO data specified to display in the data record	If no argument entered, mm = "ABCDEFGHSTI"

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Value	Description	Notes
<b>mm</b>	A-H	Output axes A-H data record block	
	S	Output coordinated axis S data block	
	T	Output coordinated axis T data block	
	I	Output General IO data block	

## REMARKS

- The data returned by the QR command is in binary format and is unreadable in programs such as Galiltools.
  - The Galiltools API has specialized commands to parse the data record packet. See the Galiltools User Manual for more details.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
QR A; ' Return the data record with A axis block only
QR BI; ' Return the data record with B axis block and IO block
QR ST; ' Return the data record with S and T coordinated axis blocks
QR ;' Return the data record for all axes, including IO and S and T axis blocks
```

QR is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QS</b>	Error Magnitude	INTERROGATION, STEPPER MOTOR
The QS command reports the magnitude of error, in drive step counts, for axes in Stepper Position Maintenance mode		

QS A	Argument is an axis mask
QSm	Operand has special meaning see Remarks

## DESCRIPTION

The QS command reports the magnitude of error, in drive step counts, for axes in Stepper Position Maintenance mode. A step count is directly proportional to the micro-stepping resolution of the stepper drive.

## ARGUMENTS

**QS mm**

**QSm= ?**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to query for step motor error magnitude	Default value used if mm is undefined.
m	A	H	N/A	Axis	Single Axis to query for error magnitude	

## REMARKS

- The result of QS is modularized so that result is never greater than 1/2 the revolution of the stepper motor.
  - Largest possible QS result =  $0.5 * YA * YB$

## Operand Usage

- \_QSm contains the error magnitude in drive step counts for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'For an SDM-20620 microstepping drive, query the error of B axis:  
:qsb= ?  
253 This shows 253 step counts of error. The SDM-20620 resolution is 64 microsteps per full motor step, nearly four full motor steps of error.  
Query the value of all axes:  
:QS  
0,253,0,0,0,0,0,0 Response shows all axes error values
```

QS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QU</b>	<i>Upload Array</i>	<b>INTERROGATION</b>
The QU command transfers array data from the controller to a host computer		

QU n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The QU command transfers array data from the controller to a host computer. The QU requires that the array name be specified along with the first element of the array and last element of the array.

## ARGUMENTS

**QU str[],n0,n1,n2**

Argument	Min	Max	Default	Resolution	Description	Notes
<b>str</b>	1 char	7 chars	N/A	String	Name of array to be uploaded	
<b>n0</b>	0	See Notes	0	1	Index of first array element	Value cannot exceed size of array - 2
<b>n1</b>	1	See Notes	See Notes	1	Index of last array element	Defaults to last element of array. Value cannot exceed size of array - 1
<b>n2</b>	0	1	0	1	Selects character delimiter between array elements	n2 = 0 selects CR delimiting, n2 = 1 select comma delimiting

## REMARKS

- Array name must be a valid, dimensioned array name followed by empty [] brackets.
- The uploaded array will be followed by a Z as an end of text marker.
- The GalilTools array upload functions can be used to upload array data in .csv format.

## EXAMPLES

```
DM test[10];' Dimension a 10 element sized array
QU test[],0,1,1;' Upload first 2 elements
QU test[],8,9,1;' Upload last 2 elements (size=2 and size=1 used for n1,n2)
EN

:DM array[5];' Dimension Array
:QU array[],0,4,1;' Upload Array
0.0000, 0.0000, 0.0000, 0.0000, 0.0000
:array[0]= 9;' Set value
:array[1]= 1
:QU array[],0,4,1
9.0000, 1.0000, 0.0000, 0.0000, 0.0000
:array[0]= ?;' Alternative method to return just one array value
9.0000
```

QU is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>QZ</b>	<i>Return Data Record information</i>	INTERROGATION
The QZ command is an interrogation command that returns information regarding the data record		

**QZ** Command takes no arguments

## DESCRIPTION

The QZ command is an interrogation command that returns information regarding the data record. The controller's response to this command will be the return of 4 integers separated by commas.

## ARGUMENTS

### QZ

QZ is an interrogation command with no parameters.

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- The four fields returned by QZ represent the following
  - First field returns the number of axes.
  - Second field returns the number of bytes to be transferred for general status
  - Third field returns the number bytes to be transferred for coordinated move status
  - Fourth field returns the number of bytes to be transferred for axis specific information

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:QZ?'
4, 52, 26, 36
```

standard DMC-4143 example response

QZ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RA</b>	Record Array	PROGRAMMING
The RA command selects the user arrays to be populated by the Record Array function		
		

RA n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The RA command selects the user arrays to be populated by the Record Array function. The data to be captured is specified by the RD command and time interval by the RC command.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**RA str[ ],str[ ],str[ ],str[ ],str[ ],str[ ],str[ ]**

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	7 chars	N/A	String	Valid array name to use in record array function	The arrays listed correspond to the source list defined by the RD command. See Remarks

## REMARKS

- The array name str must be followed by the [] brackets. Those brackets must be empty.
- The array name str must be a valid array defined by the DM command and reported by LA.

## EXAMPLES

```
' try to start record array without defining array[]
:RA array[]?
:TC 1
82 Undefined array
:DM array[100]
:RA array[]
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#record; Label
DM pos[100]; Define array
RA pos[]; Specify Record Mode
RD _TPA; Specify data type for record
RC 1; Begin recording at 2 msec intervals
PR 1000;BG ; Start motion
EN; End

'The record array mode is useful for recording the real-time motor position during motion.
'The data is automatically captured in the background and does not interrupt the program sequencer.
'The record mode can also be used for a teach or learn of a motion path.

'The GalilTools Realtime scope can often be used as an alternative to record array.
```

RA is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RC</b>	Record	PROGRAMMING
The RC command begins recording for the Automatic Record Array Mode		
		

RC n ...	Arguments specified with an implicit comma-separated order
_RC	Operand has special meaning, see Remarks

## DESCRIPTION

The RC command begins recording for the Automatic Record Array Mode. RC 0 stops recording. The record array mode loads source data specified by the RD command into the arrays defined by the RA command. The address for the array element for the next recording can be interrogated with \_RD.

## ARGUMENTS

### RC n0,n1

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
n0	0	8	0	1	Specify the record array time interval as $2^n$ samples.	n0 = 0 stops recording
n1	see Notes	see Notes	0	1	Specify the number of records to perform	n1 has special rules for the maximum setting. See Remarks.

## REMARKS

- Firmware Note: Do not allocate or deallocate arrays (DM,DA) while the Automatic Record Array Mode is running
- GalilTools Note: Do not download arrays from GalilTools, or call the arrayDownload() or arrayDownloadFile() functions while automatic record array mode is running
- n0 = non zero number automatically starts record mode.
- n0 = ? returns status of recording '1' if recording, '0' if not recording

### Second Parameter Rules

- n1 specifies the last array element to use for record mode.
- If arrays specified by RA have different sizes, the smallest array size is the maximum value for n1
- If n1 = 0 or not specified, the maximum value is used.
- A negative value for n1 specifies circular (continuous) record over array addresses 0 to (n1-1).
  - The absolute value of the minimum n1 allowed = maximum n1 allowed

### Operand Usage

- \_RC contains status of recording '1' if recording, '0' if not recording

### Setting up the record array mode

- Dimension an array/arrays for storing data. Make sure you dimension the array with the number of elements required to capture data for your application.
- Set the RA command with the arrays to be used for recording
- Set the RD command with the data sources to be applied to the arrays. The order of your arrays entered into RA will match the order of data sources set by RD
- Set the RC command to get the desired time between records and enable the recording
- Monitor the \_RC operand for a 0 to indicate recording is done.
- View the data in your embedded code, or extract the data using Galiltools software and the Upload array function.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#record;
DM torque[1000];          Record label
Define Array
RA torque[];               Specify Array to record data
RD _TTA;                   Specify Data Type
RC 2;                      Begin recording and set 4 msec between records
JG 1000:BG ;               Begin motion
#a;JP #a,_RC=1;            Loop until done
MG "DONE RECORDING";      Print message
EN;                        End program
```

RC is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RD</b>	Record Data	PROGRAMMING
The RD command specifies the data type to be captured for the Record Array (RA) mode		

RD n ...	Arguments specified with an implicit comma-separated order
_RD	Operand has special meaning see Remarks

## DESCRIPTION

The RD command specifies the data type to be captured for the Record Array (RA) mode. The data defined in this command is stored in arrays defined by the RA command at the time interval specified with the RC command.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### RD arg, arg, arg, arg, arg, arg, arg, arg

Argument	Min	Max	Default	Resolution	Description	Notes
arg	N/A	N/A	N/A	N/A	Source to be captured using the record array feature	The order of args specified in RD corresponds with the array order specified in the RA command.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

*Valid arguments for RD command*

Argument	Value	Description	Notes
arg	TIME	Time in servo samples	Value as read by the TIME command
	_AFm	Analog input digital value	Data range is -32768 to 32767. The analog inputs are limited to those which correspond to an axis on the controller.
	_DEm	2nd encoder position	
	_TPm	Encoder position	
	_TEM	Position error	
	_RPm	Commanded position	RPm and _SHm capture the same data
	_SHm	Commanded position	RPm and _SHm capture the same data
	_RLm	Latched position	
	_TI	Input status	
	_OP	Output status	
	_TSm	Switches	Only bits 0-4 valid
	_SCm	Stop code	
	_TTm	Torque command	The values recorded for torque are in the range of +/- 32767 where 0 is 0 torque, -32767 is -10 volt command output, and +32767 is +10 volt.
	_TVm	Filtered velocity	This value will be 64 times greater than TV command
	_TDm	Stepper position	

## REMARKS

- Arguments listed as \_XXm are valid when m is a valid axis mask

### Operand Usage

- \_RD contains the address for the next array element for recording

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
DM errora[50],errorb[50];' Define arrays
RA errora[],errorb[];' Specify arrays to be recorded
RD _TEA,_TEB;' Specify data source
RC 1;' Begin recording, period is once every other servo sample
JG 1000;BG ;' Begin motion
```

'The GalilTools Realtime scope can often be used as an alternative to record array.

RD is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RE</b>	<i>Return from Error Routine</i>	PROGRAMMING
The RE command is used to end subroutines in application code		
		

RE Only valid from within embedded code. No terminal.

## DESCRIPTION

The RE command is used to end subroutines in application code. An RE at the end of these routines causes a return to the main program. Specific automatic error subroutines require the use of the RE command to end the code correctly.

## ARGUMENTS

### RE n

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	1	0	1	Determines state of interrupted trippoint when returning from an automatic subroutine.	n = 1 restores the interrupted trippoint. n = 0 clears the trippoint

## REMARKS

- The RE command is used to end the following error automatic subroutines.

DMC40x0, DMC41x3, DMC21x3, DMC30010

Automatic Subroutines Used	Notes
#AMPERR	Only when using internal amps
#LIMSWI	
#POSERR	
#SERERR	Only when equipped with serial encoder firmware support
#TCPERR	

- Care should be taken to ensure the error conditions are cleared when finishing the subroutine to avoid immediate re-entering of the error routine.
- To avoid returning to the main program on an interrupt, use the ZS command to zero the subroutine stack, then use JP to return to the desired location in code.
- RE 1 restores the trippoint that was interrupted by an automatic subroutine (like WT)

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- A motion trippoint like MF or MR requires the axis to be actively profiling in order to be restored with the RE 1 command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
REM dummy loop
#a
JP #a
EN

#POSERR; ' Begin Error Handling Subroutine
MG "ERROR"; ' Print message
SB 1; ' Set output bit 1
RE; ' Return to main program and clear trippoint
```

RE is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>REM</b>	<i>Remark</i>	PROGRAMMING
<b>REM is used for comment lines</b>		
		

## DESCRIPTION

REM is used for comment lines. The REM statement is NOT a controller command. Rather, it is recognized by Galil PC software, which strips away the REM lines before downloading the DMC file to the controller.

NO (or ') should be used instead of REM for commenting in application code unless speed or program space is an issue.

## ARGUMENTS

### REM str

Argument	Value	Description	Notes
str	String	Comment to be removed from code prior to download	This comment is not limited by the character limit of the controller, as it is never downloaded

## REMARKS

- REM differs from NO (or ') in the following ways:
  - 1. NO (or ') comments are downloaded to the controller and REM comments aren't
  - 2. NO (or ') comments take up execution time and REM comments don't; therefore, REM should be used for code that needs to run fast.
  - 3. REM comments cannot be recovered when uploading a program but NO (or ') comments are recovered. Thus the uploaded program is less readable with REM.
  - 4. NO (or ') comments take up program line space and REM lines don't.
  - 5. REM comments must be the first and only thing on a line, whereas NO (or ') can be used to place comments to the right of code (after a semicolon) on the same line

### Special Strings

- REM DISABLE COMPRESSION
  - Inserting this line into the beginning of your application code disables Galiltools download compression utility. This is not a controller function.

## EXAMPLES

```
REM This comment will be stripped when downloaded to the controller
'This comment will be downloaded and takes some execution time
PRA= 1000 ;'this comment is to the right of the code
```

REM is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RI</b>	<i>Return from Interrupt Routine</i>	PROGRAMMING
The RI command is used to end the input interrupt subroutine		
		

RI Only valid from within embedded code. No terminal.

## DESCRIPTION

The RI command is used to end the input interrupt subroutine.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

The input interrupt subroutine begins with the label #ININT. An RI at the end of this routine causes a return to the main program. The RI command also re-enables input interrupts.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**RI n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	1	0	1	Determines state of interrupted trippoint when returning from an automatic subroutine.	n = 0 clears the trippoint. n = 1 restores the interrupted trippoint.

## REMARKS

- To avoid returning to the main program on an interrupt, use the command ZS to zero the subroutine stack. This turns the jump subroutine into a jump only.
- If the program sequencer was interrupted while waiting for a trippoint, such as WT, RI1 restores the trippoint on the return to the program. RI0 clears the trippoint.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- A motion trippoint like MF or MR requires the axis to be actively profiling in order to be restored with the RI1 command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
#a;II 1;JP #a;EN; ' Program label
#ININT; ' Begin interrupt subroutine
MG "INPUT INTERRUPT"; ' Print Message
SB 1; ' Set output line 1
RI 1; ' Return to the main program and restore trippoint
```

RI is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RL</b>	<i>Report Latched Position</i>	INTERROGATION
<b>The RL command will return the last position captured by the latch</b>		

RL A	Argument is an axis mask
_RLm	Operand has special meaning, see Remarks

## DESCRIPTION

The RL command will return the last position captured by the latch. The latch must first be armed by the AL command and then the appropriate input must be activated. Each axis uses a specific general input for the latch input; see the AL command for information on latch inputs.

## ARGUMENTS

### RL mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to query for latched position	

## REMARKS

- The armed state of the latch can be configured using the CN command.
- The Latch Function works with the main or auxiliary encoder.

### Capturing Stepper Position using the Latch

- When working with a stepper motor without an encoder, the latch can be used to capture the stepper position. Follow the steps below to achieve this.

  1. Place a wire from the controller Step (PWM) output into the main encoder input, channel A+.
  2. Connect the Direction (sign) output into the channel B+ input.
  3. Configure the main encoder for Step/Direction using the CE command.
  4. The latch will now capture the stepper position based on the pulses generated by the controller.

### Operand Usage

- \_RLm contains the latched position of the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:JG ,5000;' Set up to jog the B-axis
:BG B;' Begin jog
:AL B;' Arm the B latch, assume that after about 2 seconds, input goes low
:RL B;' Report the latch
10000
```

RL is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RP</b>	<i>Reference Position</i>	INTERROGATION
The RP command returns the commanded reference position of the motor(s)		

RP A	Argument is an axis mask
_RPM	Operand has special meaning, see Remarks

## DESCRIPTION

The RP command returns the commanded reference position of the motor(s). RP command is useful when operating step motors since it provides the commanded position in steps when operating in stepper mode.

## ARGUMENTS

### RP mm

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report commanded position	
	M	N	N/A	Multi-Axis Mask	Virtual axes to report commanded position	

## REMARKS

- The relationship between RP, TP and TE: TEA equals the difference between the reference position, RPA, and the actual position, TPA.
  - TE = RP - TP
- \_RPM contains the commanded reference position for the specified axis.

## EXAMPLES

```
'Assume that A axis is commanded to be at the position 200
'The returned units are in quadrature counts.
:PF 7;' Position format of 7
:RP
200
:RP A
200 Return the A motor reference position
:PF -6.0;' Change to hex format
:RP
$0000C8
:position = RPA;' Assign the variable, position, the value of RPA
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Assume that ABC and D axes are commanded to be at the positions 200, -10, 0, -110
'respectively. The returned units are in quadrature counts.
:PF 7;' Position format of 7
:RP ;' Return A,B,C,D reference positions
200,-10,0,-110
:RP A
200 Return the A motor reference position
:RP B
-10 Return the B motor reference position
:PF -6.0;' Change to hex format
:RP
$0000C8,$FFFFF6,$000000,$FFF93 Return A,B,C,D in hex
:position = RPA;' Assign the variable, position, the value of RPA
```

```
:GA n;' make A axis slave to N imaginary axis
:GR -1;' 1:-1 gearing
:SPN= 10000
:PRN= 10000
:BG N;' Begin motion
:RP N;' Get master position
10000
:RP A;' Get slave commanded position
-10000
```

RP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>RS</b>	Reset	SYSTEM CONFIG
The RS command resets the state of the processor to its power-on condition		
RS	Command takes no arguments	
RS	Operand has special meaning, see Remarks	

## DESCRIPTION

The RS command resets the state of the processor to its power-on condition. The previously saved state of the hardware, along with parameter values and saved program, are restored.

## ARGUMENTS

### RS n

Argument	Min	Max	Default	Resolution	Description	Notes
n	-1	0	0	1	Set behavior of RS command	n = 0 performs normal reset. n = -1 performs soft master reset. See Remarks.

## REMARKS

- A soft master reset performed by issuing RS-1 restores factory default settings without erasing the EEPROM. To restore saved EEPROM settings use RS with no arguments, or RS0.

### Operand Usage

- \_RS returns the state of the processor on its last power-up condition. The value returned is the decimal equivalent of the 4 bit binary value shown below.
  - Bit 3 For master reset error
  - Bit 2 For program checksum error
  - Bit 1 For parameter checksum error
  - Bit 0 For variable checksum error
- At startup the controller operating system verifies the firmware sector. If there is a checksum error shown by \_RS in firmware, it is not loaded and the controller will boot to monitor mode.
  - The #AUTOERR automatic subroutine will run if this error occurs and the subroutine is located in the program space.

## EXAMPLES

```
:RS; ' Reset the hardware
:RS-1; ' Perform a soft master reset
:
```

RS is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SA</b>	<i>Send Command</i>	ETHERNET
<b>SA</b> sends a command, and optionally receives a response, from one controller to another via Ethernet		
		
SAA=n	Arguments specified with an axis mask and an assignment (=)	
San0, SAN1, SAN2, SAN3, SAN4, SAN5, SAN6, SAN7	Operand has special meaning see Remarks	

## DESCRIPTION

SA sends a command, and optionally receives a response, from one controller to another via Ethernet.

## ARGUMENTS

**SAm= str**

**SAm= str,n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Handle	Handle to specify for message output	
str	1 char	74 chars	""	String	String to send over handle	
n	-2,147,483,648	2,147,483,647	0	1	Value to send for the specified parameter	

## REMARKS

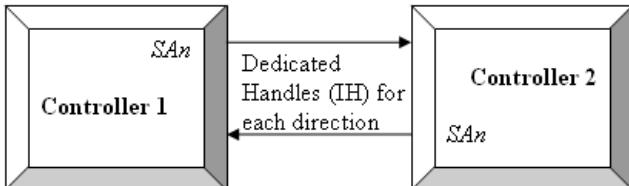
- Strings are encapsulated by quotations. This will typically begin an SA command.
- n is a number, controller operand, variable, mathematical function, or string. The range for numeric values is 4 bytes of integer followed by two bytes of fraction.
- Typical usage would have the first argument as a string such as "KI" and the subsequent arguments as the arguments to the command:
  - Example SAF="KI", 1, 2 would send the command: KI1,2
  - SA automatically adds commas between two number values being sent.

DMC40x0, DMC41x3, DMC21x3

- There is a 78 character maximum payload length for the SA command.

## Operational Notes

- SA is non-blocking. A wait (e.g. WT10) must occur between successive calls to SA.
- SA is not valid over a handle configured for Modbus (port 502).
- When writing multi-threaded DMC code, send all traffic from only one thread.
- The Galil that establishes the connection and issues the SA command is called the master. The Galil that receives the connection and answers the SA is the slave.
  - For both controllers in a connection to be both masters and slaves, open two Ethernet handles. Each of the controllers is a master over one of the handles, and a slave on the other.



## Operand Usage

- SAm** gives the value of the response to the command sent with an SA command.
  - The m value represents the handle A thru H and the n value represents the specific field returned from the controller (0-7).
  - If the specific field is not used, the operand will be -2^31.

## EXAMPLES

```

#a
IHA= 10,0,0,12;          Configures handle A to be connected to a controller with IP 10.0.0.12
#b;JP #b,_IHA2<>-2;   Wait for connection
SAA= "KI", 1, 2;         Sends the command to handle A (slave controller): KI 1,2
WT 10
SAA= "TE";               Sends the command to handle A (slave controller): TE
WT 10
MG _SAA0;                Display the content of the operand SAA (first response to ;'TE command)
MG _SAA1;                Display the content of the operand SAA (2nd response to TE ;'command)
SAA= "TEMP=", 16;         Sets variable temp equal to 16 on handle A controller
EN;                      End Program
  
```

SA is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SB</b>	<i>Set Bit</i>	<b>IO</b>
The SB command sets a particular digital output		

SB n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The SB command sets a particular digital output. The SB and CB (Clear Bit) instructions can be used to control the state of output lines.

## ARGUMENTS

### SB n

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	16	N/A	1	General output bit to be set	Max value is 8 for 1-4 axis controllers
n	1,000	8,999	N/A	1	Set Modbus slave bit	See "SB via Modbus Slave" in Remarks

## REMARKS

- The state of the output can be read with the @OUT command

### SB via Modbus Slave

DMC40x0, DMC41x3, RIO, DMC21x3, DMC30010

- n0 = (SlaveAddress\*10000) + (HandleNum\*1000) + ((Module-1)\*4) + (Bitnum-1)
  - Slave Address is used when the ModBus device has slave devices connected to it and specified as Addresses 0 to 255. Please note that the use of slave devices for modbus are very rare and this number will usually be 0.
  - HandleNum is the handle specifier from A to H.
  - Module is the position of the module in the rack from 1 to 16.
  - BitNum is the I/O point in the module from 1 to 4

## EXAMPLES

```
#main
SB 5;' Set digital output 5
SB 1;' Set digital output 1
CB 5;' Clear digital output 5
CB 1;' Clear digital output 1
EN
```

```
#modbus
REM connect to modubs slave at IP address 192.168.1.50
IHH= 192,168,1,50<502>2
WT 100
SB 8001;'set bit 1 on modbus slave
WT 10
CB 8003;'set bit 3 on modbus slave
EN
```

For detailed information on connecting to a Modbus slave, see:  
<http://www.galilmc.com/techtalk/io-control/setting-up-and-rio-as-extended-io-for-a-controller/>

SB is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SC</b>	<i>Stop Code</i>	INTERROGATION
The Stop Code command returns a number indicating why a motor has stopped		

SC A	Argument is an axis mask
SCm	Operand has special meaning, see Remarks

## DESCRIPTION

The Stop Code command returns a number indicating why a motor has stopped.

## ARGUMENTS

### SC mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	N/A	Multi-Axis Mask	Axis to query stop code	Omitting argument shows stop code for all axes

## REMARKS

- When SC is issued, the controller responds with a number for the axis queried. The number is interpreted as follows:

*Stop Code Table*

Stop Code Number	Meaning
0	Motors are running, independent mode
1	Motors decelerating or stopped at commanded independent position
2	Decelerating or stopped by FWD limit switch or soft limit FL
3	Decelerating or stopped by REV limit switch or soft limit BL
4	Decelerating or stopped by Stop Command (ST)
6	Stopped by Abort input
7	Stopped by Abort command (AB)
8	Decelerating or stopped by Off on Error (OEI)
9	Stopped after finding edge (FE)
10	Stopped after homing (HM) or Find Index (FI)
11	Stopped by selective abort input
12	Decelerating or stopped by encoder failure (OA1) (For controllers supporting OA/OV/OT)
15	Amplifier Fault (For controllers with internal drives)
16	Stepper position maintenance error
30	Running in PVT mode
31	PVT mode completed normally
32	PVT mode exited because buffer is empty
50	Contour Running
51	Contour Stopped
60	ECAM Running
61	ECAM Stopped
99	MC timeout
100	Vector Sequence running
101	Vector Sequence stopped

- SCm contains the value of the stop code for the specified axis.

## EXAMPLES

```
tom = _SCA;          Assign the Stop Code of A axis to variable tom
```

```
:JG 10000
:BG A
:SC A
0          //Axis is running in independent mode
:ST A
:SC A
4          //Axis is stopped by ST command
:
```

SC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SD</b>	<i>Switch Deceleration</i>	INDEPENDENT MOTION
The Limit Switch Deceleration command (SD) sets the linear deceleration rate of the motors when a limit switch has been reached		

SDA=n	Arguments specified with an axis mask and an assignment (=)
SD n ...	Arguments specified with an implicit comma-separated order
SDm	Operand holds the value set in the command

## DESCRIPTION

The Limit Switch Deceleration command (SD) sets the linear deceleration rate of the motors when a limit switch has been reached.

## ARGUMENTS

**SDm=n**

**SD n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	1,024	1,073,740,800	256,000	1,024	Value of switch deceleration	Resolution changes with TM, see Remarks

## REMARKS

- The resolution of the SD command is dependent upon the update rate setting (TM). With the default rate of TM 1000 the resolution is 1024 cnts/second^2. The equation to calculate the resolution of the AC command is:
  - Resolution =  $1024^2(1000/TM)^2$
  - Example:
    - With TM 500 the minimum AC setting and resolution is 4096 cnts/second^2
    - resolution =  $1024^2(1000/500)^2 = 4096$
- The SD command may be changed during the move in JG move, but not in PR or PA move.

## EXAMPLES

```
#main
PR 10000; ' Specify position
AC 2000000; ' Specify acceleration rate
DC 1000000; ' Specify deceleration rate
SD 5000000; ' Specify Limit Switch Deceleration Rate
SP 5000; ' Specify slew speed
EN
```

SD is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SH</b>	Servo Here	FILTER/CONTROL
The SH commands tells the controller to use the current motor position as the command position and to enable servo control here		

SH A Argument is an axis mask

## DESCRIPTION

The SH commands tells the controller to use the current motor position as the command position and to enable servo control here.

## ARGUMENTS

### SH mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to enable	

## REMARKS

- The SH command changes the coordinate system.
  - Therefore, all position commands given prior to SH, must be repeated. Otherwise, the controller produces incorrect motion.
- This command can be useful when the position of a motor has been manually adjusted following a motor off (MO) command.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
SH ;' Servo A,B,C,D motors
SH A; ' Only servo the A motor, the B,C and D motors remain in its previous state.
SH B; ' Servo the B motor, leave the A,C and D motors unchanged
SH C; ' Servo the C motor, leave the A,B and D motors unchanged
SH D; ' Servo the D motor, leave the A,B and C motors unchanged
```

```
'show how issuing SH clears position error
'by resetting the coordinate system
:MO A; ' disable the A axis
:TE A; ' check error on A axis
-12435  large error due to manual motion
:TP A; ' Check position
12435
:SH A; ' enable A axis, doing so clears the error
:TE A; ' check error again
0
:TP A; ' confirm position hasn't changed
12435
```

SH is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SI</b>	Configure the special Galil SSI feature	SYSTEM CONFIG
The SI command enables and configures the controller to read SSI encoder data		
		

SIA=n Arguments specified with an axis mask and an assignment (=)

## DESCRIPTION

The SI command enables and configures the controller to read SSI encoder data. Synchronous Serial Interface (SSI) allows for serial transmission of absolute position data (either binary or Gray code) from the encoder based on a timed clock pulse train from the controller. Connection between the controller and encoder is based on two signal lines, clock and data, which are usually differential for increased noise immunity. For each sequential clock pulse of the controller, the encoder transmits one data bit from shift registers on the encoder.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010

**SI**m=n0, n1, n2, n3 <o>p

DMC40x0, DMC41x3, DMC21x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to configure	
<b>n0</b>	0	2	0	1	Register to hold SSI	0=off; 1=Main encoder; 2=Aux Encoder
<b>n1</b>	-31	31	N/A	1	Total number of bits of SSI	Negative=Rollover, Positive=No Rollover (see Remarks)
<b>n2</b>	0	31	N/A	1	Number of single turn bits	n2 <= n1
<b>n3</b>	-8	8	N/A	1	Number of status bits	Negative=Prepended status bits, Positive=Postpended status bits (see Remarks)
<b>o</b>	4	26	N/A	1	Clock frequency divider	See table below
<b>p</b>	1	2	N/A	1	Data Encoding	1 = Binary encoding 2 = Gray Code

DMC40x0, DMC41x3, DMC30010

*Clock Frequency (o)*

Value	Frequency (MHz)
4	2
5	1.7
6	1.4
7	1.3
8	1.1
9	1.0
10	0.91
11	0.83
12	0.77
13	0.71
14	0.67
15	0.63
16	0.59
17	0.56
18	0.53
19	0.50
20	0.48
21	0.45
22	0.43
23	0.42
24	0.40
25	0.38
26	0.37

## REMARKS

- Slm=? Returns the configuration parameters
- n1: A positive number designates No Rollover. A negative number will cause the controller to act as an incremental encoder, allowing the encoder to count past the max value of the encoder. This prevents a discontinuity in servo error at the ends of the absolute data. When the controller is powered down, the rollover values are lost.
- n3: A negative number designates status bits as leading the SSI data. A positive number designates status bits as trailing the SSI data,
- See Application Note 2438 for more information

DMC40x0, DMC41x3, DMC30010

- There are two items required when connecting an SSI encoder to the controller, special SSI firmware and a controller ordered with -SSI or -SER option.
- Clocking in SSI data has a timing overhead which may be non-negligible. In the event that clocking in data may have a negative effect on servo performance (e.g. using multiple encoders with a lowered TM sample rate) the controller will respond with an error mode. See #AUTOERR for more information. This error mode is very rare, and is expected to occur only in development.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC30010

```
SIA= 1,25,25,0<10>1; ' Encoder on axis A replaces main encoder (TP), 25 bits total, all single turn, no status
```

```
SIA= 0; ' Disable SSI on axis A
```



<b>SL</b>	<i>Single Step</i>	PROGRAMMING
The SL command is used to single-step through a program for debugging purposes		

SL n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The SL command is used to single-step through a program for debugging purposes. SL can be used after execution has paused at a breakpoint (BK). The argument n allows user to specify the number of lines to execute before pausing again.

## ARGUMENTS

### SL n

Argument	Min	Max	Default	Resolution	Description	Notes
n	1	255	1	1	Number of lines to execute before pausing	If n is omitted, default value used.

## REMARKS

- The BK command resumes normal program execution.

## EXAMPLES

```
BK 3; ' Pause at line 3 (the 4th line) in thread 0
BK 5; ' Continue to line 5
SL; ' Execute the next line
SL 3; ' Execute the next 3 lines
BK; ' Resume normal execution
```

SL is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SM</b>	<i>Subnet Mask</i>	ETHERNET
<b>The SM command assigns a subnet mask to the controller</b>		
		

SM n ...	Arguments specified with an implicit comma-separated order
_SM0	Operand has special meaning, see Remarks

## DESCRIPTION

The SM command assigns a subnet mask to the controller. All packets sent to the controller whose source IP address is not on the subnet will be ignored by the controller. For example, for SM 255,255,0,0 and IA 10,0,51,1, only packets from IP addresses of the form 10.0.xxx.xxx will be accepted.

## ARGUMENTS

### SM n0,n1,n2,n3

#### SM n

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	0	255	0	1	Byte 3 of the Subnet mask	
<b>n1</b>	0	255	0	1	Byte 2 of the Subnet mask	
<b>n2</b>	0	255	0	1	Byte 1 of the Subnet mask	
<b>n3</b>	0	255	0	1	Byte 0 of the Subnet mask	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	The full subnet mask specified as a signed 32 bit two's complement integer	

## REMARKS

- n = ? will return the subnet mask of the controller as n0,n1,n2,n3
- \_SM0 contains the subnet mask representing a 32 bit signed number (Two's complement)
- Use the following equation to change the 4 byte subnet (n0,n1,n2,n3) to a single 32 bit number, n
  - $n = (n0*2^{24}) + (n1*2^{16}) + (n2*2^8) + n3$

## EXAMPLES

```
SM 255,255,255,255; ! Ignore all incoming Ethernet packets
SM 0,0,0,0; ! Process all incoming Ethernet packets
```

SM is supported on DMC40x0, DMC41x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SP</b>	<i>Speed</i>	INDEPENDENT MOTION
The SP command sets the slew speed of any or all axes for independent moves		

SPA= n	Arguments specified with an axis mask and an assignment (=)
SP n ...	Arguments specified with an implicit comma-separated order
SPm	Operand holds the value set in the command

## DESCRIPTION

The SP command sets the slew speed of any or all axes for independent moves.

## ARGUMENTS

**SPm= n**

**SP n,n,n,n,n,n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
	M	N	N/A	Axis	Virtual axis to assign value	
<b>n</b>	0	15,000,000	25,000	2	Value of jog speed in cnts/second	For MT settings of 1,-1,1.5 and -1.5 (Servos) - See Remarks for Resolution details
	0	3,000,000	25,000	2	Value of jog speed in cnts/second	For MT settings of 2,-2,2.5 and -2.5 (Steppers) - See Remarks for Resolution details

## REMARKS

- Negative values will be interpreted as the absolute value

DMC40x0, DMC41x3, DMC18x6, DMC30010

## Resolution

DMC40x0, DMC41x3, DMC18x6, DMC30010

- The resolution of the SP command is dependent upon the update rate setting (TM).
  - With the default rate of TM 1000 the resolution is 2 cnts/second.
  - The equation to calculate the resolution of the SP command is:
    - resolution =  $2 * (1000/TM)$
  - example:
    - With TM 250 the resolution of the SP command is 8 cnts/second
    - resolution =  $2 * (1000/250) = 8$

## EXAMPLES

```

PR 2000,3000,4000,5000;          Specify a,b,c,d parameter
SP 5000,6000,7000,8000;          Specify a,b,c,d speeds
BG ;                            Begin motion of all axes
AM C;                           After C motion is complete
'
'
'
'For vector moves, use the vector speed command (VS) to change the speed.
'SP is not a "mode" of motion like JOG (JG).
>Note: 2 is the minimum non-zero speed.

```

SP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SS</b>	Configure the special Galil BiSS feature	SYSTEM CONFIG
The SS command enables and configures the controller to read BiSS encoder data		
		

SS=n	Arguments specified with an axis mask and an assignment (=)
_SSm	Operand has special meaning, see Remarks

## DESCRIPTION

The SS command enables and configures the controller to read BiSS encoder data. BiSS is an open-standard, digital interface for sensors and actuators. BiSS is hardware compatible to the industrial standard SSI (Serial Synchronous Interface). It allows serial transmission of absolute position data from BiSS encoders based on a master clock signal from the controller.

## ARGUMENTS

**SSm = n0,n1,n2,n3 < o**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to specify	
<b>n0</b>	0	2	0	1	Register to hold BiSS position	0=None, 1=Main encoder, 2=Aux encoder
<b>n1</b>	-31	31	N/A	1	Number of single-turn bits	n2 < 0 enables rollover. See Remarks.
<b>n2</b>	0	31	N/A	1	Number of total bits before error bits	See Example for BiSS example
<b>n3</b>	0	8	N/A	1	Number of zero padding bits	
<b>o</b>	4	26	N/A	1	Clock divider	Read data on rising clock edge. See Table
	-26	-4	N/A	1	Clock divider	Read data on falling clock edge. See Table

DMC40x0, DMC41x3, DMC30010

*Clock Frequency (o)*

Value	Frequency (MHz)
4	2
5	1.7
6	1.4
7	1.3
8	1.1
9	1.0
10	0.91
11	0.83
12	0.77
13	0.71
14	0.67
15	0.63
16	0.59
17	0.56
18	0.53
19	0.50
20	0.48
21	0.45
22	0.43
23	0.42
24	0.40
25	0.38
26	0.37

## REMARKS

- SSm=? Returns the configuration parameters
- n1: A positive number designates No Rollover. A negative number will cause the RIO to act as an incremental encoder, allowing the encoder to count past the max value of the encoder. Note, when the controller is powered down, the rollover values are lost
- BiSS clock (MA) frequency is set with the p argument and has the following form:
  - MA freq= 20 MHz / (2 \* (p+1))

DMC40x0, DMC41x3

- 20 MHz frequency is hardware dependent with a range of 18Mhz to 26Mhz. Contact Galil if tolerances must be tighter for a particular application (this is rare).
- Clocking in BiSS data has a timing overhead which may be non-negligible. In the event that clocking in data may have a negative effect on servo performance (e.g. using multiple encoders with a lowered TM sample rate) the controller will respond with an error mode. See #AUTOERR for more information.
  - This error mode is very rare, and is expected to occur only in development.

## Operand Usage

- \_SSm Returns 4 bits of axis status data where n is the axis designator used.
  - #SERERR is an automatic sub which will run in the event of an encoder problem. See SY for setting up the active high/low status of bits 2 and 3.
  - Note: The encoder manufacturer may name the Error and Warning bits differently. Consult the encoder documentation for the naming convention.
  - Galil defines the Warning bit as the bit directly preceding the CRC. The Error bit is defined as the bit directly preceding the Warning bit. See "\_SSm Bit Map" table below.

*SSm Bit Map*

Bit Position	Bit Meaning	Description
No timeout = 0, timeout occurred		

0	<small>ENO TIMEOUT = 0, timeout occurred = 1</small>	The BiSS decoding hardware will timeout if the encoder doesn't set the start bit within 30uS
1	CRC valid = 0, invalid = 1	BiSS employs a Cyclic Redundancy Check to verify data after transmission
2	Error bit* (active state set with SY)	When SY is set correctly, this bit should be low when there is no active warning. Consult the encoder documentation for the Warning bit definition
3	Warning bit* (active state set with SY)	When SY is set correctly, this bit should be low when there is no active alarm/error. Consult the encoder documentation for the Alarm bit definition

## EXAMPLES

*SS Example for Hengstler 12 bit MT 10 bit ST*

Bit sequence:	T-2	T-1 (Delay)	T0	T1... T12	T13... T22	T23... T26	T27	T28	T29... T34	T35
Data (Data/SLO line):	1	0	1	M11... M0	S9... S0	0	E	W	C5... C0	MCD
Data Description:	Idle	Encoder acquiring	Start Bit	Multi-turn data	Single-turn data	Zero padding	Error Bit	Warning Bit	CRC	Multi-Cycle Data
SS command details:	-	-	-	-	ss1=10	ss3=4	ss2=26, E bit read in SSn	W read in SSn	CRC valid bit read in SSn	Ignored by default

```
'BiSS setup command for the Hengstler 12 bit MT 10 bit ST
'Data will be available in TP and for servo feedback
SSA= 1,10,26,4<13
```

```
'Configuration for 26 bit Renishaw Resolute single-turn encoder
SYA= 0;'           Warning and Alarm bits are active low
SSA= 1,26,27,0<14
'The 27 includes the Resolute single leading zero bit
```

```
'Configuration for 36 bit Hengstler multi-turn encoder
SYA= 3;'           Warning and Alarm bits are active high
SSA= 1,19,36,5<14
'19 bits single turn, 12 bits multi turn, 5 zero padding bits
```

SS is supported on DMC40x0, DMC41x3, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ST</b>	Stop	INDEPENDENT MOTION, VECTOR/LINEAR
The ST command stops motion on the specified axis		

ST A Argument is an axis mask

## DESCRIPTION

The ST command stops motion on the specified axis. Motors will come to a decelerated stop.

## ARGUMENTS

### ST mm

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>mm</b>	A	ABCDEFGHMNST	ABCDEFGH	Multi-Axis Mask	Axes to command to stop motion	

## REMARKS

- If ST is sent from the host without an axis specification, program execution will stop in addition to motion.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
ST A; ' Stop A-axis motion
ST S; ' Stop coordinated sequence
ST ABCD; ' Stop A,B,C,D motion
ST ; ' Stop ABCDEFGH motion
ST SCD; ' Stop coordinated AB sequence, and C and D motion
'Use the after motion complete command, AM, to wait for motion to be stopped.
```

```
:ST A; ' Stop motion on the A axis
:SC A; ' Query A axis status
 4 Indicates stopped by ST command
:MG _NO; ' Check if code is running
 1 Thread 0 running
:ST ; ' General stop
:MG _NO; ' check code again
 0 Thread 0 stopped
```

ST is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>SY</b>	Serial encoder BiSS active level	SYSTEM CONFIG
This command is used to designate the active level of the Error and Warning bits when using the Galil BiSS upgrade		

SYA= n	Arguments specified with an axis mask and an assignment (=)
SY n ...	Arguments specified with an implicit comma-separated order
SYm	Operand holds the value set in the command

## DESCRIPTION

This command is used to designate the active level of the Error and Warning bits when using the Galil BiSS upgrade. The BiSS protocol defines two bits which can be used by the encoder to signal various events.

## ARGUMENTS

DMC40x0, DMC41x3, DMC30010

**SYm= n**

DMC40x0, DMC41x3, DMC30010

**SY n,n,n,n,n,n,n**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	

Argument	Value	Description	Notes
n	0	Warning bit = Active Low; Error bit = Active Low	
	1	Warning bit = Active Low; Error bit = Active High	
	2	Warning bit = Active High; Error bit = Active Low	
	3	Warning bit = Active High; Error bit = Active High	Default

## REMARKS

DMC40x0, DMC41x3, DMC30010

- The SY mask should be set appropriately to ensure that the #SERERR automatic subroutine will run when the bits are active, and that the \_SSn operand reports the fault state of the encoder correctly.
- Example of Warning and Alarm/Error bit use, Quoted from Renishaw Data Sheet L-9709-9005-03-A
  - Error (1 bit) "The error bit is active low: "1" indicates that the transmitted position information has been verified by the readhead's internal safety checking algorithm and is correct; "0" indicates that the internal check has failed and the position information should not be trusted. The error bit is also set to "0" if the temperature exceeds the maximum specification for the product."
  - Warning(1 bit) "The warning bit is active low: "0" indicates that the encoder scale (and/or reading window) should be cleaned. Note that the warning bit is not an indication of the trustworthiness of the position data. Only the error bit should be used for this purpose."

## EXAMPLES

```
'configure SY for Renishaw Resolute encoder
SYA= 0
```

SY is supported on DMC40x0, DMC41x3, RIO, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TA</b>	<i>Tell Amplifier error status</i>	INTERROGATION, ERROR CONTROL
The command returns the amplifier error status		
TA n ...	Arguments specified with an implicit comma-separated order	
_TA0, _TA1, _TA2, _TA3	Operand has special meaning see Remarks	

## DESCRIPTION

The command returns the amplifier error status. The value is decimal and represents an 8 bit value. Bit 7 is most significant bit, 0 is least.

## ARGUMENTS

### TA n

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	3	N/A	1	Selects amp status byte to return	

DMC40x0, DMC41x3, DMC21x3

*Tell Amplifier Error Status Bit Definition*

	TA0	TA1	TA2	TA3	
BIT #:	STATUS:	STATUS:	STATUS:	STATUS:	BIT #
7	Under Voltage (E-H Axes) (2)	Hall Error H Axis (1)	Peak Current H Axis	0	7
6	Over Temperature (E-H Axes) (2)	Hall Error G Axis (1)	Peak Current G Axis	0	6
5	Over Voltage (E-H Axes) (1)	Hall Error F Axis (1)	Peak Current F Axis	0	5
4	Over Current (E-H Axes) (3)	Hall Error E Axis (1)	Peak Current E Axis	0	4
3	Under Voltage (A-D Axes) (2)	Hall Error D Axis (1)	Peak Current D Axis	0	3
2	Over Temperature (A-D Axes) (1)	Hall Error C Axis (1)	Peak Current C Axis	0	2
1	Over Voltage (A-D Axes) (1)	Hall Error B Axis (1)	Peak Current B Axis	ELO Active (E-H Axes) (4)	1
0	Over Current (A-D Axes) (3)	Hall Error A Axis (1)	Peak Current A Axis	ELO Active (A-D Axes) (4)	0

DMC40x0, DMC41x3

1. Valid for AMP-43040 (-D3040), AMP-43240 (-D3240), & AMP-43540 (-D3540)
2. Valid for -D3040, -D3240, -D3540 & Valid for SDM-44140 (-D4140)
3. Valid for -D3040, -D3240, -D3540, -D4140 & Valid for SDM-44040 (-D4040)
4. Valid for -D3040, -D3240, -D3540, -D4140, -D4040 & Valid for AMP-43140 (-D3140) & AMP-43640 (-D3640)

## REMARKS

- TA n Contains the Amplifier error status. n = 0,1,2, or 3
- If a Brushed-type servo motor is disabling and TA1 shows a hall error, use the BR command to set the axis as a Brushed axis. This causes the controller to ignore invalid Hall states.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3

:TA 1 5 Hall Error for Axis A and C
:TA 1 1 'bit 0 means hall error for A axis :TA 0 8 'bit 3 means under voltage error for amp

TA is supported on DMC40x0, DMC41x3, DMC21x3, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TB</b>	<i>Tell Status Byte</i>	INTERROGATION
The TB command returns status information from the controller as a decimal number		

TB	Command takes no arguments
_TB	Operand has special meaning, see Remarks

## DESCRIPTION

The TB command returns status information from the controller as a decimal number. Each bit of the status byte denotes an active condition when the bit is set (high):

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**TB ?**

**TB**

The following table describes the specific conditions reported with each bit of the TB report.

DMC40x0, DMC41x3, DMC21x3, DMC30010

*Tell Status Byte Response Bit Description*

Bit #	Status
Bit 7	Executing application program
Bit 6	N/A
Bit 5	Contouring
Bit 4	Executing error or limit switch routine
Bit 3	Input Interrupt enabled
Bit 2	Executing input interrupt routine
Bit 1	N/A
Bit 0	Echo on

## REMARKS

- \_TB Contains the status byte reported by the TB command

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:TB
33' Contouring on and Echo is on (2^5 + 2^0 = 32 + 1 = 33)
```

```
:TB;^ Tell status information
129' Executing program and echo on (2^7 + 2^0 = 128 + 1 = 129)
```

TB is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TC</b>	Tell Error Code	INTERROGATION, ERROR CONTROL
The TC command reports programming or command errors detected by the controller		
		

TC n ...	Arguments specified with an implicit comma-separated order
_TC	Operand has special meaning, see Remarks

## DESCRIPTION

The TC command reports programming or command errors detected by the controller. The TC command returns a number between 1 and 255. This number is a code that reflects why a command was not accepted by the controller. This command is useful when the controller halts execution of a program or when the response to a command is a question mark.

## ARGUMENTS

### TC n

Argument	Value	Description	Notes
n	0	Return the numerical code only	Default
	1	Return the numerical code and human-readable message	

DMC40x0, DMC41x3, DMC18x6, DMC30010

TC Error Code List

Tell Code Number	Description	Notes
1	Unrecognized command	
2	Command only valid from program	
3	Command not valid in program	
4	Operand error	
5	Input buffer full	
6	Number out of range	
7	Command not valid while running	
8	Command not valid while not running	
9	Variable error	
10	Empty program line or undefined label	
11	Invalid label or line number	
12	Subroutine more than 16 deep	
13	JG only valid when running in jog mode	
14	EEPROM check sum error	
15	EEPROM write error	
16	IP incorrect sign during position move or IP given during forced deceleration	
17	ED, BN and DL not valid while program running	
18	Command not valid when contouring	
19	Application strand already executing	
20	Begin not valid with motor off	
21	Begin not valid while running	
22	Begin not possible due to Limit Switch	
24	Begin not valid because no sequence defined	
25	Variable not given in IN command	
28	S operand not valid	
29	Not valid during coordinated move	
30	Sequence Segment Too Short	
31	Total move distance in a sequence > 2 billion	
32	Segment buffer full	
33	VP or CR commands cannot be mixed with LI commands	
39	No time specified	
41	Contouring record range error	
42	Contour data being sent too slowly	
46	Gear axis both master and follower	
50	Not enough fields	
51	Question mark not valid	
52	Missing " or string too long	
53	Error in {}	
54	Question mark part of string	
55	Missing [ or ]	
56	Array index invalid or out of range	
57	Bad function or array	
58	Bad command response	i.e. GNX
59	Mismatched parentheses	
60	Download error - line too long or too many lines	
61	Duplicate or bad label	
62	Too many labels	
63	IF statement without ENDIF	
65	IN command must have a comma	
66	Array space full	

67	Too many arrays or variables	
71	IN only valid in thread #0	
80	Record mode already running	
81	No array or source specified	
82	Undefined Array	
83	Not a valid number	
84	Too many elements	
90	Only A B C D valid operand	
97	Bad Binary Command Format	
98	Binary Commands not valid in application program	
99	Bad binary command number	
100	Not valid when running ECAM	
101	Improper index into ET	
102	No master axis defined for ECAM	
103	Master axis modulus greater than 256 EP value	
104	Not valid when axis performing ECAM	
105	EB1 command must be given first	
106	Privilege Violation	
110	No hall effect sensors detected	
111	Must be made brushless by BA command	
112	BZ command timeout	
113	No movement in BZ command	
114	BZ command runaway	
118	Controller has GL1600 not GL1800	
119	Not valid for axis configured as stepper	
120	Bad Ethernet transmit	not valid for PCI
121	Bad Ethernet packet received	not valid for PCI
123	TCP lost sync	not valid for PCI
124	Ethernet handle already in use	not valid for PCI
125	No ARP response from IP address	not valid for PCI
126	Closed Ethernet handle	not valid for PCI
127	Illegal Modbus function code	not valid for PCI
128	IP address not valid	not valid for PCI
130	Remote IO command error	not valid for PCI
131	Serial Port Timeout	not valid for PCI, See Remarks
132	Analog inputs not present	
133	Command not valid when locked / Handle must be UDP	not valid for PCI
134	All motors must be in MO for this command	
135	Motor must be in MO	
136	Invalid Password	
137	Invalid lock setting	
138	Passwords not identical	
140	Serial encoder missing	Valid for BiSS support
141	Incorrect ICM Configuration	
143	TM timed out	Valid on SER firmware (SSI and BiSS)
160	BX failure	
161	Sine amp axis not initialized	
164	Exceeded maximum sequence length, BGS or BGT is required	
166	Unable to set analog output	30000 Hardware, see AO

## REMARKS

- TC command accepts ? as a query. This is equivalent to TC or TC 0
- After TC has been read, the error code is set to zero.
- \_TC contains the value of the error code. Use of the operand does not clear the error code.

DMC41x3, RIO, DMC21x3, DMC30010

- Note: Error code 131 means that an RS232/USB timeout is being generated while trying to transmit data to the serial port.
  - This is usually caused by MG. Numerous timeouts on serial communication can cause a slowdown in DMC code execution and should be avoided.

## EXAMPLES

```
:GF32;` Bad command
?
:TC 1;` Tell error code
1     Unrecognized command
:
```

TC is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TD</b>	Tell Dual Encoder	INTERROGATION
The TD command returns the current position of the dual (auxiliary) encoder input		

TD A	Argument is an axis mask
_TDm	Operand has special meaning, see Remarks

## DESCRIPTION

The TD command returns the current position of the dual (auxiliary) encoder input. When operating with stepper motors, the TD command returns the number of counts that have been output by the controller.

## ARGUMENTS

### TD mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report dual (auxiliary) encoder position.	

## REMARKS

- Auxiliary encoders are not available for a stepper axis or for the axis where output compare is used.

### Operand Usage

- \_TDm reports the dual encoder position for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:TD ;' Return A,B,C,D Dual encoders
200, -10, 0, -110
:TD A; ' Return the A motor Dual encoder
200
:dual= TDA;' Assign the variable, DUAL, the value of TDA
```

TD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TE</b>	Tell Error	INTERROGATION
The TE command returns the current error in the control loop		
		

TE A	Argument is an axis mask
_TEm	Operand has special meaning, see Remarks

## DESCRIPTION

The TE command returns the current error in the control loop.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

The command returns the position error of the motor(s), which is the difference between commanded (RP) and actual (TP) position.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### TE mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report position error	

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- Under normal operating conditions with servo control, the position error should be small. The position error is typically largest during acceleration and deceleration.
- The Tell Error command is not valid for step motors since they operate open-loop.

### Operand Usage

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- \_TEm contains the current position error value for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:TE ;' Return all position errors
5, -2, 0, 6
:TE A; ' Return the A motor position error
5
:TE B; ' Return the B motor position error
-2
:error = _TEA; ' Sets the variable, Error, with the A-axis position error
```

TE is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TH</b>	Tell Ethernet Handle	INTERROGATION
The TH command returns a list of data pertaining to the Galil's Ethernet connection		
		

TH Command takes no arguments

## DESCRIPTION

The TH command returns a list of data pertaining to the Galil's Ethernet connection. This list begins with the IP address and Ethernet address (physical address), followed by the status of each handle indicating connection type and IP address.

## ARGUMENTS

### TH

TH is an interrogation command with no parameters

## REMARKS

- This command only shows valid ethernet handles to the controller.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3

```
:TH
CONTROLLER IP ADDRESS 10,51,0,87 ETHERNET ADDRESS 00-50-4C-08-01-1F
IHA TCP PORT 1050 TO IP ADDRESS 10,51,0,89 PORT 1000
IHB TCP PORT 1061 TO IP ADDRESS 10,51,0,89 PORT 1001
IHC TCP PORT 1012 TO IP ADDRESS 10,51,0,93 PORT 1002
IHD TCP PORT 1023 TO IP ADDRESS 10,51,0,93 PORT 1003
IHE TCP PORT 1034 TO IP ADDRESS 10,51,0,101 PORT 1004
IHF TCP PORT 1045 TO IP ADDRESS 10,51,0,101 PORT 1005
IHG AVAILABLE
IHH AVAILABLE
```

TH is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilinc.com](mailto:documentation@galilinc.com)

<b>TI</b>	<i>Tell Inputs</i>	INTERROGATION, IO
The TI command returns the state of the inputs in banks of 8 bits, or 1 byte		

TI n ...	Arguments specified with an implicit comma-separated order
_TIn	Operand has special meaning see Remarks

## DESCRIPTION

The TI command returns the state of the inputs in banks of 8 bits, or 1 byte. The value returned by this command is decimal and represents an 8 bit value (decimal value ranges from 0 to 255). Each bit represents one input where the LSB is the lowest input number and the MSB is the highest input bit.

## ARGUMENTS

### TI n

#### DMC41x3

Argument	Value	Description	Notes
n	0	Report status of Inputs 1-8	Default
	1	Report status of Inputs 9-16	Only valid for 5-8 axis controllers
	10	Report status of Inputs 81-88	Auxiliary encoder inputs. See Remarks
	11	Report status of Inputs 89-96	Auxiliary encoder inputs. Only valid for 5-8 axis controllers. See Remarks

## REMARKS

#### DMC41x3

- For n = 10 and n = 11, the auxiliary encoder channels A and B can be used as additional IO. Only 2 \* the number of axes worth of inputs are available.
  - See the User manual for more details.

## Operand Usage

- \_TIn contains the status byte of the input block specified by 'n'.
  - Note that the operand can be masked to return only specified bit information - see section on Bit-wise operations.

## EXAMPLES

DMC40x0, DMC41x3, RIO, DMC21x3, DMC18x6, DMC18x2

```
:TI 1          tell input state on bank 1
8             Bit 3 is high, others low
:TI 0          All inputs on bank 0 low
0             :input = _TII    Sets the variable, Input, with the TII value
:input= ?
8.0000
```

TI is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TIME</b>	<i>Time Operand</i>	<b>OPERAND ONLY</b>
The TIME operand returns the value of the internal free running, real time clock		
		
variable= TIME	Holds a value	
TIME	Operand has special meaning see Remarks	

## DESCRIPTION

The TIME operand returns the value of the internal free running, real time clock.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

The returned value represents the number of servo loop updates and is based on the TM command. The default value for the TM command is 1000. With this update rate, the operand TIME will increase by 1 count every update of approximately 1000usec. The clock is reset to 0 with a standard reset or a master reset.

## ARGUMENTS

### TIME

TIME is an operand and has no parameters

## REMARKS

\*The keyword, TIME, does not require an underscore (\_) as with the other operands.

\*TIME will increment up to +2,147,483,647 before rolling over to -2,147,483,648 and continuing to count up.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

- TIME rollover occurs after ~24-25 days of on-time at TM 1000 with no reset.
- TM 1000 will actually set an update rate of 976 microseconds. Thus the value returned by the TIME operand will be off by 2.4% of the actual time.

## EXAMPLES

```
MG TIME; ' Display the value of the internal clock
t1= TIME; ' Sets the variable t1 to the TIME value
```

TIME is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TK</b>	<i>Peak Torque Limit</i>	FILTER/CONTROL
The TK command sets the peak torque limit on the motor command output		

TKA= n	Arguments specified with an axis mask and an assignment (=)
TK n ...	Arguments specified with an implicit comma-separated order
TKm	Operand holds the value set in the command

## DESCRIPTION

The TK command sets the peak torque limit on the motor command output. This command works with the TL command which sets the continuous torque limit. When the average torque is below TL, the motor command signal can go up to the TK (Peak Torque) limit for a short amount of time.

## ARGUMENTS

**TKm= n**

**TK n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	9.9982	0	20/65,536	Value of peak torque limit	n = 0 disables the peak torque limit

## REMARKS

- TK provides the absolute value of the peak torque limit for +/- torque outputs
- Peak torque can be achieved for approximately 1000 samples upon initial command from 0V torque.
- If TK is set lower than TL, then TL is the maximum command output under all circumstances.

## EXAMPLES

```
TLA= 7; ' Limit A-axis to a 7 volt average torque output
TKA= 9.99; ' Limit A-axis to a 9.99 volt peak torque output
```

TK is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TL</b>	Torque Limit	Filter/Control
The TL command sets the limit on the motor command output		

TLA=n	Arguments specified with an axis mask and an assignment (=)
TL n ...	Arguments specified with an implicit comma-separated order
TLm	Operand holds the value set in the command

## DESCRIPTION

The TL command sets the limit on the motor command output. This limit is designed to prevent over current to motors with lower current rating than the drive.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC30010

TL works along with the TK (Peak torque) command to control output current to the motor.

## ARGUMENTS

**TLm= n**

**TL n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	0	9.9982	9.9982	20/65,536	Value of torque limit	

## REMARKS

- TL sets the absolute torque maximum for negative and positive torque
  - For example, TL of 5 limits the motor command output to 5 volts maximum and -5 volts minimum

DMC40x0, DMC41x3, DMC21x3, DMC30010

- The maximum torque limit is different for certain amplifier configurations at a specific AG setting. These cases are listed below.

DMC40x0, DMC41x3

Amplifier	AG setting (current rating)	TL.limit
AMP-430x0	2 (1.0 A/V)	7.0000
AMP-43240	2 (2.0 A/V)	5.0000
AMP-43540	2 (1.6 A/V)	5.0000
AMP-43640	N/A (0.2 A/V)	5.0000

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:TL 1,5,9,7.5; ' Limit A-axis to 1 volt. Limit B-axis to 5 volts. Limit C-axis to 9 volts. Limit D-axis to 7.5 volts.
:TL ?,?,?,?; ' Return limits
1.0000,5.0000,9.0000,7.5000
:TL ?; ' Return A-axis limit
1.0000
```

TL is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)

Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TM</b>	Update Time	Filter/Control
The TM command sets the sampling period of the control loop		
		

TM n ...	Arguments specified with an implicit comma-separated order
_TM	Operand holds the value set in the command

## DESCRIPTION

The TM command sets the sampling period of the control loop. The units of this command are microseconds. A negative number turns off the servo loop.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### TM n

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
n	62.5	10,000	1,000	31.25	Set the sample time in usecs	The minimum value varies based on axis count and firmware usage. See Remarks

## REMARKS

- TM 1000 will actually set an update rate of 976 microseconds. Thus the value returned by the TIME operand will be off by 2.4% of the actual time.
- The minimum allowed TM setting for the controller is listed in the tables below:

DMC41x3

Firmware	Axis Count	Minimum TM
Standard	1-2	125
	3-4	250
	5-6	375
	7-8	500
Fast	1-2	62.5
	3-4	125
	5-6	187.5
	7-8	250

### Fast Firmware Limitations

- In the Fast firmware mode the following functions are disabled:
  - Dual encoder support
  - Peak Torque (TK)
  - Notch function (NB, NZ, NF)
  - Second field of EI
  - Electronic Gearing
  - ECAM
  - Pole filter (PL)
  - Analog Feedback (AF)
  - Steppers
  - Trippoints in all but threads 0 and 1
  - Data Record
  - TV command

## EXAMPLES

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
TM -1000; ' Turn off internal clock
TM 2000; ' Set sample rate to 2000 msec
TM 1000; ' Return to default sample rate
```

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

See <http://www.galilmc.com/support/firmware-downloads.php> to download fast firmware.

TM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TN</b>	<i>Vector Tangent</i>	VECTOR/LINEAR
The TN command describes the tangent axis to the coordinated motion path		
		

<b>TN n ...</b>	Arguments specified with an implicit comma-separated order
<b>_TNm</b>	Operand holds the value set in the command

## DESCRIPTION

The TN command describes the tangent axis to the coordinated motion path. n0 is the scale factor in counts/degree of the tangent axis. n1 is the absolute position of the tangent axis where the tangent axis is aligned with zero degrees in the coordinated motion plane. The tangent function is useful for cutting applications where a cutting tool must remain tangent to the part.

## ARGUMENTS

### TN n0,n1

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-127	127	0	0.004	Scale factor in counts/degree of the tangent axis	
<b>n1</b>	-8388608	8388607	0	1	Absolute position of tangent axis where the tangent angle is 0	

## REMARKS

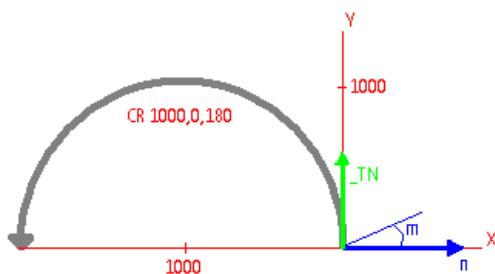
- When operating with stepper motors, n0 is the scale factor in steps / degree
- The tangent axis is specified with the VMm0mlm2 command where m2 is the tangent axis.
  - For example, VMABD specifies the D axis as the tangent axis

## Operand Usage

- \_TNm** (where m = S or T) contains the first position value for the tangent axis in the specified vector plane. This allows the user to correctly position the tangent axis before the motion begins.
  - \_TNm** will change based upon the vector path described in the VM declaration. See the example below.
  - n0 = ?** also reports this value

## EXAMPLES

Use a 2D table with a tangent cutting blade to cut a half circle. Ensure that the blade is oriented before turning on the saw. The saw is activated with output 1.



```
#example
VM ABC; '          Z axis is tangent
VSS= 500; '         Set vector speed
m= 1000/360; '     Z axis encoder is 1000 counts per full revolution
n= 0; '             When TPZ=0, blade is oriented to cut along X axis
TN m,n; '           Set these tangent characteristics
CR 1000,0,180; '   Profile a circle with radius 1000 counts,
                   starting at 0 degrees
                   and spanning 180 degrees
VE; '               End the vector path
MG _TN; '           Print the calculated initial tangent entry point (250)
PAC=_TN; '          Profile a move to orient the Z axis to begin
BG C; '             Move the blade into place
AM C; '             Wait until the blade motion is done
SB 1; '              Turn on the saw
WT 1000; '          Wait for saw to spin up
BG S; '              Begin vector motion, saw will stay tangent
AM S; '              Wait for the cut to complete
CB 0; '              Turn off the saw
MG "ALL DONE"; '   Print a message
EN
```

TN is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TP</b>	Tell Position	INTERROGATION
The TP command returns the current position of the motor		

TP A	Argument is an axis mask
_TPm	Operand has special meaning, see Remarks

## DESCRIPTION

The TP command returns the current position of the motor.

## ARGUMENTS

### TP mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report motor position	

## REMARKS

- \_TPm contains the current position value for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Assume the A-axis is at the position 200 (decimal), the B-axis is at the position -10 (decimal)
'the C-axis is at position 0, and the D-axis is at -110 (decimal). The returned parameter units are in quadrature counts.
:PF 7;' Position format of 7
:TP ;' Return A,B,C,D positions
200, -10, 0, -110
:TP A;' Return the A motor position
200
:TP B;' Return the B motor position
-10
:PF -6.0;' Change to hex format
:TP ;' Return A,B,C,D in hex
$0000C8,$FFFFF6,$000000,$FFFF93
:position = TPA;' Assign the variable, Position, the value of TPA
```

TP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TR</b>	<i>Trace</i>	INTERROGATION
The TR command causes each instruction in a program to be sent out the communications port prior to execution		
		

TR n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The TR command causes each instruction in a program to be sent out the communications port prior to execution. The trace command is useful in debugging programs.

## ARGUMENTS

DMC40x0, DMC41x3, RIO, DMC30010

### TR n0, n1

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
n0	0	1	0	1	Set status of trace function	n0 = 0 or null disables Trace. n0 = 1 enables trace.
n1	0	255	255	1	Set threads to trace by bitmask	See Remarks

## REMARKS

DMC40x0, DMC41x3, DMC18x6, DMC30010

- n1 sets a 1-byte bitmask which determines which threads will run. Bit n set corresponds to thread n traced.
  - For example, setting bit 2 and 3 sets TR to trace threads 2 and 3. ( $2^2 + 2^3 = 4 + 8 = 12$ . TR 1,12 is issued)
- Omitting n1 sets it to the default maximum value to enable trace on all threads.

## EXAMPLES

```
:Turn on trace during a program execution
:LS
0 MGTIME
1 WT1000
2 JPO
3
:XQ
:
18003461.0000
18004461.0000
18005461.0000

:TR 1
:
2 JPO
0 MGTIME
18006461.0000
1 WT1000
2 JPO
0 MGTIME
18007461.0000
1 WT1000

:TR 0
:
18008461.0000
18009461.0000

:ST
:
```

TR is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TS</b>	Tell Switches	INTERROGATION
The TS command returns information including axis-specific IO status, error conditions, motor condition and state		

TS A	Argument is an axis mask
_TSm	Operand has special meaning, see Remarks

## DESCRIPTION

The TS command returns information including axis-specific IO status, error conditions, motor condition and state. The value returned by this command is decimal and represents an 8 bit value (decimal value ranges from 0 to 255).

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

### TS mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report axis switches	

## REMARKS

- Each bit of the TS response represents the following status information when the bit is set (1).

Bit #	Status
Bit 7	Axis in motion
Bit 6	Position error exceeds error limit
Bit 5	Motor off
Bit 4	Reserved (0)
Bit 3	Forward Limit switch inactive
Bit 2	Reverse Limit switch inactive
Bit 1	Home switch status
Bit 0	Position Latch has occurred

- For active high or active low configuration (CN command), the limit switch bits are '1' when the switch is inactive and '0' when active.

### Operand Usage

- \_TSm contains the current status of the switches for the specified axis.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
:v1=_TSB;' Assigns value of TSB to the variable V1
:v1='; Interrogate value of variable V1
15 (returned value) Decimal value corresponding to bit pattern 00001111
Y axis not in motion (bit 7 - has a value of 0)
Y axis error limit not exceeded (bit 6 has a value of 0)
Y axis motor is on (bit 5 has a value of 0)
Y axis forward limit is inactive (bit 3 has a value of 1)
Y axis reverse limit is inactive (bit 2 has a value of 1)
Y axis home switch is high (bit 1 has a value of 1)
Y axis latch is not armed (bit 0 has a value of 1)
```

TS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TT</b>	Tell Torque	INTERROGATION
The TT command reports the value of the analog output signal, which is a number between -9		
		

TT A	Argument is an axis mask
_TTm	Operand has special meaning, see Remarks

## DESCRIPTION

The TT command reports the value of the analog output signal, which is a number between -9.998 and 9.998 volts.

## ARGUMENTS

### TT mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report output torque command	

## REMARKS

- Torque output is limited by the value set for the TL command.
- \_TTm contains the value of the torque for the specified axis.

## EXAMPLES

```
:vl=_TTA; ' Assigns value of TTA to variable, vl
:TT A; ' Report torque on A
-0.2843
```

TT is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TV</b>	Tell Velocity	INTERROGATION
The TV command returns the actual velocity of the axes in units of encoder count/s		

TV A	Argument is an axis mask
_TVM	Operand has special meaning, see Remarks

## DESCRIPTION

The TV command returns the actual velocity of the axes in units of encoder count/s. The value returned includes the sign bit for direction.

## ARGUMENTS

### TV mm

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
mm	A	ABCDEFGH	ABCDEFGH	Multi-Axis Mask	Axes to report velocity	

## REMARKS

- The TV command is computed using a special averaging filter (over approximately 0.25 sec for TM1000). Therefore, TV will return average velocity, not instantaneous velocity.
- \_TVM contains the value of the velocity for the specified axis.

## EXAMPLES

```
:vela=_TVA;          Assigns value of A-axis velocity to the variable VELA
:TV A;              Returns the A-axis velocity
3420
```

TV is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TW</b>	Timeout for IN Position (MC)	ERROR CONTROL
The TW command sets the timeout time for the MC tripoint		
		

TWA= n	Arguments specified with an axis mask and an assignment (=)
TW n ...	Arguments specified with an implicit comma-separated order
_TWM	Operand holds the value set in the command

## DESCRIPTION

The TW command sets the timeout time for the MC tripoint. The TW command sets the timeout in msec to declare an error if the MC command is active and the motor is not at or beyond the actual position within n msec after the completion of the motion profile. If a timeout occurs, then the MC tripoint will clear and the stop code will be set to 99. A running program will jump to the special label #MCTIME, if located in the application code.

## ARGUMENTS

**TWm= n**

**TW n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
m	A	H	N/A	Axis	Axis to assign value	
n	-1	32,767	32,766	1	Set the timeout in msec for the MC command	n = -1 disables the timeout

## REMARKS

- The EN command should be used to return from the #MCTIME subroutine.

## EXAMPLES

```
TWA= 1000;' set timeout time for MC to 1000 for A axis
var= _TWA;' set value of TW for A axis to variable, var
```

```
TWA= 5000;'      set MC timeout to 5 seconds
PRA= 10000;'    set move length
BG A
MC A
MG "Move done";' message when move completes
EN
'
#MCTIME
'code when motor doesn't reach final pos in time
MG "Move didn't finish"
MG "Longer than ",_TWA," msecs"
ST A
AM A
MO A;'        shut off axis
EN
```

TW is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>TZ</b>	<i>Tell I/O Configuration</i>	INTERROGATION
The TZ command is used to request the I/O status of the controller		
		

TZ Command takes no arguments

## DESCRIPTION

The TZ command is used to request the I/O status of the controller. This is returned to the user as a human-readable text string.

## ARGUMENTS

### TZ

TZ is an interrogation command with no parameters

## REMARKS

- The data reported by TZ is also accessible through the TI (inputs) and OP (outputs) command

## EXAMPLES

### DMC41x3

```
:TZ
Block 0 (8-1) dedicated as input - value 255 (1111_1111)
Block 0 (8-1) dedicated as output - value 0 (0000_0000)
Block 1 (16-9) dedicated as input - value 255 (1111_1111)
Block 1 (16-9) dedicated as output - value 0 (0000_0000)
Block 10 (88-81) dedicated as input - value 255 (1111_1111)
Block 11 (96-89) dedicated as input - value 191 (1011_1111)
:
```

TZ is supported on DMC40x0, DMC41x3, DMC21x3, RIO firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>UI</b>	User Interrupt	SYSTEM CONFIG
The UI command allows user-defined interrupts to be created		

UI n ... Arguments specified with an implicit comma-separated order

## DESCRIPTION

The UI command allows user-defined interrupts to be created. UI can generate 16 different status bytes, \$F0 to \$FF (240-255), corresponding to UI0 to UI15.

DMC40x0, DMC41x3, DMC30010

UI pushes a user-defined status byte into the EI queue. When the UI command (e.g. UI5) is executed, the status byte value (e.g. \$F5 or 245) is queued up for transmission to the host, along with any other interrupts.

## ARGUMENTS

UI n

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	15	0	1	Set the status byte for the interrupt	

Status Byte (dec)	Condition	Status Byte (dec)	Condition
\$F0 (240)	UI0 was executed	\$F8 (248)	UI8 was executed
\$F1 (241)	UI1 was executed	\$F9 (249)	UI9 was executed
\$F2 (242)	UI2 was executed	\$FA (250)	UI10 was executed
\$F3 (243)	UI3 was executed	\$FB (251)	UI11 was executed
\$F4 (244)	UI4 was executed	\$FC (252)	UI12 was executed
\$F5 (245)	UI5 was executed	\$FD (253)	UI13 was executed
\$F6 (246)	UI6 was executed	\$FE (254)	UI14 was executed
\$F7 (247)	UI7 was executed	\$FF (255)	UI15 was executed

## REMARKS

DMC40x0, DMC41x3, DMC30010

- The UDP interrupt packet dispatch may be delayed depending on the number of interrupts in the queue
  - If immediate packet dispatch is required, use the message command (MG) to send a unique message to the host software.
- EI,h must be set to a valid UDP port (set by the host, not the DMC code, is recommended) before any interrupt packet will be dispatched.

## EXAMPLES

```
JG 5000; ' Jog at 5000 counts/s
BG A; ' Begin motion
AS A; ' Wait for at speed
UI 1; ' Cause an interrupt with status byte $F1 (241)
'The program above interrupts the host PC with status byte $F1 (241)
'when the motor has reach its target speed of 5000 counts/s
```

UI is supported on DMC40x0, DMC41x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>UL</b>	<i>Upload</i>	<b>PROGRAMMING</b>
<b>The UL command transfers data from the controller to a host computer</b>		
		

<b>UL</b>	Command takes no arguments
<b>_UL</b>	Operand has special meaning, see Remarks

## DESCRIPTION

The UL command transfers data from the controller to a host computer. Programs are sent without line numbers. The Uploaded program will be followed by a <control>Z or a '' as an end of text marker.

## ARGUMENTS

### UL

UL is a command with no parameters

## REMARKS

- In the Galil software, the UL command is not necessary because the UL command is handled by the graphical interface (Upload Program).
- In a terminal utility such as HyperTerminal or Telnet, the UL command will bring the uploaded program to screen.
- From there, the user can copy it and save it to a file.

### Operand Usage

- When used as an operand, \_UL gives the number of available variables.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
:UL; ' Begin upload
#A Line 0
NO This is an Example Line 1
NO Program Line 2
EN Line 3
{ctrl}Z Terminator
:
```

UL is supported on All firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VA</b>	<i>Vector Acceleration</i>	VECTOR/LINEAR
<b>The VA command sets the acceleration rate of the vector in a coordinated motion sequence</b>		

VAA= n	Arguments specified with an axis mask and an assignment (=)
VA n ...	Arguments specified with an implicit comma-separated order
_VAm	Operand holds the value set in the command

## DESCRIPTION

The VA command sets the acceleration rate of the vector in a coordinated motion sequence.

## ARGUMENTS

**VAm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

**VA n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	S	T	S	Axis	Coordinate plane to be specified	
n	1,024	1,073,740,800	256,000	1,024	Vector acceleration for the coordinate system	

## REMARKS

- \_VAm contains the value of the vector acceleration for the specified coordinate system

## EXAMPLES

```
:VA 1024;' Set vector acceleration to 1024 counts/sec2
:VA ?;' Return vector acceleration
1024
:VA 20000;' Set vector acceleration
:VA ?;'      Return vector acceleration
19456
:accel=_VAS;' Assign variable, accel, the value of VA
```

VA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)

Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VD</b>	<i>Vector Deceleration</i>	VECTOR/LINEAR
The VD command sets the deceleration rate of the vector in a coordinated motion sequence		

VDA= n	Arguments specified with an axis mask and an assignment (=)
VD n ...	Arguments specified with an implicit comma-separated order
VDm	Operand has special meaning see Remarks

## DESCRIPTION

The VD command sets the deceleration rate of the vector in a coordinated motion sequence.

## ARGUMENTS

**VDm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

**VD n,n**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
m	S	T	S	Axis	Coordinate plane to be specified	
n	1,024	1,073,740,800	256,000	1,024	Vector deceleration for the coordinate system	

Argument	Min	Max	Default	Resolution	Description	Notes
m	S	S	S	Axis	Coordinate plane to be specified	
n	1,024	1,073,740,800	256,000	1,024	Vector deceleration for the coordinate system	

## REMARKS

- VDm contains the value of the vector deceleration for the specified coordinate system

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#vector; ' Vector Program Label
VM AB; ' Specify plane of motion
VA 1000000; ' Vector Acceleration
VD 5000000; ' Vector Deceleration
VS 2000; ' Vector Speed
VP 1000,2000; ' Vector Position
VE; ' End Vector
BG S; ' Begin Sequence
AM S; ' Wait for Vector sequence to complete
EN; ' End Program
```

VD is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VE</b>	<i>Vector Sequence End</i>	VECTOR/LINEAR
<b>The VE command is used to mark the end segment of a coordinated move sequence</b>		

VE n ...	Arguments specified with an implicit comma-separated order
_VEm	Operand has special meaning see Remarks

## DESCRIPTION

The VE command is used to mark the end segment of a coordinated move sequence. VE is required to specify the end segment, and would follow the final VP or CR command in a sequence. VE is equivalent to the LE command for linear interpolation mode.

## ARGUMENTS

### VE n

Argument	Value	Description	Notes
n	0	Specify the end of a vector segment	Also occurs when n = 'null'
?		Returns the length of the vector in counts	

## REMARKS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The VE command will apply to the selected coordinate system, S or T. To select the coordinate system, use the command CAS or CAT.
- \_VEm contains the length of the vector in counts for the specified coordinate system, S or T

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#vector; ' Vector Program Label
VM AB; ' Specify plane of motion
VA 1000000; ' Vector Acceleration
VD 5000000; ' Vector Deceleration
VS 2000; ' Vector Speed
VP 1000,2000; ' Vector Position
VE; ' End Vector
BG S; ' Begin Sequence
AM S; ' Wait for Vector sequence to complete
EN; ' End Program
```

VE is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VF</b>	Variable Format	SYSTEM CONFIG
The VF command formats the number of digits to be displayed when interrogating the controller		

VF n ...	Arguments specified with an implicit comma-separated order
_VF	Operand has special meaning see Remarks

## DESCRIPTION

The VF command formats the number of digits to be displayed when interrogating the controller. If a number exceeds the format, the number will be displayed as the maximum possible positive or negative number (i.e. 999.99, -999, \$8000 or \$7FF).

## ARGUMENTS

### VF n0.n1

Argument	Min	Max	Default	Resolution	Description	Notes
n0	-8	10	10	1	Specify the number of digits displayed before the decimal point.	A negative value specifies hexadecimal format. See Remarks
n1	0	1023.875	6	0.125	Specify the number of digits displayed after the decimal point.	

## REMARKS

- A negative n0 specifies hexadecimal format. When in hexadecimal, the string will be preceded by a \$ and Hex numbers are displayed as 2's complement with the first bit used to signify the sign.
- A positive n0 specifies standard decimal format.
- A ? is only valid for querying n0. When queried, the value reported will be the value of the format for variables and arrays specified by n0 and n1
  - eg VF 10.4 would respond to VF ? with 10.4
- \_VF contains the value of the format for variables and arrays

## EXAMPLES

```
:VF 5.3;!  
' Sets 5 digits of integers and 3 digits after the decimal point  
:VF 8.0;!  
' Sets 8 digits of integers and no fractions  
:VF -4.0;!  
' Specify hexadecimal format with 4 bytes to the left of the decimal
```

```
:VF 8,4;!  
' set vf to 8 digits of integers and 4 digits of fraction  
:VF ?;!  
' query the value of VF  
8.4  
:MG _VF;!  
' query again  
8.4
```

VF is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VM</b>	<i>Vector Mode</i>	VECTOR/LINEAR
<b>The VM command enables the coordinated motion mode and specifies the plane of motion</b>		

VM A	Argument is an axis mask
_VMm	Operand has special meaning see Remarks

## DESCRIPTION

The VM command enables the coordinated motion mode and specifies the plane of motion. This mode may be specified for motion on any set of two axes, including a combination of real and virtual axes for single-axis operation. The motion is specified by the instructions VP and CR, which specify linear and circular segments.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Up to 511 segments may be given before the Begin Sequence (BGS or BGT) command. The number of available segments is queriable via the \_LMm operand.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

### VM m0m1m2

DMC40x0, DMC41x3, DMC21x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m0</b>	A	H	A	Axis	First axis specified for vector motion	
<b>m1</b>	A	H	B	Axis	Second axis specified for vector motion	
	M	N	B	Axis	Virtual axis specified for vector mode	Used when performing vector mode for a single real axis
<b>m2</b>	A	H	N/A	Axis	Tangent axis specified for vector mode.	m2 = null if tangent mode is not desired.
	M	N	N/A	Axis	Virtual axis specified for vector mode.	Used to disable the tangent function if already enabled. Otherwise, use m2 = null.

## REMARKS

- Specifying one axis for vector mode is useful for obtaining sinusoidal motion on 1 axis using the CR command.
- The Vector End (VE) command must be given after the last segment. This allows the controller to properly decelerate.
- Additional segments may be given during the motion when the buffer frees additional spaces for new segments.
- It is the responsibility of the user to keep enough motion segments in the buffer to ensure continuous motion.
- The first vector in a coordinated motion sequence defines the origin for that sequence. All other vectors in the sequence are defined by their endpoints with respect to the start of the move sequence.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The VM command will apply to the selected coordinate system, S or T. To select the coordinate system, use the command CAS or CAT.
- \_VMm contains instantaneous commanded vector velocity for the specified coordinate system, S or T.

### Enabling Vector Mode

- Specify the desired coordinate system to use with the CA command. S is default.
- Specify the vector plane to be used with the VMm0m1 command. If using tangent axis include that as the m2 parameter
  - EG. for a AB vector plane with the D axis used as a tangent axis, issue VM ABD
  - If only the vector plane is desired for the above example, then issue VM AB
- Specify vector speed with VS, vector acceleration with VA, and vector deceleration with VD
- Specify vector segments with the VP command, or circular segments with the CR command
- When finished with the sequence of moves, issue VE
- Issue BGS to begin motion for the S coordinate system
- You can now wait for motion to complete, issue additional segments as buffer space is cleared, or start a new move on the T coordinate plane by specifying CAT and starting from step 2.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#a;' Program Label
VM AB;' Specify motion plane
VP 1000,2000;' Specify vector position 1000,2000
VP 2000,4000;' Specify vector position 2000,4000
CR 1000,0,360;' Specify arc
VE;' Vector end
BG S;' Begin motion sequence
AM S;' Wait for vector motion to complete
EN;' End Program
```

```
#a;' Program Label
VM AN;' Specify motion plane
VP 1000,2000;' Specify vector position 1000,2000
VP 2000,4000;' Specify vector position 2000,4000
CR 1000,0,360;' Specify arc
VE;' Vector end
BG S;' Begin motion sequence
AM S;' Wait for vector motion to complete
EN;' End Program
```

VM is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VP</b>	<i>Vector Position</i>	VECTOR/LINEAR
<b>The VP command defines a vector move segment for the VM mode of motion</b>		

VP n ...	Arguments specified with an implicit comma-separated order
_VPm	Operand has special meaning see Remarks

## DESCRIPTION

The VP command defines a vector move segment for the VM mode of motion. The VP command defines the target coordinates of a straight line segment in a 2 axis motion sequence. The units are in quadrature counts, and are a function of the elliptical scale factor set using the command ES. For three or more axes in linear interpolation mode, use the LI command.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**VP n0,n1 < o > p**

DMC41x3, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	-2,147,483,648	2,147,483,647	0	1	Specify the target position for the first vector axis	See Remarks
<b>n1</b>	-2,147,483,648	2,147,483,647	0	1	Specify the target position for the second vector axis	See Remarks
<b>o</b>	2	15,000,000	N/A	2	Specifies a vector speed for the beginning of the segment	For MT 1,-1,1.5, and -1.5. If omitted, segment uses speed defined by VS command.
	2	3,000,000	N/A	2	Specifies a vector speed for the beginning of the segment	For MT 2,-2,2.5, and -2.5. If omitted, segment uses speed defined by VS command.
<b>p</b>	2	15,000,000	N/A	2	Specifies a vector speed for the end of the segment	For MT 1,-1,1.5, and -1.5. If omitted, segment uses speed defined by VS command.
	2	3,000,000	N/A	2	Specifies a vector speed for the end of the segment	For MT 2,-2,2.5, and -2.5. If omitted, segment uses speed defined by VS command.

DMC40x0, DMC41x3, DMC18x6, DMC30010

Argument	Value	Description	Notes
<b>o</b>	-1	Specifies vector speed to be set by Vector Speed Variable (VV command)	See VV command

## REMARKS

- The first vector in a coordinated motion sequence defines the origin for that sequence. All other vectors in the sequence are defined by their endpoints with respect to the start of the move sequence.
- Vector moves are defined as absolute positions from the origin of the sequence.
- The length of each vector segment must be limited to 8,388,607.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- The VM command will apply to the selected coordinate system, S or T. To select the coordinate system, use the command CAS or CAT.
- \_VPm where m = axis designator A,B,C,D,E,F,G or H and contains the absolute coordinate of the axes at the last intersection along the sequence.
  - For example, during the first motion segment, this instruction returns the coordinate at the start of the sequence.
  - The use of \_VPm as an operand is valid in the linear mode, LM, and in the Vector mode, VM.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#a;' Program Label
VM AB;' Specify motion plane
VP 1000,2000;' Specify vector position 1000,2000
VP 2000,4000;' Specify vector position 2000,4000
CR 1000,0,360;' Specify arc
VE;' Vector end
BG S;' Begin motion sequence
AM S;' Wait for vector motion to complete
EN;' End Program
```

DMC40x0, DMC41x3, DMC18x6, DMC30010

```
REM VP n,m p
REM 'o' and 'p' are in counts/sample rather than counts/second as the VS command.
REM This means that when TM <> 1000, commanded speed for VS will be different than
REM values for 'o' and 'p'
REM To get counts/second for 'o' and 'p', divide them by a ratio of 1000/_TM
REM
REM #vs and #vsop result in the same profile
#vs
TM 250
VM AN
VS 100000
VA 2560000
VD 2560000
VP 20000,20000
VE
BG S
```

```
AM S
EN
'
#vsop
TM 250
VM AN
n= 1000/_TM
'VS 100000
VA 2560000
VD 2560000
VP 20000,20000<(100000/n)
VE
BG S
AM S
EN
```

VP is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VR</b>	<i>Vector Speed Ratio</i>	VECTOR/LINEAR
The VR sets a ratio to be used as a multiplier of the current vector speed		

VR n ...	Arguments specified with an implicit comma-separated order
_VRm	Operand holds the value set in the command

## DESCRIPTION

The VR sets a ratio to be used as a multiplier of the current vector speed. The vector speed can be set by the command VS or the operators < and > used with CR, VP and LI commands. VR takes effect immediately and will ratio all the previous vector speed commands.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

**VR n0,n1**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
n0	0	10	1	1/65,536	Vector ratio specified for the S coordinate plane	
n1	0	10	1	1/65,536	Vector ratio specified for the T coordinate plane	

## REMARKS

- VR doesn't ratio acceleration or deceleration, but the change in speed is accomplished by accelerating or decelerating at the rate specified by VA and VD.
- VR is useful for feedrate override, particularly when specifying the speed of individual segments using the operator '<' and '>'.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

- \_VRm contains the vector speed ratio of the specified coordinate system - where m = S or T.
- \_VRS contains the vector speed ratio of the specified coordinate system

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
#a;'          Vector Program
VM AB;'      Vector Mode
VP 1000,2000;' Vector Position
CR 1000,0,360;' Specify Arc
VE;'          End Sequence
VS 2000;'      Vector Speed
BG S;'        Begin Sequence
AM S;'        After Motion
JP #a;'        Repeat Move
#speed;'      Speed Override
VR (@AN[1]*.1);' Read analog input compute ratio
vr= _VRS;'    Store vector ratio in variable 'vr'
JP #speed;'  Loop
XQ #a,0
XQ #speed,1;' Execute task 0 and 1 simultaneously
```

VR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VS</b>	Vector Speed	VECTOR/LINEAR
The VS command specifies the speed of the vector in a coordinated motion sequence in either the LM or VM modes		

VSA= n	Arguments specified with an axis mask and an assignment (=)
VS n ...	Arguments specified with an implicit comma-separated order
_VSm	Operand has special meaning, see Remarks

## DESCRIPTION

The VS command specifies the speed of the vector in a coordinated motion sequence in either the LM or VM modes. This speed is in place when individual segment speeds for VP, LI and CR are not specified.

## ARGUMENTS

**VSm= n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

**VS n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	S	T	S	Axis	Coordinate plane to be specified	
<b>n</b>	2	15,000,000	25,000	2	Vector speed applied to the coordinate system	

## REMARKS

- Vector speed can be attached to individual vector segments using the operators '<' and '>'. For more information, see description of VP, CR, and LI commands. The VV command allows for variables to be specified during vector segments.
- Vector Speed can be calculated by taking the square root of the sum of the squared values of speed for each axis specified for vector or linear interpolated motion.
- \_VSm contains the vector speed of the specified coordinate system

## EXAMPLES

```
:VS 2000;" Define vector speed of S coordinate system
:VS ?;" Return vector speed of S coordinate system
2000
:
```

VS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>VV</b>	<i>Vector Speed Variable</i>	VECTOR/LINEAR
The VV command sets the speed of the vector variable in a coordinated motion sequence in either the LM or VM modes		
VVA= n	Arguments specified with an axis mask and an assignment (=)	
VV n ...	Arguments specified with an implicit comma-separated order	
_Vvm	Operand has special meaning see Remarks	

## DESCRIPTION

The VV command sets the speed of the vector variable in a coordinated motion sequence in either the LM or VM modes. The VV command is used to set the "o" vector speed argument for segments that exist in the vector buffer for LI, CR and VP commands. By defining a vector segment begin speed as a negative 1 (i.e. "<-1"), the controller will utilize the current vector variable speed as the segment is profiled from the buffer.

## ARGUMENTS

**VVm= n**

DMC40x0, DMC41x3, DMC18x6

**VV n,n**

DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	S	T	S	Axis	Coordinate plane to assign value	
<b>n</b>	0	15000000	0	2	Variable vector speed	For MT 1,-1,1.5 and -1.5
	0	3000000	0	2	Variable vector speed	For MT 2,-2,2.5 and -2.5

## REMARKS

- VV command is useful when vector segments exist in the buffer that use the "<" and ">" speed indicators for specific segment and corner speed control and the host needs to be able to dynamically change the nominal return operating speed.
- \_Vvm contains the vector speed variable of the specified coordinate system

## EXAMPLES

```
:VVS= 20000; Define vector speed variable to 20000 for the S coordinate system
:VP 1000,2000<-1>100; Define vector speed variable for specific segment.
:VVS= ?
20000
:
```

VV is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>WH</b>	<i>Which Handle</i>	INTERROGATION
The WH command is used to identify the handle from which the command was received		
		

WH	Command takes no arguments
_WH	Operand has special meaning see Remarks

## DESCRIPTION

The WH command is used to identify the handle from which the command was received. This is useful for determining what interface or handle you are connected to.

## ARGUMENTS

### WH

WH is an interrogation command with no parameters

## REMARKS

- WH contains the numeric representation of the handle from which the command was received.
- The following table lists the possible string returned by WH, and the numerical value returned by \_WH

DMC41x3

Communication Channel	WH	_WH
Main USB Port	USB	-1
Ethernet Handle A	IHA	0
Ethernet Handle B	IHB	1
Ethernet Handle C	IHC	2
Ethernet Handle D	IHD	3
Ethernet Handle E	IHE	4
Ethernet Handle F	IHF	5
Ethernet Handle G	IHG	6
Ethernet Handle H	IHH	7

## EXAMPLES

```
:WH;^ Request incoming handle identification
IHC
:MG _WH
2
```

DMC41x3

```
:WH;^ Request incoming handle identification
USB Command received from USB port
:MG _WH
-1
```

WH is supported on DMC40x0, DMC41x3, DMC21x3, RIO, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>WT</b>	Wait	PROGRAMMING, TRIPPOINT
The WT command is a trippoint used to time events		

WT n ... | Arguments specified with an implicit comma-separated order

## DESCRIPTION

The WT command is a trippoint used to time events. When this command is executed, the controller will wait for the amount of time specified before executing the next command.

DMC40x0, DMC41x3, DMC30010

The amount of time in the WT command is specified to be either samples or milliseconds, depending on the second argument of WT

## ARGUMENTS

DMC40x0, DMC41x3, DMC30010

**WT n0,n1**

DMC40x0, DMC41x3, DMC30010

Argument	Min	Max	Default	Resolution	Description	Notes
<b>n0</b>	2	2,147,483,646	N/A	2	Specify amount of time to hold execution of code	
<b>n1</b>	0	1	0	1	Specify the type of WT	n = 0 or null specifies WT in msec. n = 1 specifies WT in samples

## REMARKS

DMC40x0, DMC41x3, DMC30010

- If n1=1 for WTn0,n1 then the controller will wait for the number of samples specified before executing the next command.
- By default, WT is specified in milliseconds. If n1 is omitted, then n1 = 0 is used and WT is timed in milliseconds

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

```
'10 seconds after a move is complete, turn on a relay for 2 seconds
#a;'          Program A
PR 50000;      Position relative move
BG A;          Begin the move
AM A;          After the move is over
WT 10000;      Wait 10 seconds
SB 1;          Turn on relay (set output 1)
WT 2000;       Wait 2 seconds
CB 1;          Turn off relay (clear output 1)
EN;           End Program
```

WT is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>XQ</b>	<i>Execute Program</i>	PROGRAMMING
-----------	------------------------	-------------

The XQ command begins execution of a program residing in the program memory of the controller



XQ n ...	Arguments specified with an implicit comma-separated order
_XQ0, _XQ1, _XQ2, _XQ3, _XQ4, _XQ5, _XQ6, _XQ7	Operand has special meaning see Remarks

## DESCRIPTION

The XQ command begins execution of a program residing in the program memory of the controller. Execution will start at the label or line number specified.

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Up to 8 programs may be executed simultaneously to perform multitasking

## ARGUMENTS

**XQ #str,n1**

**XQ n0,n1**

DMC40x0, DMC41x3, DMC18x6

Argument	Min	Max	Default	Resolution	Description	Notes
str	1 char	7 chars	See Notes	String	Label to begin code execution	If omitted, start from line 0 (n0=0)
n0	0	1999	0	1	Line number to begin code execution	
n1	0	7	0	1	Thread number to execute code	

## REMARKS

- XQn contains the current line number of execution for thread n, and -1 if thread t is not running
- If using ED to add code, you must exit ED mode before executing code.

## EXAMPLES

```
XQ #apple,0; Start execution at label apple, thread zero
XQ #data,2; Start execution at label data, thread two
XQ : Start execution at line 0
```

XQ is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>YA</b>	<i>Step Drive Resolution</i>	STEPPER MOTOR
The YA command specifies the resolution of the step drive, in step counts per full motor step, for Stepper Position Maintenance (SPM) mode		

YAA= n	Arguments specified with an axis mask and an assignment (=)
YA n ...	Arguments specified with an implicit comma-separated order
YAm	Operand holds the value set in the command

## DESCRIPTION

The YA command specifies the resolution of the step drive, in step counts per full motor step, for Stepper Position Maintenance (SPM) mode.

## ARGUMENTS

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**YA** **m**= **n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010

**YA** **n,n,n,n,n,n,n**

DMC40x0, DMC41x3

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	9,999	2	1	Drive resolution in step counts/motor step for SPM mode	YA has special functionality for certain hardware configurations. See Notes below.
	1	16	2	see Notes	Valid settings for SDM-44040 (-D4040)	1,2,4 and 16 set the step resolution of the SDM-44040 to full, half, 1/4th and 1/16th microstepping respectively
	64	64	2	0	Valid setting for SDM-44140 (-D4140)	The SDM-44140 is always configured for 64th microstepping. YA must be set to 64 for SPM mode

## REMARKS

- None

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6

```
'Set the step drive resolution for a 1/64 Microstepping Drive:  
:YA 64,64,64,64  
:'Query the D axis value  
:MG _YAD  
64.0000 Response shows D axis step drive resolution
```

```
'Set the step drive resolution for a 1/256 Microstepping Drive:  
:YA 256  
:'Query the A axis value  
:MG _YA  
256.0000 Response shows A axis step drive resolution
```

YA is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>YB</b>	<i>Step Motor Resolution</i>	STEPPER MOTOR
-----------	------------------------------	---------------

The YB command specifies the resolution of the step motor, in full steps per full revolution, for Stepper Position Maintenance (SPM) mode



YBA= n	Arguments specified with an axis mask and an assignment (=)
YB n ...	Arguments specified with an implicit comma-separated order
YBm	Operand holds the value set in the command

## DESCRIPTION

The YB command specifies the resolution of the step motor, in full steps per full revolution, for Stepper Position Maintenance (SPM) mode.

## ARGUMENTS

**YBm= n**

**YB n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	9,999	200	1	Motor resolution in full steps/revolution	

## REMARKS

- This command is only required if using SPM mode with stepper motors with an attached encoder.
- A 1.8 degree step motor is 200 steps/revolution.

## EXAMPLES

```
'Set the step motor resolution of the A axis for a 1.8 degree step motor:  
:YBA= 200  
:'Query the A axis value  
:YBA= ?  
200 Response shows A axis step motor resolution
```

YB is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>YC</b>	<i>Encoder Resolution</i>	STEPPER MOTOR
The YC command specifies the resolution of the encoder, in counts per revolution, for Stepper Position Maintenance (SPM) mode		

YCA= n	Arguments specified with an axis mask and an assignment (=)
YC n ...	Arguments specified with an implicit comma-separated order
YCm	Operand holds the value set in the command

## DESCRIPTION

The YC command specifies the resolution of the encoder, in counts per revolution, for Stepper Position Maintenance (SPM) mode.

## ARGUMENTS

**YCm= n**

**YC n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	32,766	2,000	1	Encoder resolution in counts/revolution	

## REMARKS

- This command is only required if using SPM mode with stepper motors with an attached encoder.

## EXAMPLES

```
'Set the encoder resolution of the A axis
:YCA= 2000
:'Query the A axis value
:YBA= ?
2000 Response shows A axis encoder resolution
```

YC is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>YR</b>	Error Correction	STEPPER MOTOR
The YR command allows the user to correct for position error in Stepper Position Maintenance mode		

YRA= n	Arguments specified with an axis mask and an assignment (=)
YR n ...	Arguments specified with an implicit comma-separated order

## DESCRIPTION

The YR command allows the user to correct for position error in Stepper Position Maintenance mode. This correction acts like an IP command, moving the axis or axes the specified quantity of step counts. YR will typically be used in conjunction with QS.

## ARGUMENTS

**YRm= n**

**YR n,n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2,147,483,648	2,147,483,647	0	1	Number of step pulses to increment position by	

## REMARKS

- Users will typically use the value of QS to increment motor by the number of step pulses of error.
  - EG. YRm= \_QSm increments the specified axis by the error magnitude.
- The sign of YR depends on the polarity of the position encoder
  - If the encoder increments when the stepper moves forward (increasing TD), the correction is YRm= \_QSm. This is typical.
  - If the encoder decrements when the stepper moves forward, the correction is YRm= -\_QSm. See CE to invert the polarity of the position encoder, if desired.

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Query the error of the B axis:  
:qsb= ?  
253 This shows 253 step counts of error  
:'Correct for the error:  
:YRB= _QSB;' The motor moves _QS step counts to correct for the error  
:'and YS is set back to 1
```

```
'Query the error of the A axis:  
:qsa= ?  
253 This shows 253 step counts of error  
:'Correct for the error:  
:YRA= _QSA;' The motor moves _QS step counts to correct for the error  
:'and YS is set back to 1
```

YR is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>YS</b>	<i>Stepper Position Maintenance Mode Enable, Status</i>	STEPPER MOTOR
The YS command enables and disables the Stepper Position Maintenance Mode function		
YSA= n	Arguments specified with an axis mask and an assignment (=)	
YS n ...	Arguments specified with an implicit comma-separated order	
YSm	Operand holds the value set in the command	

## DESCRIPTION

The YS command enables and disables the Stepper Position Maintenance Mode function. YS also reacts to excessive position error condition as defined by the QS command.

## ARGUMENTS

**YSm= n**

**YS n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	0	1	0	1	Setting of the SPM mode	n = 0 disables SPM mode, n = 1 Enables SPM mode. See Remarks

## REMARKS

- Both YSm=? and \_YSm contain the value of n. n is 1 when SPM mode is enabled and no error has occurred. If a position error has occurred, n becomes 2.
- If n = 2, this indicates a position error condition detected by QS has occurred.
  - Issuing an n = 1 will clear the error

## EXAMPLES

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

```
'Enable the mode:  
:YSH= 1  
:'Query the value:  
:YST= ?  
0,0,0,0,0,0,0,1 Response shows H axis is enabled
```

```
'Enable the mode:  
:YSA= 1  
:'Query the value:  
:YSA= ?  
1 Response shows A axis is enabled
```

YS is supported on DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

<b>ZA</b>	User Data Record Variables	PROGRAMMING
ZA sets the user variables in the data record		

ZAA= n	Arguments specified with an axis mask and an assignment (=)
ZA n ...	Arguments specified with an implicit comma-separated order
ZAm	Operand holds the value set in the command

## DESCRIPTION

ZA sets the user variables in the data record. The user variables (one per axis) are automatically sent as part of the status record from the controller to the host computer. These variables provide a method for specific controller information to be passed to the host automatically.

## ARGUMENTS

**ZAm= n**

**ZA n,n,n,n,n,n,n**

DMC40x0, DMC41x3, DMC21x3, DMC18x6, DMC18x2

Argument	Min	Max	Default	Resolution	Description	Notes
<b>m</b>	A	H	N/A	Axis	Axis to assign value	
<b>n</b>	-2147483648	2147483647	0	1	Value of user variable for data record	

## REMARKS

- n is an integer and can be a number, controller operand, variable, mathematical function, or string
- Only 4 bytes are available for n. Fractional values are not stored or sent via the data record

## EXAMPLES

```
#thread
ZAA= myvar; 'constantly update ZA with variable myVar
WT 10
JP #thread
```

ZA is supported on DMC40x0, DMC41x3, DMC18x6, DMC30010 firmware platform(s)  
 Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)

The ZS command is used to clear the stack when finishing or leaving a subroutine



ZS	Only valid from within embedded code. No terminal.
ZS0, ZS1, ZS2, ZS3, ZS4, ZS5, ZS6, ZS7	Operand has special meaning, see Remarks

## DESCRIPTION

The ZS command is used to clear the stack when finishing or leaving a subroutine. This command is used to avoid returning from an interrupt (either input or error). This turns the jump to subroutine into a jump. The status of the stack can be interrogated with the operand \_ZS, see Remarks.

## ARGUMENTS

### ZS n

Argument	Min	Max	Default	Resolution	Description	Notes
n	0	1	0	1	Sets zero stack operation	n = 0 clears the entire stack. n = 1 clears one level of the stack.

## REMARKS

- Do not use RI (Return from Interrupt) when using ZS.
  - To re-enable interrupts, you must use II command again.

## Operand Usage

DMC40x0, DMC21x3, DMC18x6, DMC18x2, DMC41x3

- \_ZSn contains the stack level for the specified thread where n = 0 to 7.
  - The response, an integer between zero and sixteen, indicates zero for beginning condition and sixteen for the deepest value.

## EXAMPLES

DMC40x0, DMC21x3, DMC18x6, DMC18x2, DMC41x3, DMC30010

```
#a;          Main Program
II 1;        Input Interrupt on 1
#b;JP #b;EN ;' Loop
#ININT;      Input Interrupt
MG "INTERRUPT";'Print message
s= _ZS0;     Interrogate stack
s= ?;        Print stack
ZS;          Zero stack
s= _ZS0;     Interrogate stack
s= ?;        Print stack
EN;          End
```

ZS is supported on All firmware platform(s)  
Corrections, Feedback: [documentation@galilmc.com](mailto:documentation@galilmc.com)