Navigator[™] Motion Processor

Programmer's Reference



Performance Motion Devices, Inc. 12 Waltham St. Lexington, MA 02421

NOTICE

This document contains proprietary and confidential information of Performance Motion Devices, Inc., and is protected by federal copyright law. The contents of this document may not be disclosed to third parties, translated, copied, or duplicated in any form, in whole or in part, without the express written permission of PMD.

The information contained in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form, by any means, electronic or mechanical, for any purpose, without the express written permission of PMD.

Copyright 1998, 1999 by Performance Motion Devices, Inc. Navigator and C-Motion are trademarks of Performance Motion Devices, Inc

Navigator Motion Processor Programmer's Reference

Warranty

PMD warrants performance of its products to the specifications applicable at the time of sale in accordance with PMD's standard warranty. Testing and other quality control techniques are utilized to the extent PMD deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Performance Motion Devices, Inc. (PMD) reserves the right to make changes to its products or to discontinue any product or service without notice, and advises customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

Safety Notice

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage. Products are not designed, authorized, or warranted to be suitable for use in life support devices or systems or other critical applications. Inclusion of PMD products in such applications is understood to be fully at the customer's risk.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent procedural hazards.

Disclaimer

PMD assumes no liability for applications assistance or customer product design. PMD does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of PMD covering or relating to any combination, machine, or process in which such products or services might be or are used. PMD's publication of information regarding any third party's products or services does not constitute PMD's approval, warranty or endorsement thereof.

Related Documents

Navigator Motion Processor User's Guide (MC2000UG)

How to set up and use all members of the Navigator Motion Processor family.

Navigator Motion Processor Programmer's Reference (MC2000PR)

Descriptions of all Navigator Motion Processor commands, with coding syntax and examples, listed alphabetically for quick reference.

Navigator Motion Processor Technical Specifications

Four booklets containing physical and electrical characteristics, timing diagrams, pinouts, and pin descriptions of each series:

MC2100 series, for brushed servo motion control (MC2100TS); MC2300 series, for brushless servo motion control (MC2300TS); MC2400 series, for microstepping motion control (MC2400TS); MC2500 series, for stepping motion control (MC2500TS).

Navigator Motion Processor Developer's Kit Manual (DK2000M)

How to install and configure the DK2000 developer's kit PC board.

Table of Contents

Warranty	iii
Safety Notice	iii
Disclaimer	iii
Related Documents	iv
Table of Contents	v
1 The Navigator Family	7
2 Instruction Reference	
2.1 How to use this reference	9
3 Instruction Summary Tables	
3.1 Descriptions by Functional Category	
3.2 Alphabetical Listing	
3.3 Numeric Listing	

Navigator Motion Processor Programmer's Reference

1 The Navigator Family

	MC2100 Series	MC2300 Series	MC2400 Series	MC2500 Series
# of axes	4, 2, or 1	4, 2 or 1	4, 2 or 1	4, 2, or 1
Motor type supported	Brushed servo	Brushless servo	Stepping	Stepping
Output format	Brushed servo (single phase)	Commutated (6- step or sinusoidal)	Microstepping	Pulse and direction
Incremental encoder input	\checkmark	\checkmark	√	\checkmark
Parallel word device input	\checkmark	\checkmark	\checkmark	√
Parallel communication	\checkmark	\checkmark	\checkmark	√
Serial communication	\checkmark	\checkmark	√	\checkmark
Diagnostic port	\checkmark	\checkmark	\checkmark	√
S-curve profiling	\checkmark	\checkmark	\checkmark	\checkmark
Electronic gearing	\checkmark	\checkmark	\checkmark	\checkmark
On-the-fly changes	\checkmark	\checkmark	\checkmark	\checkmark
Directional limit switches	\checkmark	\checkmark	\checkmark	\checkmark
Programmable bit output	\checkmark	\checkmark	\checkmark	\checkmark
Software-invertable signals	\checkmark	\checkmark	\checkmark	\checkmark
PID servo control	\checkmark	\checkmark	-	-
Feedforward (accel & vel)	\checkmark	\checkmark	-	-
Derivative sampling time	\checkmark	\checkmark	-	-
Data trace/diagnostics	\checkmark	\checkmark	\checkmark	\checkmark
PWM output	\checkmark	\checkmark	\checkmark	-
Motion error detection	\checkmark	\checkmark	\checkmark (with encoder)	\checkmark (with encoder)
Axis settled indicator	\checkmark	\checkmark	\checkmark (with encoder)	\checkmark (with encoder)
DAC-compatible output	\checkmark	\checkmark	\checkmark	-
Pulse & direction output	-	-	-	\checkmark
Index & Home signals	\checkmark	\checkmark	\checkmark	\checkmark
Position capture	\checkmark	\checkmark	\checkmark	\checkmark
Analog input	\checkmark	\checkmark	\checkmark	\checkmark
User-defined I/O	\checkmark	\checkmark	\checkmark	\checkmark
External RAM support	\checkmark	\checkmark	\checkmark	\checkmark
Chipset part numbers	MC2140 (4 axes)	MC2340 (4 axes)	MC2440 (4 axes)	MC2540 (4 axes)
	MC2120 (2 axes)	MC2320 (2 axes)	MC2420 (2 axes)	MC2520 (2 axes)
	MC2110 (1 axis)	MC2310 (1 axis)	MC2410 (1 axis)	MC2510 (1 axis)
Developer's Kit p/n's:	DK2100	DK2300	DK2400	DK2500

Introduction

This manual describes the format of instructions supported by the Navigator family of Motion Processors from PMD. These devices are members of PMD's second-generation motion processor family, which consists of 12 separate products organized into 4 series.

Each of these devices are complete chip-based motion processors. They provide trajectory generation and related motion control functions. Depending on the type of motor controlled they provide servo loop closure, on-board commutation for brushless motors, and high speed pulse and

direction outputs. Together these products provide a software-compatible family of dedicated motion processors that can handle a large variety of system configurations.

Each of these chips utilize a similar architecture, consisting of a high-speed DSP (Digital Signal Processor) computation unit, along with an ASIC (Application Specific Integrated Circuit). The computation unit contains special on-board hardware that makes it well suited for the task of motion control.

Along with similar hardware architecture these chips also share most software commands, so that software written for one chipset may be re-used with another, even though the type of motor may be different.

Each chipset consists of two PQFP (Plastic Quad Flat Pack) ICs: a 100-pin Input/Output (I/O) chip, and a 132-pin Command Processor (CP) chip.

The four different series in the Navigator family are designed for a particular type of motor or control scheme. Here is a summary description of each series.

Family Summary

MC2100 Series (MC2140, MC2120, MC2110) – This series outputs motor commands in either Sign/Magnitude PWM or DAC-compatible format for use with brushed servo motors, or with brushless servo motors having external commutation.

MC2300 Series (MC2340, MC2320, MC2310) – This series outputs sinusoidally commutated motor signals appropriate for driving brushless motors. Depending on the motor type, the output is a two-phase or three-phase signal in either PWM or DAC-compatible format.

MC2400 Series (MC2440, MC2420, MC2410) – This series provides microstepping signals for stepping motors. Two phased signals per axis are generated in either PWM or DAC-compatible format.

MC2500 Series (MC2540, MC2520, MC2510) – These chipsets provide high-speed pulse and direction signals for stepping motor systems.

2 Instruction Reference

2.1 How to use this reference

This document is in two parts: first, a detailed description of all host instructions, and second, a set of summary tables listing the instructions by functional group, alphabetically by instruction mnemonic, and numerically by hexadecimal code.

In the reference section, instructions are arranged alphabetically, **except** that all "Set/Get" pairs (for example, **SetVelocity** and GetVelocity) are described together. Each description begins on a new page; most occupy no more than a page. The page is organized as follows:

Name	The instruction mnemonic is shown at the left, its hexadecimal code at the right.				
Syntax	The instruction mnemonic and its required arguments are shown with all arguments separated by spaces.				
Arguments	There are two types of arguments: encoded-field and numeric.				
	Encoded-field arguments are packed into a single 16-bit data word, except for axis, which occupies bits 11-8 of the instruction word. The Name of the argument is that shown in the generic syntax. Instance mnenomic used to represent the data value. Encoding is the value assigned to the field for that instance.				
	For numeric arguments, the parameter Value , the Type (signed or unsigned integer) and Range of acceptable values are given. Numeric arguments may require one or two data words. For 32-bit arguments, the high-order part is transmitted first.				
Buffered	Certain parameters and other data written to the chipset are buffered, that is, they are not acted upon until the next Update or MultiUpdate command is executed. These parameters are identified by the word buffered in the instruction heading.				
Packet structure	This is a graphic representation of the 16-bit words transmitted in the packet: the instruction, which is identified by its name, followed by 1, 2, or 3 data words. Bit numbers are shown directly below each word. For each field in a word, only the high and low bits are shown. For 32-bit numeric data, the high-order bits are numbered from 31 to 16, the low-order bits from 15 to 0.				
	The hex code of the instruction is shown in boldface.				
	Argument names are shown in their respective words or fields.				
	For data words, the direction of transfer—read or write—is shown at the left of the word's diagram.				
	Unused bits are shaded. In data words and instructions sent (written) to the motion processor, all unused bits must be 0.				
Description	Describes what the instruction does and any special information relating to the instruction.				
Restrictions	Describes the circumstances in which the instruction is not valid, that is, when it should not be issued. For example, velocity, acceleration, deceleration, and jerk parameters may not be issued while an S-curve profile is being executed.				
see	Refers to related instructions.				

AdjustActualPosition

Syntax	AdjustActualPosition axis position							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
	position	Type signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> unity	Units counts steps			
Packet structure		AdjustA	ctualPosition					
	0	axis		F5 h				
	15	12 11 First	8 7 data word		0			
write	position (high-order p	part)						
	31	Secon	d data word		16			
write	position (low-order pa	art)						
	15				0			
Description	The <i>position</i> spe	cified as the parar	neter to AdiustA	ctualPosition i	s summed with			

The *position* specified as the parameter to AdjustActualPosition is summed with the actual position register (encoder position) for the specified *axis*. This has the effect of adding or subtracting an offset to the current actual position. At the same time, the current commanded position is replaced by the new actual position value minus the current actual position error. This prevents a servo "bump" when the new axis position is established. The destination position (see **SetPosition**) is also modified by this amount so that no trajectory motion will occur when the update instruction is issued. In effect, this instruction establishes a new reference position from which subsequent positions can be calculated. It is commonly used to set a known reference position after a homing procedure.

Note: On the MC2400 and MC2500 series, the current actual position error is zeroed.

AdjustActualPosition takes effect immediately, it is not buffered.

Restrictions

see

GetPositionError; GetActualVelocity, Set/GetActualPositionUnits, Set/GetActualPosition

ClearInterrupt

Syntax	ClearInterrupt
Arguments	none
Packet structure	ClearInterrupt
	15 8 7 0
Description	ClearInterrupt resets the HostInterrupt signal to its inactive state. If interrupts are still pending, the HostInterrupt line will return to its active state within one cycle. It is used after an interrupt has been recognized and processed by the host. This command does not affect the Event Status Register. If this command is executed when no interrupts are pending it has no effect.
Restrictions	
see	GetInterruptAxis, Set/GetInterruptMask

ClearPositionError

Syntax	ClearPositionError axis						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
Packet structure			Cl	earPos	itionError		
	15	0	axis		_	47 h	
	15	12	11	8	1		U
Description	ClearPos actual po axis. Th it is used destinatio comman	sitionError osition (enc is comman when the on position d.	sets the curr oder input), nd can be use axis is movin n (used in traj	ent pr therel d who g the pezoio	ofile's comman oy clearing the en the axis is at host should be dal and s-curve	nded position position error rest, or when aware that th modes) is no	equal to the for the specified it is moving. If he trajectory t changed by this
Restrictions	ClearPositionError is a buffered command. The new value set will not take effect until the next Update or MultiUpdate instruction is entered. This command cannot be executed while the chip is performing an s-curve profile.						
see	GetPosit	tionError, I	MultiUpdate	, Set	GetPositionEr	rorLimit, Upo	date

GetActivityStatus

Syntax	GetActivityStatus axis															
Arguments	Name axis	Instan Axis Axis Axis Axis	2 1 2 3 4	<i>En</i> 0 1 2 3	ncodin	ng										
Returned data	status			se	ee be	elow										
Packet structure GetActivityStatus							-									
		0			ax	cis					A6	i h				
	15		12	11			8	7							0	
							Data	а								-
read															1	
	15	13	12	11	10	9	8	7	6	5		3	2	1	0	-
Description	GetActivit Each of the without as state of the The follow	t yStatu he bits ny acti nese bi wing ta	IS re in tl on o ts, si able	ads t nis re n the nce t show	he 10 egiste e par hey a	6 bit er con t of t are co e enc	activ ntinu he h ontro odin	ity stored iously ost. olled g of	tatus y ind The by the the c	regis icate re is ne ch lata 1	ster fo the s no di nip se returr	or th state rect t. ned h	ne sp of tl way by th	ecifie he ch to se is co	ed ax lipse et or mma	(is . t clear the and.

Name	Bit Number	Description							
Phasing initialized	0	Set to 1 if phasing is initialized							
		(MC2300/MC2800 series only)							
At maximum	1	Set to 1 when the trajectory is at maximum							
velocity		velocity. This bit is determined by the							
		trajectory generator, not the actual encoder							
		position.							
Iracking	2	Set to 1 when the axis is within the							
<u> </u>	0.5	tracking window							
Current profile	3-5	Contains trajectory mode encoded as							
mode		bit 5 bit 4 bit 2 Profile Mode							
		DIL 5 DIL 4 DIL 5 FIOTILE MODE							
		0 1 1 electronic gear							
reserved	6	not used, may be 0 or 1							
Axis settled	7	Set to 1 when the axis is settled							
Motor on/off	8	Set to 1 when motor mode is on, 0 when							
		off.							
Position capture	9	Set to 1 when a value has been captured							
		by the high speed position capture							
		hardware but has not yet been read. The							
		GetCaptureValue command must be							
		executed before another capture can occur.							
In-motion	10	Set to 1 when the trajectory generator is							
		executing a profile on the axis.							
In positive limit	11	Set to 1 when the positive limit switch is							
		active							

Name	Bit Number	Description
In negative limit	12	Set to 1 when the negative limit switch is active
Profile segment	13-15	Only used during S-curve profile mode. Contains value of 0 when the profile is at rest. Contains phase number 1-7 when profile is in motion.

Restrictions

GetEventStatus, GetSignalStatus see

GetActualVelocity

Syntax		GetActualVelocity axis						
Arguments		Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
Returned data		velocity	<i>Түре</i> signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	<i>Units</i> counts/cycle		
Packet structure			GetActu	alVelocity				
		0	axis	7	AD h			
		15	First da	ata word		U		
	read	Actual velocity (hig	gh-order part)					
		31	Second	data word		16		
	read	Actual velocity (low	w-order part)					
		15				0		
Description		GetActualVelocity reads the current actual velocity for the specified axis . This value is the result of the last encoder input, so it will be accurate to within one cycle.						
		Scaling exampl command (hig of -1,234,567/	le: If a value of -1,234 h word: 0FFEDh, low 65,536 or -18.8380 co	,567 is retrieved v word: 2979h) t unts/cycle.	by the GetA his correspo	actualVelocity nds to a velocity		
Restrictions								
see		Set/GetActua	IPosition					

GetCaptureValue

Syntax	GetCaptureValue axis						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3				
Returned data	captured position	Түре signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> unity	Units counts		
Packet structure	GetCaptureValue						
	0	axis		36 h			
	15	12 11 First c	8 7 lata word		0		
read	cantured position (high-order part)						
1000	31				16		
		Second	data word				
read	captured position (lov	v-order part)					
	15				U		
Description	GetCaptureValue specified <i>axis</i> . T capture to occur.	e returns the conte 'his command also	nts of the Position resets the captur	on Capture Reg re hardware to a	ister for the llow another		
Restrictions							

see Set/GetCaptureSource

GetChecksum



Restrictions

see

GetCommandedAcceleration

Syntax	GetCommanded	GetCommandedAcceleration axis							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3						
Returned data	acceleration	<i>Type</i> signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	<i>Units</i> counts/cycle ²				
Packet structure		GetCommanded	Acceleration						
	0	axis		A7 h					
	15	12 11 8 7 First data	, word		0				
read	acceleration (high-orde	er part)	, word						
	31	Second da	ta word		16				
read	acceleration (low-orde	r part)							
	15	· ·			0				
Description	GetCommanded, for the specified a value output by th	Acceleration returns t axis. Commanded ac ne trajectory generator	he current com celeration is the r.	manded acco instantaneo	eleration value us acceleration				
	Scaling example: 1 corresponds to 11	If a value of 114,688 i 4,688/65,536 = 1.75	s retrieved using 0 counts/cycle ²	g this comm acceleration	and then this value.				
Restrictions	This command fu Velocity Contour gearing.	inctions when the pro ing. It does not functi	file mode is set on when the pr	to Trapezoi ofile mode is	dal, S-curve, or s set to electronic				
see	GetCommanded	Position, GetComma	ndedVelocity						

GetCommandedPosition

Syntax	GetCommandedPosition axis						
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
Returned data	position	Түре signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	Scaling unity	Units counts		
Packet structure		GetCommand	edPosition				
	0	axis	,	1D h			
	15	First data	word		U		
read	position (high-order pa	art)					
	31 16 Second data word						
read	position (low-order part)						
	15				0		
Description	GetCommandedPosition returns the current commanded position for the specified <i>axis</i> . Commanded position is the instantaneous position value output by the trajectory generator. This command functions in all profile modes.						
Restrictions							
see	GetCommanded	GetCommandedAcceleration, GetCommandedVelocity					

GetCommandedVelocity

Syntax	GetCommandedVelocity axis						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
Returned data	velocity	<i>Type</i> signed integer	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	Units counts/cycle		
Packet structure		GetCommand	edVelocity				
	0	axis	-	1E h			
	15	12 11 8 First data	word		0		
read	velocity (high-order pa	art)					
	31	Second dat	ta word		16		
read	velocity (low-order par	rt)					
	15				0		
Description	GetCommanded specified axis. Co the trajectory gen	Velocity returns the cu ommanded velocity is the erator.	arrent command the instantaneou	led velocity us velocity v	value for the alue output by		
	Scaling example: If a value of $-1,234,567$ is retrieved using this command (FFEDh in high word, 2979h in low word) then this corresponds to $-1,234,567/65,536 = -18.8380$ counts/cycle velocity value.						
	This command fu	unctions in all profile r	nodes.				
Restrictions		-					
see	GetCommanded	Acceleration, GetCo	mmandedPosit	ion			

1Eh

GetCurrentMotorCommand

Syntax	GetCurrentMotorCommand axis					
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
Returned data	motor output command	<i>Type</i> signed 16 bits	<i>Range</i> -2 ¹⁵ <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> 100/2 ¹⁵	Units % output	
Packet structure		GetCurrentMot	orCommand			
	0	axis	7	3A h		
	15	First data	word		0	
read	motor output comman	od			0	
Description	GetCurrentMotorCommand returns the current motor output command for the specified <i>axis</i> . In closed-loop mode, this is the output of the servo filter; in open-loop mode it is the contents of the motor output command register. Scaling example: To convert the retrieved value to units of % of full scale motor output multiply by 100/32,768. For example if the value -123 is retrieved by the GetCurrentMotorCommand, this represents -123*100/32,768 or3754 % of full scale output.					
Restrictions	This command is	not available on the M	MC2500 chipset.			
see	Set/GetMotorCommand					

Syntax	GetDerivative axis					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
Returned data	derivative	<i>Туре</i> signed 16 bits	<i>Range</i> -2 ¹⁵ to 2 ¹⁵ -1	Scaling unity	Units counts/cycle	
Packet structure		GetDe	rivative			
	0	axis		9 B h		
	Data					
read	derivative				0	
Description	GetDerivative returns the derivative of the current position error as calculated by the servo filter. The derivative value is defined as the previous position error subtracted from the current position error. See SetDerivativeTime for details on setting the derivative sampling time.					
Restrictions	This value is avai	lable only when the	chipset is in close	ed-loop oper	ration.	
	This command is	s not valid on the M	C2400 and MC25	00.		
see	GetIntegral, Set	/GetDerivativeTime)			

9Bh

GetEventStatus

Syntax	GetEven	tStatus <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
Returned data	see belo	w					
Packet structure	15 15 15 14	0 12 1 12 1 13 12 11	axis 1 10 8	SetEventStatus 31h 8 7 0 Data 1 1 8 7 6 5 4 3 2 1 0			
Description	GetEven The follo	t Status reads owing table sh	the event ows the e	register for the specified axis . ncoding of the data returned by this command.			
	Name		Rit(s)	Description			
	Motion of Wrap-are	complete	0	Set to 1 when motion is completed. SetMotionCompleteMode determines if this bit is based on the trajectory generator position or the encoder position. Set to 1 when the actual (encoder) position			
				minimum or vice versa			
	Breakpoi	int 1	2	Set to 1 when breakpoint 1 is triggered			
	Capture	received	3	Set to 1 when a position capture occurs			
	<u>Motion e</u> In positiv	error ve limit	4 5	Set to 1 when a motion error occurs Set to 1 when the axis enters a positive limit switch condition			
	In negat	ive limit	6	Set to 1 when the axis enters a negative limit switch condition			
	Instructi	on error	7	Set to 1 when instruction error occurs			
	reserved	/	8-10	Not used, may be 0 or 1.			
	Commut	tation error	11	Set to 1 when a commutation error occurs			
	<i>reserved</i>	/ :=== 0	12-13	Not used, may be 0 or 1.			
	гесотисс	πτ∠	14	Not used may be 0 or 1			
Restrictions	All of the	e bits in this s se bits use the	tatus word ResetEv	d are set by the chipset and cleared by the host. To entStatus command.			
see	GetActiv	vityStatus, G	etSignalS	Status			

31h

GetHostIOError

Syntax	GetHostIOErr	or					
Arguments	none						
Returned data	Name error code	Instance No error Processor Reset Invalid instruction Invalid axis Invalid parameter Trace running reserved Block out of bounds Trace buffer zero Bad serial checksum Not primary port Invalid negative value Invalid parameter change Invalid move after limit condition Invalid move into limit	<i>Encoding</i> O 1 2 3 4 5 6 7 8 9 Ah Bh Ch Dh Eh				
Packet structure		GetHostlOError					
	15 read	0 8 7 Data	A5h 0 4 3 0				
Description	GetHostIOE both the err command is indicates the	GetHostIOError returns the code for the last Host I/O error, then resets to 0 both the <i>error</i> and the Host I/O bit in the Status-Read word. Generally this command is issued only after the Host I/O error bit in the Status-read word indicates there was an I/O error.					
Restrictions							

GetEventStatus

GetIntegral (Servo products only)

Syntax	GetIntegral axis							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
Returned data	integral	<i>Type</i> signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ⁸	<i>Units</i> count*cycles			
Packet structure	0 15	GetInte axis 12 11 8 First date	egral	9A h	0			
read	Integrated position er	ror (high-order part)	a word					
	31	Socond da	ata word		16			
read	Integrated position er	ror (low-order part)						
	15	· · · ·			0			
Description	GetIntegral return specified <i>axis</i> . Get changes in the ax	ns the current integral etIntegral can be used is loading can be refle	ted position error l to monitor loa cted in the value	or of the serv ding on the a e of the integ	vo filter for the axis, because gration limit.			
	Scaling example:							
	If a constant position error of 100 counts is present for 256 cycles than the total accumulated integral value will be 100 (100*256/256). Alternatively a returned value of 1,000 indicates a total stored value of 256,000 count*cycles (1,000*256).							
Restrictions	The integrated period mode (SetMotor	osition error is availab M ode command).	le only when the	e chipset is in	n closed-loop			
	This command is	s not valid on the MC2	2400 and MC25	00.				
see	GetDerivative, S	Set/GetIntegrationLin	nit					

GetInterruptAxis

Syntax	GetInterruptAxis					
Arguments	none					
Returned data	Name axisMask	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 1 2 4 8			
Packet structure		GetInterru	ıptAxis			
	45	0	7	E1 h		
	Data					
read			~	axisMask	7	
	15		4	4 3	0	
Description	GetInterruptAxis returns a field which identifies all axes with pending interrupts. Axis numbers are assigned to the low-order four bits of the returned word; bits corresponding to interrupting axes are set to 1. If the host interrupt signal has not been set, the returned word is 0.					
Restrictions						
see	ClearInterrupt, S	Set/GetInterruptMasI	κ			

Syntax	GetPhase	Command <i>axis</i>			
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
Returned data	phase	PhaseA PhaseB PhaseC	0 1 2		
	motor command	<i>Type</i> signed 16 bit	<i>Range</i> -2 ¹⁵ to 2 ¹⁵ -1	<i>Scaling</i> 100/2 ¹⁵	Units % output
Packet structure		GetPł	naseCommand		
	15	12 11 axis	8 7	EAh	0
		Fir	st data word		
	write	0		phase	0
		Sec	ond data word	ů -	
	read motor comma	and			
	15				U
Description	GetPhase phase A, B the motor	Command returns the v o, or C of the specified a after commutation.	value of the current xis. These are the p	motor outpu hase values c	t command for lirectly output to
	Scaling exa	imple:			
	If a value of correspond	of -4,489 is retrieved (El ds to -4,489*100/32,768	E77h) for a given as $3 = -13.7$ % of full-s	xis and phase scale output.	then this
Restrictions					
see	InitializePh	nase, Set/GetNumberF	hases		

GetPositionError

Syntax	GetPositionError axis						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3				
Returned data	position error	<i>Type</i> signed 32 bit	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> unity	<i>Units</i> counts steps		
Packet structure		GetPositio	onError				
	0	axis		99 h			
	15 12 11 8 7 0 First data word						
read	position error (high-or	der part)					
	31		16				
read	position error (low-order part)						
	15				0		
Description	GetPositionError returns the current position error of the specified <i>axis</i> . The error is the difference between the actual position (encoder position) and the commanded position (instantaneous output of the trajectory generator). Refer to the User's Guide for more information on this command when it is used with the stepping motor chipsets.						
Restrictions							
see	Set/GetPosition	, Set/GetPositionErro	orLimit				

99h

GetSignalStatus

Syntax	GetSigna	lStatus <i>axis</i>				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	Encoding O 1 2 3			
Returned data	status	Description Encoder A Encoder B Encoder Index Encoder Home Positive limit Negative limit AxisIn Hall A Hall B Hall C AxisOut reserved	<i>Bit Number</i> O 1 2 3 4 5 6 7 8 9 10 10 11-15			
Packet structure	15 read	G 0 axis 12 11 11 10 9	ietSignalStatus A4h 0 8 7 0 Data 0 8 7 6 5 4 3 2 1 0			
Description	15 11 10 9 8 7 6 5 4 3 2 1 0 GetSignalStatus returns the contents of the signal status register for the specified <i>axis</i> . The signal status register contains the current value of the various hardware signals connected to each axis of the chipset. The value read is combined with the signal sense register (SetSignalSense command) and then returned to the user. For each bit in the Signal Sense register that is set to 1 the corresponding bit in the GetSignalStatus command will be inverted, so that a low signal will be read as 1 and a high signal will be read as a 0. Conversely for each bit in the signal sense register that is set to 0 the corresponding bit in the GetSignalStatus command is not inverted, so that a low signal will be read as 0 and a high signal will be read as a 1. All of the bits in the GetSignalStatus command are inputs except for AxisOut. The value read for this bit is equal to the current value output by the axis out mechanism. See SetAxisOutSource command for more details.					
Restrictions						
see	GetAct	ivityStatus, GetEvent	tStatus			

GetTime

Syntax	GetTime				
Arguments	none				
Returned data	Name current chipset time	<i>Type</i> unsigned 32 bit	<i>Range</i> O <i>to</i> 2 ³² -1	<i>Scaling</i> unity	<i>Units</i> cycles
Packet structure		GetT	ime		
		0		3E h	
	15	8	7		0
		First dat	a word		
read	current chipset time (h	nigh-order part)			
	31	× i ć			16
		Second da	ata word		
read	current chipset time (le	ow-order part)			
	15				0
Description	Returns the numl initialized or reset	per of cycles that have	e occurred since	e the processo	or was last

Restrictions

see

GetTraceCount

Syntax	GetTraceCount						
Arguments	none						
Returned data	<i>Value</i> trace count	<i>Type</i> unsigned 32 bit	<i>Range</i> O <i>to</i> 2 ³² -1	<i>Scaling</i> unity	Units samples		
Packet structure		GetTrac	eCount				
		0		BB h			
	15	8 First dat	7		0		
road	FIISE data Word						
Teau	31	er part)			16		
	01	Second da	ata word		10		
read	trace count (low-orde	r part)					
	15				0		
Description	GetTraceCount returns the number of points (variable values) stored in the trace buffer since the beginning of the trace.						
Restrictions							
see	ReadBuffer, Set	/GetTraceStart, Set	/GetTraceStop				

BBh

GetTraceStatus

Syntax	GetTr	GetTraceStatus				
Arguments	none					
Returned data	Name mask	<i>Bit</i> 0 1 2	<i>Instance</i> Mode Activity Data wrap	Description Set to 0 when trace is in one-time mode, 1 when in rolling mode. Set to 1 when trace is active (currently tracing), 0 if trace not active Set to 1 when trace has wrapped, 0 if it has not wrapped. If 0, the buffer has not yet been filled and all recorded data are intact. If 1, the trace has wrapped to the beginning of the buffer; any previous data may have been overwritten if not explicitly retrieved by the host using the ReadBuffer command while the trace is active.		
Packet structure Description	read 15 15 Get	TraceStat	0 us returns the	GetTraceStatus 8 7 First data word 0 3 2 1 0 0 1 0 2 1 0 2 1 0		

Restrictions

Set/GetTraceStart, Set/GetTraceMode see

GetVersion

Syntax GetVersion

Arguments none

Returned data	Product in	Encoding	
	product family	Navigator	2
	motor type	Servo	1
		Brushless	3
		Microstepping	4
		Pulse & Direction	5
	axes supported		1, 2, <i>or</i> 4
	special attributes		0 <i>to</i> 15
	customization code	none	0
		other	1 <i>to</i> 255
	major s/w version		0 <i>to</i> 15
	minor s/w version		0 <i>to</i> 15

Packet structure

re				GetVe	ersion			
		0				8F h		
	15			8	7			0
				First da	ta word			
read		product family	motor	type	number of axes	;	special attributes	
	15	12	11	8	7	4 3		0
				Second of	lata word			
read		customizat	ion code		major s/w versio	n	minor s/w version	
	15			8	7	4 3		0

Description GetVersion returns product information encoded as shown above.

Restrictions

see

InitializePhase (MC2300 and MC2800 only)

Syntax	InitializePhase axis						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3				
Packet structure	InitializePhase						
	15	0 12	axis	8	7	7A h	0
Description	InitializePhase initializes the phase angle for the specified axis using the mode (Hall-based or Algorithmic) specified by the SetPhaseInitializationMode command.						
Restrictions	Warning: If the phase initialization mode has been set to algorithmic then after this command is sent the motor can move suddenly in an uncontrolled manner.						
see	GetPhaseCommand, Set/GetNumberPhases						

MultiUpdate

Syntax	MultiUpo	date <i>mask</i>				
Arguments	Name mask	<i>Instance</i> None Axis1mask Axis2mask Axis3mask Axis4mask	<i>Encoding</i> O 1 2 4 8			
Packet structure			MultiUpdate			
	15	0	8 7	51	Bh	
	10		Data			0
write			0		mask	
	15			4	3	U
Description	MultiUpdate causes an Update to occur on all axes whose corresponding bit is set to 1 in the mask argument. After this command is executed, and for those axes which are selected using the mask, all buffered data parameters are copied into the corresponding run-time registers.					ng bit is set hose axes pied into the
	The following instruction is buffered: ClearPositionError. The following trajectory parameters are buffered: Acceleration, Deceleration, GearRatio, Jerk, Position, ProfileMode, StartVelocity, StopMode, and Velocity. The following PID filter parameters are buffered: DerivativeTime, IntegrationLimit Kaff, Kd, Ki, Kp, and Kvff.					
	The following Motor Command parameter is buffered: MotorCommand					nd
Restrictions						

see

Update

NoOperation

Syntax	NoOperation	
Arguments	none	
Packet structure	NoOpera	tion
	15 8	7 0
Description	The NoOperation command has no afference operation to verify communications with	ect on the chipset. It is useful as a "null" h the Motion Processor.
Restrictions		

see
ReadAnalog

Syntax	Re	ReadAnalog <i>portID</i>								
Arguments	Nai poi	ne rtID	<i>Type</i> unsigned 1	6 bit	<i>Range</i> O to 7	Scaling unity	Units -			
Returned data	val	lue	e unsigned 16 bit 0 to 2^{16} -1 1/ 2^{16}							
Packet structure	•	ReadAnalog								
			0			EFh				
	15		8 First data	7 word		0				
١	write		0		portID					
	15	15 Second data word								
	read val	ue								
	15						0			
Description	Re	adAnalog retu	rns a 16-bit	value repr	esenting the vo	ltage (read l	oy an on-chip			

ReadAnalog returns a 16-bit value representing the voltage (read by an on-chip 10 bit A/D) presented to the specified analog input. See User's Guide for more information on analog input and scaling. The value returned is the result of shifting the 10-bit value 6 bits left.

Restrictions

see

ReadBuffer

Syntax		ReadBuffer <i>bufferID</i>								
Arguments		Name bufferID	<i>Type</i> unsigned 16 bit	<i>Range</i> O to 31	Scaling unity	Units -				
Returned data		value	signed 32 bit	-2 ³¹ to 2 ³¹ -1	unity	-				
Packet structure		ReadBuffer								
			0		C9 h					
		15	8 First data	7 word		0				
	write		0		bufferID					
		15	15 4 3							
	read	buffer contents (high-	order part)							
		31	Third data	a word		16				
	read	buffer contents (low-order part)								
		15	• •			0				
Description		¹⁵ ReadBuffer returns the 32-bit contents of the current location in the specified buffer. The current location is determined by adding the base address of the buffer (set by SetBufferStart), to the buffer's Read Index (set by SetBufferReadIndex). After the contents have been read, the Read Index is incremented by 1; if the result is equal to the buffer length (set by SetBufferLength), the Index is reset to 0.								

Some commands automatically change the read index such as at the completion of a trace when in rolling mode. Refer to Section 6.4 of the User's Guide for details.

Restrictions

see Set/GetBufferReadIndex, WriteBuffer **C9**h

Syntax		ReadIO address								
Arguments		Name address	<i>Type</i> unsigned	8 bit	<i>Range</i> O to 255	Scaling unity	Units -			
Returned data		value	unsigned	16 bit	0 <i>to</i> 2 ¹⁶ -1	unity	-			
Packet structure		-		Read	10					
		0	10 11	axis	_	83h				
		15	12 11	8 First data	word		U			
Ň	write		0	Thot date	address					
		15	•	8	7		0			
				Second da	ta word					
I	read	data								
		15					U			
Description		ReadIO reads one 16-bit word of data from the device whose address is calculated by adding 1000h to <i>address</i> . (<i>address</i> is an offset from the base address, 1000h, of the MC2000's memory-mapped I/O space.) The format and interpretation of the 16-bit data word are dependent on the user-defined device being addressed. User-defined I/O can be used to implement a								
	number of features including additional parallel I/O, flash memory for non-volatile configuration information storage, or display devices such as LED arrays.									
Restrictions										
see		WriteIO								

ΠΕϿΕΙ	R	es	et
-------	---	----	----

Jerk

Kaff

Kd

Ki

Kout

Kp Kvff

LimitMode

MotionCompleteMode

Syntax	Reset			
Arguments	none			
Packet structure			Reset	
	0		39 h	
	15		8 7	0
Description	Reset restores the cl default values. These	hipset to its in e default value	itial condition, setting all s are shown in the follow	chipset variables to their ring table:
	Acceleration	0	MotorBias	0
	ActualPosition	0	MotorCommand	0
	AutoStopMode	0	MotorLimit	32767
	AxisMode	1	MotorMode	1
	AxisOutSource	0	NumberPhases	see note 1
	Breakpoint 1	0	OutputMode	see note 2
	Breakpoint 2	0	PhaseAngle	65535
	BreakpointValue 1	0	PhaseCorrectionMode	1
	BreakpointValue 2	0	PhaseCounts	0
	BufferLength	0	PhaseInitializeMode	0
	BufferReadIndex	0	PhaseInitializeTime	0
	BufferStart	200h	PhaseOffset	65535
	BufferWriteIndex	0	PhasePrescale	0
	CaptureSource	0	Position	0
	CommutationMode	0	PositionErrorLimit	32767
	Deceleration	0	ProfileMode	0
	DerivativeTime	1	SampleTime	see note 3
	EncoderModulus	0	SettleTime	0
	EncoderSource	0	SettleWindow	0
	GearMaster	0	SignalSense	0
	GearRatio	0	Stop	0
	IntegrationLimit	0	TraceMode	0
	InterruptMask	0	TracePeriod	1

TraceStart

TraceStop

Velocity

TraceVariable 1

TraceVariable 2

TraceVariable 3

TraceVariable 4

TrackingWindow

0

0

0

0

0

0

0

0

0

0

0

0

0

0

1

0

65535

Notes:

- 1. The reset value for the number of phases is dependent on the Motion Processor series, as follows:
 - MC2100
 - 1 MC2300 3
 - MC2400 2
- 2. The reset value for the output mode is dependent on the Motion Processor series, as follows:
 - MC2100 1
 - MC2300 2
 - MC2400 1
- 3. The reset value for SampleTime depends on the number of axes and the motion processor series, as follows:
 - MC2100 102 x number of axes MC2300 154 x number of axes
 - MC2400 154 x number of axes

All axes supported by the motion processor are enabled at reset.

Profile, servo filter, and other axis-specific parameters are reset on all axes. External-memory buffer parameters are reset for all buffers. BufferStart is reset to (200h), the lowest user-accessible address.

Axis-specific conditions are reset on all axes. External-memory buffer conditions are reset on all 16 memory buffers.

Restrictions For the MC2400/MC2500: AutoStopMode Off EncoderSource None

see

ResetEventStatus

Syntax	ResetEve	ResetEventStatus axis mask							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3						
	mask	Motion complete Wrap-around Breakpoint 1 Capture received Motion error In positive limit In negative limit Instruction error Commutation error Breakpoint 2	0001h 0002h 0004h 0008h 0010h 0020h 0040h 0080h 0800h 4000h						
Packet structure		ResetEventStatus							
	15	12 11	8 7 Data	340	0				
write	0	0 0 0 0	0	mask					
Description	ResetEve Status Re Event Sta	entStatus clears (sets to 0 gister that has a value of atus register bits (bits whi), for the specified 0 in the mask sent ch have a mask va	d axis , each bit in the with this command use of 1) are unaffect	he Event 1. All other cted.				
Restrictions									
see	GetEven	tStatus							

Syntax	SetAcceleratior GetAcceleratior	SetAcceleration <i>axis acceleration</i> GetAcceleration <i>axis</i>									
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3								
	acceleration	<i>Type</i> unsigned 32 bit	<i>Range</i> O <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	Units counts/cycle ²						
Packet structure		SetAccel	eration								
	0	axis		90 h							
	15	12 11 8 First dat	7 a word		0						
write	acceleration (high-or	der part)									
	31	Second da	ata word		16						
write	e acceleration (low-ord	er part)									
	15				0						
	0	GetAccel	eration	1Ch							
	15	12 11 8	7	4611	0						
read	acceleration (high-or	First data word									
	31				16						
read	Second data word										
	15				0						
Description	SetAcceleration axis. This comm curve profiling n	loads the maximum a nand is used with the ' nodes.	cceleration buff Frapezoidal, Ve	fer register f locity Conto	or the specified ouring, and S-						
	GetAcceleration SetAcceleration	reads the maximum a command.	cceleration but	ter register s	set by the previous						
	Scaling example: To load a value of 1.750 counts/cycle ² multiply by 65,536 (giving 114,688) and load the resultant number as a 32 bit number, giving 0001 in the high word and C000h in the low word. Values returned by GetAcceleration must correspondingly be divided by 65,536 to convert to units of counts/cycle ² .										
Restrictions	SetAcceleration profile.	may not be issued wh	ile an axis is in	motion with	the S-curve						
	SetAcceleration	is not valid in Electro	nic Gearing pro	ofile mode.							
	SetAcceleration not take effect us	is a buffered comman ntil the next Update o	id. The value so r MultiUpdate i	et using this nstruction.	command will						
see	Set/GetDecelera MultiUpdate, U	ation, Set/GetJerk, S pdate	Set/GetPositior	n, Set/GetV	elocity,						

Syntax	SetActualPositi GetActualPosit	on <i>axis position</i> ion <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	Encoding O 1 2 3				
	position	Type signed 32 bits	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	Scaling unity	<i>Units</i> counts steps		
Packet structure		SetAc	tualPosition				
	0	axis		4D h			
	15	12 11 Firs	8 7 t data word		0		
writ	e position (high-order	part)					
	31	Seco	nd data word		16		
writ	e position (low-order p	part)					
	15				0		
		GetAd	tualPosition	071			
	15	12 11 axis	8 7	37n	0		
	d	Firs	t data word				
rea	a position (nign-order) 31	part)			16		
Second data word							
rea	a position (low-order p	art)			0		
Description	SetActualPosition specified axis. A loaded value min when the new axis is also modified update instruction position from w set a known refer Note: On the M SetActualPosition Value will be the cycle (as determined	on loads the actual At the same time, the nus the current actual is position is estable by this amount so on is issued. In effect hich subsequent po- erence position after IC2400 and MC250 on takes effect imm on reads the conten- result of the last en- ined by Set/GetSa	position register ne current comm al position error dished. The dest that no trajectory ect, this instruction sitions can be can r a homing proce 00 series, the post nediately, it is not not sof the encoder ncoder input, wh mpleTime).	(encoder posi anded positio . This preven ination positio y motion will o on establishes loculated. It is edure. ition error is z buffered. r's actual posi ich will be acc	tion) for the n is replaced by the its a servo "bump" on (see SetPosition) occur when the a new reference commonly used to zeroed.		
Restrictions							
<i>See</i>	GetPositionErro AdjustActualPos	r; GetActualVeloci sition	ty, Set/GetActua	IPositionUnit	S,		

SetActualPositionUnits (MC2400 and MC 2500 only) GetActualPositionUnits (MC2400 and MC 2500 only)

Syntax	SetActualPositionUnits <i>axis mode</i> GetActualPositionUnits <i>axis</i>									
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3							
	mode	Counts Steps	0 1							
Packet structure			SetAct	ualPositionUn	its					
		0	axis		BEh					
	15	12 11		8 7 Data		0				
write			0	Dulu		mode				
	15					1 0				
	GetActualPositionUnits									
		0	axis		BFh					
	15	12 11		8 7 Data		0				
read				Data		mode				
	15					1 0				
Description	SetActualPositionUnits determines the units used by the Set/GetActualPosition, AdjustActualPosition and GetCaptureValue for the specified <i>axis</i> . When set to <i>Counts</i> position units are in encoder counts. When set to <i>Steps</i> GetActualPosition position units are in steps.									
Restrictions	This com	imand is only	v available or	n the MC24	00 and MC2500 seri	ies.				
see	Set/Get/ GetCapt	ActualPositic ureValue	on, Set/Getl	EncoderTo	StepRatio, AdjustA	ctualPosition,				

Syntax	SetAuto GetAuto	SetAutoStopMode <i>axis mode</i> GetAutoStopMode <i>axis</i>							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3						
	mode	Disable Enable	0 1						
Packet structure			SetAuto	oStopMode					
	15	0	axis		D2 h				
	15	12 1	י ו	8 / Data		U			
write			0	Jala		mode			
	15					0			
			GetAuto	oStopMode					
	15	0 12 1	axis	0 7	D3 h				
	15	12 1	' I	° ′		0			
read				Jala		mode			
	15				1	0			
Description	SetAutos error occ into oper (SetAutos GetAutos	StopMode de urs. When au i-loop mode StopMode D StopMode re	termines the b ito stop is enal when a motion i sable), the axi turns the curre	ehavior of the sp bled (SetAutoSto n error occurs. W is is not affected ent state of the Au	ecified axis wh pMode Enable hen Auto-Stop by a motion er ato-Stop mode	then a motion (a), the axis goes (b) is disabled (ror.			
Restrictions									

see

GetEventStatus

Syntax	SetAxisl GetAxisl	etAxisMode <i>axis mode</i> SetAxisMode <i>axis</i>								
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3							
	mode	off on	0 1							
Packet structure				SetAxis	lode					
		0	axis			87 h				
	15	12 11		8 7			0			
		Data								
writ	e			0			mode			
	15					1	0			
		^		GetAxis	lode	00				
	15	0 12 11	axis	0 7		88h	0			
	15	12 11		o / Dete			U			
roo	4			Data			modo			
Iea	15					1	noue			
Description	SetAxisM not respo GetAxisM	lode enables and to profile lode returns	(On) or d or other the currer	isables motion nt status	(Off) the speci commands. s of the specifi	fied axis . A d ed axis.	isabled ax	as will		
Restrictions	Disabled encoder	axes do not j feedback ever	provide er n though 1	ncoder f no profi	eedback. If it ling or servo o	is desired that control is to b	it an axis j be used, th	provide nat axis		

must be left enabled.

see

87h **88**h

Syntax	SetAxisOutSource <i>axis source,</i> GetAxisOutSource <i>axis</i>					s bit regis	ster		
Arguments	Name Ins axis Ax Ax Ax Ax Ax Ax Ax Ax Ax Ax Ax Ax			:e	Encoding O 1 2 3				
					0 1 2 3				
	bit	ee below			0 to 15				
	register (no Ev Ac Sig		one) rentStatus ctivityStatus gnalStatus			0 1 2 3			
Packet structure	SetAxisOutSource								
	0			axis				ED h	
	15	12	11		8 П	7 ata			
write	0			register		b b	it		sourceAxi
	15	12	11	•	8	7	4	3	
	GetAxisOutSource								
	0			axis				EEh	

 0
 axis
 EEh

 15
 12
 11
 8
 7
 0

 Data

 register
 bit
 sourceAxis

 15
 12
 11
 8
 7
 4
 3
 0

EDh

EEh

Description SetAxisOutSource maps the specified *bit* of the specified status *register* of *axisn* to the AxisOut pin for the specified *axis*. The state of the AxisOut pin will thereafter track the state of *bit*. If *register* is absent (encoding of 0), *bit* is ignored, and the specified AxisOut pin is, in effect, turned off (inactive).

GetAxisOutSource reads the mapping of the AxisOut pin of axis.

encoding of "bit"	register = event status	register = activity status	register = signal
			status
0	Motion Complete	Phasing Initialized	Encoder A
1	Wrap-around	At maximum velocity	Encoder B
2	Breakpoint 1	Tracking	Encoder index
3	Position capture		Home
4	Motion error		Positive limit
5	In positive limit		Negative limit
6	In negative limit		AxisIn
7	Instruction error	Axis settled	Hall sensor 1
8		Motor on/off	Hall sensor 2
9		Position capture	Hall sensor 3
0Ah		In motion	
0Bh	Commutation error	In positive limit	
0Ch		In negative limit	
0Dh			
0Eh	Breakpoint 2		
0Fh			

The table below shows the corresponding value for combinations of *bit* and *register*.

Restrictions

see

SetSignalSense

Syntax	SetBreakpoir GetBreakpoir	SetBreakpoint <i>axis breakpoint sourceAxis action trigger</i> GetBreakpoint <i>axis breakpoint</i>				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	breakpoint	Breakpoint1 Breakpoint2	0 1			
	sourceAxis	Axis1 Axis2 Axis3 Axis4	0 1 2 3			
	action	(none) Update AbruptStop SmoothStop MotorOff	0 1 2 3 4			
	trigger	(none) PositiveCommandedPosition NegativeCommandedPosition PositiveActualPosition NegativeActualPosition CommandedPositionCrossed ActualPositionCrossed Time EventStatus ActivityStatus Signal	0 1 2 3 4 5 6 7 8 9 Ah			



Navigator Motion Processor Programmer's Reference

D4h **D5**h **Description** SetBreakpoint establishes a breakpoint for the specified *axis* to be triggered by a condition or event on *sourceAxis*, which may be the same as or different from *axis*. Up to two concurrent breakpoints can be set for each axis.

The six **Position** breakpoints and the **Time** breakpoint are *threshold-triggered*; the breakpoint occurs when the indicated value reaches or crosses a threshold. The **Status** breakpoints are *level-triggered*; the breakpoint occurs when a specific bit or combination of bits in the indicated status register changes state. Thresholds and bit specifications are both set by the **SetBreakpointValue** instruction.

action determines what the Navigator does when the breakpoint occurs, as follows:

	Action	Resultant command sequence			
	none	no action			
	Update	Update axis			
	AbruptStop	SetStop <i>axis</i> , AbruptStop Update <i>axis</i>			
	SmoothStop	SetStop <i>axis</i> , SmoothStop Update <i>axis</i>			
	MotorOff	SetMotorMode axis, Off			
	axis is the axis for	which the breakpoint has been set.			
	GetBreakpoint ret (1 or 2) of the ind	Breakpoint returns the condition, axis, and action for the specified breakpoint r 2) of the indicated axis.			
	Two completely separate breakpoints are supported, each of which may breakpoint type and comparison value. The <i>breakpoint</i> field specifies w breakpoint the SetBreakpoint and GetBreakpoint commands will addre				
Restrictions	Before setting a new breakpoint condition (SetBreakpoint command) ALWAYS load the comparison value first (SetBreakpointValue command). This is because as soon as the breakpoint condition is set the chipset will start using the breakpoint value register, and if it is not yet defined the breakpoint will not behave as expected.				
see	Set/GetBreakpoir	ntValue			

Syntax	SetBreakpoi GetBreakpoi	SetBreakpointValue <i>axis breakpoint value</i> GetBreakpointValue <i>axis breakpoint</i>						
Arguments	Name	Instance	Encoding					
•	axis	Axis1	0					
		Axis2	1					
		Axis3	2					
		Axis4	3					
	breakpoint	Breakpoint1	0					
		Breakpoint2	1					
			Type	Range	Units			
	value	PositiveCommandedPosition	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		NegativeCommandedPosition	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		PositiveActualPosition	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		NegativeActualPosition	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		CommandedPositionCrossed	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		ActualPositionCrossed	signed 32 bit	-2 ³¹ to 2 ³¹ -1	counts			
		Time	unsigned 32 bit	0 <i>to</i> 2 ³² -1	cycles			
		EventStatus	2 word mask*	-	-			
		ActivityStatus	2 word mask*	-	-			
		SignalStatus	2 word mask*	-	-			

D6h

D7h

* see description section below for more details on mask format



Navigator Motion Processor Programmer's Reference

Description	SetBreakpointValue sets the breakpoint comparison value for the specified axis . For the position and time breakpoints this is a threshold comparison value.				
	For level-triggered breakpoints, the high-order part of <i>value</i> is the selection mask, and the low-order word is the sense mask. For each selection bit that is set to 1, the corresponding bit of the specified status register is conditioned to cause a breakpoint when it changes state. The sense-mask bit determines which state causes the break. It is 1, the corresponding status-register bit will cause a break when it is set to 1. If it is 0, the status-register bit will cause a break when it is set to 0.				
	For example assume it is desired that the breakpoint type will be set to "EventStatus" and that a breakpoint should be recognized whenever the motion complete bit (bit 0 of event status register) is set to 1, and the commutation error bit (bit 11 of event status register) is set to 0. In this situation the high and low words for <i>value</i> would be high word: 0x801 (hex) and low word: 1.				
	GetBreakpointValue returns the current breakpoint value for the specified breakpoint.				
	Two completely separate breakpoints are supported, each of which may have its own breakpoint type and comparison value. The <i>breakpoint</i> field specifies which breakpoint the SetBreakpointValue and GetBreakpointValue commands will address.				
Restrictions	Before setting a new breakpoint condition (SetBreakpoint command) ALWAYS load the comparison value first (SetBreakpointValue command). This is because as soon as the breakpoint condition is set the chipset will start using the breakpoint value register, and if it is not yet defined the breakpoint will not behave as expected.				
see	Set/GetBreakpoint				

Syntax	SetBufferFunction axis function bufferID GetBufferFunction axis function				
Arguments	Name	Instance	Encoding		
·	axis	Axis1	0		
		Axis2	1		
		Axis3	2		
		Axis4	3		
	function	Position	0		
		Velocity	1		
		Acceleration	2		
		Jerk	3		
		Time	4		
	Name	Туре	Range	Scaling	Units
	bufferID	signed 16 bits	-1 <i>to</i> 31	unity	-



Restrictions

Set/GetProfileMode see

CAh

CBh

Syntax		SetBufferLength <i>bufferID length</i> GetBufferLength <i>bufferID</i>					
Arguments		Name bufferID	<i>Type</i> unsigned 16 bits	<i>Range</i> O to 31	<i>Scaling</i> unity	Units -	
		length	unsigned 32 bits	1 to 2 ³⁰ -1	unity	-	
Packet structu	ure		SetBuffe	erLength			
		45	0	7	C2 h		
		15	, First da	ita word		U	
	write		0		buff	ferID	
		15	Second of	data word	4	0	
	write	e length (high-order pa	art)				
		31	Third da	ata word		16	
	write	e length (low-order par	rt)				
		15				0	
			GetBuffe	erLength			
		15	0 8	7	C3h	0	
			First da	ta word			
	write	15	0		buff	ferID	
		13	Second of	data word	+ 0		
	read	I length (high-order pa	art)			16	
		51	Third da	ata word		10	
	read	l length (low-order par	rt)				
		15				0	
Description		SetBufferLength memory block is	n sets the length, in nu dentified by <i>bufferID</i> .	umber of 32-bit	elements, of	the buffer in the	
		Note: SetBuffer	Length resets the bu	iffers read and	write index	xes to 0.	
		GetBufferl ength returns the length of the specified buffer					
		0	0	1			
Restrictions		If the specified l SetBufferLength exceeded.	If the specified length extends beyond the end of addressable memory, SetBufferLength is not executed, and returns host-I/O error code 7, <i>buffer bound exceeded</i> .				
		Note: Setting the cause the chip	he buffer length bey set to unexpectedly	rond the end of reset during o	f physical m peration.	nemory could	
see		Set/GetBufferR	eadIndex; Set/GetB	ufferStart; Set,	/GetBufferW	/riteIndex	

SetBufferReadIndex GetBufferReadIndex

Syntax		SetBufferReadIndex <i>bufferID index</i> GetBufferReadIndex <i>bufferID</i>					
Arguments		Name bufferID	<i>Туре</i> unsigned 16 bits	<i>Range</i> O <i>to</i> 31	Scaling unity	Units -	
		index	unsigned 32 bits	0 to buffer length-1	unity	double words (32 bit)	
Packet structur	e		SetBufferR	eadIndex			
			0		C6 h		
		15	8 First date	7		0	
	write			a word	bufforl	0	
	white	15	0	4	3	0	
			Second da	ata word			
	write	index (high-order part))				
		31				16	
			Third dat	a word			
	write	index (low-order part)				0	
		15	0-4DffrD			U	
		15	8	7	U III	0	
			First data	a word			
	write	15	0		bufferl	D	
		15	Second da	ata word	. 3	U	
	read	index (high-order part))				
		31				16	
		index (low order rout)	I hird dat	a word			
	reau					0	
						Ū	
Description		SetBufferReadIn the read index is a	dex sets the address of set to an address beyo	of the Read Inde	ex for the spe f the buffer, t	cified buffer. If he command	
		will not be execut	ted and will return an	error.			
		GetBufferReadIn	dex returns the curre	nt Read Index f	or the specifi	ed buffer.	
					*		
Restrictions							

see

Set/GetBufferLength, Set/GetBufferStart, Set/GetBufferWriteIndex

Syntax		SetBufferStart <i>bufferID address</i> GetBufferStart <i>bufferID</i>					
Arguments		Name	Туре	Range	Scaling	Units	
-		bufferID	unsigned 16 bit	0 <i>to</i> 31	unity	-	
		address	unsigned 32 bit	2 ⁹ <i>to</i> 2 ³¹ -1	unity	double words (32 bit)	
Packet structure)		SetBuf	ferStart			
			0		C0 h		
		15	8 First da	7 ata word		0	
,	write	45	0		buffer	rID	
		15	Second of	data word	4 3	U	
,	write	address (high-order p	part)			10	
		31	Third da	ata word		10	
	write	address (low-order pa	art)			0	
		10	GetBuf	ferStart		Ū	
			0	_	C1 h		
		15	8 First da	7 ata word		0	
,	write	45	0		buffer	rID	
		15	Second of	data word	4 3	0	
	read	address (high-order p	oart)			16	
		51	Third da	ata word		10	
	read	address (low-order pa	art)			0	
		10				Ū	
Description		SetBufferStart se address must be	ets the starting addres e 200h or greater.	ss for the specifi	ed buffer. T	he buffer start	
		Note: SetBuffer	Start resets the buff	fers read and w	rite indexes	to 0.	
		GetBufferStart ro	eturns the starting ad	dress for the spe	ecified buffer.		
			0	*			
Restrictions		If the specified length extends beyond the end of addressable memory, SetBufferStart is not executed, and returns host-I/O error code 7, <i>buffer bound exceeded</i> .					
		Note: Setting the cause the chip s	ne buffer start beyon set to unexpectedly	nd the end of p reset during of	hysical mem peration.	nory could	
see		Set/GetBufferLe	ength, Set/GetRead	Index, Set/GetE	BufferWriteIr	ndex	

SetBufferWriteIndex GetBufferWriteIndex

Syntax		SetBufferWriteIndex <i>bufferID index</i> GetBufferWriteIndex <i>bufferID</i>					
Arguments		Name bufferID	<i>Түре</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 31	Scaling unity	Units -	
		index	unsigned 32 bit	0 <i>to</i> buffer length-1	unity	long words (32 bits)	
Packet structur	e		SetBufferW	riteIndex			
			0		C4 h		
		15	8 7 First data	7 word		0	
	write		0	i word	bufferl	D	
		15		4	3	0	
			Second da	ta word			
	write	index (high-order part)				
		31	Third data	a word		16	
	write	index (low-order part)					
		15				0	
			GetBufferW	riteIndex			
		0 C5h					
		15	First data	a word		U	
	write		0		bufferl	D	
		15	Second de	4 to word	3	0	
	read	index (high-order part		la woru			
		31	/			16	
		Third data word					
	read	15				0	
						Ŭ	
Description		SetBufferWriteIn the write index is will not be execut	dex sets the address of set to an address beyo ted and will return an	of the write index and the length o error.	x for the spec f the buffer,	tified buffer. If the command	
		GetBufferWriteIn	idex returns the current	nt write index fo	or the specifie	d buffer.	
Restrictions							

C4h C5h

Set/GetBufferLength, Set/GetBufferReadIndex, Set/GetBufferStart see

Syntax	SetCaptureSource <i>axis source</i> GetCaptureSource <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encodin</i> g O 1 2 3	7			
	source	Index Home	0 1				
Packet structure			:	SetCaptureS	ource		
		0	axi	s		D8 h	
	15	12	11	87 Data			0
write				0			source
	15					1	0
		-		GetCaptureS	ource		
	15	0 12	11 axi	S 8 7		D9 h	0
		12		Data			
read							source
	15					1	0
Description	SetCaptu used to t axis. GetCaptu	ureSource d rigger the hig ureSource r	etermines gh-speed eturns the	which of capture of e capture s	two encoder si the actual axis ignal source fo	gnals, Index position for r the selecte	a or Home, is r the specified ed axis .
Restrictions							
see	GetCapt	ureValue					

SetCommutationMode (MC2300 and MC2800 only) GetCommutationMode (MC2300 and MC2800 only)

Syntax	SetCommutationMode <i>axis mode</i> GetCommutationMode <i>axis</i>					
Arguments	Name Instance axis Axis1 Axis2 Axis3 Axis4 mode Sinusoidal Hall-Based Microstepping	<i>Encoding</i> O 1 2 3 O 1 1 2 2				
Packet structure		SetCommutationMode				
	0	axis	E2 h			
	15 12 11	8 7 Data	0			
write	e	0	mode			
	15		2 1 0			
		GetCommutationMode				
	0	axis	E3h			
	13 12 11	Data	U			
read	d		mode			
	15		2 1 0			
Description	SetCommutationMode set When set to sinusoidal, a calculate the phase angle. outputs to each motor we	ets the phase commutation m as the motor turns, the encod This angle is in turn used to inding.	ode for the specified axis . er input signal is used to generate sinusoidally varying			
	When set to Hall-based to motor windings using a "	When set to Hall-based the hall effect sensor inputs are used to commutate the motor windings using a "six-step" or "trapezoidal" waveform method.				
	When set to microstepping the output of the trajectory generator is used to calculate the phase angle. This angle is in turn used to generate sinusoidally varying outputs to each motor phase.					
	GetCommutationMode returns the current commutation mode.					
	When operating with brushless servo motors either sinusoidal or Hall-based are typically used for motor commutation.					
	Microstepping is sometin motor before phase initia used with step motors or that is required to rotate	nes used with brushless moto lization has occurred. Alterna- with AC induction motors w the motor.	rs to "manually" move the atively, Microstepping can be where frequency synthesis is all			
Restrictions	-					
see	Set/GetCommutationPr Set/GetPhase command	escale, Set/GetCommutatic ds	onCounts,			

Syntax	SetDeceleration <i>axis deceleration</i> GetDeceleration <i>axis</i>						
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4 Type unsigned 32 bits	<i>Encoding</i> 0 1 2 3 <i>Range</i> 0 <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	Units		
	deceleration		0102 -1	1/2	counts/cycle		
Packet structure	0	SetDece axis	leration	91 h			
	15	12 11 8 First dat	7 a word		0		
write	deceleration (high-or	rder part)			16		
	51	Second d	ata word		10		
write	deceleration (low-ord	ler part)			0		
	-	GetDece	leration				
	15	12 11 axis	7	92 h	0		
		First dat	a word				
read	31 deceleration (nign-or	der part)			16		
read	deceleration (low-ord	Second d ler part)	ata word				
	15	- 1 - 7			0		
Description	SetDeceleration axis. This comm has a negative sig	loads the maximum on nand sets the magnitu gn.	deceleration but de of the decele	ffer register f eration regist	for the specified er, which always		
	GetDeceleration	reads the Maximum	Deceleration by	uffer.			
	Scaling example: 114,688) and loa word and C000h correspondingly	To load a value of 1.7 d the resultant number in the low word. Ref be divided by 65,536	750 counts/cyc er as a 32 bit nu trieved number to convert to u	le ² multiply l mber, giving s (GetDecele nits of count	by 65,536 (giving 0001 in the high eration) must cs/cycle ²		
Restrictions	This is a buffered Update or Multil	d command. The new Jpdate instruction is a	v value set will i entered.	not take effe	ct until the next		
	These command profile modes. T	ls are used with the Tr hey are not used with	apezoidal, S-cu the electronic g	rve, and Vel gearing profi	ocity contouring le mode.		
	Note: If deceleration	ation is set to zero, th n) will automatically be	en the value spe e used to set the	ecified for ac e magnitude	celeration of deceleration.		
see	Set/GetAccelera MultiUpdate, U	ation, Set/GetJerk, S pdate	Set/GetPositio	n, Set/GetV	elocity,		

Syntax	SetDerivativeTime <i>axis time</i> GetDerivativeTime <i>axis</i>					
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	time	<i>Type</i> unsigned 16 bits	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	<i>Units</i> cycles	
Packet structure		SetDerivat	iveTime			
	0	axis		9 C h		
	15	12 11 8 Dat:	7 a		0	
write	time	Dat				
	15				0	
	-	GetDerivat	iveTime			
	15	axis	7	9D h	0	
	15	Data	a		U	
read	time					
	15				0	
Description	SetDerivativeTim filter to use in cal GetDerivativeTim	ne sets the sampling ti culating the derivative ne returns the derivati	me, in number e term for the sp ve sampling tin	of servo cycle pecified axis . ne.	es, for the servo	
Restrictions	This command is immediately after This command do	NOT buffered. The the command is sent oes not affect the ove	new sampling ti to the chipset. rall cycle time c	me value will f the chipset,	take effect only the	
see	derivative sampling time. The overall cycle time of the chipset is set using the command SetSampleTime. GetDerivative, GetIntegral, MultiUpdate, Update					

SetDiagnosticPortMode GetDiagnosticPortMode

Syntax	SetDiagnosticPortMode <i>mode</i> GetDiagnosticPortMode							
Arguments	Name mode	<i>Instance</i> Limited Full	<i>Encoding</i> O 1					
Packet structure			SetDi	agnosticPortM	ode			
	45	0		0 7		89h		
	10			₀ / Data			U	
write				0			mode	
	15					1	0	
			GetD	agnosticPortM	ode			
	15	0		8 7		8 A h	0	
	10			Data			U	
read							mode	
	15					1	0	
Description	SetDiagn through t may be ex a T When set GetDiagn	osticPortMc he diagnosti accuted: Il Get instru The SetBuffe to Full, all in costicPortMc	ode determ c port. Wh ctions erReadInde nstructions ode returns	ines the insta en set to Lin ex instruction may be exec the current	ruction set t ni ted , only t n cuted. mode of th	that can be e he following e diagnostic	executed g instruction	ιS
Restrictions								
see	Set/GetS	erialPort						

89h **8A**h

Syntax	SetEncoderModulus <i>axis modulus</i> GetEncoderModulus <i>axis</i>						
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	modulus	<i>Type</i> unsigned 16 bit	<i>Range</i> 1 <i>to</i> 2 ¹⁶ -1	<i>Scaling</i> unity	Units counts		
Packet structure		SetEncoder	Modulus				
	0	axis	7	8Dh			
	15	Data	a		U		
write	modulus						
	15				0		
	0	GetEncoder	Modulus	0 E h			
	15	12 11 8 Data	7 a		0		
read	modulus						
	15				0		
Description	SetEncoderModulus sets the parallel word range for the specified <i>axis</i> when parallel-word feedback is used. <i>Modulus</i> determines the range of the connected device. The value provided should be one-half of the actual <i>modulus</i> of the axis. For example if the parallel-word input is used with a linear potentiometer connected to an external A/D (Analog to Digital converter) which has 12 bits of resolution, than the total range is 4,096 and a value of 2,048 should be loaded with this command. GetEncoderModulus returns the current encoder modulus.						
Restrictions	These commands encoder feedback	s are only used if paral s is used then these co	lel-word feedba mmands are no	ack is used. If ot required.	fincremental		
see	Set/GetEncoder	Source					

Syntax	SetEnco GetEnco	derSource <i>axis</i> derSource <i>axis</i>	source						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3						
	source	Incremental Parallel None	0 1 2						
Packet structure			SetEn	coderSource					
		0	axis		DA h				
	15	12 11		8 7		0			
wite			٥	Data		001///00			
write	15		0		1				
		Co4Emandon Comment							
		0	GetEn	coderSource	DBh				
	15	12 11	UXI5	8 7	DDI	0			
				Data					
read						source			
	15					0			
Description	SetEnco parallel-v chip set o parallel-v connecte position used but GetEnco	derSource sets vord) for the spe expects A and B vord is selected d to the chip set value for each a the unused bits derSource retur	the type o ecified axi quadratur the chipse t's externa xis. Extern must be s rns the coo	f feedback (incre s. When increme e signals to be in t expects user-de l bus to load a 1 hal feedback dev ign extended or de for the curren	emental quadra ental quadratur nput at the I/C efined external 6-bit word con ices with less t 'zeroed'. nt type of feedl	ture encoder or e is selected the) chip. When circuitry taining the current han 16 bits may be pack.			
Restrictions									
see	Set/GetI	EncoderModulu	S						

DAh

SetEncoderToStepRatio (MC2400 and MC2500 only) GetEncoderToStepRatio (MC2400 and MC2500 only)

Syntax		SetEncoderToStepRatio <i>axis counts steps</i> GetEncoderToStepRatio <i>axis</i>						
Arguments		Name axis		n <i>stance</i> Axis1 Axis2 Axis3 Axis4		<i>Encoding</i> 0 1 2 3		
		counts	1 s	ype igned	16 bit	<i>Range</i> -2 ¹⁵ <i>to</i> 2 ¹⁵ -1	Scaling unity	<i>Units</i> encoder counts
		steps	s	igned	16 bit	-2 ¹⁵ to 2 ¹⁵ -1	unity	steps
Packet structur	e				SetEncoder	oStepRatio		
		0			axis	-	DEh	
		15	12	11	8 First dat	7 a word		0
	write	counts						
		31			Second d	ata word		16
	write	steps						
		15						0
		0			GetEncoder	oStepRatio	DEh]
		15	12	11	8	7		0
	read	counts			First dat	a word		
	Teau	31			Second d	ata word		16
	read	steps						
		15						0
Description		 SetEncoderToStepRatio sets the ratio of number of encoder counts to the number of output steps per motor rotation used by the motion processor to convert encoder counts into steps/microsteps. <i>Counts</i> is the number of encoder counts per full rotation of the motor. <i>Steps</i> is the number of steps/microsteps output by the motion processor per full rotation of the motor. Since this command sets a ratio, the parameters do not have to be for a full rotation as long as they correctly represent the encoder count to step ratio. GetEncoderToStepRatio gets the ratio of number of encoder counts to the number of output steps per motor rotation. 						
Restrictions								
see		Set/GetActua	IPos	itionU	nits			

DEh DFh

Syntax	ter <i>axis masterA</i> ter <i>axis</i>	rAxis source		
Arguments	Name	Instance	Encoding	
-	axis	Axis1	0	
		Axis2	1	
		Axis3	2	
		Axis4	3	
	masterAxis	Axis1	0	
		Axis2	1	
		Axis3	2	
		Axis4	3	
	source	Actual	0	
		Commanded	1	



see Set/GetGearRatio

SetGearRatio buffered 14h **GetGearRatio 59**h SetGearRatio slaveAxis ratio Syntax GetGearRatio Arguments Name Encodina Instance Axis1 slaveAxis 0 Axis2 1 2 Axis3 Axis4 3 Type Range Scaling Units -2³¹ to 2³¹-1 $1/2^{16}$ ratio signed 32 bits SlaveCounts/ MasterCounts Packet structure SetGearRatio **14**h 0 slaveAxis 12 11 8 First data word write ratio (high-order part) 16 31 Second data word write ratio (low-order part) 15 ٥ GetGearRatio slaveAxis **59**h 0 12 11 0 8 7 First data word read ratio (high-order part) 16 Second data word read ratio (low-order part) Description SetGearRatio sets the ratio between the master and slave axes for the electronic gearing profile for the current **axis**. Positive ratios cause the slave to move in the same direction as the master, negative ratios in the opposite direction. The specified ratio has a unity scaling of 65,536. GetGearRatio returns the gear ratio set for the specified slave axis. Scaling examples: ratio value resultant ratio -32.768.5 negative slave counts for each positive master count 1,000,000 15.259 positive slave counts for each positive master count 123 .0018 positive slave counts for each positive master count Restrictions This is a buffered command. The new value set will not take effect until the next Update or MultiUpdate instruction is entered.

see Set/GetGearMaster, MultiUpdate, Update

buffered SetIntegrationLimit (Servo products only) GetIntegrationLimit (Servo products only) SetIntegrationLimit axis limit Syntax GetIntegrationLimit axis Arguments Encodina Name Instance Axis1 0 axis Axis2 1 2 Axis3 Axis4 3 Type Range Scaling Units 0 to 2³¹-1 $1/2^{8}$ limit signed 32 bits count*cycles **Packet structure SetIntegrationLimit** 0 axis 95h 12 11 8 7 Λ First data word write *limit* (high-order part) 16 Second data word write *limit* (low-order part) 15 ٥ GetIntegrationLimit 0 96h axis 12 11 8 7 First data word read *limit* (high-order part) 16 Second data word read *limit* (low-order part) Description SetIntegrationLimit loads the integration-limit register of the digital servo filter for the specified axis. GetIntegrationLimit returns the value of the current integration limit. Scaling example: The scaling is the same as for the **GetIntegral** command, namely that (for example) a constant position error of 100 counts which is present for 256 cycles will result in an integral value of 100 (100*256/256), and therefore an IntegrationLimit value of 100 will limit the total accumulated integration error to

95h 96h

Restrictions This is a buffered command. The value set using this command will not take effect until the next **Update** or **MultiUpdate** instruction.

This command is not valid on the MC2400 and MC2500.

25,600 count*cycles.

see GetIntegral, GetDerivative, Set/GetDerivativeTime, MultiUpdate, Update

Syntax	SetInterruptMas GetInterruptMas	sk axis interruptMask sk axis		
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3	
	interruptMask	Motion complete Wrap-around Breakpoint 1 Capture received Motion error In positive limit In negative limit Instruction error Commutation error Breakpoint 2	0001h 0002h 0004h 0008h 0010h 0020h 0040h 0080h 0800h 4000h	
Packet structure		SetInterrupt	Mask	
	0	axis	2F h	
	15	mask		U
write	e 0 0 0			
	14	GetInterrupt	Mask	0
	0	axis	56h	
	15	12 11 8 7 mask		0
read	L L			
	14	11 8 7		0
Description	SetInterruptMask axis will cause a licorresponding Evregister bit goes a interrupts. GetInterruptMask Example: The int "in positive limit" active (set to 1).	 K determines which bits host interrupt. For each vent Status register bit wattive (is set to 1). Interr K returns the current matterrupt mask value 28h wattive value receiption 	in the Event Status regis interrupt mask bit that is zill cause an interrupt wh upt mask bits set to 0 wi tsk for the specified axis. vill generate an interrupt eived" bit of the event sta	ter of the specified s set to 1, the en that status ll not generate when either the atus register goes
Restrictions				
see	ClearInterrupt, G	GetInterruptAxis		

SetJerk GetJerk				b	ouffered	13h 58h
Syntax	SetJerk <i>axi</i> GetJerk <i>axi</i>	s jerk is				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	jerk	<i>Type</i> unsigned 32 bits	<i>Range</i> 0 <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ³²	<i>Units</i> counts/cycle ³	
Packet structure	15 write <i>ierk</i> (high-or	Se axis 12 11 First of der part)	tJerk	13 h	0	
	$\frac{31}{15}$ write $\int \frac{jerk}{15}$ (low-ord	er part)	data word		16 0	
	15	12 11 axis	5 7	58h	0	
	read <u>jerk (high-or</u> 31	Hirst o der part) Secono	lata word		16	
Description	read <u>jerk (low-ord</u>	er part)			0	
Description	SetJerk for GetJerk ru Scaling ex .012345 co a value to 0329h and	ads the jerk register in the eads the contents of the ample: To load a jerk val punts/cycle ³ multiply by load of 53,021,371 (deci l a low word of 0ABBh y	ne parameter bu Jerk register. ue (time rate of 2 ³² or 4,294,967 mal) which corr when loading ea	change of ac ,296. In this of esponds to a ch word in ho	pecified axis . celeration) of example this gives high word of exadecimal.	
Restrictions	SetJerk is effect unti This com trapezoida	a buffered command. The next Update or Mu nand is used only with the l, velocity contouring, or	The value set usi ItiUpdate instru ne S-curve profi e electronic gear	ing this comm ction. le mode. It is profile mode	nand will not take not used with the es.	:
see	Set/GetA Set/GetV	cceleration, Set/GetDe elocity, MultiUpdate, L	celeration, Set Ipdate	/GetPosition	,	

Syntax	SetKaff <i>axis Ka</i> GetKaff <i>axis</i>	ff					
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	Kaff	<i>Түре</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	Units -		
Packet structure		SetK	aff				
	0	axis		93h			
	15	12 11 8 7	7		0		
write	Kaff	Data	a				
	15				0		
		GetKa	aff				
	15	12 11 8 7	7	94 h	0		
		Data	a		0		
read	Kaff						
	15				0		
Description	SetKaff sets the a specified axis.	acceleration feedforwa	rd gain of the d	igital servo fi	lter for the		
	GetKaff reads the	e current value of the a	acceleration fee	dforward gain	n.		
Restrictions	SetKaff is a buffered command. The value set using this command will not take effect until the next Update or MultiUpdate instruction.						
	This command is	not valid on the MC2	2400 and MC25	00.			
see	Set/GetKd, Set/ Update	GetKi, Set/GetKout,	Set/GetKp, Se	et/GetKvff, I	MultiUpdate,		
etKd (Se	ervo products	only)					
----------	---------------	-------					
----------	---------------	-------					

Syntax	SetKd <i>axis Kd</i> GetKd <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	Kd	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	Units -		
Packet structure		SetK	(d				
	0	axis		27 h			
	15	12 11 8 Dat	7		0		
write	Kd	Date	a				
	15				0		
		Get	(d	501			
	15	12 11 8	7	52 h	0		
		Data	a		-		
read	Kd				0		
Description	SetKd sets the de	erivative gain of the di	gital servo filter	r for the spec	tified axis.		
	Conta reads the	current value of the u	envative gam.				
Restrictions	SetKd is a buffered command. The value set using this command will not take effect until the next Update or MultiUpdate instruction.						
	This command is	s not valid on the MC2	2400 and MC25	500.			
<i>See</i>	Set/GetKaff, Se Update	t/GetKi, Set/GetKou	t, Set/GetKp,	Set/GetKvff	, MultiUpdate,		

Syntax	SetKi <i>axis Ki</i> GetKi <i>axis</i>				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
	Кі	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	Units -
Packet structure		Set	Ki		
	0	axis		26 h	
	15	12 11 8 7 Dat	a		0
write	Ki				
	15	Cot	V :		0
	0	axis	NI	51 h	
	15	12 11 8 7 Dat	2		0
read	Ki	Dat	a		
	15				0
Description	SetKi sets the in	tearal min of the digit	al servo filter f	or the specify	ad avis
Description				or the specifi	
	Geini reads the	current value of the in	tegral gain.		
Restrictions	This is a buffered until the next Up	d command. The valu odate or MultiUpdate i	ie set using this nstruction.	s command w	vill not take effect
	This command is	s not valid on the MC	2400 and MC2	500.	
see	Set/GetKaff, Se Update	et/GetKd, Set/GetKo	ut, Set/GetKp	, Set/GetKvt	ff, MultiUpdate,

SetKout axis Kout **Syntax** GetKout axis Arguments Name Instance Encoding Axis1 0 axis Axis2 1 Axis3 2 Axis4 3 Type Range Scaling Units 0 to 2¹⁶-1 $100/2^{16}$ Kout unsigned 16 bit % output **Packet structure** SetKout 9Eh 0 axis 12 11 8 7 Data 15 Λ write Kout GetKout **9F**h 0 axis 12 11 8 7 Data read Kout Description SetKout sets the output scale factor of the digital servo filter for the specified axis. The default value of Kout is 65535. GetKout reads the current value of the output scale factor. Example: To set the output scaling of the servo filter to half, set the Kout register to 32767. Restrictions This command is NOT buffered. It will take affect immediately after it is sent. This command is not valid on the MC2400 and MC2500. see Set/GetKaff, Set/GetKd, Set/GetKi, Set/GetKp, Set/GetKvff

9Eh 9Fh

Syntax	SetKp <i>axis Kp</i> GetKp <i>axis</i>				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
	Кр	<i>Туре</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	Units -
Packet structure		Set	Kp		
	0	axis	_	25 h	
	15	12 11 8 Dat	7 a		0
write	Кр				
	15	0.4			0
	0	axis	\ ρ	50 h	
	15	12 11 8	7		0
read	Κρ	Dai	a		
	15				0
Description	Sotk pasta than	non-ontional sain of the	dicital come f	lton fon the	an actived axia
Description	Seikp sets the p	roportional gain of the	e digital servo f	liter for the s	specified axis.
	GetKp reads the	current value of the p	roportional gai	n.	
Restrictions	SetKp is a buffer effect until the n	red command. The va ext Update or MultiUp	llue set using th odate instructio	is command n.	l will not take
	This command i	s not valid on the MC	2400 and MC25	500.	
see	Set/GetKaff, Se Update	et/GetKd, Set/GetKi,	Set/GetKout,	Set/GetKvf	f, MultiUpdate,

Syntax	SetKvff <i>axis Kvff</i> GetKvff <i>axis</i>							
Arguments	Name axis	Axis1 Axis2 Axis3 Axis4						
	Kvff	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	Units -			
Packet structure		SetKy	/ff					
	0	axis		2B h				
	15 12	2 11 8 7 Data	à		0			
write	Kvff							
	15	GetK	uff		0			
	0	axis		54 h				
	15 12	2 11 8 7 Data	3		0			
read	Kvff		~					
Description	¹⁵ SetKvff sets the v	elocity feedforward g	ain of the digita	l servo filter	o for the specified			
	CotKuff reads the	our of the r	rologity foodfor	ward onin				
Restrictions	SetKvff is a buffe	red command. The v	alue set using t	his command	l will not take			
	effect until the ne	ext Update or MultiUp	date instruction	n.				
	This command is	not valid on the MC2	2400 and MC25	500.				
see	Set/GetKaff, Se Update	t/GetKd, Set/GetKi,	Set/GetKout,	Set/GetKp, I	MultiUpdate,			

SetLimitSwitchMode GetLimitSwitchMode

Syntax	SetLimits GetLimits	SwitchMoo SwitchMoo	de <i>axis mode</i> de <i>axis</i>			
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	mode	off on	0 1			
Packet structure			SetLir	nitSwitchMode		
		0	axis		80 h	
	15	12	11	8 7 Data		0
write			0	Bata		mode
	15				1	0
			GetLir	nitSwitchMode		
		0	axis		81 h	
	15	12	11	8 7 Data		0
read						mode
	15				1	0
Description	SetLimitS specified	SwitchMode axis. When	enables (On) n the mode is	or disables (C	Dff) limit-switch ser xis will cause the co	using for the prresponding
	when it en immediat whether t	ch bits in the nters either ely stopped he axis is it	the positive of the positive o	is register and or negative lim lisabled these	it switches and the bits are not set, reg	axis will be ardless of

GetLimitSwitchMode returns the code for the current state of the limit-sensing mode.

Restrictions

see

GetActivityStatus, GetEventStatus

set

SetMotionCompleteMode GetMotionCompleteMode

Syntax	SetMotic GetMotic	SetMotionCompleteMode <i>axis mode</i> GetMotionCompleteMode <i>axis</i>						
Arguments	Name axis mode	Instance Axis1 Axis2 Axis3 Axis4 commanded actual	<i>Encodin</i> , 0 1 2 3 0	g				
Packet structure write	15 15	0 12 11	SetMotic axis	nCompleteMode	EBh	1	0 <u>mode</u> 0	
			GetMotic	onCompleteMode				
	15	12 11	axis	8 7	EC h		0	
				Data				
read	15					1	mode 0	
Description	SetMotion determine command reaches ze This mod When set condition Window (the SetSet trajectory after the t GetMotio	nCompleteMode es the motion-con ded mode the mode ero and no furthe e is unaffected b to actual mode to is true AND the (SetSettleWindor ettleTime comma profile motion s trajectory profile nCompleteMode	establist mplete s otion is of er motio y the act the moti e actual e w comm and. The o at a m motion e returns	hes the source tatus for the sp considered com n will occur wi ual encoder loo on complete b encoder positio and) for the m settle "timer" inimum a delay is complete. the current mo	for the comp becified axis . In plete when the thout an addition. it will be set within the it within the set within the set within the it within the set within the set within the it within the set within the set within the set within the it within the set within the set within the set within the it within the set within the set within the set within the it within the set within the set within the set within the it within the set within the set within the set within the set within the it within the set with	arison when she prof tional h when the ithin the voloop ero at t e cycle te mod	which set to ile velocity nost command. the above se Settle s specified by he end of the s will occur e.	
Restrictions								
see	Set/GetS	ettleTime, Set/0	GetSett	eWindow				

EBh **EC**h

Syntax	SetMotorBias <i>axis bias</i> GetMotorBias <i>axis</i>							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
	bias	<i>Туре</i> signed 16 bit	<i>Range</i> -2 ¹⁵ to 2 ¹⁵ -1	<i>Scaling</i> 100/2 ¹⁵	<i>Units</i> % output			
Packet structure		SetMoto	orBias					
	0	axis		0F h				
	15	12 11 8 Dat	7 A		0			
write	e bias		а 					
	15 0							
	GetMotorBias							
	15	12 11 8	7	2011	0			
		Dat	а					
read	1 <i>bias</i> 15				0			
Description	SetMotorBias se	ts the bias voltage of t	he digital servo	filter for the	specified axis.			
		eads the current bias vo	oltage of the dig	ital servo m	er.			
	Scaling example:							
	If it is desired th filter output thar 819 (decimal). Th	at a motor bias value on this register should b his corresponds to a lo	of -2.5 % of full e loaded with a baded hexadecin	scale be plac value of -2.5 nal value of (ed on the servo *32,768/100 = - FCCDh.			
Restrictions	This command is	s not valid on the MC.	2400 and MC25	00.				
see	Set/GetMotorC	ommand, Set/GetMo	otorLimit					

_

Syntax	SetMotorComn GetMotorComn	SetMotorCommand <i>axis value</i> GetMotorCommand <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	value	<i>Type</i> signed 16 bit	<i>Range</i> -2 ¹⁵ <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> 100/2 ¹⁵	<i>Units</i> % output		
Packet structure	0	SetMoto	rCommand	77h			
	15	12 11	3 7	11	0		
write			Data				
	15				0		
	0	GetMoto	rCommand	60h			
	15	12 11 axis	3 7	090	0		
read	Value		Data				
Teac	15				0		
Description	SetMotorComm For the MC2400 output waveform GetMotorComm	and loads the motor) series, this commann. nand reads the conte	e-command buffer ad is used to contr nts of the motor-o	register of the magning command bu	ne specified axis . tude of the ffer register.		
	Scaling example:	:					
	If it is desired th motor than this (decimal). This c	at a motor comman register should be lo corresponds to a hex	d value of 13.7 % aded with a value adecimal value of	of full scale k of 13.7 *32,7 1189h.	be output to the 68/100 = 4,489		
Restrictions	SetMotorComm MC2300 series.	and is valid only wh	en the motor is "o	off" for the M	IC2100 and		
	SetMotorComm will not take effe	and is a buffered co ect until the next Up	mmand. The valu date or MultiUpda	ie set using the instruction	nis command n.		
	This command i	is not available on th	e MC2500 series.				
see	Set/GetMotorB Update	ias, Set/GetMotorL	imit, Set/GetMot	torMode, Mu	ultiUpdate,		

Syntax	SetMotorLimit <i>axis limit</i> GetMotorLimit <i>axis limit</i>							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
	limit	Type unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> 100/2 ¹⁵	Units % output			
Packet structure		SetMot	orl imit					
	0	axis		06 h				
	15	12 11 8 Da	7 ta		0			
write	limit							
	15	0.48.4			0			
	0	axis	orlimit	07 h				
	15	12 11 8	7		0			
read	limit	Da	ta					
	15				0			
Description	SetMotorLimit set the digital servo value will be clip motor limit was ouput value show the output value useful for protect that a motor cor GetMotorLimit r Scaling example: If it is desired the register should be corresponds to a	ets the maximum valu filter of the specified of oped to the specified n set to 1,000 and the se uld be 1,100 the actual were -1,100 then it w cting amplifiers, motor nmand exceeding a ce eads the current motor that a motor limit of 75 be loaded with a value a hexadecimal value of	e for the motor axis. Motor com- notor command ervo filter deter- l output value w ould be clipped rs, or system mo- rtain value will or limit value. 5 % of full scale of 75.0 *32,768	output comm nmand values l limit. For ex mined that the vould be 1,000 to -1,000. The chanisms wh cause damage be established /100 = 24,57	nand allowed by s beyond this ample if the e current motor 0. Conversely if his command is hen it is known c. ed than this 6 (decimal). This			
Restrictions	This command of chipset is in ope This command i	only affects the motor n loop mode this com is not valid on the MC	ouput when in mand has no af 2400 and MC2!	closed loop 1 ffect. 500.	mode. When the			
see	Set/GetMotorB	ias, Set/GetMotorCo	mmand					

06h 07h

Syntax	SetMotor GetMotor	Mode <i>axis i</i> Mode <i>axis</i>	mode			
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	mode	Off On	0 1			
Packet structure			SetM	lotorMode		
	15) 12 11	axis	8 7	DCh	
		12 11		Data		
write	15		0		1	mode
			GetM	lotorMode	•	
	15) 12 11	axis	8 7	DDh	
	10	12 11		Data		0
read						mode
	15				1	0
Description	SetMotor	Mode determ	nines the mod	le of motor operat	ion. For serv	o products,
	when set to servo filte controls to	to On, the ax r. On the M he motor ou	tis is in <i>closed-i</i> IC2400 series tput.	<i>bop</i> mode, and is co and MC2500 serie	ontrolled by the s, the trajecto	he output of the ry generator
	When set	to Off , the a	xis is in <i>open-l</i>	oop mode, and is co	ontrolled by co	ommands placed
	directly in MC2500 s to Off.	to the motor series the traj	output regis	ter by the host. On tor is switched off	n the MC2400 when the mo) series and otor mode is set
	GetMotor	Mode return	s the current	motor mode		
				motor mode.		
Restrictions						
see	GetActivi	ityStatus, S	et/GetMotor	Command		

DCh DDh

83

SetNumberPhases (MC2300, MC2400 and MC2800 only) GetNumberPhases (MC2300, MC2400 and MC2800 only)

Syntax	SetNumberPhases <i>axis phases</i> GetNumberPhases <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3			
	phases	1Phase 2Phases 3Phases	1 2 3			
Packet structure			SetNur	nberPhases		
	0		axis		85 h	
	15	12 11		8 7 Data		0
write			0	Dala		phases
	15				2	0
			GetNu	mberPhases		
	0		axis		86 h	
	15	12 11		8 7 Data		0
read				Dulu		phases
	15				2	0
Description	SetNumber the specifie	Ph ases estab d axis .	lishes the	number of phase	es, 1, 2 or 3, for	commutation of
	GetNumber	Phases retur	ns the nur	nber of phases s	et for the axis .	
				Ĩ		
Restrictions	In PWM Sig	gn/Magnitude	e output m	ode, the number	r of phases can	be set to 1 or 2.
	In PWM 50	950 output mo	de, the nu	mber of phases	can be set to 1,	2 or 3.
see	GetPhaseC commands	ommand, Ini	tializePha	se, Set/GetPha	se Set/GetOut	putMode

Syntax	SetOutp GetOutp	etOutputMode <i>axis mode</i> ietOutputMode <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4			<i>Encoding</i> 0 1 2 3			
	mode	DAC PWMSigr PWM505	nMagnitud OMagnitud	e de	0 1 2			
Packet structure				SetOut	putMode			
		0	axis			E0 h		
	15	12	11	} ח	8 7 Jata			0
write				0	vala		mode	
	15					2	1	0
				GetOut	putMode			
	15	0	axis	s	8 7	6E h		
	15	12	11	D	ata			0
read							mode	
	15					2	1	0
Description	SetOutpu <i>axis</i> .	utMode dete	ermines the	e form	n of the motor of	output signal	of the s	pecified
	GetOutpu	utMode retu	urns the co	de for	the current mo	otor output r	node.	
Restrictions	This com	mand is no	ot available	on the	e MC2500.			
	If the nur available.	nber of ph	ases is set t	o 3, P	WM Sign/Mag	nitude outpu	ıt mode	is not

see

Syntax	SetPhaseAngle <i>axis angle</i> GetPhaseAngle <i>axis</i>							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>encoding</i> O 1 2 3					
	angle	Түре unsigned integer	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	Units counts			
Packet structure	-	SetPhase	Angle					
	15	12 11 8 7	7	84 h	0			
write	angla	Data	a					
write	15				0			
	GetPhaseAngle							
	0	12 11 axis	7	2C h	0			
	10	Data	a					
read	angle							
Description	SetPhaseAngle GetPhaseAngle an actual phase a and multiply by 3	sets the instantaneous returns the value of th ngle divide by the num 660.	commutation a le current phase lber of encoder	ngle for the spe angle. To conv counts per elec	cified axis . ert counts to trical cycle			
	For example if a value of 500 is retrieved using GetPhaseAngle and the counts electrical cycle value has been set to $2,000$ (SetPhaseCounts command) this corresponds to an angle of $(500/2,000)*360 = 90$ degrees current phase angle position.							
Restrictions	The specified ang the SetPhaseCo	gle must not exceed th unts command.	e number of co	ounts per electric	al cycle set by			
see	GetPhaseCommand, InitializePhase, Set/GetNumberPhases							

SetPhaseCorrectionMode (MC2300 and MC2800 only) GetPhaseCorrectionMode (MC2300 and MC2800 only)

Syntax	SetPhase GetPhase	SetPhaseCorrectionMode <i>axis mode</i> GetPhaseCorrectionMode <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	mode	Disabled Enabled	0 1				
Packet structure			SetPhaseC	orrectionMode			
	15	0 12 11	axis	8 7	E8 h		
				Data			
write	15		0		1	mode	
	GetPhaseCorrectionMode						
	(0	axis		E9 h		
	15	12 11		8 7 Data		0	
read				Dala		mode	
	15				1	0	
Description	SetPhase either 0 (C index sigr This ensu counts are counts/el GetPhase	eCorrectionM disabled) or f and is used to ares that the of e lost due to ectrical phase eCorrectionM	Node sets the p 1 (enabled). W update the co commutation a electrical noise e not being an Node returns t	bhase correction n When phase correction phase angle will remain of e, or due to the nu integer. he current phase of	node for the s tion is enabled angle each mo correct even if imber of enco correction mo	pecified axis to l, the encoder otor revolution. some encoder der de.	
Restrictions							
<i>see</i>	GetPhase Set/GetP	eCommand, haseCounts	InitializePhas	e, Set/GetNumb	erPhases,		

SetPhaseCounts (MC2300, MC2400 and MC2800 only) GetPhaseCounts (MC2300, MC2400 and MC2800 only)

Syntax	SetPhaseCounts <i>axis counts</i> GetPhaseCounts <i>axis</i>							
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	encoding O 1 2 3					
	counts	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	<i>Units</i> counts			
Packet structure	SetPhaseCounts							
	0	axis		75 h				
	15	12 11 8 7 Data	I		0			
write	count							
	15 0 GetPhaseCounts							
	0	axis	Jounts	7D h				
	15	12 11 8 7 Data			0			
read	count							
	15				0			
Description	SetPhaseCounts motor. If this val and phase correct command). The r poles. GetPhaseCounts	sets the number of er lue is not an integer th ion mode should be e number of electrical cy returns the number of	ncoder count p en the closest i nabled (See Se cles is equal to of counts per el	er electrical p nteger value s tPhaseCorre 1/2 the num ectrical cycle.	hase of the should be used, ctionMode ber of motor			
Restrictions	For MC2400:							
	The number of m SetPhaseCounts microsteps per ele example, to set 64 should be used. T step is 256, giving	hicrosteps per full step . The parameter used ectrical cycle (4 times t 4 microsteps per full st 'he maximum number g a maximum paramete	is set using the for this comma the desired num tep, the comma of microsteps er for this comma	e command and represent and SetPhase that can be g mand of 1024	s the number of osteps). So for e Counts 256 enerated per ful 4.			
see	GetPhaseComma	and, InitializePhase,	Set/GetNumb	erPhases				

SetPhaseInitializeMode (MC2300 and MC2800 only) GetPhaseInitializeMode (MC2300 and MC2800 only)

Syntax	SetPhaseInitializeMode <i>axis mode</i> GetPhaseInitializeMode <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> 0 1 2 3			
	mode	Algorithmic Hall-based	0 1			
Packet structure			SetPhasel	nitializeMode		
	()	axis		E4 h	
	15	12 11	Ē	Data		U
write			0			mode
	15				1	0
			GetPhasel	nitializeMode	F Eh	
	15	12 11	axis 8	7	EON	0
]	Data		
read	15				1	mode
Description	SetPhase initialized algorithm initial pha mode the GetPhase	elnitializeMode for commutation ic mode the chi sing based on the 3 Hall sensor se	establishes t on. The opt pset briefly he observed ignals are us returns the	he mode in which ions are Algorithm stimulates the mot motor response. red to determine th current initializatio	the specified ic and Hall-l or windings In Hall-base ie motor pha on mode.	d axis is to be based. In and sets the d initialization using.
			iciuilis ult		ii iii)uc.	
Restrictions	Algorithm move in b motor, me	nic mode should both directions, echanism, and h	l only be sel and that a b oad.	ected if it is known rief uncontrolled r	n that the axi nove can be	is is free to tolerated by the
see	GetPhase	Command, Ini	tializePhas	e, Set/GetNumbe	rPhases	

SetPhaseInitializeTime (MC2300 and MC2800 only) GetPhaseInitializeTime (MC2300 and MC2800 only)

Syntax		SetPhaseInitializeTime <i>axis time</i> SetPhaseInitializeTime <i>axis</i>					
Arguments		Name axis	Instance Axis1 Axis2 Axis3 Axis4	<i>encoding</i> O 1 2 3			
		time	Түре unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	<i>Units</i> cycles	
Packet structure			SetPhaseIniti	ializeTime			
	[0	axis	7	72 h	0	
		time					
write		time				0	
		GetPhaseInitializeTime					
		0	axis		7C h		
	-	15	12 11 8 7 Data	7		0	
r	ead	time	Duu	4			
		15				0	
Description		SetPhaseInitializ algorithmic phase each of the four s for more informa GetPhaseInitializ	eTime sets the time va e initialization procedu segments in the phase ation on algorithmic in reTime returns the cur	alue (in cycles) f ire. This value initialization alg itialization. rrent phase initi	to be used du determines th gorithm. See alization time	ring the ne duration of the User's guide	
Restrictions							

see

 $GetPhaseCommand,\ InitializePhase,\ Set/GetNumberPhases$

Syntax	SetPhaseOffset <i>axis offset</i> GetPhaseOffset <i>axis</i>					
Arguments	Name axis	Instance Axis1 Axis2 Axis3 Axis4 Type upsigned 16 bit	<i>Encoding</i> 0 1 2 3 <i>Range</i> 0 <i>to</i> 2 ¹⁵ -1	<i>Scaling</i>	Units	
Dackat structura	0//361	CatDhase		unity	counts	
	0	ovic	Unset	76h		
	15	12 11 8 7	,	7011	0	
		Data	1			
write	offset					
	15				0	
	0	GetPhase	Offset	7Dh		
	15	12 11 8 7	,		0	
		Data	1			
read	offset					
Description	SetPhaseOffset a maximum output on the commutat encountered. GetPhaseOffset To convert count counts per electri specified using Se set to 2,000 (SetF (500/2,000)*360	sets the offset from the value of phase A. This ion angle but will have returns the current value ts to a phase angle in d cal cycles and multiply etPhaseOffset and the PhaseCounts commant = 90 degrees phase and	e index mark of is command wil e an affect once lue of the phase legrees, divide b by 360. For ex e counts per ele- nd) this correspo- gle at the index	f the specified l have no imp the index put e offset. by the number ample if a val ctrical cycle v onds to an an mark.	l axis to the nediate effect lse is r of encoder ue of 500 is alue has been gle of	
Restrictions <i>see</i>	GetPhaseComm	and, InitializePhase,	Set/GetNumbe	erPhases		

Syntax		SetPhase GetPhase	Prescale <i>ax</i> Prescale <i>ax</i>	kis scale kis					
Arguments		Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
		mode	Off On	0 1					
Packet structure	•			S	etPhase	Prescale			
		0)	axis		7	E6 h		
		15	12 11	1	First dat	ta word		U	
	write				0			mode	
		15						1 0	
				G	etPhase	Prescale	F7 b		
		15	12 11	1 axis	8	7	E1 11	0	
					First dat	ta word			
	read	15			0			mode	
		15						1 0	
Description		SetPhase	Prescale O	n causes tl	he nun	nber of enco	der counts to b	e scaled by a	a
•		factor of -	1 before bei	ng used to	, calcu	late a commi	itation angle fo	r the specifi	ed
				· _1	1		ination angle 10	i the speem	·.1
		a high nur encoders.	nber of cou	nts per ele	ectrical	cycle, such a	is motors with	very high ac	curacy
		SetPhase	Prescale O	ff removes	s the so	cale factor.			
		GetPhase	Prescale re	eturns the	curren	t scaling mo	de		
					curren	i scanng moi			
Restrictions									
see		GetPhase	Command,	Initializel	Phase,	, Set/GetNu	mberPhases		

Syntax	SetPosition a GetPosition a	SetPosition <i>axis position</i> GetPosition <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
	position	<i>Type</i> signed 32 bit	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	Scaling unity	Units counts			
Packet structure		Set	Position					
	0	axis		10 h				
	15	12 11 First	8 7 data word		0			
,	write position (high-ord	ler part)						
	31	0	d datad		16			
,	write position (low-orde	Secon	d data word					
	15				0			
		Get	Position					
	0	axis		4A h				
	15	12 11 First	8 7 data word		0			
	read position (high-ord	ler part)						
	31	Sooon	d data word		16			
	read position (low-orde	er part)						
	15				0			
Description	SetPosition s Trapezoidal a	pecifies the trajectory on nd S-curve profile mo	destination of the s des.	specified ax	t is. It is used in the			
	Gerrosition f	eaus the contents of th	ie bullered positio	in register.				
Restrictions	SetPosition is take effect un	a buffered command. til the next Update or	The value set usi MultiUpdate instru	ng this com action.	nmand will not			
see	Set/GetAcce GetPositionE	leration, Set/GetDec rror, Set/GetPosition	eleration, Set/Get ErrorLimit, MultiL	tJerk, Set/ Jpdate, Up	GetVelocity, date			

SetPosition

GetPosition

SetPositionErrorLimit GetPositionErrorLimit

Syntax	SetPositionErro GetPositionErro	SetPositionErrorLimit <i>axis limit</i> GetPositionErrorLimit <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3					
	limit	Type unsigned 32 bit	<i>Range</i> O <i>to</i> 2 ³¹ -1	Scaling unity	Units counts			
Packet structure		SetPosition	nErrorLimit					
	0	axis		97 h				
	15	12 11 8	7		0			
	urito <i>limit</i> (high order par	FIRST Da	ta word]			
v		y Second (lata word		16			
	urite limit (low-order part)	rite limit (low order part)						
v	15				0			
		GetPositio	nErrorLimit					
	0	axis		98h				
	15	12 11 8	7		0			
		First da	ta word					
r	ead limit (high-order par	t)			16			
	51	Second of	lata word		10			
r	ead limit (low-order part)							
	15				0			
Description	SetPositionErro allowable by the limit, a motion o stop moving de GetPositionErro	orLimit sets the absolut e chipset for the specif error occurs. Such a n pending on the value s orLimit returns the cur	te value of the n fied axis . If the notion error ma set using the Se rent position er	naximum po position err y or may no tAutoStopM ror limit valu	sition error or exceeds this t cause the axis to ode command. 1e.			
Restrictions								

see GetPositionError, GetActualPosition, Set/GetPosition

Syntax	SetProfileMode <i>axis profile</i> GetProfileMode <i>axis</i>						
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3				
	profile	Trapezoidal Velocity contouring S-curve Electronic gear External	0 1 2 3 4				
Packet structure		Set	ProfileMode				
	15	0 axis	A	0 h	0		
	15	12 11	Data		0		
write	15	0		3 2	profile		
	10	Ge	ProfileMode	0 2	Ŭ		
		0 axis	A	1 h			
	15	12 11	8 7 Data		0		
read					profile		
	15			32	0		
Description	SetProfileMode sets the profile mode, selecting Trapezoidal, Velocity Contouring, S-curve, Electronic gear or External for the specified <i>axis</i> .						
	GetProfil specified	eMode returns the conte axis.	nts of the buffered pro	ofile-mo	de register for the		
Restrictions	SetProfile take effec	eMode is a buffered com	mand. The value set u or MultiUpdate instruct	ising this	s command will not		
see	Set/Get0 Update	GearMaster, Set/GetGea	arRatio, Set/GetBuffe	rFuncti	on, MultiUpdate,		

Syntax	SetSampleTime <i>time</i> GetSampleTime						
Arguments	Name time	Type unsigned 16 bit	<i>Range</i> 1 <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	Units μsec/cycle		
Packet structure		SetSam	pleTime				
write	15 [8 Da	7 ta	38 h	0		
	GetSampleTime						
	15	8 Da	7 ta	61 h	0		
read	<i>time</i> 15				0		
Description	SetSampleTime loop updates an Only certain val	e sets the cycle time fo d trajectory calculation ues are allowed as follo	or the chipset. The value is ows:	This is the tir expressed in	ne between servo microseconds.		
	Product	Allowed values					
	MC2100 series	multiples of 102 and	at least 102 µse	c per enable	d axis		
	MC2300 series	multiples of 154 and	at least 154 µse	c per enable	d axis		
	MC2400 series	multiples of 154 and	at least 154 μse	c per enable	d axis		
	GetSampleTime	e returns the current s	ample time valu	ıe.			
Restrictions	This command This command	is not available with th affects the cycle time f	e MC2500 serie for all axes.	es.			

see

SetSerialPortMode GetSerialPortMode



 Description
 SetSerialPortMode sets the configuration for the asynchronous serial port.

 Note: It is recommended that two stop bits be used for baud rates greater than 19200bps.

 Output

GetSerialPortMode returns the configuration for the asynchronous serial port.

8Bh

8Ch

Bit Number	Name	Instance	Encoding
0-3	transmission rate	1200 baud	0
		2400	1
		9600	2
		19200	3
		57600	4
		115200	5
		250000	6
		416667	7
·			
4-5	parity	none	0
		odd	1
		even	2
6	stop bits	1	0
		2	1
7-8	protocol	Point-to-point	0
		Multi-drop using address bit	2
		Multi-drop using idle-line	3
		detection	
11-15	multi-drop	Address 0	0
	address	Address 1	1
		Address 31	31

The following table shows the encoding of the data used by this command.

Restrictions

see

Set/GetDiagnosticPortMode

Syntax	SetSettleTime <i>axis time</i> GetSettleTime <i>axis</i>				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
	time	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	<i>Units</i> cycles
Packet structure		SetSettle	eTime		
	0	axis		AA h	
	15	12 11 8 Dat	7 a		0
write	time				
	15				0
	-	GetSettl	eTime		1
	0	axis	7	AB h	
	15	Dat	a		0
read	time				
	15				0
Description	SetSettleTime so remain within th status register) is GetSettleTime re	ets the time, in numbe e settle window before s set. eturns the current sett	r of cycles, that e the axis-settle le time for the	t the specified d indicator (i specified axi s	d <i>axis</i> must n the activity s.
Restrictions					
see	Set/GetMotion(CompleteMode, Set/0	GetSettleWind	low, GetAct	ivityStatus

Syntax	SetSettleWindow <i>axis window</i> GetSettleWindow <i>axis</i>					
Arguments	Name II axis A A A A A	<i>nstance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	1 window U	ype Insigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	Scaling unity	<i>Units</i> counts	
Packet structure		SetSettleV	Vindow			
	0	axis	-	BC h		
	15 12	11 8 7 Data	7		0	
write	window		-			
	15				0	
		GetSettleV	Vindow			
	0	axis		BDh		
	15 12	Data	7 A		U	
read	window	200	~			
	15				0	
Description	SetSettleWindow s remain for the dura (in the activity state GetSettleWindow	sets the position rang ation specified by Se as register) is set. returns the current v	ge within which e tSettleTime be value of the sett	n the specified efore the axis- tle window.	l axis must settled indicator	
Restrictions						
see	Set/GetMotionCo	mpleteMode, Set/G	GetSettleTime,	, GetActivity	Status	

SetSignalSense GetSignalSense

Syntax	SetSigr GetSigi	nalSense <i>axis mask</i> nalSense <i>axis</i>		
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3	
		Indicator		Bit Number
	mask	Encoder A	0001h	0
		Encoder B	0002h	1
		Encoder Index	0004h	2
		Encoder Home	0008h	3
		Positive limit	0010h	4
		Negative limit	0020h	5
		AxisIn	0040h	6
		Hall A	0080h	7
		Hall B	0100h	8
		Hall C	0200h	9
		AxisOut	0400h	10
		StepOutput	0800h	11
		MotorOutput	1000h	12
		reserved		13 - 15



Description SetSignalSense establishes the sense of the signals connected to the Signal Sense register by using a bitwise mask that corresponds to the bits of the Signal Status register, for the specified axis.

For each sense bit that is 0, the input is active low, or not inverted.

For each sense bit that is 1, the input is active high, or inverted.

Inverting the MotorOutput has the effect of reversing the direction of motion when a positive or negative motor command is given.

Inverting the StepOutput has the effect of reversing step signal generated by the MC2500 chipset. Refer to the User's Guide for more information.

GetSignalSense returns the current signal sense mask.

Restrictions Inverting ther encoder A,B, or index may prevent the index capture mechanism from operating correctly. Refer to the Navigator Technical Specifications for the index capture electrical requirements.

see GetSignalStatus

Syntax	SetStartVelocity <i>axis velocity</i> GetStartVelocity <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
	velocity	<i>Type</i> unsigned 32 bit	<i>Range</i> O <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	<i>Units</i> counts/cycle	
Packet structure		SetStart	Velocity			
	0	axis	7	6A h		
	15	First da	ta word		U	
write	e velocity (high-order p	part)				
	31	Second of	lata word		16	
write	e velocity (low-order pa	art)				
	15				0	
		GetStart	Velocity			
	15	12 11 8	7	6Bh	0	
		First da	ta word		•	
read	d velocity (high-order p	part)			16	
		Second of	lata word			
read	15 low-order pa	art)			0	
Description	SetStartVelocity GetStartVelocity	loads the starting vel reads the starting ve	ocity buffer reg locity buffer reg	ister for the s rister.	specified axis.	
	Scaling example: 65,536 (giving 11 0001 in the high (GetStartingVelo units of counts/	To load a starting ve 14,688) and load the r word and C000h in t ocity) must correspon cycle.	locity value of 1 esultant numbe he low word. R dingly be divide	.750 counts/ r as a 32 bit r etrieved num ed by 65,536	Cycle multiply by number, giving bers to convert to	
Restrictions	SetStartVelocity SetVelocity is a l take effect until	has no effect when t ouffered command. ⁷ the next Update or M	he chip is in S-c I'he value set us IultiUpdate instr	curve profile ing this comp cuction.	mode. mand will not	
see	Set/GetAcceler	ation, Set/GetDecel	eration, Set/Ge	etPosition		

Syntax	SetStepRar GetStepRar	SetStepRange <i>axis frequency</i> GetStepRange <i>axis</i>					
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4		<i>Encoding</i> O 1 2 3			
	frequency	5 MHz 625 kHz 156.25 kHz 39.062 kHz		1 4 6 8			
Packet structure			SetStepRa	nge			
	0	i	axis		CFh		
	15	12 11	8 7 Data			0	
write			0			freq	
	15				4	0	
			GetStepRa	nge			
	0	i	axis	-	CEh		
	15	12 11	8 7 Data			0	
read			Data			frea	
	15				4	0	
Description	SetStepRan example, if t command S	ge set the maxi he desired maxi etStepRange 4	imum pulse ra imum pulse ra should be iss	ate frequency ate is 200,000 sued.	for the specif pulses/secon	ied axis . d, the	For
	GetMaxStep	Rate returns th	ne maximum	pulse rate free	quency for the	e specifie	ed axis .
Restrictions	This comma	and is only avail	able on the M	IC2500 series.			

Syntax	SetStop GetStop	axis mode axis					
Arguments	Name axis mode	Instance Axis1 Axis2 Axis3 Axis4 NoStop AbruptStop SmoothStop	<i>Encoding</i> 0 1 2 3 0 1 2				
Packet structure			SetSto	opMode			
	45	0	axis	7	D0 h		
	15	12 11	8 D	ata			0
write	15		0		2	mode	
	15		0.404	Mada	2	I	0
		0	axis	риюае	D1 h		
	15	12 11	8	7			0
read				ala		mode	
	15				2	1	0
Description	SetStopM which ins uses the p mode to a set stop c	Node stops the spatial stops the spatial without a brogrammed dec stop the axis, or command.	pecified ax is ny decelera eleration va NoStop wh	s . The available s tion phase) stops lue and profile sl ich is generally u	stop modes s the axis, S hape for the used to turn	are Abr moothS e current off a pr	uptStop, top which t profile eviously
	Note: Afreset to the followed retrieved	ter an Update a ne NoStop conde by an Update co stop mode will b	buffered sto ition. In oth ommand and oe NoStop.	op command (Se ler words if the c l then by a GetS	tStopMode command S topMode co	e comma SetStopN ommand	nd) will lode is l, the
	GetStop	Mode returns the	e stop mode	set using SetSto	pMode.		
Restrictions	SmoothS SetStopM take effec	top mode is not lode is a buffere at until the next U	available in ed command Jpdate or M	the electronic-g l. The value set IultiUpdate instr	earing profi using this c uction.	lle. commane	d will not
see	MultiUpo	late, Update					

Syntax	SetTraceMode <i>mode</i> GetTraceMode				
Arguments	Name mode	<i>Instance</i> OneTime RollingBuffer	<i>Encoding</i> O 1		
Packet structure			SetTrace	eMode	
		0		B0 h	
	15		8 Dat	7 a	0
write	9		0		mode
	15			1	0
			GetTrace	eMode	
		0		B1 h	
	15		8 Dat	7 a	0
read	ł				mode
	15			1	0
Description	SetTraceMode sets the buffer usage for the next trace. In OneTime mode, the trace continues until the buffer is filled, then stops. In Rolling mode, the trace continues from the beginning of the buffer after the end is reached. Values stored when in the rolling mode are lost if they are not read before being overwritten by the wrapped data being traced and stored. GetTraceMode returns the code for the current buffer mode.				

Restrictions

see GetTraceStatus

B0h

SetTracePeriod GetTracePeriod

Syntax	SetTracePe GetTracePe	riod <i>period</i> riod							
Arguments	Name period	<i>Type</i> unsigned 16 bit	<i>Range</i> 1 <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	<i>Units</i> cycles				
Packet structure		SetTrac	cePeriod						
		0		B8 h					
	15	8	7		0				
		D	ata						
v	vrite period				0				
	15	15 0							
		GetTracePeriod							
	45	0	7	B9n	0				
	15	ů	ata		U				
r	read period		ala						
	15				0				
Description	SetTracePerson Successive tr	riod sets the time period, ace points.	expressed in nu	umber of cyc	les, between				
	GetTracePe	riod returns the current t	race period.						
Restrictions									
see	Set/GetSam	pleTime, Set/GetTrace	Start, Set/Get	TraceStop					

Syntax	tax SetTraceStart <i>triggerAxis condition triggerBit triggerSta</i> GetTraceStart				
Arguments	Name triggerAxis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
	condition	Description Immediate Next update Event Status register bit Activity Status register bit Signal Status register bit	0 1 2 3 4		
	triggerBit	Status register bit	0 <i>to</i> 15		
	triggerState	Triggering state of the bit	0 (value = 0) 1 (value = 1)		



Description SetTraceStart sets the condition for starting the trace. The *Immediate* condition requires no axis to be specified and the trace will begin upon execution of this instruction. The other four conditions require an axis to be specified; and when the condition for that axis is attained, the trace will begin.

When a status register bit is the trigger, the bit number and state must be included in the argument. The trace is started when the indicated bit reaches the specified state (0 or 1).

Once a trace has started, the trace-start indicator is reset and the **SetTraceStart** instruction must be reentered before another trace can be started.

GetTraceStart returns the the current trace-start condition.

Examples:

If it is desired that the trace begin on the next **Update** for axis 3, then a "1" is set for the condition, a "2" is set for the axis number, and bit number and state can be loaded with zeroes since they are not used.

B2h
If it is desired that the trace begin when bit 7 of the Activity Status register for axis 2 goes to 0 then the trace start is loaded as follows: A "3" is loaded for condition, a "1" is loaded for axis number, a "7" is loaded for bit number, and a "0" is loaded for state.

encoding of	register = event status	register = activity status	register = signal
"triggerBit"			status
0	Motion Complete	Phasing Initialized	Encoder A
1	Wrap-around	At maximum velocity	Encoder B
2	Breakpoint 1	Tracking	Encoder index
3	Position capture		Home
4	Motion error		Positive limit
5	In positive limit		Negative limit
6	In negative limit		AxisIn
7	Instruction error	Axis settled	Hall sensor 1
8		Motor on/off	Hall sensor 2
9		Position capture	Hall sensor 3
0Ah		In motion	
0Bh	Commutation error	In positive limit	
0Ch		In negative limit	
0Dh			
0Eh	Breakpoint 2		
0Fh			

The table below shows the corresponding value for combinations of *triggerBit* and *register*.

Restrictions

see

Set/GetBufferLength, GetTraceCount, Set/GetTraceMode, Set/GetTracePeriod, Set/GetTraceStop

Syntax	SetTraceSto GetTraceSto	p <i>triggerAxis condition triggerL</i> p	Bit triggerState	
Arguments	Name triggerAxis	<i>Instance</i> Axis1 Axis2 Axis3	<i>Encoding</i> O 1 2	
	condition	Axis4 Description Immediate Next update Event Status register bit Activity Status register bit Signal register bit	3 0 1 2 3 4	
	triggerBit	Status register bit	0 <i>to</i> 15	
	triggerState	Triggering state of the bit	0 (value = 0) 1 (value = 1)	



Description SetTraceStop sets the condition for stopping the trace. The *Immediate* condition requires no axis to be specified and the trace will stop upon execution of this instruction. The other four conditions require an axis to be specified; and when the condition for that axis is attained, the trace will stop.

When a status register bit is the trigger, the bit number and state must be included in the argument. The trace stops when the indicated bit reaches the specified state (0 or 1).

Once a trace has stopped, the trace-stop indicator is reset and the **SetTraceStop** instruction must be reentered before another trace can be stopped.

GetTraceStop returns the code for the current trace-stop condition.

Examples:

If it is desired that the trace stop on the next Update for axis 3, then a "1" is set for the condition, a "2" is set for the axis number, and bit number and state can be loaded with zeroes since they are not used.

If it is desired that the trace stop when bit 7 of the Activity status for axis 2 goes to 0 then the trace stop is loaded as follows: A "3" is loaded for condition, a "1" is loaded for axis number, a "7" is loaded for bit number, and a "0" is loaded for state.

encoding of	register = event status	register = activity status	register = signal
"triggerBit"			status
0	Motion Complete	Phasing Initialized	Encoder A
1	Wrap-around	At maximum velocity	Encoder B
2	Breakpoint 1	Tracking	Encoder index
3	Position capture		Home
4	Motion error		Positive limit
5	In positive limit		Negative limit
6	In negative limit		AxisIn
7	Instruction error	Axis settled	Hall sensor 1
8		Motor on/off	Hall sensor 2
9		Position capture	Hall sensor 3
0Ah		In motion	
0Bh	Commutation error	In positive limit	
0Ch		In negative limit	
0Dh			
0Eh	Breakpoint 2		
0Fh			

The table below shows the corresponding value for combinations of *triggerBit* and *register*.

Restrictions

see

Set/GetTraceCount, Set/GetTraceStart, Set/GetTraceStatus

Syntax	SetTraceVariable <i>variableNumber traceAxis variable</i> GetTraceVariable <i>variableNumber</i>					
Arguments	Name variableNumber	<i>Instance</i> Variable1 Variable2 Variable3 Variable4	<i>Encoding</i> O 1 2 3			
	axis	Axis1 Axis2 Axis3 Axis4	0 1 2 3			
	variable	None (disable the variable) Position error (32 bits) Commanded position (32 bits) Commanded velocity (32 bits) Commanded acceleration (32 bits) Actual position (32 bits) Actual velocity (32 bits) Motor command (16 bits) Chipset time (32 bits) Capture register (32 bits) Integral (32 bits) Derivative (16 bits) Event Status register (16 bits) Activity Status register (16 bits) Signal Status register (16 bits) Phase angle (16 bits) Phase offset (16 bits) Phase A (16 bits) Phase B (16 bits) Phase C (16 bits) Analog input 1 (16 bits) Analog input 3 (16 bits) Analog input 5 (16 bits) Analog input 5 (16 bits) Analog input 7 (16 bits) Analog input 8 (16 bits) PID Servo Error	0 1 2 3 4 5 6 7 8 9 Ah Bh Dh Eh 10 h 11 h 12 h 14 h 15 h 18 h 18 h 18 h 18 h 18 h 18 h 18			



see

Set/GetTrace commands

Syntax		SetTrackingWindow <i>axis window</i> GetTrackingWindow <i>axis</i>				
Arguments		Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3		
		window	Type unsigned 16 bit	<i>Range</i> O <i>to</i> 2 ¹⁵ -1	<i>Scaling</i> unity	Units counts
Packet structur	e		SetTracki	ngWindow		
		0	axis		A8 h	
		15	12 11 8 D	ata		0
	write	window				
		15	GetTracki	ingWindow		0
		0	axis		A9 h	
		15	12 11 8 D	7 ata		0
	read	window	U	ata		
		15				0
Description		SetTrackingWir the axis crosses 2 of the activity window, the trac GetTrackingWin	ndow sets boundaries the window boundar Status register) is set cking indicator is set t ndow returns the valu	for the actual portion of the sector of the	osition of the ion, the Trac axis returns t tracking win	e specified axis. If king indicator (bit o within the ndow.
Restrictions						

see

GetActivityStatus, GetActualPosition

SetVelocity GetVelocity						b	uffered	1 4
Syntax		SetVelocity ax GetVelocity ax	SetVelocity <i>axis velocity</i> GetVelocity <i>axis</i>					
Arguments		Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4		<i>Encoding</i> O 1 2 3			
		velocity	<i>Type</i> signed 32 bi	t	<i>Range</i> -2 ³¹ <i>to</i> 2 ³¹ -1	<i>Scaling</i> 1/2 ¹⁶	Units counts/cycle	
Packet structur	e			SetVel	ocity			
		0 15	12 11 axis	8	7	11 h	0	
	write	velocity (high-order	part)	First data	a word			
		31		Second da	ita word		16	
	write	velocity (low-order p	part)				0	
		10		GetVel	ocity		0	
		0	axis	Getven	ocity	4B h		
		15	12 11	8 First data	7 a word		0	
	read	velocity (high-order 31	part)				16	
	read	velocity (low-order r	oart)	Second da	ita word			
		15					0	
Description		SetVelocity load	ds the Maximun	n Veloc	ity buffer registe	er for the sp	pecified axis.	
		GetVelocity retu	urns the Maxim	um Vele	ocity buffer regi	ster.		
		Scaling example (giving 114,688) the high word a correspondingly	:: To load a velo) and load the re nd C000h in the 7 be divided by (ocity val esultant e low w 65,536 t	ue of 1.750 cour number as a 32 ord. Retrieved r to convert to un	nts/cycle m bit numbers numbers (G its of coun	nultiply by 65,536 r, giving 0001 in etVelocity) must ts/cycle.	
Restrictions		SetVelocity may SetVelocity is n The velocity mu SetVelocity is a take effect until	y not be issued v ot valid in Elect ist not be < 0 end buffered comment the next Updat	while ar tronic C xcept in nand. T e or M u	a axis is in motio Gearing profile n the Velocity-Co he value set usin Il tiUpdate instru	on with the node. ontouring p ng this com action.	S-curve profile. profile mode. mand will not	
see		Set/GetAccele MultiUpdate, U	ration, Set/Get Jpdate	Decele	ration, Set/Get	Jerk, Set/	GetPosition,	

Syntax	Update	axis				
Arguments	Name axis	<i>Instance</i> Axis1 Axis2 Axis3 Axis4	<i>Encoding</i> O 1 2 3			
Packet structure				Update		
	15	0 12	axis	8 7	1A h	0
Description	Update causes all buffered data parameters are copied into the corresponding rutime registers on the specified axis .					responding run-
	The following instruction is buffered: ClearPositionError.					
	The following trajectory parameters are buffered: Acceleration, Deceleration, GearRatio, Jerk, Position, ProfileMode, StartVelocity, Stop, and Velocity.					eceleration, elocity.
	The foll Kaff, K	lowing PID f d, Ki, Kp, an	ilter paramet d Kvff.	ers are buff	ered: DerivativeTime,	IntegrationLimit,
	The fol	lowing Motor	Command	parameters	is buffered: MotorCo	mmand.
Restrictions						
see	MultiUp	odate				

WriteBuffer

Syntax		WriteBuffer <i>bufferID value</i>					
Arguments		Name bufferID	<i>Type</i> unsigned 16 bit	<i>Range</i> O <i>to</i> 15	<i>Scaling</i> unity	Units -	
		value	signed 32 bit	-2 ³¹ to 2 ³¹ -1	unity	-	
Packet structu	re		WriteB	uffer			
		15	0	-	C8 h		
		15	First data	a word		0	
	write		0		bufferID		
		15	Second da	4 ata word	3	0	
	write	value (high-order par	i)				
		31	Third dat	a word		16	
	write	value (low-order part)					
		15				0	
Description		WriteBuffer writes the 32-bit <i>value</i> into the current location in the specified buffer. The current location is determined by adding the base address of the buffer (set by SetBufferStart), to the buffer's Write Index (set by SetBufferWriteIndex). After the contents have been read, the Write Index is incremented by 1; if the result is equal to the buffer length (set by SetBufferLength), the Index is reset to 0. Some chipset operations automatically change the write index such as during a trace. See the User's Guide for more details.					
Restrictions							
see		ReadBuffer, Set	/GetBufferWriteInde	x			

Writel0

Syntax		WriteIO address data					
Arguments		Name address	<i>Type</i> unsigned 8 bit	Range O <i>to</i> 255	Scaling unity	Units -	
		data	unsigned 16 bit	0 <i>to</i> 2 ¹⁶ -1	unity	-	
Packet structure	e		Write	elO			
		0	axis		82 h		
		15	First dat	a word		0	
	write		0	address			
		15	8 Second da	7 ata word		0	
	write	data					
		15				0	
Description		WritelO writes one 16-bit word of data to the device whose address is calculated by adding 1000h to <i>address</i> . (<i>address</i> is an offset from the base address, 1000h, of the Navigator's memory-mapped I/O space.) The format and interpretation of the 16-bit data word are dependent on the user-defined device being addressed. User-defined I/O can be used to implement a variety of features such as additional parallel I/O, flash memory for non-volatile configuration information storage, or display devices such as LED arrays.					
Restrictions							
see		ReadIO					

3 Instruction Summary Tables

3.1 Descriptions by Functional Category

Breakpoints and Interrupts

ClearInterrupt Reset interrupt line GetBreakpoint Get breakpoint type GetBreakpointValue Get breakpoint comparison value GetInterruptAxis Get the axes with pending interrupts GetInterruptMask Get interrupt mask SetBreakpoint Set breakpoint type SetBreakpointValue Set breakpoint comparison value SetInterruptMask Set interrupt mask Commutation GetCommutationMode Get the commutation mode GetNumberPhases Get the number of phases GetPhaseAngle Get current commutation phase angle GetPhaseCommand Get the motor output command for a given phase A, B, or C GetPhaseCorrectionMode Get phase correction mode GetPhaseCounts Get number of encoder counts per commutation cycle GetPhaseInitializeMode Get phase initialization mode GetPhaseInitializeTime Get the time parameters for algorithmic phase initialization GetPhaseOffset Get phase offset value GetPhasePrescale Get phasing prescaler InitializePhase Perform phase initialization procedure SetCommutationMode Set the commutation mode (Hall-based, sinusoidal, or microstepping) **SetNumberPhases** Set the number of phases (1, 2, or 3) SetPhaseAngle Set current commutation phase angle SetPhaseCorrectionMode Set phase correction mode (on or off) SetPhaseCounts Set number of encoder counts per commutation cycle SetPhaseInitializeMode Set phase initialization method (hall-based or algorithmic) SetPhaseInitializeTime Set the time parameters for algorithmic phase initialization SetPhaseOffset Set phase offset value SetPhasePrescale Set commutation prescaler mode (enable or disable) **Digital Servo Filter** ClearPositionError Set position error to 0 GetAutoStopMode Get auto stop mode GetDerivative Get the derivative of the error signal GetDerivativeTime Get derivative sampling time GetIntegral Get integrated position error value GetIntegrationLimit Get integration limit GetKaff Get acceleration feedforward gain GetKd Get derivative gain GetKi Get integral gain GetKout Get servo filter output scaler GetKp Get proportional gain GetKvff Get velocity feedforward gain **GetMotorBias** Get motor output bias GetMotorLimit Get motor output limit

GetPositionError GetPositionErrorLimit SetAutoStopMode SetDerivativeTime SetIntegrationLimit SetKaff SetKd SetKi SetKout SetKvff SetMotorBias SetMotorLimit SetPositionErrorLimit

Encoder

AdjustActualPosition GetActualPosition GetActualPositionUnits GetActualVelocity GetCaptureSource GetCaptureValue GetEncoderModulus GetEncoderSource GetEncoderToStepRatio SetActualPositionUnits SetCaptureSource SetEncoderModulus SetEncoderModulus SetEncoderSource SetEncoderSource SetEncoderSource

External RAM

GetBufferFunction GetBufferLength GetBufferReadIndex GetBufferStart GetBufferWriteIndex ReadBuffer SetBufferFunction SetBufferLength SetBufferReadIndex SetBufferStart SetBufferWriteIndex WriteBuffer

Motor Output

GetCurrentMotorCommand GetMotorCommand GetMotorMode GetOutputMode SetStepRange SetMotorCommand SetMotorMode SetOutputMode Get actual position error Get position error limit Set auto stop on position error (on or off) Set derivative sampling time Set integration limit Set acceleration feedforward gain Set derivative gain Set integral gain Set servo filter output scaler Set proportional gain Set velocity feedforward gain Set motor output bias Set motor output limit Set maximum position error limit

Sums the specified offset with the actual encoder position Get the actual encoder position Get the unit type returned for the actual encoder position Get the actual encoder velocity Get capture source Get current axis position capture value and reset the capture Get the full scale range of the parallel-word encoder Get encoder type Get encoder count to step ratio Set the actual encoder position Set the unit type returned for the actual encoder position Set the full scale range of the parallel-word encoder Set the full scale range of the parallel-word encoder Set the full scale range of the parallel-word encoder Set encoder type (incremental or 16-bit parallel word) Set encoder count to step ratio

Returns the buffer ID for a specified function Get the length of a memory buffer Get the buffer read pointer for a particular buffer Get the start location of a memory buffer Get the buffer write pointer for a particular buffer Read a long word value from a buffer memory location Assigns a buffer to the specified function Set the length of a memory buffer Set the buffer read pointer for a particular buffer Set the start location of a memory buffer Set the buffer write pointer for a particular buffer Write a long word value to a buffer memory location

Read the current motor command value Read buffered motor output command Get motor loop mode Get output mode Sets the allowable range (in KHz) for step output generation Set direct value to motor output register Set motor loop mode (on or off) Set motor output mode (PWM sign-magnitude, PWM 50%, or DAC)

Profile generation GetAcceleration GetCommandedAcceleration GetCommandedPosition GetCommandedVelocity GetDeceleration GetGearMaster GetGearRatio GetJerk GetPosition GetProfileMode GetStartVelocity GetStop GetVelocity MultiUpdate SetAcceleration SetDeceleration SetGearMaster SetGearRatio SetJerk SetPosition SetProfileMode SetStartVelocity SetStop SetVelocity Update Servo loop control GetAxisMode GetLimitSwitchMode GetMotionCompleteMode GetSampleTime GetSettleTime GetSettleWindow GetTime GetTrackingWindow SetAxisMode SetLimitSwitchMode SetMotionCompleteMode SetSampleTime SetSettleTime SetSettleWindow SetTrackingWindow Status Registers and AxisOut Indicator **GetActivityStatus** GetAxisOutSource GetEventStatus GetSignalStatus GetSignalSense ResetEventStatus SetAxisOutSource SetSignalSense

Traces

GetTraceCount

GetTraceMode

Get acceleration limit Get commanded (instantaneous desired) acceleration Get commanded (instantaneous desired) position Get commanded (instantaneous desired) velocity Get deceleration limit Get the electronic gear mode master axis and source Get command electronic gear rate Get jerk limit Get destination position Get current profile mode set using SetProfileMode Get start velocity Get stop command; abrupt, smooth, or none Get velocity limit Multiple axis immediate parameter update Set acceleration limit Set deceleration limit Set the master axis and source (actual or target-based) Set command electronic gear ratio Set jerk limit Set position limit Set profile mode (S-curve, trapezoidal, velocity-contouring, or electronic gear) Set start velocity Set stop command. (abrupt stop, smooth stop, or none) Set velocity limit Immediate parameter update

Get axis mode
Get limit switch mode
Get the motion complete mode
Get servo loop sample time
Get the axis-settled time
Get the settle-window boundary value
Get current chip set time (number of servo loops)
Get the tracking window boundary value
Set axis operation mode (enabled or disabled)
Set limit switching (on or off)
Set the motion complete mode (target-based or actual)
Set servo loop sample time
Set the axis-settled time
Set the settle-window boundary
Set the settle-window boundary
Set the tracking window boundary

Get Activity Status Get axis out signal monitor source Get event status word Get the current axis Signal Status register Get the interpretation of the Signal Status bits Reset bits in event status word Set axis out monitor signal source Set the interpretation of the Signal Status bits Get the number of traced data points Get the trace mode

GetTracePeriod GetTraceStart GetTraceStatus GetTraceStop GetTraceVariable SetTraceMode SetTracePeriod SetTraceStart SetTraceStop SetTraceVariable Miscellaneous	Get the trace period Get the trace start condition Get the trace status word Get the trace stop condition Get a trace variable setting Set the trace mode (rolling or one-time) Set the trace period Start the trace Stop the trace Set variable (i.e., data) to be traced
GetChecksum	Reads the internal chip checksum
GetDiagnosticPortMode	Get the diagnostic port valid instruction mode
GetHostIOError	Get the most recent I/O error code
GetSerialPort	Read serial-port configuration data
GetVersion	Get chipset software version information
NoOperation	Perform no operation, used to verify communications
ReadIO	Read user defined I/O value
Reset	Reset chipset
SetDiagnosticPortMode	Set the diagnostic port valid instruction mode (limited or full)
SetSerialPort	Set serial-port configuration data
WriteIO	Write user-defined I/O value

Alphabetical Listing 3.2

Note: Get/Set instruction pairs are shown together on the same line of the table

Instruction	Code	Instruction	Code
AdjustActualPosition	F5		
ClearInterrupt	AC		
ClearPositionError	47		
GetAcceleration	4C	SetAcceleration	90
GetActivityStatus	A6		
GetActualPosition	37	SetActualPosition	4D
GetActualPositionUnits	BF	SetActualPositionUnits	BE
GetActualVelocity	AD		
GetAutoStopMode	D3	SetAutoStopMode	D2
GetAxisMode	88	SetAxisMode	87
GetAxisOutSource	EE	SetAxisOutSource	ED
GetBreakpoint	D5	SetBreakpoint	D4
GetBreakpointValue	D7	SetBreakpointValue	D6
GetBufferFunction	СВ	SetBufferFunction	CA
GetBufferLength	C3	SetBufferLenath	C2
GetBufferReadIndex	C7	SetBufferReadIndex	C6
GetBufferStart	C1	SetBufferStart	CO
GetBufferWriteIndex	C5	SetBufferWriteIndex	C4
GetCaptureSource	D9	SetCaptureSource	01
GetChecksum	F8		20
GetCantureValue	36		
GetCommandedAcceleration	Δ7		
GetCommandedPosition	10		
CotCommanded Valacity	10		
CotCommutationMode		SatCommutationMada	⊏2
CetCurrentMeterCommand	20	SelCommutationwode	LZ
GetCurrentinotorCommand	3A 02	SatDocoloration	01
GetDeceleration	92	SelDeceleration	91
GelDerivetiveTime	9D	SatDariy (ati) (aTima	00
GetDerivative I Ime	9D	SetDerivative I Ime	90
	8A		89
	8E		8D
	DB	SetEncoderSource	DA
GetEncoderToStepRatio		SetEncoder I oStepRatio	DE
GetEventStatus	31		. –
GetGearMaster	AF	SetGearMaster	AE
GetGearRatio	59	SetGearRatio	14
GetHostIOError	A5		
GetIntegral	9A		0-
GetIntegrationLimit	96	SetIntegrationLimit	95
GetInterruptAxis	E1		
GetInterruptMask	56	SetInterruptMask	2F
GetJerk	58	SetJerk	13
GetKaff	94	SetKaff	93
GetKd	52	SetKd	27
GetKi	51	SetKi	26
GetKout	9F	SetKout	9E
GetKp	50	SetKp	25
GetKvff	54	SetKvff	2B
GetLimitSwitchMode	81	SetLimitSwitchMode	80
GetMotionCompleteMode	EC	SetMotionCompleteMode	EB
GetMotorBias	2D	SetMotorBias	0F
GetMotorCommand	69	SetMotorCommand	77
GetMotorLimit	07	SetMotorLimit	06

Navigator Motion Processor Programmer's Reference 123

Instruction	Code	Instruction	Code
GetMotorMode	DD	SetMotorMode	DC
GetNumberPhases	86	SetNumberPhases	85
GetOutputMode	6E	SetOutputMode	E0
GetPhaseAngle	2C	SetPhaseAngle	84
GetPhaseCommand	EA	, , , , , , , , , , , , , , , , , , ,	
GetPhaseCorrectionMode	E9	SetPhaseCorrectionMode	E8
GetPhaseCounts	7D	SetPhaseCounts	75
GetPhaseInitializeMode	E5	SetPhaseInitializeMode	E4
GetPhaseInitializeTime	7C	SetPhaseInitializeTime	72
GetPhaseOffset	7B	SetPhaseOffset	76
GetPhasePrescale	E7	SetPhasePrescale	E6
GetPosition	4A	SetPosition	10
GetPositionError	99		
GetPositionErrorLimit	98	SetPositionErrorLimit	97
GetProfileMode	A1	SetProfileMode	A0
GetSampleTime	61	SetSampleTime	38
GetSerialPort	8C	SetSerialPort	8B
GetSettleTime	AB	SetSettleTime	AA
GetSettleWindow	BD	SetSettleWindow	BC
GetSignalStatus	A4		
GetSignalSense	A3	SetSignalSense	A2
GetStartVelocity	6B	SetStartVelocity	6A
GetStop	D1	SetStop	D0
GetTime	3E		
GetTraceCount	BB		
GetTraceMode	B1	SetTraceMode	B0
GetTracePeriod	B9	SetTracePeriod	B8
GetTraceStart	B3	SetTraceStart	B2
GetTraceStatus	BA		
GetTraceStop	B5	SetTraceStop	B4
GetTraceVariable	B7	SetTraceVariable	B6
GetTrackingWindow	A9	SetTrackingWindow	A8
GetVelocity	4B	SetVelocity	11
GetVersion	8F		
InitializePhase	7A		
MultiUpdate	5B		
NoOperation	00		
ReadAnalog	EF		
ReadBuffer	C9		
ReadIO	83		
Reset	39		
ResetEventStatus	34		
Update	1A		
WriteBuffer	C8		
WriteIO	82		

3.3 Numeric Listing

Code	Instruction	Code	Instruction	Code	Instruction
00	NoOperation	80	SetLimitSwitchMode	B6	SetTraceVariable
06	SetMotorLimit	81	GetLimitSwitchMode	B7	GetTraceVariable
07	GetMotorLimit	82	WriteIO	B8	SetTracePeriod
0F	SetMotorBias	83	ReadIO	B9	GetTracePeriod
10	SetPosition	84	SetPhaseAngle	BA	GetTraceStatus
11	SetVelocity	85	SetNumberPhases	BB	GetTraceCount
13	SetJerk	86	GetNumberPhases	BC	SetSettleWindow
14	SetGearRatio	87	SetAxisMode	BD	GetSettleWindow
1A	Update	88	GetAxisMode	BE	SetActualPositionUnits
1D	GetCommandedPosition	89	SetDiagnosticPortMode	BF	GetActualPositionUnits
1E	GetCommandedVelocity	8A	GetDiagnosticPortMode	C0	SetBufferStart
25	SetKp	8B	SetSerialPort	C1	GetBufferStart
26	SetKi	8C	GetSerialPort	C2	SetBufferLength
27	SetKd	8D	SetEncoderModulus	C3	GetBufferLength
2B	SetKvff	8E	GetEncoderModulus	C4	SetBufferWriteIndex
2C	GetPhaseAngle	8F	GetVersion	C5	GetBufferWriteIndex
2D	GetMotorBias	90	SetAcceleration	C6	SetBufferReadIndex
2F	SetInterruptMask	91	SetDeceleration	C7	GetBufferReadIndex
31	GetEventStatus	92	GetDeceleration	C8	WriteBuffer
34	ResetEventStatus	93	SetKaff	C9	ReadBuffer
36	GetCaptureValue	94	GetKaff	CA	SetBufferFunction
37	GetActualPosition	95	SetIntegrationLimit	CB	GetBufferFunction
38	SetSampleTime	96	GetIntegrationLimit	D0	SetStop
39	Reset	97	SetPositionErrorLimit	D1	GetStop
3A	GetCurrentMotorCommand	98	GetPositionErrorLimit	D2	SetAutoStopMode
3E	GetTime	99	GetPositionError	D3	GetAutoStopMode
47	ClearPositionError	9A	GetIntegral	D4	SetBreakpoint
4A	GetPosition	9B	GetDerivative	D5	GetBreakpoint
4B	GetVelocity	9C	SetDerivativeTime	D6	SetBreakpointValue
4C	GetAcceleration	9D	GetDerivativeTime	D7	GetBreakpointValue
4D	SetActualPosition	9E	SetKout	D8	SetCaptureSource
50	GetKp	9F	GetKout	D9	GetCaptureSource
51	GetKi	A0	SetProfileMode	DA	SetEncoderSource
52	GetKd	A1	GetProfileMode	DB	GetEncoderSource
54	GetKvff	A2	SetSignalSense	DC	SetMotorMode
56	GetInterruptMask	A3	GetSignalSense	DD	GetMotorMode
58	GetJerk	A4	GetSignalStatus	E0	SetOutputMode
59	GetGearRatio	A5	GetHostIOError	E1	GetInterruptAxis
5B	MultiUpdate	A6	GetActivityStatus	E2	SetCommutationMode
61	GetSampleTime	A7	GetCommandedAcceleration	E3	GetCommutationMode
68	SetEncoderToStepRatio	A8	SetTrackingWindow	E4	SetPhaseInitializeMode
69	GetMotorCommand	A9	GetTrackingWindow	E5	GetPhaseInitializeMode
6A	SetStartVelocity	AA	SetSettleTime	E6	SetPhasePrescale
6B	GetStartVelocity	AB	GetSettleTime	E7	GetPhasePrescale
6E	GetOutputMode	AC	ClearInterrupt	E8	SetPhaseCorrectionMode
6F	GetEncoderToStepRatio	AD	GetActualVelocity	E9	GetPhaseCorrectionMode
72	SetPhaseInitializeTime	AE	SetGearMaster	EA	GetPhaseCommand
75	SetPhaseCounts	AF	GetGearMaster	EB	SetMotionCompleteMode
76	SetPhaseOffset	B0	SetTraceMode	EC	GetMotionCompleteMode
77	SetMotorCommand	B1	GetTraceMode	ED	SetAxisOutSource
7A	InitializePhase	B2	SetTraceStart	EE	GetAxisOutSource
7B	GetPhaseOffset	B3	GetTraceStart	EF	ReadAnalog
7C	GetPhaseInitializeTime	B4	SetTraceStop	F5	AdjustActualPosition
7D	GetPhaseCounts	B5	GetTraceStop	F8	GetChecksum